® 2025 ANSYS, Inc. or its affiliated companies
Unauthorized use, distribution, or duplication is prohibited.

Ansys CFX Reference Guide

Ansys

ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 2025 R1
January 2025

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

Copyright and Trademark Information

© 2025 ANSYS, Inc. Unauthorized use, distribution or duplication is prohibited.

ANSYS, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. ICEM CFD is a trademark used by ANSYS, Inc. under license. CFX is a trademark of Sony Corporation in Japan. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXlm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the legal information in the product help files for the complete Legal Notice for ANSYS proprietary software and third-party software. If you are unable to access the Legal Notice, contact ANSYS, Inc.

Published in the U.S.A.

Table of Contents

1. Ansys CFX Launcher 29
1.1. The Ansys CFX Launcher Interface 29
1.1.1. Menu Bar 29
1.1.1.1. File Menu 29
1.1.1.1.1. Save As 29
1.1.1.1.2. Quit 30
1.1.1.2. Edit Menu 30
1.1.1.2.1. Clear 30
1.1.1.2.2. Find 30
1.1.1.2.3. Options 30
1.1.1.2.3.1. Graphical User Interface Style 30
1.1.1.2.3.2. Font and Formatted Font 30
1.1.1.3. CFX Menu 30
1.1.1.3.1. CFX-Pre 30
1.1.1.3.2. CFX-Solver Manager 30
1.1.1.3.3. CFD-Post 30
1.1.1.3.4. Other CFX Applications 31
1.1.1.4. Show Menu 31
1.1.1.4.1. Show Installation 31
1.1.1.4.2. Show All 31
1.1.1.4.3. Show System 31
1.1.1.4.4. Show Variables 31
1.1.1.4.5. Show Patches 31
1.1.1.5. Tools Menu 31
1.1.1.5.1. Command Line 31
1.1.1.5.2. Configure User Startup Files (Linux only) 32
1.1.1.5.3. Edit File 32
1.1.1.5.4. Edit Site-wide Configuration File 32
1.1.1.6. User Menu 32
1.1.1.7. Help Menu 32
1.1.2. Toolbar 33
1.1.3. Working Directory Selector 33
1.1.4. Output Window 33
1.2. Customizing the Ansys CFX Launcher 33
1.2.1. CCL Structure 34
1.2.1.1. GROUP 34
1.2.1.2. APPLICATION 34
1.2.1.2.1. Including Environment Variables 36
1.2.1.3. DIVIDER 37
1.2.2. Example: Adding the Windows Calculator 37
2. Volume Mesh Import API 39
2.1. Valid Mesh Elements in CFX 39
2.2. Creating a Custom Mesh Import Executable for CFX-Pre 40
2.2.1. Compiling Code with the Mesh Import API 41
2.2.2. Linking Code with the Mesh Import API 41
2.2.2.1. Linking a Customized Mesh Import Executable on a Windows Platform 41
2.2.2.1.1. Linking a Customized C Mesh Import Executable on a Windows Platform 42
2.2.2.1.2. Linking a Customized Fortran Mesh Import Executable on a Windows Platform 42
2.2.2.2. Linking a Customized Mesh Import Executable on a Linux Platform 42

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.2.2.2.1. Linking a Customized C Mesh Import Executable on a Linux Platform 42
2.2.2.2.2. Linking a Customized Fortran Mesh Import Executable on a Linux Platform 43
2.3. Details of the Mesh Import API 43
2.3.1. Defined Constants 44
2.3.1.1. Element Types 44
2.3.1.2. Region Types 44
2.3.2. Initialization Routines 45
2.3.2.1. cfxImportStatus 45
2.3.2.2. cfxImportInit 45
2.3.2.3. cfxImportTest 45
2.3.3. Termination Routines 46
2.3.3.1. cfxImportDone 46
2.3.3.2. cfxImportTotals 46
2.3.4. Error Handling Routines 46
2.3.4.1. cfxImportError 46
2.3.4.2. cfxImportFatal 47
2.3.5. Node Routines 47
2.3.5.1. cfxImportNode 47
2.3.5.2. cfxImportGetNode 47
2.3.5.3. cfxImportNodeList 47
2.3.6. Element Routines 48
2.3.6.1. cfxImportElement 48
2.3.6.2. cfxImportGetElement 49
2.3.6.3. cfxImportElementList 49
2.3.6.4. cfxImportGetFace 50
2.3.6.5. cfxImportFindFace 51
2.3.7. Primitive Region Routines 51
2.3.7.1. cfxImportBegReg 51
2.3.7.2. cfxImportAddReg 51
2.3.7.3. cfxImportEndReg 52
2.3.7.4. cfxImportRegion 52
2.3.7.5. cfxImportRegionList 52
2.3.7.6. cfxImportGetRegion 52
2.3.8. Composite Regions Routines 53
2.3.8.1. cfxImportBegCompRegion 53
2.3.8.2. cfxImportAddCompRegComponents 53
2.3.8.3. cfxImportEndCompReg 53
2.3.8.4. cfxImportCompositeRegion 53
2.3.9. Explicit Node Pairing 53
2.3.9.1. cfxImportMap 54
2.3.10. Fortran Interface 54
2.3.10.1. cfxinit 54
2.3.10.2. cfxtest 54
2.3.10.3. cfxunit 54
2.3.10.4. cfxwarn 54
2.3.10.5. cfxfatl 54
2.3.10.6. cfxdone 55
2.3.10.7. cfxnode 55
2.3.10.8. cfxnodg 55
2.3.10.9. cfxnods 55
2.3.10.10. cfxelem 55

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.10.11. cfxeleg 55
2.3.10.12. cfxeles 56
2.3.10.13. cfxfacd 56
2.3.10.14. cfxface 56
2.3.10.15. cfxffac 56
2.3.10.16. cfxregn 56
2.3.10.17. cfxregb 56
2.3.10.18. cfxrega 57
2.3.10.19. cfxrege 57
2.3.10.20. cfxregs 57
2.3.10.21. cfxregg 57
2.3.10.22. cfxcmpb 57
2.3.10.23. cfxcmpa 58
2.3.10.24. cfxcmpe 58
2.3.11. Unsupported Routines Previously Available in the API 58
2.4. An Example of a Customized C Program for Importing Meshes into CFX-Pre 58
2.5. Import Programs 58
2.5.1. Ansys 59
2.5.2. CFX Def/Res 59
2.5.3. CFX-4 60
2.5.4. CFX-5.1 60
2.5.5. CFX-TfC 61
2.5.6. CGNS 62
2.5.7. Fluent 63
2.5.8. GridPro/az3000 63
2.5.9. I-DEAS 64
2.5.10. ICEM CFX 64
2.5.11. PATRAN 64
2.5.12. NASTRAN 64
2.5.13. CFX-TASCflow 65
3. Mesh and Results Export API 67
3.1. Creating a Customized Export Program 67
3.1.1. An Example of an Export Program 68
3.1.1.1. File Header 68
3.1.1.2. Allowed Arguments 68
3.1.1.3. Main Program Initialization 69
3.1.1.4. Checking File Names 70
3.1.1.5. Opening the CFX Results File 71
3.1.1.6. Timestep Setup 72
3.1.1.7. Geometry File Output 72
3.1.1.8. Template Results File 74
3.1.1.9. Creating Files with Results for Each Variable 76
3.1.2. Example of Output Produced 77
3.1.2.1. example.geom 78
3.1.2.2. example.res 79
3.1.2.3. example.s01 79
3.1.3. Source Code for getargs.c 79
3.2. Compiling Code with the Mesh and Results Export API 80
3.3. Linking Code with the Mesh and Results Export API 80
3.3.1. Linking a Customized Mesh and Results Export Executable on a Windows Platform 81
3.3.1.1. Linking a Customized C Mesh and Results Export Executable on a Windows Platform 81

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.3.1.2. Linking a Customized Fortran Mesh and Results Export Executable on a Windows Platform 81
3.3.2. Linking a Customized Mesh and Results Export Executable on a Linux Platform 82
3.3.2.1. Linking a Customized C Mesh and Results Export Executable on a Linux Platform 82
3.3.2.2. Linking a Customized Fortran Mesh and Results Export Executable on a Linux Platform 82
3.4. Details of the Mesh Export API 83
3.4.1. Defined Constants and Structures 83
3.4.1.1. Element Types 83
3.4.1.2. Volume List Types 83
3.4.1.3. Region List Types 84
3.4.1.4. Count Entries 84
3.4.1.5. Node Data Structure 84
3.4.1.6. Element Data Structure 85
3.4.2. Initialization and Error Routines 85
3.4.2.1. cfxExportInit 85
3.4.2.2. cfxExportDone 85
3.4.2.3. cfxExportError 85
3.4.2.4. cfxExportFatal 86
3.4.3. Zone Routines 86
3.4.3.1. cfxExportZoneCount 86
3.4.3.2. cfxExportZoneSet 86
3.4.3.3. cfxExportZoneGet 86
3.4.3.4. cfxExportZoneFree 87
3.4.3.5. cfxExportZoneIsRotating 87
3.4.3.6. cfxExportZoneMotionAction 87
3.4.4. Node Routines 87
3.4.4.1. cfxExportNodeCount 88
3.4.4.2. cfxExportNodeList 88
3.4.4.3. cfxExportNodeGet 88
3.4.4.4. cfxExportNodeFree 88
3.4.4.5. cfxExportNodeUnits 88
3.4.5. Element Routines 88
3.4.5.1. cfxExportElementCount 89
3.4.5.2. cfxExportElementList 89
3.4.5.3. cfxExportElementGet 89
3.4.5.4. cfxExportElementFree 90
3.4.6. Region Routines 90
3.4.6.1. cfxExportRegionCount 90
3.4.6.2. cfxExportRegionSize 90
3.4.6.3. cfxExportRegionName 90
3.4.6.4. cfxExportRegionList 91
3.4.6.5. cfxExportRegionGet 91
3.4.6.6. cfxExportRegionFree 91
3.4.7. Face Routines 91
3.4.7.1. cfxExportFaceNodes 91
3.4.8. Volume Routines 93
3.4.8.1. cfxExportVolumeCount 93
3.4.8.2. cfxExportVolumeSize 93
3.4.8.3. cfxExportVolumeName 93
3.4.8.4. cfxExportVolumeList 93

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.8.5. cfxExportVolumeGet 93
3.4.8.6. cfxExportVolumeFree 94
3.4.9. Boundary Condition Routines 94
3.4.9.1. cfxExportBoundaryCount 94
3.4.9.2. cfxExportBoundaryName 94
3.4.9.3. cfxExportBoundaryType 95
3.4.9.4. cfxExportBoundarySize 95
3.4.9.5. cfxExportBoundaryList 95
3.4.9.6. cfxExportBoundaryGet 95
3.4.9.7. cfxExportBoundaryFree 96
3.4.10. Variable Routines 96
3.4.10.1. cfxExportVariableCount 96
3.4.10.2. cfxExportVariableSize 96
3.4.10.3. cfxExportVariableName 96
3.4.10.4. cfxExportVariableList 97
3.4.10.5. cfxExportVariableGet 97
3.4.10.6. cfxExportVariableFree 97
3.4.10.7. cfxExportVariableQuantityDimensions 97
3.4.10.8. cfxExportVariableUnitsString 98
3.4.11. Timestep Routines 98
3.4.11.1. cfxExportTimestepCount 98
3.4.11.2. cfxExportTimestepTimeGet 98
3.4.11.3. cfxExportTimestepNumGet 98
3.4.11.4. cfxExportTimestepSet 98
4. Remeshing Guide 101
4.1. User Defined Remeshing 102
4.1.1. Remeshing with Key-Frame Meshes 103
4.1.2. Remeshing with Automatic Geometry Extraction 104
4.2. ICEM CFD Replay Remeshing 104
4.2.1. Steps to Set Up a Simulation Using ICEM CFD Replay Remeshing 106
4.3. Directory Structure and Files Used During Remeshing 107
4.4. Additional Considerations 108
4.4.1. Mesh Re-Initialization During Remeshing 108
4.4.2. Software License Handling 109
4.4.3. Results File Option 109
5. Reference Guide for Mesh Deformation and Fluid-Structure Interaction 111
5.1. Mesh Deformation 111
5.1.1. Mesh Folding: Negative Sector and Element Volumes 111
5.1.2. Applying Large Displacements Gradually 111
5.1.3. Consistency of Mesh Motion Specifications 112
5.1.4. Solving the Mesh Displacement Equations and Updating Mesh Coordinates 112
5.1.5. Mesh Displacement Diffusion Scheme 112
5.1.6. Mesh Displacement vs. Total Mesh Displacement 115
5.1.7. Simulation Restart Behavior 115
5.2. Fluid Structure Interaction 116
5.2.1. Unidirectional (One-Way) FSI 116
5.2.1.1. Using CFX Only 116
5.2.1.2. Using CFX and the Mechanical Application 117
5.2.1.2.1. Importing Data from the Mechanical Application Solver 117
5.2.1.2.2. Mechanical Import/Export Example: One-Way FSI Data Transfer 117
5.2.1.3. Using CFX and Other CAE Software 118

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5.2.2. Bidirectional (Two-Way) FSI 118
5.2.2.1. Using CFX Only 118
5.2.2.2. Using CFX and Other CAE Software 118
6. CFX Best Practices Guide for Numerical Accuracy 119
6.1. An Approach to Error Identification, Estimation and Validation 119
6.2. Definition of Errors in CFD Simulations 120
6.2.1. Numerical Errors 121
6.2.1.1. Solution Errors 121
6.2.1.2. Spatial Discretization Errors 122
6.2.1.3. Time Discretization Errors 123
6.2.1.4. Iteration Errors 124
6.2.1.5. Round-off Error 124
6.2.1.6. Solution Error Estimation 124
6.2.2. Modeling Errors 127
6.2.3. User Errors 128
6.2.4. Application Uncertainties 128
6.2.5. Software Errors 129
6.3. General Best Practice Guidelines 129
6.3.1. Avoiding User Errors 129
6.3.2. Geometry Generation 130
6.3.3. Grid Generation 130
6.3.4. Model Selection and Application 131
6.3.4.1. Turbulence Models 132
6.3.4.1.1. One-equation Models 133
6.3.4.1.2. Two-equation Models 133
6.3.4.1.3. Second Moment Closure (SMC) Models 133
6.3.4.1.4. Large Eddy Simulation Models 134
6.3.4.1.5. Wall Boundary Conditions 135
6.3.4.1.5.1. Wall Function Boundary Conditions 135
6.3.4.1.5.2. Integration to the wall (low-Reynolds number formulation) 135
6.3.4.1.5.3. Mixed formulation (automatic near-wall treatment) 136
6.3.4.1.5.4. Recommendations for Model Selection 136
6.3.4.2. Heat Transfer Models 136
6.3.4.3. Multi-Phase Models 136
6.3.5. Reduction of Application Uncertainties 137
6.3.6. CFD Simulation 137
6.3.6.1. Target Variables 137
6.3.6.2. Minimizing Iteration Errors 138
6.3.6.3. Minimizing Spatial Discretization Errors 138
6.3.6.4. Minimizing Time Discretization Errors 139
6.3.6.5. Avoiding Round-Off Errors 140
6.3.7. Handling Software Errors 140
6.4. Selection and Evaluation of Experimental Data 140
6.4.1. Verification Experiments 141
6.4.1.1. Description 141
6.4.1.2. Requirements 141
6.4.2. Validation Experiments 142
6.4.2.1. Description 142
6.4.2.2. Requirements 142
6.4.3. Demonstration Experiments 143
6.4.3.1. Description 144

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6.4.3.2. Requirements 144
7. CFX Best Practices Guide for Cavitation 145
7.1. Approaches to Modeling Cavitation 145
7.2. Liquid Pumps 146
7.2.1. Pump Performance without Cavitation 146
7.2.2. Pump Performance with Cavitation 146
7.2.3. Procedure for Plotting Performance Curve 148
7.2.4. Setup 148
7.2.5. Convergence Tips 149
7.2.6. Postprocessing 149
8. CFX Best Practices Guide for Combustion 151
8.1. Gas Turbine Combustors 151
8.1.1. Setup 151
8.1.1.1. Steady-state vs. Transient 151
8.1.1.2. Turbulence Model 151
8.1.1.3. Reference Pressure 151
8.1.1.4. Combustion Model 151
8.1.2. Reactions 152
8.1.3. Convergence Tips 152
8.1.4. Postprocessing 153
8.2. Combustion Modeling in HVAC Cases 153
8.2.1. Set Up 154
8.2.2. Convergence Tips 154
8.2.3. Postprocessing 154
9. CFX Best Practices Guide for HVAC 155
9.1. HVAC Simulations 155
9.1.1. Setting Up HVAC Simulations 155
9.1.1.1. Buoyancy 155
9.1.1.2. Thermal Radiation 156
9.1.1.2.1. Thermal Radiation Model 156
9.1.1.2.2. Spectral Model 156
9.1.1.2.3. Scattering Model 156
9.1.1.3. CHT (Conjugate Heat Transfer) Domains 157
9.1.1.4. Mesh Quality 158
9.1.1.5. Fans 158
9.1.1.6. Thermostats 158
9.1.1.7. Collections of Objects 158
9.2. Convergence Tips 158
10. CFX Best Practices Guide for Multiphase 159
10.1. Bubble Columns 159
10.1.1. Setup 159
10.1.2. Convergence Tips 160
10.1.3. Postprocessing 160
10.2. Mixing Vessels 160
10.2.1. Setup 160
10.3. Free Surface Applications 161
10.3.1. Setup 161
10.3.2. Convergence Tips 161
10.4. Multiphase Flow with Turbulence Dispersion Force 162
11. CFX Best Practices Guide for Turbomachinery 163
11.1. Gas Compressors and Turbines 163

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11.1.1. Setup for Simulations of Gas Compressors and Turbines 163
11.1.2. Convergence Tips 164
11.1.3. Computing Speedlines for a Machine 164
11.1.4. Postprocessing 165
11.2. Liquid Pumps and Turbines 166
11.2.1. Setup for Simulations of Liquid Pumps and Turbines 166
11.2.2. Convergence Tips 167
11.2.3. Postprocessing 167
11.3. Fans and Blowers 167
11.3.1. Setup for Simulations of Fans and Blowers 167
11.3.2. Convergence Tips 168
11.3.3. Postprocessing 168
11.4. Frame Change Models 168
11.4.1. Frozen Rotor 169
11.4.2. Stage (Mixing-Plane) 169
11.4.3. Transient Rotor-Stator 169
11.5. Domain Interface Setup 169
11.5.1. General Considerations 170
11.5.2. Case 1: Impeller/Volute 170
11.5.3. Case 2: Step Change Between Rotor and Stator 171
11.5.4. Case 3: Blade Passage at or Close to the Edge of a Domain 171
11.5.5. Case 4: Impeller Leakage 172
11.5.6. Case 5: Domain Interface Near Zone of Reversed Flow 173
11.6. Transient Blade Row 174
11.6.1. Steady versus Transient Blade Row Analysis 174
11.6.2. Full Model Simulation versus Reduced Geometry Simulation (Pitch Change Models) 175
11.6.3. Selecting an Appropriate Transient Blade Row Model with Pitch Change 175
11.6.3.1. Profile Transformation 175
11.6.3.2. Time Transformation 175
11.6.3.3. Fourier Transformation 176
11.6.4. Convergence and Solution Monitoring of Transient Blade Row Flow Problems 177
11.6.5. Boundary Conditions in Blade Row Simulation 177
11.6.5.1. Steady-state Analysis 177
11.6.5.2. Transient Analysis 178
11.6.6. Transient versus Harmonic Solution Method 178
12. CFX Best Practices Guide for Turbulence 181
12.1. Scale-Resolving Simulations in Ansys CFD 181
12.1.1. Scale-Resolving Simulation (SRS) Models – Basic Formulations 183
12.1.1.1. Scale-Adaptive Simulation (SAS) 184
12.1.1.2. Detached Eddy Simulation (DES) 186
12.1.1.3. Shielded Detached Eddy Simulation (SDES) 187
12.1.1.4. Stress-Blended Eddy Simulation (SBES) 190
12.1.1.5. Large Eddy Simulation (LES) 191
12.1.1.5.1. Limitations of Large Eddy Simulation (LES) 191
12.1.1.6. Wall Modeled Large Eddy Simulation (WMLES) 200
12.1.1.7. Embedded/Zonal LES (ELES, ZLES) 202
12.1.1.8. Unsteady Inlet/Interface Turbulence 203
12.1.2. Generic Flow Types and Basic Model Selection 204
12.1.2.1. Globally Unstable Flows 204
12.1.2.1.1. Flow Physics 204
12.1.2.1.2. Modeling 206

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.1.2.1.3. Meshing Requirements 206
12.1.2.1.4. Numerical Settings 207
12.1.2.1.5. Examples 207
12.1.2.1.5.1. Flow around a Fighter Aircraft 207
12.1.2.1.5.2. Flow around a Triangular Cylinder 208
12.1.2.1.5.3. ITS Combustion Chamber 210
12.1.2.2. Locally Unstable Flows 215
12.1.2.2.1. Flow Physics 215
12.1.2.2.2. Modeling 215
12.1.2.2.3. Meshing Requirements 217
12.1.2.2.4. Numerical Settings 218
12.1.2.2.5. Examples 218
12.1.2.2.5.1. Mixing Layer 218
12.1.2.2.5.2. Backward-Facing Step I 222
12.1.2.3. Stable Flows and Wall Boundary Layers 228
12.1.2.3.1. Flow Physics 228
12.1.2.3.2. Modeling 228
12.1.2.3.3. Meshing Requirements 229
12.1.2.3.4. Numerical Settings 230
12.1.2.3.5. Examples 230
12.1.2.3.5.1. Periodic Channel 230
12.1.2.3.5.2. Wall Boundary Layer 236
12.1.2.3.5.3. NASA Hump Flow 241
12.1.2.3.5.4. T-Junction with Thermal Mixing 243
12.1.3. Numerical Settings for SRS 251
12.1.3.1. Spatial Discretization 251
12.1.3.1.1. Momentum 251
12.1.3.1.2. Turbulence Equations 253
12.1.3.1.3. Gradients (Ansys Fluent) 253
12.1.3.1.4. Pressure (Ansys Fluent) 253
12.1.3.2. Time Discretization 253
12.1.3.2.1. Time Integration 253
12.1.3.2.2. Time Advancement and Under-Relaxation (Ansys Fluent) 254
12.1.4. Initial and Boundary Conditions 255
12.1.4.1. Initialization of SRS 255
12.1.4.2. Boundary Conditions for SRS 255
12.1.4.2.1. Inlet Conditions 255
12.1.4.2.2. Outlet Conditions 255
12.1.4.2.3. Wall Conditions 256
12.1.4.3. Symmetry vs. Periodicity 256
12.1.5. Postprocessing and Averaging 256
12.1.5.1. Visual Inspection 256
12.1.5.2. Averaging 258
12.1.6. Summary 259
12.1.6.1. Acknowledgment 260
12.1.6.2. Appendix 1: Summary of Numerics Settings with Ansys Fluent 260
12.1.6.3. Appendix 2: Summary of Numerics Settings With Ansys CFX 261
12.1.6.4. Appendix 3: Models 262
12.1.6.5. Appendix 4: Generic Flow Types and Modeling 265
12.1.7. References for Scale-Resolving Simulations 269
12.2. RANS Turbulence Modeling in Ansys CFD 275

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.1. General Considerations 276
12.2.2. Best Practice RANS 276
12.2.2.1. Managing Uncertainty 277
12.2.2.2. Steady vs Unsteady vs Convergence 278
12.2.2.3. Turbulence Model Selection 279
12.2.2.3.1. Spalart-Allmaras (SA) One Equation Model 280
12.2.2.3.2. Two-Equation Models 280
12.2.2.3.3. Wallin-Johansson Explicit Algebraic Reynolds Stress Models (WJ-EARSM) 283
12.2.2.3.4. Reynolds Stress Models (RSM) 283
12.2.2.3.5. Limiters 284
12.2.2.4. Additional Physics 284
12.2.2.4.1. Laminar-turbulent transition 284
12.2.2.4.2. Curvature Correction 285
12.2.2.4.3. Corner Correction 286
12.2.2.4.4. Buoyancy Correction 287
12.2.2.4.5. Wall Roughness Correction 288
12.2.3. Model Evaluation 288
12.2.3.1. Flat Plate Flow 289
12.2.3.2. Adverse Pressure Gradients and Flow Separation 294
12.2.3.2.1. NASA CS0 Diffuser 294
12.2.3.2.2. Airfoil Flows 300
12.2.3.2.3. Transonic Bump Flow 306
12.2.3.3. Corner Flows 308
12.2.3.3.1. Developed Flow in Square Duct 309
12.2.3.3.2. Flow in Rectangular Diffusers 310
12.2.3.3.3. Flow around DLR F6 aircraft 319
12.2.3.3.4. Conclusions 323
12.2.3.4. Swirl Flows 323
12.2.3.4.1. NACA-0012 Wing Tip Vortex 323
12.2.3.4.2. Flow in Hydro-Cyclone 328
12.2.3.5. Reattachment Flows 332
12.2.3.6. Impinging Flows 337
12.2.3.7. Buoyancy Flows 340
12.2.3.7.1. Stratified Mixing Layer 340
12.2.3.8. Effect of Limiters 343
12.2.3.9. Mesh Resolution Requirements 345
12.2.3.9.1. Inviscid Flow 345
12.2.3.9.2. Free Shear Flows 345
12.2.3.9.3. Fully Turbulent Boundary Layers 347
12.2.3.9.4. Transitional Boundary Layers 355
12.2.3.9.5. Corner Flows 358
12.2.4. Numerical Settings 364
12.2.4.1. Example: High-Lift Aircraft 364
12.2.5. Summary 366
12.2.6. Acknowledgment 366
12.2.7. References 367
12.2.8. Appendix A: Theory 376
12.2.8.1. The Closure Problem 376
12.2.8.1.1. Averaging 376
12.2.8.1.2. The Eddy-Viscosity Assumption 377
12.2.8.1.3. Reynolds Stress Modeling (RSM) 378

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.8.1.4. Explicit Algebraic Reynolds Stress Modeling (EARSM) 378
12.2.8.1.5. The Equation for the Turbulent Kinetic Energy 378
12.2.8.1.6. The Turbulent Scale-Equation 379
12.2.8.2. Two-Equation Models 382
12.2.8.2.1. The $k-\omega$ Models 382
12.2.8.2.2. The $k-\varepsilon$ Models 387
12.2.8.2.3. Limiters 389
12.2.8.3. Near-Wall Treatment 390
12.2.8.3.1. Standard Wall Functions 393
12.2.8.3.2. Scalable Wall Functions 395
12.2.8.3.3. Viscous Sublayer Model (VSM) 395
12.2.8.3.4. Y+-Insensitive Wall Treatments 396
12.2.8.4. Appendix A: Boundary Layer Parameters 400
12.2.8.4.1. Laminar Flow: 400
12.2.8.4.2. Turbulent Flow: 400
12.3. Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD 401
12.3.1. The Generalized k-omega (GEKO) Model Formulation 402
12.3.1.1. Basic Formulation 402
12.3.1.2. Limiters and Realizability 404
12.3.1.3. Near Wall Treatment 405
12.3.1.4. Terminology 407
12.3.2. The Influence of the Free GEKO Parameter 407
12.3.2.1. The 'Separation' Parameter C_{SEP} 408
12.3.2.2. 3.2 The 'Near Wall' Parameter C_{NW} 416
12.3.2.3. The 'Mixing' Parameter C_{MIX} 420
12.3.2.4. The 'Jet' Parameter C_{JET} 422
12.3.2.5. The 'Corner' Parameter CCORNER 424
12.3.2.6. The 'Curvature' Parameter CCURV 427
12.3.2.7. The Blending Function 430
12.3.2.8. Other Special Coefficients 437
12.3.3. Strategies for Model Optimization 438
12.3.3.1. GEKO Defaults 438
12.3.3.2. Optimizing Coefficients 439
12.3.4. Summary 440
12.3.5. Example UDFs 441
12.3.6. 7 References 441
13. CFX Command Language (CCL) 447
13.1. CFX Command Language (CCL) Syntax 447
13.1.1. Basic Terminology 448
13.1.2. The Data Hierarchy 448
13.1.3. Simple Syntax Details 448
13.1.3.1. Case Sensitivity 448
13.1.3.2. CCL Names Definition 449
13.1.3.3. Indentation 449
13.1.3.4. End of Line Comment Character 449
13.1.3.5. Continuation Character 449
13.1.3.6. Named Objects 450
13.1.3.7. Singleton Objects 450
13.1.3.8. Parameters 450
13.1.3.9. Lists 450
13.1.3.10. Parameter Values 450

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

13.1.3.10.1. String 450
13.1.3.10.2. String List 451
13.1.3.10.3. Integer 451
13.1.3.10.4. Integer List 451
13.1.3.10.5. Real 451
13.1.3.10.6. Real List 452
13.1.3.10.7. Logical 452
13.1.3.10.8. Logical List 452
13.1.3.11. Escape Character 452
14. CFX Expression Language (CEL) 453
14.1. CEL Fundamentals 453
14.1.1. Values and Expressions 454
14.1.1.1. Using Locators in Expressions 454
14.1.2. CFX Expression Language Statements 455
14.1.2.1. Use of Constants 455
14.1.2.2. Expression Syntax 455
14.1.2.3. Multiple-Line Expressions 456
14.2. CEL Operators, Constants, and Expressions 456
14.2.1. CEL Operators 457
14.2.2. Conditional if Statement 458
14.2.3. CEL Constants 458
14.2.4. Using Expressions 459
14.2.4.1. Use of Offset Temperature 459
14.3. CEL Examples 459
14.3.1. Example: Reynolds Number Dependent Viscosity 460
14.3.2. Example: Feedback to Control Inlet Temperature 460
14.3.3. Examples: Using Expressions in CFD-Post 462
14.4. CEL Technical Details 462
15. Functions in Ansys CFX 465
15.1. CEL Mathematical Functions 465
15.2. Quantitative CEL Functions in Ansys CFX 467
15.3. Functions Involving Coordinates 470
15.4. CEL Functions with Multiphase Flow 471
15.5. Quantitative Function List 472
15.5.1. area 479
15.5.1.1. Tools > Command Editor Example 479
15.5.1.2. Tools > Function Calculator Example 480
15.5.2. areaAve 480
15.5.2.1. Tools > Command Editor Example 480
15.5.2.2. Tools > Function Calculator Examples 481
15.5.3. areaInt 481
15.5.3.1. Tools > Command Editor Example 482
15.5.3.2. Tools > Function Calculator Examples 482
15.5.4. ave 482
15.5.4.1. Tools > Command Editor Example 483
15.5.4.2. Tools > Function Calculator Example 483
15.5.5. count 483
15.5.5.1. Tools > Command Editor Example 483
15.5.5.2. Tools > Function Calculator Example 484
15.5.6. countTrue 484
15.5.6.1. Tools > Command Editor Examples 484

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.6.2. Tools > Function Calculator Example 484
15.5.7. force 484
15.5.7.1. Tools > Command Editor Example 485
15.5.7.2. Tools > Function Calculator Examples 485
15.5.8. forceNorm 486
15.5.8.1. Tools > Command Editor Example 486
15.5.8.2. Tools > Function Calculator Example 486
15.5.9. inside 486
15.5.9.1. Tools > Command Editor Example 487
15.5.10. length 487
15.5.10.1. Tools > Command Editor Example 487
15.5.10.2. Tools > Function Calculator Example 487
15.5.11. lengthAve 488
15.5.11.1. Tools > Command Editor Example 488
15.5.11.2. Tools > Function Calculator Example 488
15.5.12. lengthInt 488
15.5.12.1. Tools > Command Editor Example 488
15.5.13. lineCloudAve 488
15.5.13.1. Tools > Command Editor Example 489
15.5.13.2. Tools > Function Calculator Example 489
15.5.14. mass 489
15.5.15. massAve 489
15.5.16. massFlow 490
15.5.16.1. Mass Flow Sign Convention 491
15.5.16.2. Tools > Command Editor Example 491
15.5.16.3. Tools > Function Calculator Example 491
15.5.17. massFlowAve 491
15.5.18. massFlowAveAbs 492
15.5.19. Details on Mass Flow Related Functions 493
15.5.20. massFlowInt 494
15.5.20.1. Tools > Command Editor Example 494
15.5.20.2. Tools > Function Calculator Example 494
15.5.21. massInt 494
15.5.22. maxVal 494
15.5.22.1. Tools > Command Editor Example 495
15.5.22.2. Tools > Function Calculator Example 495
15.5.23. minVal 495
15.5.23.1. Tools > Command Editor Example 495
15.5.23.2. Tools > Function Calculator Example 495
15.5.24. probe 495
15.5.24.1. Tools > Command Editor Example 496
15.5.24.2. Tools > Function Calculator Example 496
15.5.25. rbstate 496
15.5.25.1. Expressions Details View Example 497
15.5.26. rmsAve 497
15.5.27. sum 497
15.5.27.1. Tools > Command Editor Example 498
15.5.27.2. Tools > Function Calculator Example 498
15.5.28. torque 498
15.5.28.1. Tools > Command Editor Example 498
15.5.28.2. Tools > Function Calculator Example 498

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.29. volume 498
15.5.29.1. Tools > Command Editor Example 499
15.5.29.2. Tools > Function Calculator Example 499
15.5.30. volumeAve 499
15.5.30.1. Tools > Command Editor Example 499
15.5.30.2. Tools > Function Calculator Example 499
15.5.31. volumeInt 499
15.5.31.1. Tools > Command Editor Example 500
15.5.31.2. Tools > Function Calculator Example 500
16. Variables in Ansys CFX 501
16.1. Hybrid and Conservative Variable Values 501
16.1.1. Solid-Fluid Interface Variable Values 502
16.1.1.1. Conservative Values at 1:1 Interface 502
16.1.1.2. Hybrid Values at 1:1 Interface 503
16.1.1.3. Conservative Values on a GGI Interface 503
16.1.1.4. Hybrid Values on a GGI Interface 503
16.2. List of Field Variables 503
16.2.1. Common Variables Relevant for Most CFD Calculations 504
16.2.2. Variables Relevant for Turbulent Flows 509
16.2.3. Variables Relevant for Buoyant Flow 511
16.2.4. Variables Relevant for Compressible Flow 512
16.2.5. Variables Relevant for Particle Tracking 513
16.2.6. Variables Relevant for Calculations with a Rotating Frame of Reference 513
16.2.7. Variables Relevant for Parallel Calculations 514
16.2.8. Variables Relevant for Multicomponent Calculations 514
16.2.9. Variables Relevant for Multiphase Calculations 515
16.2.10. Variables Relevant for Radiation Calculations 516
16.2.11. Variables for Total Enthalpies, Temperatures, and Pressures 518
16.2.12. Variables and Predefined Expressions Available in CEL Expressions 519
16.2.12.1. System Variable Prefixes 531
16.2.12.2. CEL Variables r and theta 531
16.2.12.3. CEL Variable rNoDim 532
16.2.12.4. CEL Variable "subdomain" and CEL Function "inside" 532
16.2.12.5. Timestep, Timestep Interval, and Iteration Number Variables 532
16.2.12.5.1. Steady-State Runs 532
16.2.12.5.2. Transient Runs 533
16.2.12.5.3. Timestep Variables in CFD-Post 533
16.2.12.6. Expression Names 533
16.2.12.7. Scalar Expressions 533
16.2.12.8. Expression Properties 534
16.2.12.9. Available and Unavailable Variables 534
16.3. Particle Variables Generated by the Solver 534
16.3.1. Particle Track Variables 535
16.3.2. Particle Field Variables 538
16.3.2.1. Particle Sources into the Coupled Fluid Phase 538
16.3.2.2. Particle Radiation Variables 539
16.3.2.3. Particle Vertex Variables 539
16.3.2.3.1. Variable Calculations 541
16.3.2.4. Particle Boundary Vertex Variables 542
16.3.2.5. Particle RMS Variables 543
16.3.2.5.1. Variable Calculations 544

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.4. Miscellaneous Variables 545
17. Power Syntax in Ansys CFX 559
17.1. Examples of Power Syntax 560
17.1.1. Example 1: Print the Value of the Pressure Drop Through a Pipe 560
17.1.2. Example 2: Using a for Loop 561
17.1.3. Example 3: Creating a Simple Subroutine 562
17.1.4. Example 4: Creating a Complex Quantitative Subroutine 562
17.2. Predefined Power Syntax Subroutines 564
17.2.1. Power Syntax Subroutine Descriptions 564
17.2.2. Power Syntax Usage 565
17.2.3. Power Syntax Subroutines 565
17.2.3.1. area(Location, Axis) 565
17.2.3.2. areaAve(Variable, Location, Axis) 565
17.2.3.3. areaInt(Variable, Location, Axis) 565
17.2.3.4. ave(Variable, Location) 565
17.2.3.5. calcTurboVariables() 566
17.2.3.6. calculate(function,...) 566
17.2.3.7. calculateUnits(function,...) 566
17.2.3.8. collectTurboInfo() 566
17.2.3.9. comfortFactors() 566
17.2.3.10. compressorPerform(Location, Location, Location, Var, Args) 566
17.2.3.11. compressorPerformTurbo() 566
17.2.3.12. copyFile(FromPath, ToPath) 566
17.2.3.13. count(Location) 566
17.2.3.14. countTrue(Expression, Location) 567
17.2.3.15. cpPolar(Location, Var, Arg, Var, Location, Arg) 567
17.2.3.16. evaluate(Expression) 567
17.2.3.17. evaluateInPreferred(Expression) 568
17.2.3.18. exprExists(Expression) 568
17.2.3.19. fanNoiseDefault() 568
17.2.3.20. fanNoise() 568
17.2.3.21. force(Location, Axis) 568
17.2.3.22. forceNorm(Location, Axis) 568
17.2.3.23. getBladeForceExpr() 568
17.2.3.24. getBladeTorqueExpr() 568
17.2.3.25. getCCLState() 568
17.2.3.26. getChildrenByCategory(Object Path, Category) 569
17.2.3.27. getChildren(Object Path, Child Type) 569
17.2.3.28. getExprOnLocators() 569
17.2.3.29. getExprString(Expression) 569
17.2.3.30. getExprVal(Expression) 569
17.2.3.31. getObjectName(Object Path) 569
17.2.3.32. getParameterInfo(Object Path, Parameter Name, Info Type) 570
17.2.3.33. getParameters(Object Path) 570
17.2.3.34. getTempDirectory() 570
17.2.3.35. getType(Object Path) 570
17.2.3.36. getValue(Object Path, Parameter Name) 570
17.2.3.36.1. Example 570
17.2.3.37. getViewArea() 571
17.2.3.38. isCategory(Object Path, Category) 571
17.2.3.39. Length(Location) 571

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.3.40. lengthAve(Variable, Location) 571
17.2.3.41. lengthInt(Variable, Location) 571
17.2.3.42. liquidTurbPerformTurbo() 572
17.2.3.43. liquidTurbPerform() 572
17.2.3.44. massFlow(Location) 572
17.2.3.45. massFlowAve(Variable, Location) 572
17.2.3.46. massFlowAveAbs(Variable, Location) 572
17.2.3.47. massFlowInt(Variable, Location) 572
17.2.3.48. maxVal(Variable, Location) 572
17.2.3.49. minVal(Variable, Location) 572
17.2.3.50. objectExists(Object Path) 572
17.2.3.51. probe(Variable, Location) 573
17.2.3.52. pumpPerform() 573
17.2.3.53. pumpPerformTurbo() 573
17.2.3.54. range(Variable, Location) 573
17.2.3.55. reportError(String) 573
17.2.3.56. reportWarning(String) 573
17.2.3.57. showPkgs() 573
17.2.3.58. showSubs(packageName) 573
17.2.3.59. showVars(packageName) 574
17.2.3.60. spawnAsyncProcess(command, arguments) 574
17.2.3.61. sum(Variable, Location) 574
17.2.3.62. torque(Location, Axis) 574
17.2.3.63. turbinePerform() 574
17.2.3.64. turbinePerformTurbo() 574
17.2.3.65. verboseOn() 574
17.2.3.66. volume(Location) 574
17.2.3.67. volumeAve(Variable, Location) 575
17.2.3.68. volumeInt(Variable, Location) 575
18. Bibliography 577
18.1. References 1-20 577
18.2. References 21-40 580
18.3. References 41-60 583
18.4. References 61-80 586
18.5. References 81-100 589
18.6. References 101-120 592
18.7. References 121-140 595
18.8. References 141-160 598
18.9. References 161-180 601
18.10. References 181-200 604
18.11. References 201-220 607
18.12. References 221- 610
Glossary 615

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

List of Figures

1.1. Ansys CFX Launcher 29
4.1. Integration of a Remeshing Loop into the General Simulation Workflow 101
4.2. Schematic for User Defined remeshing 102
4.3. Schematic for ICEM CFD Replay remeshing 105
5.1. Original Undeformed Mesh 113
5.2. Deformed Mesh with the Default Diffusion Scheme 114
5.3. Deformed Mesh with "meshdisp diffusion scheme = 3" 115
7.1. Flow Rate vs Pressure Rise for a Liquid Pump 146
7.2. Cavitation Performance at Constant RPM and Flow Rate 147
10.1. An exaggerated view of three inflation layers on each side of the uppermost subdomain boundary surface. 162
11.1. Flow Rate vs Pressure Rise for a Gas Compressor 166
11.2. Element Aspect Ratio at Domain Interface 170
11.3. Impeller/Volute 171
11.4. Possible Domain Interface Positions with Step Change in Passage Height 171
11.5. Radial Compressor 172
11.6. Flow Leakage Through Gap Near Impeller Inlet 173
11.7. Domain Interface Between Blade Rows in an Axial Machine 174
12.1. URANS computations of a flow past a circular cylinder (SST model) 184
12.2. SAS simulation of flow past a circular cylinder (SAS-SST model) 185
12.3. The domain and grid for the separated flow in CS0 diffuser 188
12.4. Contours of blending functions overset by vorticity iso-lines for CS=0 diffuser for SBES and DDES models 189
12.5. Contours of eddy viscosity ratio for CS0 diffuser for SBES and DDES models 190
12.6. Turbulence spectrum for DIT test case after t=2. Comparison of results without Sub-Grid Scale model (no LES) with WALE and Smagorinsky LES model simulations 194
12.7. Sketch of turbulence structures for wall-bounded channel flow with viscous sublayer (a) Low Re number (b) High Re number (Grey area: viscous sublayer) 195
12.8. Turbulence structures in a channel flow 196
12.9. Details of grid around a NACA 0012 airfoil (a) Grid topology (b) Leading edge area (c) Trailing edge area 198
12.10. Turbulence structures of WALE LES computation around a NACA 0012 airfoil (a) Leading edge (b) Trailing edge (Q-criterion, color- spanwise velocity component) 199
12.11. Wall pressure coefficient C_p on the suction side of a NACA 0012 airfoil: comparison of RANS-SST and LES-WALE results 199
12.12. Wall shear stress coefficient C_f on the suction side of a NACA 0012 airfoil: comparison of RANS-SST and LES-WALE results 200
12.13. Concept of WMLES for high Re number flows (a) Wall-resolved LES. (b) WMLES 201
12.14. Sketch of boundary layer profile with thickness δ, x-streamwise direction, y-normal direction, and z-spanwise direction 202
12.15. Turbulence structures for flow around a generic fighter aircraft (Q-criterion) as computed by SAS-SST model. Courtesy - EADS Germany GmbH - Military Air Systems DESIDER Project 208
12.16. Grid around cylinder in crossflow 209
12.17. Turbulence structures for flow around a cylinder in crossflow 209
12.18. Velocity profiles and turbulence RMS profiles for three different stations downstream of the triangular cylinder (x/a=0.375, x/a=1.53, x/a=3.75). Comparison of SAS-SST, DES-SST models, and experiment. (a) U-velocity, (b) u_{rms}, (c) v_{rms}, (d) u'v' 210
12.19. SAS-SST simulation for flow around a triangular cylinder using the BCD and the CD scheme for the convective fluxes 210

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.20. Computational domain for the ITS swirl burner 211
12.21. Unstructured grid on the symmetry plane and boundary locations for the ITS swirl burner and relevant length scale, L 212
12.22. SAS solution for ITS combustion chamber (a) Non-reacting, (b) Reacting flow 213
12.23. Reacting flow velocity profiles at the axial distance from the inlet x=103 mm (a) Axial velocity, (b) Tangential velocity 214
12.24. Schematic of locally unstable flow: Mixing layer originating from a flat plate with two boundary layers of different freestream velocity. Full yellow circles are boundary layer turbulence. Dashed yellow circles are remains of the boundary layer turbulence. Red arrows are new mixing layer turbulence 215
12.25. The domain and grid for the single stream mixing layer flow 219
12.26. Isosurfaces of the Q-criterion colored with the velocity magnitude 220
12.27. Profiles of the mean velocity at different sections (x/θ=19.3, 29.6, 40.6, and 54.2) for mixing layer test case 221
12.28. The domain and grid for the backstep test case 222
12.29. Isosurfaces of the Q-criterion colored with the velocity magnitude for backstep test case 223
12.30. Contours of the blending function with the vorticity iso-lines for backstep test case 224
12.31. Contours of the eddy viscosity ratio for backstep test case 225
12.32. Distribution of the skin friction coefficient for backstep test case 226
12.33. Profiles of the mean and RMS velocity at different sections (x/H=3.2, 4.55, 5.87, and 7.2) for backstep test case 227
12.34. Sketch of embedded LES (ELES) domain 229
12.35. Computational domain and grid for WMLES of channel flow 231
12.36. Turbulence structures for WMLES of channel flow at lowest Reynolds number (Q=350 [s^{-2}]) 232
12.37. Flow visualization for WMLES of channel flow (a) Vorticity rate Omega, (b) Absolute value of velocity U, (c) Ratio of eddy-viscosity to molecular viscosity 233
12.38. Resolved normal stresses, turbulent kinetic energy, and mean velocity profiles for WMLES at different Reynolds numbers 234
12.39. Isosurfaces of the Q-criterion colored with the velocity magnitude 235
12.40. Profiles of the eddy viscosity ratio for different models 236
12.41. Profiles of the mean velocity for different models 236
12.42. Profiles of the total turbulence kinetic energy for different models 236
12.43. Computational domain for a Wall Boundary Layer test case 237
12.44. Computational grid (a), (c) and applied boundary conditions (b) 238
12.45. Isosurfaces of Q-criterion (Q=200 [s^{-2}]) colored with velocity for a flat plate at two different Reynolds numbers 239
12.46. Skin friction distributions along a flat plate predicted by WMLES at two Reynolds numbers (a) Re theta=1000 with different numerical settings (b) Re theta =10 000 240
12.47. Profiles of resolved normal and shear Reynolds stresses and mean velocity in the flat plate boundary layer predicted by WMLES at two Reynolds numbers (a) Re theta=1000 with different numerical settings (b) Re theta=10 000 with the second order pressure interpolation 240
12.48. Experimental set-up for NASA hump flow experiment 241
12.49. (a) Grid used for the NASA hump simulation (b) Turbulent structures in the LES domain (Q-criterion colored with spanwise velocity component) 242
12.50. (a) Skin-friction, c_f, and (b) Wall pressure coefficients, c_p, from NASA hump flow simulations. Comparison of WMLES and WALE LES methods in the LES domain 243
12.51. Geometry and grid of T-Junction test case with measurement planes 245
12.52. Turbulence structures for SAS-SST model (a) Central Difference (CD) scheme, (b) Bounded Central Difference (BCD) scheme 246
12.53. Vorticity contours for ELES/WMLES simulation (a) CD scheme, (b) BCD scheme 247

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.54. Comparison of the experimental and computational velocity profiles for T-Junction flow for different turbulence models (a) CD scheme (b) BCD scheme (note that scales of coordinate axes change by large factors between curves) 248
12.55. Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Top wall (0 degrees, see Figure 12.51) of the main pipe 249
12.56. Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Front wall (90 degrees, see Figure 12.51) of the main pipe 249
12.57. Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Bottom wall (180 degrees, see Figure 12.51) of the main pipe 250
12.58. Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Rear wall (270 degrees, see Figure 12.51) of the main pipe 250
12.59. Comparison of turbulence structures on the Top wall downstream of the pipe intersection (a) DDES model (b) ELES model 251
12.60. Example of scheme oscillations in T-Junction flow shown by vorticity: (a) CD, (b) BCD 252
12.61. Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow (12). 281
12.62. Schematic of the boundary layer on curved surfaces. 286
12.63. Schematic of the flow over a flat plate. 289
12.64. Computational domain with boundary conditions and mesh for the flow over a flat plate 290
12.65. Comparison of wall shear stress coefficient, C_f (Left) and wall heat transfer coefficient, St. (Right) for flat plate boundary layer (28) and y^+~1. 291
12.66. Comparison of velocity profiles in wall-law for flat plate boundary layer at $Re_x=8.7\cdot10^6$. 291
12.67. Comparison of wall shear stress coefficient (Left) and velocity profile at $Re_x=8.7\cdot10^6$ (Right) predicted with the GEKO model without and with EARSM/RSM options for the flat plate boundary layer (28). 292
12.68. Comparison of wall shear stress coefficient, C_f (upper left), wall heat transfer coefficient, St. (upper right) and log-law velocity profile at $Re_x=8.7\cdot10^6$ (lower right) for the flat plate boundary layer using the SST model [28]. 293
12.69. Distribution of the skin friction coefficient over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28]. 293
12.70. Distribution of the Stanton number over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28]. 294
12.71. Schematic of the flow in CS0 diffuser. 295
12.72. Figure 12: Computational domain with boundary conditions and mesh for CS0 diffuser. 295
12.73. Comparison of velocity profiles predicted with different turbulence models for CS0 diffuser [30] 296
12.74. Comparison of velocity profiles for k-ω model family and GEKO models for CS0 diffuser [30]. 297
12.75. Prediction of wall characteristics for k-ω family and GEKO models for CS0 diffuser [30]. Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p. 297
12.76. Flow structured visualized with the streamwise velocity field and streamlines (black lines) for CS0 diffuser [30]. 298
12.77. Figure 17: Impact of variation in C_{SEP} coefficient of the GEKO model for CS0 diffuser flow [30]. Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p. 298
12.78. Figure 18: Impact of variation in C_{SEP} coefficient of the GEKO model on velocity profiles for CS0 diffuser flow [30]. 299
12.79. Figure 19: Impact of variation in C_{SEP} coefficient of the GEKO model on turbulence kinetic energy profiles for CS0 diffuser flow [30]. 299
12.80. Figure 20: Impact of variation in C_{SEP} coefficient of the GEKO model on turbulence shear stress profiles for CS0 diffuser flow [30]. 299
12.81. Figure 21: Impact of non-linear terms for the GEKO model for CS0 diffuser flow (30). Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p. 300
12.82. Figure 22: Schematic of the flow around an airfoil for stall regime. 300

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.83. Prediction of lift coefficient in wide range of angle of attacks for different turbulence models for S809 airfoil at Re=$2 \cdot 10^6$ [33] 302
12.84. Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 [35] (Left) and DU-97-W-300 [35] (Right) airfoil at Re=$3 \cdot 10^6$ 303
12.85. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha = 12°$ (Right) with GEKO model for S805 airfoil at Re=$1 \cdot 10^6$ [31] 303
12.86. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha = 17°$ (Right) with GEKO model for S825 airfoil at Re=$2 \cdot 10^6$ [32] 304
12.87. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha = 10°$ (Right) with GEKO model for S809 airfoil at Re=$2 \cdot 10^6$ [33] 304
12.88. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha = 18°$ (Right) with GEKO model for S814 airfoil at Re=$1.5 \cdot 10^6$ [34] 305
12.89. NACA-4412 airfoil near the trailing edge with streamwise velocity profiles at measurement locations predicted with linear and non-linear GEKO-1.75 model compared with the experimental data [38] 305
12.90. Scheme of the transonic bump flow 306
12.91. Computational domain with boundary conditions and mesh for the NASA Transonic Bump [39] 307
12.92. Comparison of wall pressure coefficient, C_p for the transonic bump flow [39]. The bump starts at X/C=0 location. Left: comparison of the different turbulence models, right: versatility of the GEKO model with the C_{SEP} variation (GEKO-1.75: C_{SEP}=1.75) 308
12.93. Comparison of different C_{SEP} settings and ERSM inclusion for the transonic bump flow [39]. Left: wall shear stress coefficient, C_f, right: wall pressure stress coefficient C_p 308
12.94. Flow topology in the square duct case predicted using the GEKO model with CFC (left) and without CFC (right). Blue lines: iso-surfaces of the streamwise velocity, red lines in the corners: streamlines of the secondary motion 309
12.95. Streamwise (Left) and lateral (Right) velocity profiles on the diagonal of the square duct case [40] 310
12.96. DLR diffuser configuration [41] 311
12.97. Computational domain with the boundary conditions for the DLR diffuser [41] 311
12.98. The surface streamlines near the inclined and side walls plotted for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41]. 312
12.99. Flow topology for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41] 312
12.100. Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for turbulence models with and without non-linear terms for the DLR diffuser [41] 312
12.101. Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the DLR diffuser [41] 313
12.102. Geometry of Stanford diffuser [42], [43]. 314
12.103. Computational domain with boundary conditions for the Stanford diffuser [42], [43]. 314
12.104. Flow topology for the linear (left) and non-linear CFC-GEKO model (middle) for the Stanford diffuser. Experimental data (right) [42], [43]. 315
12.105. Streamwise velocity contours predicted by the linear and non-linear GEKO (C_{SEP}=1.00) model at X5 section. Blue colors indicate recirculation zone. 316
12.106. Streamwise velocity contours predicted be the linear and non-linear GEKO (C_{SEP}=1.75) model at X5 section. Blue colors indicate recirculation zone. 318
12.107. Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the Stanford diffuser [42], [43]. 319
12.108. Geometry DLR F6 Wing-Body-Pylon-Nacelle testcase 320
12.109. Lift-Drag Polar for Wing body without and with engine pylon + nacelle 321

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.110. Separation zone at the upper wing-fuselage junction (left) and lower wing surface behind the engine (right). Top - oil film visualization in the experiment. 322
12.111. NACA-0012 wing setup in a wind tunnel showing wing tip vortices as visualized by streamlines. 324
12.112. Front view of eddy-viscosity ratio contours computed with the use of SST and SST-CC turbulence models downstream of the NACA-0012 wing [44]. 325
12.113. Non-dimensional cross flow velocity (UcrossVar) and axial velocity (Uax) at three planes located downstream of the trailing edge of the NACA 0012 wing tip [44]. 326
12.114. 327
12.115. Computed distributions of the axial velocity at the plane passing through the vortex core for the NACA 0012 wing tip vortex [44]. Top view. 328
12.116. Flow structure in the hydro-cyclone visualized by the streamlines. 329
12.117. Measurement planes in the hydro cyclone [45]. 330
12.118. Time-averaged profiles of the tangential velocity in the hydro cyclone [45]. 331
12.119. Schematic of the NASA hump flow. 333
12.120. Computational meshes for the Hump flow. 334
12.121. Comparison of wall shear stress coefficient for NASA Hump flow. 335
12.122. Comparison of wall shear stress coefficient predicted with GEKO model for NASA Hump flow [48]. .. 336
12.123. Effect of the GEKO CSEP on the streamwise velocity for NASA Hump flow [46], [47]. 336
12.124. Effect of the GEKO CSEP on the Reynolds shear stresses for NASA Hump flow [46], [47]. 337
12.125. Scheme of the impinging jet flow 338
12.126. Computational domain with boundary conditions and mesh for the impinging jet. 338
12.127. Local Nusselt number computed using different turbulence models (left) and SST model with and without taking into accout the laminar turbulent transition (right). 339
12.128. Computational mesh for the fully turbulent regime (left) and regime with taking into account laminar-turbulent transition (Right). 339
12.129. Scheme of the stratified mixing layer 340
12.130. Computational domain with boundary conditions and mesh for the stratified mixing layer. 341
12.131. Density and streamwise velocity profiles at different sections for the stratified mixing layer. 341
12.132. 342
12.133. Density field for the SST (Left) and SST-BC (Right) model. 342
12.134. Turbulence kinetic energy field for the SST (Left) and SST-BC (Right) model. 342
12.135. Density (upper row) and streamwise velocity (lower row) profiles for diferent models for the stratified mixing layer at X = 10 [m] section. 343
12.136. Effect of limiters for the GEKO model near the stagnation point for flow around airfoils 344
12.137. Effect of production limiter for the $k-\varepsilon$ Standard model near the stagnation point for flow around airfoils. 344
12.138. Eddy-viscosity field predicted with the RKE turbulence model near the stagnation region for flow around airfoils. 345
12.139. Computational meshes for the mixing layer. 346
12.140. Comparison of the velocity (left), turbulence kinetic energy (middle) and Reynolds shear-stress (right) profiles on different meshes for the mixing layer. 346
12.141. Boundary layer thickness in wall units for low and high Reynolds number. 347
12.142. Computational unstructured mesh with prismatic layers (red) thinner than the boundary layer (blue line) thickness. 348
12.143. The boundary layers thickness visualized by eddy-viscosity ratio field for different meshes around NACA-4412 airfoil. Blue color shows the area of the inviscid flow. 349
12.144. 351
12.145. Comparison of velocity profiles on different meshes for flow around the NACA-4412 airfoil. 351
12.146. Local mesh refinement on the leading edge for the medium mesh. 352
12.147. Comparison of the velocity profiles on the medium and refined medium (medium+LE) meshes. 352
12.148. Optimal unstructured mesh topology for flow around airfoil. 354

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.149. Computational meshes for the fully turbulent (upper) and transitional (lower) boundary layer. 356
12.150. Effect of the resolution of transition region in streamwise direction for different transition models for the T3A flat plate boundary layer. 357
12.151. Effect of the resolution of transition region in wall-normal wall direction for different transition models for T3A the flat plate boundary layer. 358
12.152. Baseline structured mesh with fine resolution in the corners for the DLR-Diffuser [41]. 360
12.153. Baseline mesh PH-1 (left) and refinement mesh PH-2 (right) near the corners for the unstructured meshes for the DLR-diffuser [41]. 360
12.154. Cross section of typical unstructured mesh PH-1 in rectangle channel (left) and collapsing of the prismatic cells near the corners on the refined mesh PH-2 (right) for the DLR-Diffuser [41]. 361
12.155. Eddy viscosity of the developing boundary layer at the straight part of diffuser and mesh cross section mesh resolution for different meshes. 362
12.156. Streamlines of crossflow pattern at the start of the diffuser for three different meshes. 362
12.157. Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41]. 363
12.158. Distribution of the pressure coefficient along the diffuser at midsection for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41]. 363
12.159. Figure 96: Wall shear stress for high-lift airplane simulation for α =18 . Left: Simulation in 'steady-state mode'. Right: Simulation in 'unsteady mode'. 365
12.160. Lift coefficient, CI, for high-lift aircraft simulation versus angle of attack (AOA). RANS-SST with default and 'steady state mode', SST with a1=1.0 and 'steady state mode' and URANS-SST with default SST and 'unsteady mode' set-up. 365
12.161. Effect of freestream omega-levels on velocity profile for free mixing layer solution. Left: std. k-omega model without CD term. Right: GEKO model including CD term. 381
12.162. Effect of freestream ω-levels on non-dimensional eddy viscosity profiles for free mixing layer solution. Left: standard $k-\omega$ model without CD term. Right: GEKO model including CD term. 381
12.163. Blending function FBlend for introducing addtional mixing through coefficients C_{MIX}/C_{JET}. 385
12.164. Flow topology for Obi [54] diffuser. Top - SST model. Bottom $k-\varepsilon$ model. 388
12.165. Velocity profiles for Obi diffuser [54]. Comparison of SST, $k-\varepsilon$ and experimental data. 388
12.166. Near wall grid cells for a finite volume cell-centered method. 391
12.167. Universal non-dimensional velocity-profile near the wall. 393
12.168. Comparison of the skin friction coefficient and velocity profiles at $\mathrm{Re}_x=8.7\cdot10^6$ on different meshes for the flat plate boundary layer (28). 396
12.169. Contours of the Re_y (Top, a,) and the Blending Function (Middle, b,) and the ratio of eddy-viscosity/molecular viscosity $EVR=\mu_t/\mu$ (Bottom, c,) of the 2L formulation for flow around the NACA-4412 airfoil at AoA = 12° 399
12.170. Near wall y^+ insensitive wall treatment for GEKO model. 406
12.171. Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow [31] 407
12.172. Change of the eddy-viscosity ration (EVR) under changes of $C_{\mathrm{SEP},}$ for a flat plate. Top:$C_{\mathrm{SEP},}$=1.0 (all else default). Bottom $C_{\mathrm{SEP},}$=1.75 (all else default) 408
12.173. Flat plate boundary layer under variation of C_{SEP} (C_{NW}=0.5). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St 409
12.174. Velocity profiles in logarithmic plot under variation of C_{SEP} for flat plate 410
12.175. Impact of variation in C_{SEP} on boundary layer with adverse pressure gradient. Left: Velocity profile. Right Eddy-viscosity profiles 411

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.176. Impact of variation in C_{SEP} on CS0 diffuser flow [10]. Left wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p 411
12.177. Impact of variation in C_{SEP} on velocity profiles for CS0 diffuser flow [10] 412
12.178. Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 (Left) and DU-97-W-300 (Right) airfoil at $\mathrm{Re}=3\cdot10^6$ [28] 413
12.179. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=12°$ (Right) with GEKO model for S805 airfoil at $\mathrm{Re}=1\cdot10^6$ [23] 413
12.180. Prediction of lift coefficient wide range of angle of attacks (Left) and pressure coefficient for $\alpha=17°$ (Right) with GEKO model for S825 airfoil at $\mathrm{Re}=2\cdot10^6$ [24] 414
12.181. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=10°$ (Right) with GEKO model for S809 airfoil at $\mathrm{Re}=2\cdot10^6$ [25] 414
12.182. Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=18°$ (Right) with GEKO model for S814 airfoil at $\mathrm{Re}=1.5\cdot10^6$ [26] 415
12.183. Schematic of transonic axi-symmetric bump flow 416
12.184. Pressure coefficient,C_p, along wall of bump. Comparison of different GEKO settings with RKE and SST model and experimental data [2]. 416
12.185. Flat plate boundary layer under variation of C_{NW}(C_{SEP}=1.75). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St. 417
12.186. Velocity profiles in logarithmic coordinates - variation of C_{NW}(C_{SEP}=1.75) 418
12.187. Impact of variation in C_{NW} on CS0 diffuser flow [10]. Left: Wall shear-stress coefficient, C_f. Right: Wall pressure coefficient C_p (C_{SEP}=1.0, C_{MIX}=0.0, C_{JET}=0.9) 419
12.188. Impact of variation in C_{NW} on velocity profiles for CS0 diffuser flow [10] 419
12.189. Backward-facing step with heat transfer under variation of C_{NW} (Top:C_{SEP}=1.00, Bottom: C_{SEP}=1.75, Both: C_{JET}=0.9, C_{MIX}=C_{MixCor}). Left: Wall shear stress coefficient, C_f, Right: Heat transfer Stanton number 420
12.190. Impact of variation in C_{MIX} on mixing layer when using the C_{SEP}=2(C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles 421
12.191. Impact of variation in C_{SEP} on mixing layer (C_{MIX}=0, C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles 421
12.192. Impact of variation in C_{SEP} on mixing layer when using the C_{MIX}=C_{MixCor} correlation (C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles 422
12.193. Effect of C_{JET} for plane (left) and round (right) jet flow (C_{SEP}=2.0, C_{MIX}=0.35, C_{NW}=0.5) 423
12.194. Comparison of GEKO-1 and GEKO-2 model (with variation of C_{JET}=0.9) 423
12.195. Comparison of streamline velocity contours and secondary motion predicted by linear (Left) and non-linear (Center) GEKO-1.00 model with the DNS (Right) data in the fully developed square duct flow [19] 424
12.196. Comparison of turbulence models for flow in square channel. Left: GEKO with WJ-EARSM. Middle, GEKO with CCORER=0.9. Right GEKO linear (CCORNER=0.0). 425
12.197. Velocity profiles plotted along diagonal of channel for different turbulence models against DNS data. Right: Main flow, Left Secondary Flow 425
12.198. Geometry of 3D diffuser flow [4] 426
12.199. Flow topology for 3D diffuser flow shown through streamwise velocity contour at downstream end of diffuser opening section for GEKO-1.00 with different C_{CORNER} values. 426
12.200. Wall pressure coefficient,C_p, for 3D diffuser for GEKO-1.00 with different C_{CORNER} values. 427

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.201. Hydrocyclone typical flow structure. Reproduced from Cullivan et al [9]. 428
12.202. Time-averaged profiles of the tangentional velocity in the hydrocyclone. Comparison with the experiments of Hartley [13]. 429
12.203. Effect of CBF_TURB on F_{GEKO} for flow around airfoil. Left CBF_TURB=2.0. Right: CBF_TURB=4.0. 431
12.204. Effect of CBF_LAM on F_{GEKO} for flow around airfoil. Left CBF_LAM=1.0 (default for turbulent simulations). Right: CBF_LAM=25 (default for transitional simulations). 431
12.205. Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF 433
12.206. Ratio of turbulence to molecular viscosity for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF 434
12.207. Flow around triangular cylinder [37] 435
12.208. GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U. Middle: Eddy-viscosity ratio. Right: F_{GEKO} using built-in function [28] 436
12.209. GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U Middle: Eddy-viscosity ratio. Right: F_{GEKO} using UDF-based function [28] 436
12.210. Center line velocity for triangular cylinder. Left: GEKO-1.75 with built-in F_{GEKO} function. Right: GEKO-1.75 with UDF-based F_{GEKO} function [28] 437
12.211. Comparison of velocity profiles for flow around the NACA-4412 airfoil [32] 438
14.1. Temperature Feedback Loop 461
16.1. r and theta with Respect to the Reference Coordinate Frame 532

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

List of Tables

12.1. Number of cells, N_t, vs Reynolds number for channel flow 196
12.2. Computing power estimate for a single turbomachinery blade with end-walls 197
12.3. Grid resolution for WMLES channel flow simulations 231
12.4. Properties for flat plate boundary layer simulations 237
12.5. Information on grids for flat plate test case 238
12.6. Flow conditions for T-Junction test case 243
12.7. Globally Unstable Flows 265
12.8. Locally Unstable Flows 267
12.9. Stable Flows 268
12.10. Parameters of the computational meshes 290
12.11. Table 2: Considered airfoils and parameters of wind tunnel and flow in the experiments 301
12.12. Flow parameters of the stratified mixing layer 340
12.13. Coefficients for the standard $k-\varepsilon$ model (see e.g. [7]). 380
12.14. Coefficients standard $k-\omega$ model (see e.g. [7]) 380
12.15. Coefficients of BSL model. 383
12.16. Coefficients of SST model. 383
12.17. Coefficients of standard $k-\varepsilon$ model. 387
12.18. Coefficients RKE $k-\varepsilon$ model 389
14.1. CEL Operators 457
14.2. CEL Constants 458
15.1. Standard Mathematical CEL Functions 465
15.2. Examples of the Calling Syntax for an Expression 470
15.3. CEL Multiphase Examples 471
15.4. CEL Functions in CFX-Pre/CFX-Solver and in CFD-Post 472
15.5. Source of mass flow data 493
16.1. Common CEL Single-Value Variables and Predefined Expressions 520
16.2. Common CEL Field Variables and Predefined Expressions 521

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 1: Ansys CFX Launcher

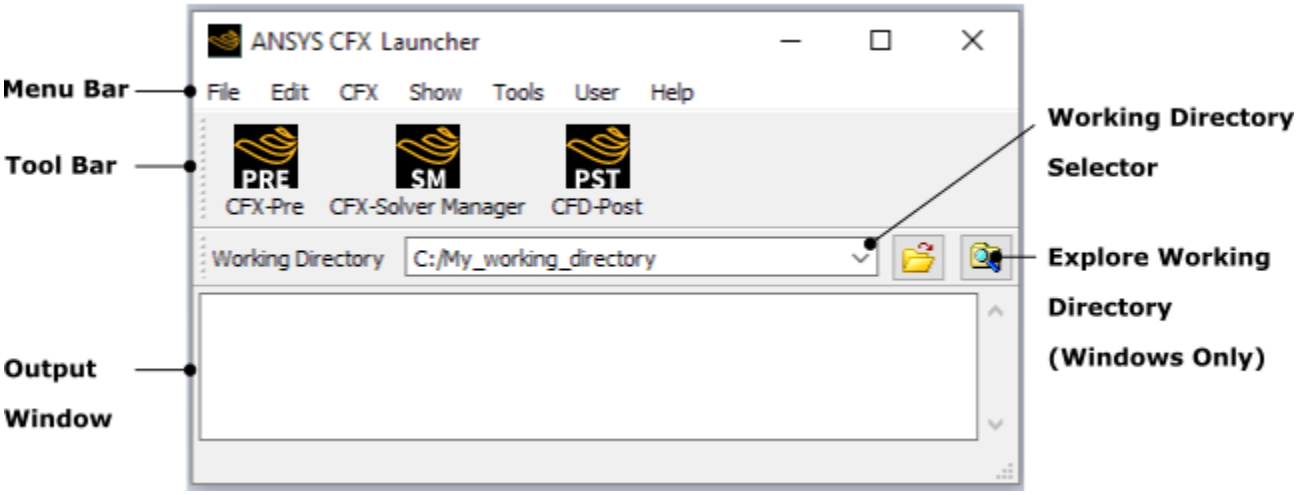
This chapter describes the Ansys CFX Launcher in detail:

1.1. The Ansys CFX Launcher Interface
1.2. Customizing the Ansys CFX Launcher

1.1. The Ansys CFX Launcher Interface

The layout of the Ansys CFX Launcher is shown below:

Figure 1.1: Ansys CFX Launcher



The launcher consists of a menu bar, a toolbar for launching applications, a working directory selector, and an output window where messages are displayed. On Windows platforms, an icon to start Windows Explorer in the working directory appears next to the directory selector.

1.1.1. Menu Bar

The Ansys CFX Launcher menus provide the following capabilities:

1.1.1.1. File Menu

Saves the contents of the text output window and to close the Ansys CFX Launcher.

1.1.1.1.1. Save As

Saves the contents of the output window to a file.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1.1.1.1.2. Quit

Shuts down the Ansys CFX Launcher. Any programs already launched will continue to run.

1.1.1.2. Edit Menu

Clears the text output window, finds text in the text output window and sets options for the Ansys CFX Launcher.

1.1.1.2.1. Clear

Clears the output window.

1.1.1.2.2. Find

Displays a dialog box where you can search the text in the output window.

1.1.1.2.3. Options

Presents the **Options** dialog box, which enables you to change the appearance of the Ansys CFX Launcher. Once you have configured the settings, click **Apply** to apply the settings tentatively. Click **OK** to accept the settings. Click **Restore** to revert the settings to the previously accepted configuration.

1.1.1.2.3.1. Graphical User Interface Style

You can choose any one of the listed interface styles to change the look and feel of the user interface.

1.1.1.2.3.2. Font and Formatted Font

The button to the right of **Font** sets the font used anywhere outside the text output window. The button to the right of **Formatted Font** applies only to the text output window. Clicking either of these buttons opens the **Select Font** dialog box.

1.1.1.3. CFX Menu

Enables you to launch CFX-Pre, CFX-Solver Manager, CFD-Post, and, if they are installed, other CFX products (such as Ansys TurboGrid).

1.1.1.3.1. CFX-Pre

Runs CFX-Pre, with the working directory as specified in Working Directory Selector (p. 33).

1.1.1.3.2. CFX-Solver Manager

Runs CFX-Solver Manager, with the working directory as specified in Working Directory Selector (p. 33).

1.1.1.3.3. CFD-Post

Runs CFD-Post, in the current working directory as specified in Working Directory Selector (p. 33).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1.1.1.3.4. Other CFX Applications

The Ansys CFX Launcher also searches for other CFX applications (for example, Ansys TurboGrid) and provides a menu entry to launch the application. If an application is not found, you can add it; for details, see Customizing the Ansys CFX Launcher (p. 33).

1.1.1.4. Show Menu

Enables you to show system, installation, and other information.

1.1.1.4.1. Show Installation

Displays information about the version of CFX that you are running.

1.1.1.4.2. Show All

Displays all of the available information, including information about your system, installation, and variables.

1.1.1.4.3. Show System

Displays information about the CFX installation and the system on which it is being run.

1.1.1.4.4. Show Variables

Displays the values of all the environment variables that are used in CFX.

1.1.1.4.5. Show Patches

Displays information about point releases and patches installed.

1.1.1.5. Tools Menu

Enables you to access license-management tools and a command line for running other CFX utilities.

1.1.1.5.1. Command Line

Starts a command window from which you can run any of the CFX commands via the command line interface. The command line will be set up to run the correct version of CFX and the commands will be run in the current working directory.

If you do not use the **Tools** > **Command Line** command to open a command window, then you will have to either type the full path of the executable in each command, or explicitly set your operating system path to include the `<CFXROOT>/bin` directory.

You may want to start components of CFX from the command line rather than by clicking the appropriate button on the Ansys CFX Launcher for the following reasons:

- CFX contains some utilities (for example, a parameter editor) that can be run only from the command line.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- You may want to specify certain command line arguments when starting up a component so that it starts up in a particular configuration.
- If you are having problems with a component, you may be able to get a more detailed error message by starting the component from the command line than you would get if you started the component from the launcher. If you start a component from the command line, any error messages produced are written to the command line window.

1.1.1.5.2. Configure User Startup Files (Linux only)

Information about creating startup files can be found in the installation documentation.

1.1.1.5.3. Edit File

Opens a platform-native text editor. Which text editor is opened is controlled by the settings in <*CFXROOT*>/etc/launcher/shared.ccl.

1.1.1.5.4. Edit Site-wide Configuration File

Opens the site-wide configuration file in a text editor. Which text editor is called is controlled by the settings in <*CFXROOT*>/etc/launcher/CFX5.ccl.

1.1.1.6. User Menu

The **User** menu is provided as an example. You can add your own applications to this menu, or create new menus; for details, see Customizing the Ansys CFX Launcher (p. 33).

1.1.1.7. Help Menu

The **Help** menu has the following commands:

CFX Launcher

Opens Using the Ansys CFX Launcher in the *CFX Introduction*.

Contents

Opens a page that lists various help resources associated with this product.

[various help resources]

Each of these commands goes directly to a particular help resource.

Ansys Product Improvement Program

Provides a brief description of, and enables you to control participation in, the Ansys Product Improvement Program.

About CFX Launcher

This gives the point releases and software patches that are installed.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Help on Help

Opens documentation about the help system: Help On Help in the *CFX Introduction*.

1.1.2. Toolbar

The toolbar contains shortcuts to the main components of CFX, for example CFX-Pre, CFX-Solver Manager and CFD-Post. Pressing any of the buttons will start up the component in the specified working directory. The equivalent menu entries for launching the components also show a keyboard shortcut that can be used to launch the component.

1.1.3. Working Directory Selector

While running CFX, all the files that are created will be stored in the working directory. To change the working directory, you can do any of the following:

- Type the directory name into the box and press **Enter**.
- Click the down-arrow icon ( ) next to the directory name. This displays a list of recently used directories.
- Click *Browse* to browse to the directory that you want.

1.1.4. Output Window

The output window is used to display information from commands in the **Show** menu. You can right-click in the output window to show a shortcut menu with the following options:

- **Find**: Displays a dialog box where you can enter text to search for in the output.
- **Select All**: Selects all the text.
- **Copy Selection**: Copies the selected text.
- **Save As**: Saves the output to a file.
- **Clear**: Clears the output window.

1.2. Customizing the Ansys CFX Launcher

Many parts of the Ansys CFX Launcher are driven by CCL commands contained in configuration files. Some parts of the launcher are not editable (such as the **File**, **Edit** and **Help** menus), but others parts enable you to edit existing actions and create new ones (for example, launching your own application from the **User** menu). The following sections outline the steps required to configure the launcher. The configuration files are located in the <*CFXROOT*>`/etc/launcher/` directory (where <*CFXROOT*> is the path to your installation of CFX). You can open these files in any text editor, but you should not edit any of the configuration files provided by CFX, other than the `User.ccl` configuration file.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1.2.1. CCL Structure

The configuration files contain CCL objects that control the appearance and behavior of menus and buttons that appear in the Ansys CFX Launcher. There are three types of CCL objects: GROUP, APPLICATION and DIVIDER objects. The fact that there are multiple configuration files is not important; applications in one file can refer to groups in other files.

An example of how to add a menu item for the Windows calculator to the launcher is given in Example: Adding the Windows Calculator (p. 37).

1.2.1.1. GROUP

GROUP objects represent menus and toolbar groups in the Ansys CFX Launcher. Each new GROUP creates a new menu and toolbar. Nothing will appear in the menu or toolbar until you add APPLICATION or DIVIDER objects to the group. An example of a GROUP object is given below:

```
GROUP: CFX
  Position = 200
  Menu Name = &CFX
  Show In Toolbar = Yes
  Show In Menu = Yes
  Enabled = Yes
END
```

- The group name is set after the colon. In this case, it is "CFX". This is the name that APPLICATION and DIVIDER objects will refer to when you want to add them to this group. This name should be different to all other GROUP objects.
- Position refers to the position of the menu relative to others. The value should be an integer between 1 and 1000. Groups with a higher Position value, relative to other groups, will have their menu appear further to the right in the menu bar. Referring to Figure 1.1: Ansys CFX Launcher (p. 29), CFX has a lower position value than the ANSYS group. The **File** and **Edit** menus are always the first two menus and the **Help** menu is always the last menu.
- The title of the menu is set under Menu Name (this menu has the title **CFX**). The optional ampersand is placed before the letter that you want to have act as a menu accelerator (for example, **Alt**+**C** displays the **CFX** menu). You must be careful not to use an existing menu accelerator.
- The creation of the menu or toolbar can be toggled by setting the Show in Menu and Show in Toolbar options to Yes or No respectively. For example, you may want to create a menu item but not an associated toolbar icon.
- Enabled sets whether the menu/toolbar is available for selection or is disabled. Set the option to No to disable it.

1.2.1.2. APPLICATION

APPLICATION objects create entries in the menus and toolbars that will launch an application or run a process. Two examples are given below with an explanation for each parameter. The first example creates a menu entry in the **Tools** menu that opens a command line window. The second example creates a menu entry and toolbar button to start CFX-Solver Manager.

```
APPLICATION: Command Line 1
  Position = 300
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
  Group = Tools
  Tool Tip = Start a window in which CFX commands can be run
  Menu Item Name = Command Line
  Command = <windir>\system32\cmd.exe
  Arguments = /c start
  Show In Toolbar = No
  Show In Menu = Yes
  Enabled = Yes
  OS List = winnt
END
APPLICATION: CFXSM
  Position = 300
  Group = CFX
  Tool Tip = Launches Ansys CFX-Solver Manager
  Menu Item Name = CFX-Solver Manager
  Command = cfx5solve
  Show In Toolbar = Yes
  Show In Menu = Yes
  Enabled = Yes
  Toolbar Name = Ansys CFX-Solver Manager
  Icon = LaunchSolveIcon.xpm
  Shortcut = Ctrl+S
END
```

- The application name is set after the colon, in the first example it is "`Command Line 1`". This name should be different from all other `APPLICATION` objects.

- `Position`: sets the relative position of the menu entry. The value should be an integer between 1 and 1000. The higher the value, relative to other applications that have the same group, the further down the menu or the further to the right in a toolbar the entry will appear. If you do not specify a position, the object assumes a high position value (so it will appear at the bottom of a menu or at the right of a group of buttons).

- `Group`: sets the `GROUP` object to which this application belongs. The value must correspond to the name that appears after "`GROUP:`" in an existing `GROUP` object. The menu and/or toolbar entry will not be created if you do not specify a valid group name. The `GROUP` object does not have to be in the same configuration file.

- `Tool Tip`: displays a message when the mouse pointer is held over a toolbar button. In the "Command Line 1" example above, the `Tool Tip` entry is not used because a toolbar button is not created. This parameter is optional.

- `Menu Item Name`: sets the name of the entry that will appear in the menu. If you do not specify a name, the name is set to the name of the `APPLICATION`: object. The optional ampersand is placed before the letter that you want to have act as a menu accelerator (for example, **Alt**+**C** then **S** will start CFX-Solver Manager. **Alt**+**C** selects the **CFX** menu and **S** selects the entry from the menu). You must be careful not to use an existing menu accelerator.

- `Command`: contains the command to run the application. The path can be absolute (that is, use a forward slash to begin the path on Linux, or a drive letter on Windows). If an absolute path is not specified, a relative path from *<CFXROOT>*`/bin/` is assumed. If no command is specified, the menu item/toolbar button will not appear in the Ansys CFX Launcher. The path and command are checked when the launcher is started. If the path or command does not exist, the menu item/toolbar button will not appear in the launcher. You may find it useful to include environment variables in a command path; for details, see Including Environment Variables (p. 36).

- `Arguments`: specifies any arguments that need to be passed to the application. The arguments are appended to the value you entered for `Command`. You do not need to include this parameter

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

as there are no arguments to pass. You may find it useful to include environment variables in the arguments; for details, see Including Environment Variables (p. 36).

Distinct arguments are space-separated. If you need to pass an argument that contains spaces (such as a Windows filepath) you should include that argument in double quotes, for example:

```
Arguments = "C:\Documents and Settings\User" arg2 arg3
```

- `Show In Toolbar`: determines if a toolbar button is created for the application. This optional parameter has a default value of **Yes**.
- `Show In Menu`: determines if a menu entry is created for the application. This optional parameter has a default value of `Yes`.
- `Enabled`: controls the menu entry and toolbar button. Set this parameter to `No` to disable the application. This optional parameter has a default value of `Yes`.
- `OS List` is an optional parameter that enables you to set which operating system the application is suitable for. If `OS List` is not supplied, the launcher will attempt to create the menu item and toolbar button on all platforms.

  For example, the command to open a command line window varies depending on the operating system. In the 'Command Line 1' example above, the application only applies to Windows platforms. To complete the OS coverage, the launcher configuration files contain more 'Command Line' applications that apply to different operating systems.
- `Toolbar Name`: sets the name that appears on the toolbar button. This parameter is optional (because you may want to show only an icon).
- `Icon`: specifies the icon to use on the toolbar button and in the menu item. The path can be absolute (that is, use a forward slash to begin the path on Linux, or a drive letter on Windows). If an absolute path is not specified, a relative path from <*CFXROOT*>`/etc/icons` is assumed. The following file formats are supported for icon image files: Portable Network Graphics (`png`) and Pixel Maps (`ppm`, `xpm`). Other icons used in the launcher are 32 pixels wide and 30 pixels high. This parameter is optional. If it is not included, an icon will not appear.
- `Shortcut`: specifies the keyboard shortcut that can be pressed to launch the application. You must be careful not to use a keyboard shortcut that is used by any other `APPLICATION` object.

1.2.1.2.1. Including Environment Variables

In can be useful to use environment variables in the values for some parameters. You can specify an environment variable value in any parameter by including its name between the < > symbols. In the 'Command Line 1' example above, <`windir`> is used in the `Command` parameter so that the command would work on different versions of Windows. <`windir`> is replaced with the value held by the `windir` environment variable. The `Command` and `Argument` parameters are the only parameters that are likely to benefit from using environment variables. Environment variables included in the `Arguments` parameter are expanded before they are passed to the application.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1.2.1.3. DIVIDER

DIVIDER objects create a divider in a menu and/or toolbar (see the **Tools** menu for an example). An example of the CCL for DIVIDER objects is shown below.

```
DIVIDER: Tools Divider 1
  Position = 250
  Group = Tools
  OS List = winnt
END
```

The Position, Group and OS List parameters are the same as those used in APPLICATION objects. For details, see APPLICATION (p. 34).

1.2.2. Example: Adding the Windows Calculator

The following CCL is the minimum required to add the Windows calculator to the Ansys CFX Launcher:

```
GROUP: Windows Apps
  Menu Name = Windows
END
APPLICATION: Calc
  Group = Windows Apps
  Command = <windir>\system32\calc.exe
  Toolbar Name = Calc
END
```

Although the parameter Toolbar Name is not strictly required, you would end up with a blank toolbar button if it were not set.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 2: Volume Mesh Import API

The Mesh Import Application Programming Interface (API) enables you to build a customized executable that reads a 3-dimensional mesh from a 3rd-party mesh file into CFX-Pre and to extend the number of file formats that CFX-Pre can understand and read beyond those supplied as part of the standard installation.

The communication between the executable and CFX-Pre is via a communications channel that is controlled by use of routines in the API provided.

For details on using the Volume Mesh Import API, see User Import.

This chapter describes:

2.1. Valid Mesh Elements in CFX

2.2. Creating a Custom Mesh Import Executable for CFX-Pre

2.3. Details of the Mesh Import API

2.4. An Example of a Customized C Program for Importing Meshes into CFX-Pre

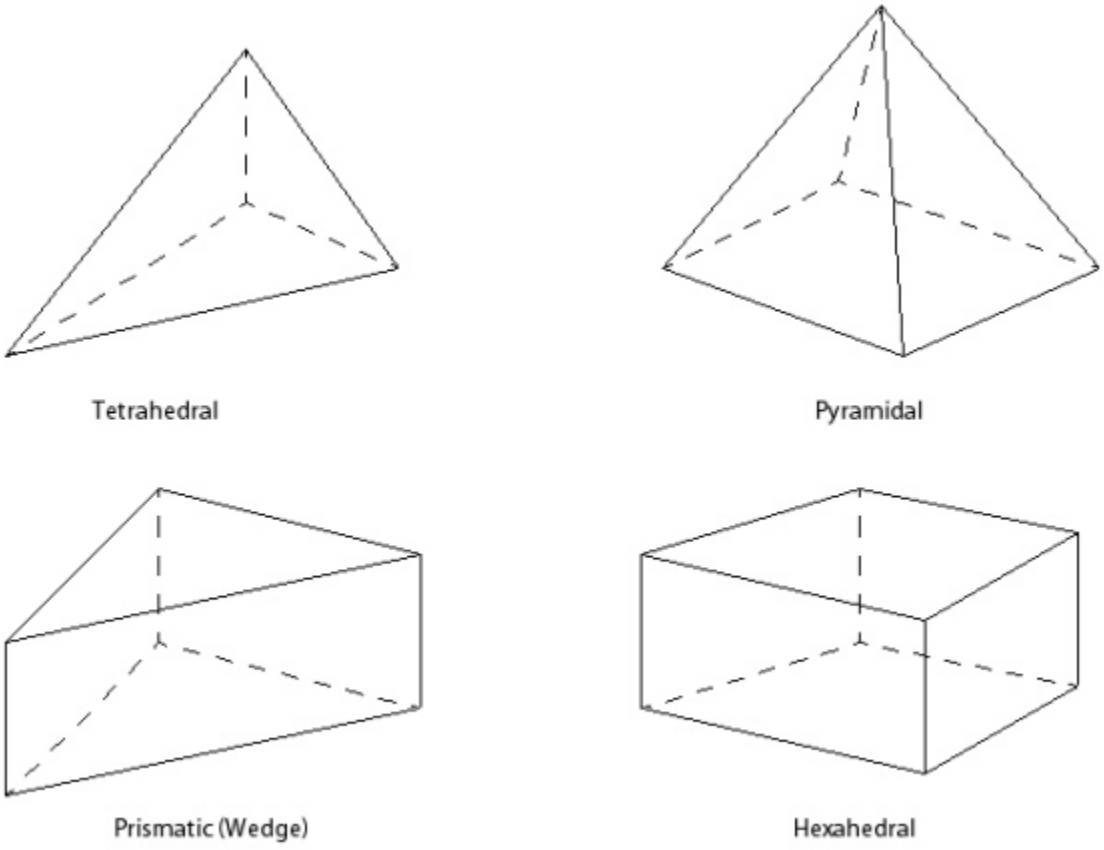
2.5. Import Programs

2.1. Valid Mesh Elements in CFX

The CFX-Solver technology works with unstructured meshes. This does not prohibit the use of structured meshes. However a structured mesh will always be dealt with internally as an unstructured mesh.

The CFX-Solver can solve flows in a mesh that involves any one or more of, and only, the following element types:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



You must write the program using the API to translate the mesh read from the 3rd-party file into a format that can be processed by CFX-Pre.

2.2. Creating a Custom Mesh Import Executable for CFX-Pre

You can create your own customized program using the 'C' programming language or Fortran programming language. A number of API functions are provided in a library supplied with the Ansys CFX installation. For details, see Details of the Mesh Import API (p. 43).

The installation contains a C source code example file that can be used as the basis of your custom executable. This file, `ImportTemplate.c`, is provided in *<CFXROOT>*`/examples/`, and is listed in: An Example of a Customized C Program for Importing Meshes into CFX-Pre (p. 58).

The basic structure of a program written to import a 3rd-party mesh into CFX-Pre is as follows:

1. Inclusion of the `cfxImport.h` header file (for C programs and not Fortran programs).
2. Initialization for import with the `cfxImportInit` routine.
3. Definition of node data with `cfxImportNode`.
4. Definition of element data with `cfxImportElement`.
5. Optionally, definitions of 2D and 3D regions with either `cfxImportRegion` or the following three functions: `cfxImportBegReg`, `cfxImportAddReg`, `cfxImportEndReg`

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6. Data transfer with `cfxImportDone`.

The header files associated with the API are located in *<CFXROOT>*`/include/`. If you do not use the header file `cfxImport.h`, the functionality of the routines contained within the API may not follow defined behavior.

After writing the program, you will need to compile the source code. For details, see Compiling Code with the Mesh Import API (p. 41).

You will also need to link your routine with the API routine libraries. For details, see Linking Code with the Mesh Import API (p. 41).

After a customized executable has been produced, it can be run in CFX-Pre. For details, see User Import.

2.2.1. Compiling Code with the Mesh Import API

Compilation of a customized executable must be performed using an appropriate compiler with appropriate compiler flags. The supported compilers are listed in Compiler Requirements for All Windows Versions in the *Ansys, Inc. Installation Guides* and Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

Note:

Windows users should note that custom mesh import programs must be compiled as multi-threaded applications.

The customized executable must be linked with the provided Mesh Import API library and the provided I/O library as detailed in Linking Code with the Mesh Import API (p. 41).

2.2.2. Linking Code with the Mesh Import API

In order to build a customized import utility routine, it must be linked with several libraries. These libraries are located in *<CFXROOT>*`/lib/`*<os>*`/`:

- `libmeshimport.lib` (on Windows), or `libmeshimport.a` (on Linux)
- `libratlas_api.lib` (on Windows), or `libratlas_api.a` (on Linux)
- `libratlas.lib` (on Windows), or `libratlas.a` (on Linux)
- `libpgtapi.lib` (on Windows), or `libpgtapi.a` (on Linux)
- `libunits.lib` (on Windows), or `libunits.a` (on Linux)
- `libcclapilt.lib` (on Windows), or `libcclapilt.a` (on Linux)
- `libio.lib` (on Windows), or `libio.a` (on Linux)

2.2.2.1. Linking a Customized Mesh Import Executable on a Windows Platform

The following topics are discussed:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.2.2.1.1. Linking a Customized C Mesh Import Executable on a Windows Platform
2.2.2.1.2. Linking a Customized Fortran Mesh Import Executable on a Windows Platform

2.2.2.1.1. Linking a Customized C Mesh Import Executable on a Windows Platform

You can build the executables on Windows systems. An example command line follows:

```
cl /MD /I "C:\Program Files\Ansys Inc\v251\CFX\include" ImportTem-
plate.c /link /libpath:"C:\Program Files\Ansys Inc\v251\CFX\lib\winnt-
amd64" libcclapilt.lib libio.lib libmeshimport.lib libunits.lib
libpgtapi.lib libratlas_api.lib libratlas.lib
```

2.2.2.1.2. Linking a Customized Fortran Mesh Import Executable on a Windows Platform

You can write the import program in Fortran and then compile and link it. An example command line follows:

```
ifort /MD /I "C:\Program Files\Ansys Inc\v251\CFX\include" /threads
/iface:mixed_str_len_arg ImportTemplate.F /exe:ImportTemplate.exe
/libs:dll /link /libpath:"C:\Program Files\Ansys
Inc\v251\CFX\lib\winnt-amd64" libcclapilt.lib libio.lib libmeshim-
port.lib libunits.lib libpgtapi.lib libratlas_api.lib libratlas.lib
```

The supported compilers are listed in Compiler Requirements for All Windows Versions in the *Ansys, Inc. Installation Guides*.

2.2.2.2. Linking a Customized Mesh Import Executable on a Linux Platform

The following topics are discussed:

2.2.2.2.1. Linking a Customized C Mesh Import Executable on a Linux Platform
2.2.2.2.2. Linking a Customized Fortran Mesh Import Executable on a Linux Platform

2.2.2.2.1. Linking a Customized C Mesh Import Executable on a Linux Platform

On most Linux systems you should be able to build the executable with the command:

```
gcc myimport.c -I<CFXROOT>/include/ -o myimport.exe -L<CFXROOT>/lib/linux-amd64 \
-lmeshimport -lratlas_api -lratlas -lpgtapi -lunits -lcclapilt -lio \
-L<AWPROOT>/tp/zlib/1.2.13/linx64/lib -lz -lm -lc -fopenmp
```

where <*CFXROOT*> is the directory in which CFX is installed, <*AWPROOT*> is the directory one level higher, and the version-specific directory under `zlib` is subject to change.

Here, `-lmeshimport`, `-lratlas_api`, `-lratlas`, `-lpgtapi`, `-lunits`, `-lcclapilt`, and `-lio` indicate the libraries mentioned above, while `-lm`, `-lc` and `-fopenmp` are system libraries.

In this example, your own import program is named `myimport.c` and the executable file will be called `myimport.exe`. You should ensure that the libraries to which you are linking (which are in the path given after `-L`) appear on the command line after the source file (or object file if you are just linking to an existing object).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The compiler flags and required libraries may vary, depending on the compiler and the custom program.

The supported compilers are listed in Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

2.2.2.2.2. Linking a Customized Fortran Mesh Import Executable on a Linux Platform

The following is an example of how to build the executable on Linux, when the source code for the executable is written in Fortran:

```
ifort myimport.F -L<CFXROOT>/lib/linux-amd64 -lmeshimport -lratlas_api -lratlas \
-lpgtapi -lunits -lcclapilt -lio -L<AWPROOT>/tp/zlib/1.2.13/linx64/lib -lz \
-lm -qopenmp -o myimport.exe
```

where <*CFXROOT*> is the directory in which CFX is installed, <*AWPROOT*> is the directory one level higher, and the version-specific directory under `zlib` is subject to change.

The supported compilers are listed in Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

2.3. Details of the Mesh Import API

This section contains information about the functions that are used to write a customized import executable in the Mesh Import API.

Before trying to use any of the routines listed in this section, it is highly recommended that you read Creating a Custom Mesh Import Executable for CFX-Pre (p. 40).

This section contains details of:

- Defined Constants (p. 44)
- Initialization Routines (p. 45)
- Termination Routines (p. 46)
- Error Handling Routines (p. 46)
- Node Routines (p. 47)
- Element Routines (p. 48)
- Primitive Region Routines (p. 51)
- Composite Regions Routines (p. 53)
- Explicit Node Pairing (p. 53)
- Fortran Interface (p. 54)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

• Unsupported Routines Previously Available in the API (p. 58)

Note:

In past releases of Ansys CFX the API has defined IDs of nodes and elements as integers (int). This release now uses a datatype ID_t to represent these quantities. This type is currently defined as an unsigned integer (unsigned int). This allows a greater number of nodes and elements to be imported than in the past.

2.3.1. Defined Constants

The following are defined in the header file `cfxImport.h`, which should be included in the import program.

2.3.1.1. Element Types

There are currently 4 types of elements, which are identified by the number of nodes: Tetrahedrons (4 nodes), pyramids (5 nodes), wedges or prisms (6 nodes), and hexahedrons (8 nodes). The element types may be identified by the defined constants:

```
#define cfxELEM_TET  4
#define cfxELEM_PYR  5
#define cfxELEM_WDG  6
#define cfxELEM_HEX  8
```

The element node ordering and local face numbering follow Patran Neutral file conventions for element descriptions.

2.3.1.2. Region Types

Regions may be defined in terms of nodes, faces or elements, based on the type argument to the `cfxImportBegReg` or `cfxImportRegion` routines. The three types are defined by the defined constants:

```
#define cfxImpREG_NODES  1
#define cfxImpREG_FACES  2
#define cfxImpREG_ELEMS  3
```

Node and Face regions define 2D regions of the imported mesh. Element regions define 3D regions of the imported mesh.

It is best to use face regions to define 2D regions of the mesh and element regions to define 3D regions of the mesh.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Node regions will be automatically transformed into a face region by the import process. This transformation requires the node IDs specified to define vertices of valid element faces. If no element faces can be constructed from the defined node region the node region will be deleted.

Note:

Due to the limited topological information recoverable from a set of nodes it is not advisable to define 2D regions internal to a 3D region using nodes. In this case it is advisable to use Face regions.

Node regions are specified by a list of node IDs.

Face regions are defined by a list of face IDs. These face IDs are a combination of an element ID and a local face number in the element.

2.3.2. Initialization Routines

The following routines check and initialize the Import API. With the exception of `cfxImportStatus` the first call to the Import API must be either `cfxImportInit` for communication with CFX, or `cfxImportTest` for testing the import routine in stand-alone mode.

2.3.2.1. cfxImportStatus

```
int cfxImportStatus ()
```

Returns 0 if descriptor is not opened and -1 if not opened for writing. In the normal case, 1 is returned if opened for writing to CFX, and 2 if opened for writing to a file.

2.3.2.2. cfxImportInit

```
void cfxImportInit ()
```

Performs initialization to begin communicating with CFX. This routine should be called early on in the import program to let CFX know that data is to be sent. If not called within 60 seconds, CFX will terminate the import process. If called and there is no connection with CFX, then the routine `cfxImportTest("/dev/null")` (UNIX) or `cfxImportTest("null")` (Windows) will be called. This routine will be automatically called by most of the API routines if not already called.

There is no return value for this routine. In the case of an error, `cfxImportFatal` will be called.

2.3.2.3. cfxImportTest

```
int cfxImportTest (filename)
    char *filename;
```

This routine allows testing of import program in isolation from CFX by writing data to a file *<file name>* instead of attempting to write it to the CFX communication channel.

The routine will return the file descriptor of the output file or will terminate with a call to `cfxImportFatal` on error.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.3. Termination Routines

With the exception of `cfxImportTotals` the last call to the Import API must always be `cfxImportDone`. This function performs the final processing of the import data, and then transfers the data to CFX.

2.3.3.1. cfxImportDone

```
long cfxImportDone ()
```

Indicate to the import API that all mesh data has been given and the API should now send the data to CFX. Except for `cfxImportTotals`, this should be last call made to the API. Returns the total number of bytes transferred to CFX by the import program.

2.3.3.2. cfxImportTotals

```
long cfxImportTotals (counts)
size_t counts[cfxImpCNT_SIZE];
```

Get the total number of nodes, elements, regions and other useful information given to the mesh import API by the program. This information is returned in the array `counts`, which should be of size at least `cfxImpCNT_SIZE` (currently defined as 9). The values returned in `counts` may be indexed by the `enum` list in `cfxImport.h`, which is:

```
counts[cfxImpCNT_NODE]    = number of nodes
counts[cfxImpCNT_ELEMENT] = number of elements
counts[cfxImpCNT_REGION]  = number of regions
counts[cfxImpCNT_UNUSED]  = number of unused nodes
counts[cfxImpCNT_DUP]     = number of duplicate nodes
counts[cfxImpCNT_TET]     = number of tetrahedral elements
counts[cfxImpCNT_PYR]     = number of pyramid elements
counts[cfxImpCNT_WDG]     = number of wedge elements
counts[cfxImpCNT_HEX]     = number of hexahedral elements
```

The return value for the function is the total number of bytes of data sent to CFX or written to the test file given when `cfxImportTest` was called.

2.3.4. Error Handling Routines

The first error handling routine allows the programmer to define an error callback function that is called when a fatal error is generated by the API or explicitly by the programmers code.

The second routine performs a method for clean termination of the program, shutting down the program and communication with Ansys CFX.

2.3.4.1. cfxImportError

```
void cfxImportError (callback)
void (*callback)(char *errmsg);
```

Define a user routine to be called before terminating due to a fatal error. `callback` is the application-supplied function to be called in the case of an error. The callback routine takes a single argument, `errmsg`, which will be passed by `cfxImportFatal` and should be processed by the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

callback function as a brief message describing the error that has occurred. If this function is not called or callback is not specified, then the normal termination behavior of the mesh import API will be that the any fatal errors will write the error message to `stderr` as well as being sent to CFX.

2.3.4.2. cfxImportFatal

```
void cfxImportFatal (errmsg)
char *errmsg;
```

Terminate with an error message, `errmsg`. This routine will send the message to CFX, shut down the communication channel or test file and call the user callback function (if specified by a call to cfxImportError).

There is no return from this call. The import program will terminate immediately after clean up tasks have been performed.

2.3.5. Node Routines

These routines define the 3D coordinates of points in space(nodes) that will be used to define elements or 2D regions that are to be imported to CFX. Each node has a unique identifier called a node ID.

2.3.5.1. cfxImportNode

```
ID_t cfxImportNode (nodeid, x, y, z)
ID_t nodeid;
double x, y, z;
```

Define a node in the import API to be subsequently imported into CFX. The unique identifier of the node is given by `nodeid`, and the coordinates of the node by x, y, and z.

Returns 0 if `nodeid` is invalid (less than 1), or `nodeid` is successfully defined. If a node with the same identity has already been defined, the coordinate values will alter to the supplied values.

2.3.5.2. cfxImportGetNode

```
ID_t cfxImportGetNode (nodeid, x, y, z)
ID_t nodeid;
double *x, *y, *z;
```

Get the coordinates for the node identified by `nodeid` and return the values in `x`, `y`, and `z`. Returns 0 if the node has not been defined or the node ID for the node.

2.3.5.3. cfxImportNodeList

```
ID_t * cfxImportNodeList ()
```

Returns an array of all node identifiers currently defined or `NULL` if no nodes have been defined. The first entry in the array is the number of nodes currently defined.

The memory for the array returned is allocated using `malloc` by the routine, consequently it should be destroyed when no longer required by calling `free`.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.6. Element Routines

The following routines define the topology of elements (using node IDs) that are to be imported to CFX. Also included here are routines that get the local face number and vertices of an element.

2.3.6.1. cfxImportElement

```
ID_t cfxImportElement (elemid, elemtype, nodelist)
ID_t elemid, *nodelist; int elemtype;
```

Define a new element to be imported to CFX. The unique identifier of the element is given by `elemid`, the element type by `elemtype` and the list of vertices by `nodelist`. If an element with the same ID has already been defined, it will be replaced by the new element being defined.

Only volume elements are currently supported by CFX; these may be tetrahedrons (4 vertices), pyramids (5 vertices), prisms (6 vertices) or hexahedrons (8 vertices). `elemtype` is the number of vertices for the element.

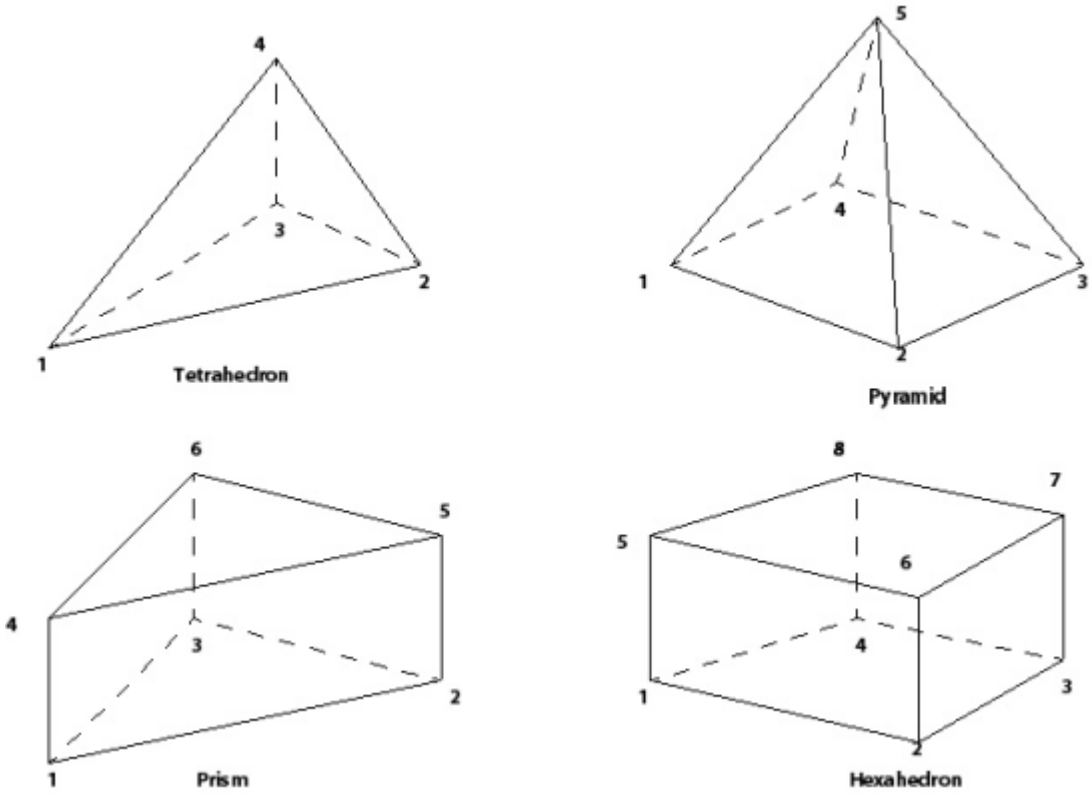
The following defines are included in the header file, `cfxImport.h` for convenience:

```
#define cfxELEM_TET  4  /* tet element (4 nodes)      */
#define cfxELEM_PYR  5  /* pyramid element (5 nodes)  */
#define cfxELEM_WDG  6  /* wedge element (6 nodes)    */
#define cfxELEM_HEX  8  /* hex element (8 nodes)      */
```

The list of vertices in `nodelist` refers to IDs of nodes that on termination of the import program by a call to `cfxImportDone` must have been defined by calls to `cfxImportNode`. If this is not the case a fatal error will be reported and the API will terminate.

The vertex ordering for the elements follows Patran Neutral File element conventions, and is shown in the following figure.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

The vertex ordering for the export API is different. For details, see cfxExportElementList (p. 89).

Returns 0 in the case of an `elemid` is invalid (less than 1) or an unsupported value is given by `elemtype`, or `elemid` if the element is successfully defined. If the element already exists the vertices of the element will be redefined.

2.3.6.2. cfxImportGetElement

```
ID_t cfxImportGetElement (elemid, nodelist)
ID_t elemid, nodelist[];
```

Get the node IDs for corresponding to the vertices of element identified by `elemid` and store in the array nodelist. This array must be at least as large the number of vertices for the element (a size of 8 will handle all possible element types).

Returns 0 if the element is not defined, or the element type (number of vertices). The node IDs will be ordered in the order expected by `cfxImportElement` if the program was to redefine the element.

2.3.6.3. cfxImportElementList

```
ID_t * cfxImportElementList ()
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns an array of all the currently defined element IDs or NULL if no elements have been defined. The first entry in the array is the number of elements.

The memory for the array returned is allocated using `malloc` by the routine, consequently it should be destroyed when no longer required by calling `free`.

2.3.6.4. cfxImportGetFace

```
ID_t cfxImportGetFace (elemid, facenum, nodelist)
ID_t elemid, nodelist[]; int facenum;
```

Gets the node IDs for the local `facenum`'th face of the element identified by `elemid`.

The node IDs are returned in `nodelist`, which should be of at least of size 4. The nodes correspond to the vertices of the face and are ordered counter-clockwise such that the normal for the face points away from the element. The face numbers and associated node indices are modeled after Patran Neutral File elements, and are tabulated here:

Element Type	**Face**	**Nodes**			
tetrahedron	1	1	3	2	
	2	1	2	4	
	3	2	3	4	
	4	1	4	3	
pyramid	1	1	4	3	2
	2	1	2	5	
	3	2	3	5	
	4	3	4	5	
	5	1	5	4	
prism	1	1	3	2	
	2	4	5	6	
	3	1	2	5	4
	4	1	4	6	3
	5	2	3	6	5
hexahedron	1	1	2	6	5
	2	3	4	8	7
	3	1	4	3	2
	4	2	3	7	6
	5	5	6	7	8
	6	1	5	8	4

Note:

The face numbers and associated node indices are different when exporting elements. For details, see cfxExportFaceNodes (p. 91).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns -1 if the element has not been defined, 0 if the face number is out of range, or the number of nodes for the face (3 or 4):

2.3.6.5. cfxImportFindFace

```
ID_t cfxImportFindFace (elemid, nnodes, nodeid)
ID_t elemid, nodeid[]; int nnodes;
```

Gets the local face number in element identified by `elemid` that contains all the nodes supplied by the calling routine in `nodeid`. `nnodes` is the number of nodes for the face (3 or 4).

Returns -1 if the element is not found or `nodeid` is not supplied or `nnodes` is greater than 4 or less than 3. Returns 0 if there is no match, or the local face number (1 to 6) of the element.

2.3.7. Primitive Region Routines

The following routines enable the specification of 2D regions as a group of nodes or faces, or a 3D region as a group of elements. In the case of nodes and faces, only those that define faces of valid imported elements will be imported; others are ignored by CFX.

2.3.7.1. cfxImportBegReg

```
int cfxImportBegReg (regname, regtype)
char *regname;
int regtype;
```

Initialize for the specification of a region. If a region is currently being defined, `cfxImportEndReg` will be called.

The name of the region is given by `regname`. If the region name is `NULL`, the name `Unnamed Region 2D` or `Unnamed Region 3D`, with a sequential integer appended, will be used. If a region named `regname` has already been defined, then additional objects will be added to the previous region.

The type of region is given by regtype, which should be one of `cfxImpREG_NODES`, `cfxImpREG_FACES` or `cfxImpREG_ELEMS` depending on whether the region is to be defined by nodes, faces or elements, respectively. It is not currently possible to mix types in a region; doing so will cause the import API to terminate with an error message.

Returns the number of objects (node, faces or elements) currently in the region.

2.3.7.2. cfxImportAddReg

```
int cfxImportAddReg (numobjs, objlist)
int numobjs, *objlist;
```

Add IDs of objects being defined to the current region.

A region must be currently defined or reactivated by `cfxImportBegReg` or an error will occur, and the API will terminate.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The number of objects to add is given by `numobjs` and the IDs of the objects are supplied in `objlist`.

The objects are interpreted as node IDs, face IDs, or element IDs, depending on the type of the region indicated when `cfxImportBegReg` was called.

On calling `cfxImportDone`, any node IDs , face IDs or element IDs specified in the object list must have been defined by the appropriate routine or they will be removed from the region.

Returns the total number of objects in the current region after the object IDs have been added.

2.3.7.3. cfxImportEndReg

```
int cfxImportEndReg ()
```

End the specification of the current region.

Returns the number of objects (nodes, faces or elements) in the region.

2.3.7.4. cfxImportRegion

```
int cfxImportRegion (regname, regtype, numobjs, objlist)
char *regname;
int regtype, numobjs, *objlist;
```

Import a region named `regname` of type `regtype`. The number of objects to add to the region is given by `numobjs`, and the list of object IDs by `objlist`. This routine combines calls to `cfxImportBegReg`, `cfxImportAddReg` and `cfxImportEndReg`.

Returns the total number of objects in the region on termination of the routine.

2.3.7.5. cfxImportRegionList

```
char ** cfxImportRegionList ()
```

Return a NULL terminated list of currently defined region names.

The memory for the array and each character string in the array returned is allocated using `malloc` by the routine, consequently each array member and the array itself should be destroyed when no longer required by calling `free`.

2.3.7.6. cfxImportGetRegion

```
int * cfxImportGetRegion (regname)
char *regname;
```

Returns a list of objects in the region named `regname`, or `NULL` if the region does not exist. The first entry in the returned list is the region type and the second entry is the number of object IDs.

The memory for the array is allocated using `malloc` by the routine, consequently the array itself should be destroyed when no longer required by calling `free`.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.8. Composite Regions Routines

The following routines enable composite regions to be defined in terms of primitive regions or other composite regions.

2.3.8.1. cfxImportBegCompRegion

```
cfxImportBegCompReg()
char *regionName;
```

Begin defining a composite region with the name `regionName`,

Returns -1 if a primitive region `regionName` is already defined or memory couldn't be allocated, or 0 if successfully created.

2.3.8.2. cfxImportAddCompRegComponents

```
int cfxImportAddCompRegComponents(componentCount,components)
int componentCount;
char **components;
```

Add a set of component region names specified in `components` to the composite region currently being defined. `componentCount` specified how many components are specified in the `components` array,

Returns -1 if a composite region is not being defined or insufficient memory is available to add the components of the composite region, or 0 if the components were successfully added.

2.3.8.3. cfxImportEndCompReg

```
int cfxImportEndCompReg()
```

Finish defining the current composite region.

Returns -1 if a composite region is not currently being defined or 0 otherwise.

2.3.8.4. cfxImportCompositeRegion

```
int cfxImportCompositeRegion(regionName, componentCount, components)
char *regionName, **components;
int componentCount;
```

Define a composite region named `regionName` with `componentCount` components supplied in character array `components`.

Returns 0 if successful or -1 if an error occurred preventing the composite region being defined.

2.3.9. Explicit Node Pairing

The following routine provides a method for explicitly marking two nodes as being identical (or in the same position in space).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.9.1. cfxImportMap

```
ID_t cfxImportMap (nodeid, mapid)
ID_t nodeid, mapid;
```

Explicitly map the node identified by `nodeid` to the node identified by `mapid`.

On calling `cfxImportDone` the Mesh Import API will update regions and elements referencing the mapped node to the node it is mapped to. This therefore reduces the total node count imported to CFX and eliminates the duplicate nodes.

Duplicate nodes may also be removed by CFX if the appropriate options are selected in the CFX interface and an appropriate tolerance set. For details, see Importing Meshes in the *CFX-Pre User's Guide*.

2.3.10. Fortran Interface

The following routines are callable from Fortran, and interface with the corresponding C routine. There are currently no return values.

2.3.10.1. cfxinit

```
call cfxinit
```

Interface to `cfxImportInit`. Initializes for import.

2.3.10.2. cfxtest

```
CHARACTER*n filename
call cfxtest(filename)
```

Interface to `cfxImportTest`. <*`filename`*> is a CHARACTER*n value that gives the name of the file to dump the output to.

2.3.10.3. cfxunit

```
CHARACTER*n units
call cfxunit(units)
```

Interface to `cfxImportUnits`. Specify the units the mesh is specified in.

2.3.10.4. cfxwarn

```
CHARACTER*n mesg
call cfxwarn(mesg)
```

Interface to `cfxImportWarning`. Emit a warning message `mesg`.

2.3.10.5. cfxfatl

```
CHARACTER*n mesg
call cfxfatl(mesg)
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Interface to `cfxImportFatal`. Emit a warning message `mesg` and terminate the program cleanly.

2.3.10.6. cfxdone

```
call cfxdone
```

Interface to `cfxImportDone`. Terminates the program and transfers the data to CFX-Pre.

2.3.10.7. cfxnode

```
INTEGER idnode
DOUBLE PRECISION x,y,z
call cfxnode(idnode,x,y,z)
```

Interface to `cfxImportNode`. Imports a node with the specified coordinates. `idnode` is an INTEGER value for the node ID, and x, y, and z are the DOUBLE PRECISION coordinates of the node.

2.3.10.8. cfxnodg

```
INTEGER idnode
DOUBLE PRECISION x,y,z
call cfxnodg(idnode,x,y,z)
```

Interface to `cfxImportGetNode`. Queries the current coordinates or a node referenced by `idnode`. `idnode` is an INTEGER value for the node ID, and x, y, and z are the DOUBLE PRECISION coordinates of the node.

2.3.10.9. cfxnods

```
INTEGER ids(*)
call cfxnods(ids)
```

Interface to `cfxImportNodeList`. Retrieves the list of all valid node IDs having been imported into the API. `ids` is an INTEGER array that must be at least as large as the number of nodes currently imported.

2.3.10.10. cfxelem

```
INTEGER idelem,itelem,nodes(*)
call cfxelem(idelem,itelem,nodes)
```

Interface to `cfxImportElement`. `idelem` is element ID, and `itelem` is the element type (number of nodes - 4,5,6, or 8). Both are of type INTEGER. `nodes` is an array of INTEGER node IDs dimensioned of size at least `itelem`.

2.3.10.11. cfxeleg

```
INTEGER idelem,itelem,nodes(*)
call cfxeleg(idelem,itelem,nodes)
```

Interface to `cfxImportGetElement`. Queries the current node ids that define the vertices of the element referenced by the id `idelem`. `idelem` is element ID, and `itelem` is the element type

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

(number of nodes - 4, 5, 6, or 8). Both are of type INTEGER. `nodes` is an array of INTEGER values that will contain the node IDs on successful return. It should be dimensioned of size at least `itelem`.

2.3.10.12. cfxeles

```
INTEGER ids(*)
call cfxeles(ids)
```

Interface to `cfxImportElemList`. Retrieves the list of all valid element IDs having been imported into the API. `ids` is an INTEGER array that must be at least as large as the number of elements currently imported.

2.3.10.13. cfxfacd

```
INTEGER eleid, elefc, id
call cfxfacd(eleid, elefc, id)
```

Interface to `cfxImportFaceID`. Defines a face id (`id`) in terms of an element ID (`eleid`) and local face (`elefc`) of that element.

2.3.10.14. cfxface

```
INTEGER eleid, elefc, vtx(*)
INTEGER cfxface(eleid, elefc, vtx)
```

Interface to `cfxImportGetFace`. Returns the node IDs of the vertices defining a face located by the element ID (`eleid`) and local face (`elefc`) of that element.

2.3.10.15. cfxffac

```
INTEGER eleid, nvtx, vtx(*), elefc
call cfxffac(eleid, nvtx, vtx, elefc)
```

Interface to `cfxImportFindFace`. Returns the local face (`elefc`) of an element (`eleid`) that is defined by the vertices (`vtx`).

2.3.10.16. cfxregn

```
CHARACTER*n regname
INTEGER type,nobjs,objs(*)
call cfxregn(regname,type,nobjs,objs)
```

Interface to `cfxImportRegion`. Regname is a CHARACTER*n string defining the region name, type is an INTEGER value specifying the type of region, either 1 for nodes, 2 for faces, or 3 for elements. `nobjs` is an INTEGER value that gives the number of objects in the region, and objs is an INTEGER array of object IDs dimensioned at least size `nobjs`.

2.3.10.17. cfxregb

```
CHARACTER*n regname
INTEGER type
call cfxregb(regname,type)
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Interface to `cfxImportBegReg`. Start defining a new region or make an existing region of the same name the current one if it already exists and is of the same type. `regname` is a CHARACTER*n string defining the region name, type is an INTEGER value specifying the type of region, either 1 for nodes, 2 for faces, or 3 for elements.

2.3.10.18. cfxrega

```
INTEGER nobjs,objs(*)
call cfxrega(nobjs,objs)
```

Interface to `cfxImportAddReg`. Add the objects (`objs`) to the current region. `nobjs` is an INTEGER value that gives the number of objects to add to the region, and `objs` is an INTEGER array of object IDs dimensioned at least size `nobjs`.

2.3.10.19. cfxrege

```
call cfxrege()
```

Interface to `cfxImportEndReg`. Finish defining the current region (after the call there will be no current region).

2.3.10.20. cfxregs

```
CHARACTER*n regname
INTEGER numobj
call cfxregs(regname,numobj)
```

Query how many objects (returned in `numobj`) are referenced by the region `regname`. `regname` is a CHARACTER*n string specifying the region name.

2.3.10.21. cfxregg

```
CHARACTER*n regname
INTEGER type, obj(*)
call cfxregg(regname, type, objs)
```

Get the type (`type`) and object IDs (`objs`) referenced by the region `regname`. `regname` is a CHARACTER*n string specifying the region name. `type` is INTEGER and `objs` is an INTEGER array at least of the size returned by `cfxregs`.

2.3.10.22. cfxcmpb

```
CHARACTER*n regname
call cfxcmpb(regname)
```

Interface to `cfxImportBegCompReg`. Start defining a new composite region or make an existing composite region of the same name as the current one if it already exists. `regname` is a CHARACTER*n string defining the region name.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.3.10.23. cfxcmpa

```
INTEGER nregs
CHARACTER*(n) regs
call cfxcmpa(nregs,regs)
```

Interface to `cfxImportAddCompReg`. Add the region names (`regs`) to the current composite region being defined. `nregs` is an INTEGER value that gives the number of regions to add to the region, and `regs` is a CHARACTER*(*) array of region names dimensioned at least size `nregs`.

2.3.10.24. cfxcmpe

```
call cfxcmpe()
```

Interface to `cfxImportEndCompReg`. Finish defining the current composite region (after the call there will be no current composite region).

2.3.11. Unsupported Routines Previously Available in the API

In Ansys CFX 2025 R1 certain functionality available in previous releases is no longer supported. These routines have been removed because they are directly implemented in CFX.

The following is a list of routines removed from the mesh import API:

```
cfxImportFixElements
cfxImportTolerance
cfxImportGetTol
cfxImportSetCheck
cfxImportRange
cfxImportCheck
cfxtol
cfxset
cfxchk
```

2.4. An Example of a Customized C Program for Importing Meshes into CFX-Pre

An example, `ImportTemplate.c`, can be found in <*CFXROOT*>`/examples`.

2.5. Import Programs

The following sections detail the standard import programs currently available within CFX-Pre and their command line equivalents.

Information about importing meshes from the CFX-Pre interface is given in Importing Meshes in the *CFX-Pre User's Guide*.

If you want to use command line options that cannot be specified through the CFX-Pre User Interface, then you may want to run these programs as user-defined mesh import programs. User Import details how to run a mesh import program.

The executables are located in <*CFXROOT*>`/bin/`<*os*>.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Ansys (p. 59)
- CFX Def/Res (p. 59)
- CFX-4 (p. 60)
- CFX-5.1 (p. 60)
- CFX-TfC (p. 61)
- CGNS (p. 62)
- Fluent (p. 63)
- GridPro/az3000 (p. 63)
- I-DEAS (p. 64)
- ICEM CFX (p. 64)
- PATRAN (p. 64)
- NASTRAN (p. 64)
- CFX-TASCflow (p. 65)

2.5.1. Ansys

Imports an Ansys file. The external import routine is `ImportANSYS`. Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-E` Import Elements of the same type as regions.

`-A` Import ANSA parts as regions.

`-S` Display a list of all supported element types.

2.5.2. CFX Def/Res

Imports the mesh from a CFX-Solver input or results file. The external import routine is `ImportDef`. Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-I` Read mesh from the initial timestep in the file.

`-L` Read mesh from the last timestep in the file.

`-T<`*`timestep`*`>` Read mesh from the timestep specified (Transient files)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.5.3. CFX-4

Imports a CFX-4 grid file. The external import routine is `ImportCFX4`.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-C` Read coordinates as being in cylindrical coordinates.

`-i` Included interfaces in regions.

`-3` Include USER3D and POROUS regions as 3D regions.

`-c` Import blocked-off conducting solid regions as 3D regions.

`-l` Include blocked-off solid regions as 3D regions.

`-X` Import axisymmetric problem with default values in geometry file.

`-a` *<nk>* Override the number of planes created in the k direction by `nk` (for example, split theta with `nk` planes) for axisymmetric import.

`-A` *<theta>* Create a total sector of *<theta>* degrees for axisymmetric import.

`-S` Rename multiple symmetry planes with the same name to conform to CFX-Solver requirements (that is, must lie in a plane).

2.5.4. CFX-5.1

Imports a CFX-5.1 results file. The external import routine is `ImportCFX5`.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-f` Input file is formatted.

`-u` Input file is unformatted (Fortran).

`-M` *<machine type>* Set the machine type in the case of a binary or unformatted file so that data conversion may be done if needed. The default file format is 32-bit IEEE (Iris, Sun, HP, IBM). The currently recognized machine types are:

- IEEE - generic 32-bit IEEE machine.
- BSIEEE - generic 32-bit byteswapped IEEE machine.
- IBM - IBM 32-bit IEEE.
- IRIS - Iris 32-bit IEEE.
- HP - HP 32-bit IEEE.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- SUN - Sun 32-bit IEEE.
- ALPHA - Compaq Tru64 UNIX Alpha 64-bit byte-swapped IEEE.
- DOS - DOS 16-bit byte-swapped IEEE.
- Compaq Tru64 UNIX - Compaq Tru64 UNIX 32-bit byte-swapped IEEE.
- CRAY - Cray 64-bit format.
- CONVEX - native Convex floating point format.
- Windows - 32-bit Windows.

The argument machine type is case insensitive, and only the first 2 characters are needed (any others are ignored).

`-M` <*machine type*> Set the machine type in the case of a binary or unformatted file so that data conversion may be done if needed. The default file format is 32-bit IEEE (Iris, Sun, HP, IBM). The currently recognized machine types are:

- IEEE - generic 32-bit IEEE machine.
- BSIEEE - generic 32-bit byteswapped IEEE machine.
- IBM - IBM 32-bit IEEE.
- IRIS - Iris 32-bit IEEE.
- HP - HP 32-bit IEEE.
- SUN - Sun 32-bit IEEE.
- ALPHA - Compaq Tru64 UNIX Alpha 64-bit byte-swapped IEEE.
- DOS - DOS 16-bit byte-swapped IEEE.
- Compaq Tru64 UNIX - Compaq Tru64 UNIX 32-bit byte-swapped IEEE.
- CRAY - Cray 64-bit format.
- CONVEX - native Convex floating point format.
- Windows - 32-bit Windows.

The argument machine type is case-insensitive, and only the first two characters are needed (any others are ignored).

2.5.5. CFX-TfC

Imports a CFX-TfC 1.3 mesh file. The external import routine is `ImportGEM`.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

`-f` Input file is formatted.

`-u` Input file is unformatted (Fortran).

`-r` Read regions from a `BFI` file.

`-b` <file> Use file as a BFI filename instead of default name.

`-M` <*`machine type`*> Set the machine type in the case of a binary or unformatted file so that data conversion may be done if needed. The default file format is 32-bit IEEE (Iris, Sun, HP, IBM). The currently recognized machine types are:

- IEEE - generic 32-bit IEEE machine.
- BSIEEE - generic 32-bit byteswapped IEEE machine.
- IBM - IBM 32-bit IEEE.
- IRIS - Iris 32-bit IEEE.
- HP - HP 32-bit IEEE.
- SUN - Sun 32-bit IEEE.
- ALPHA - Compaq Tru64 UNIX Alpha 64-bit byte-swapped IEEE.
- DOS - DOS 16-bit byte-swapped IEEE.
- Compaq Tru64 UNIX - Compaq Tru64 UNIX 32-bit byte-swapped IEEE.
- CRAY - Cray 64-bit format.
- CONVEX - native Convex floating point format.
- Windows - 32-bit Windows.

The argument machine type is case insensitive, and only the first 2 characters are needed (any others are ignored).

2.5.6. CGNS

Imports a CGNS file. The external import routine is `ImportCGNS`. Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-b` Read a grid from the specific CGNS base.

`-B` Read all CGNS bases. (default)

`-c` Read BOCO information as 2D regions.

`-f` Import Family Information as regions.

`-E` Import each Element Section as a separate region.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

`-I` Import each side of a connection as a separate region.

`-P` Do not add the Zone name as a prefix to any region being defined.

2.5.7. Fluent

Imports Fluent `msh` and `cas` files. The external import routine is `ImportFluent`. The import routine will read the mesh information from the `.cas` or `.msh` file.

Available command line options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-I` Import interior boundary conditions.

2.5.8. GridPro/az3000

Imports a GridPro/az3000 grid and connectivity file from Program Development Corporation (PDC).

The external import routine is `ImportPDC`. The import routine will attempt to determine the connectivity file associated with the grid file by appending the extension `conn` to the grid filename. If the file is not found, then the grid filename extension will be replaced by `conn` and the new file checked for. If neither of these are found, the import routine will look for a file named `conn.tmp`, and if found will use it. A command line option (`-c`) is also available to explicitly name the connectivity file.

If a connectivity file is found, the interface information in the file will be used to eliminate the duplicate nodes at block interfaces, and boundaries conditions will be imported as regions into CFX. If the boundary condition is named in the connectivity file, then that name will be used for the region name, else the default name **UnnamedRegionX** with the X replaced by a number will be used. If a connectivity file is not found, or the command line option to ignore the connectivity file is given (`-i`), then only the grid file will be imported, resulting in duplicate nodes at the block interfaces. You may then want to eliminate these duplicate nodes with the command line option (`-d` or `-D`).

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-i` Ignore the connectivity file. Duplicate nodes will result and no regions will be imported.

`-c` <*`connfile`*> Set the name of the connectivity file associated with the grid file to <*`connfile`*>.

`-p` Include periodic boundary conditions as regions. These are not normally included in the import. Setting this flag will result in these being imported as regions.

`-q` Read from the property file

`-P` <*`propfile`*> Set the name of the property file associated with the grid file to <*`propfile`*>.

`-3` Import grid blocks as 3D regions

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.5.9. I-DEAS

Imports an I-DEAS Universal file from SDRC. The external import routine is ImportIDEAS. Reads datasets 781 and 2411 (nodes) as nodes, 780 and 2412 (elements) as elements, and nodes (type 7) from datasets 752 and 2417 (permanent groups) as regions. All other datasets are read, but not processed.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-n` Import nodes in a PERMANENT group as a 2D region.

`-l` Import elements in a PERMANENT group as a 3D region.

`-f` Import faces in a PERMANENT group as a 2D region.

2.5.10. ICEM CFX

Imports a file written for CFX by ICEM Tetra. The external import routine is `ImportICEM`. Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-P` Read coordinate data from a binary file as double precision.

2.5.11. PATRAN

Imports a PATRAN Neutral file. The external import routine is `ImportPatran`. Reads packet 01 (nodes) as nodes, packet 02 (elements) as elements, and nodes (type 5) from packet 21 (named groups) as regions. A command line option is available to read packet 06 (loads) as regions also. All other packets are read, but not processed.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-l` Import packet 06 (distributed loads) as regions. The regions will be assigned the name Patran-Load<*X*> where the <*X*> is replaced by the load ID number.

2.5.12. NASTRAN

Imports a NASTRAN file. The external import routine is `ImportMSC`. Currently reads only nodes (GRID), tet (CTETRA) and hex (CHEXA) elements.

Available options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-l` Import PLOAD4 datasets as 2D regions.

`-s` Import PSOLID datasets as 3D regions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2.5.13. CFX-TASCflow

Imports TASCflow Version 2 files. The external import routine is `ImportGRD`. The import routine will read the mesh information from the GRD file and automatically remove duplicate nodes where interfaces are defined and are 1:1.

Available command line options are:

`-v` Verbose output. Echo additional data to `stdout` during the import.

`-V` More verbose output.

`-i` Ignore the blockoff file (BCF).

`-c` Ignore GCI file.

`-o` Old style 2.4 format.

`-b <file>` Specifies a `bcf` file that contains blocked-off regions (boundary condition information is ignored). For details, see CFX-TASCflow Files in the *CFX-Pre User's Guide*.

`-g <file>` Specifies the `gci` file to import. For details, see CFX-TASCflow Files in the *CFX-Pre User's Guide*.

`-f` Formatted (ASCII) GRD file.

`-u` Fortran unformatted GRD file.

`-3` Import labelled 3D regions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 3: Mesh and Results Export API

This chapter describes how to create a custom program for exporting mesh and results data. Information on using such a program is given in Using a Customized Export Program.

This chapter describes:

3.1. Creating a Customized Export Program

3.2. Compiling Code with the Mesh and Results Export API

3.3. Linking Code with the Mesh and Results Export API

3.4. Details of the Mesh Export API

3.1. Creating a Customized Export Program

The mesh and results contained within an Ansys CFX results file can be exported in many formats, ready for input into postprocessing software other than CFD-Post, MSC/PATRAN, EnSight and Fieldview. To do this, you would write a customized export program that calls routines from the Export Application Programming Interface (API). However, this is recommended only for advanced users, because it involves at least some knowledge of C or C++ programming language.

Once an export program has been created, it can be used by any number of users; so if other Ansys CFX users at a site regularly use a different postprocessor, it may be worth contacting a system administrator to find out if such a format has already been defined.

Note:

There is no support for the export of cases that use Transient Blade Row modeling.

To define a new format, use the export API. The general steps to follow are:

1. Create a file that contains instructions needed to build the format in C.

   This is most easily done by editing the template file provided (which is written in C). For details, see An Example of an Export Program (p. 68).

2. Compile your C program.

   For details, see Compiling Code with the Mesh and Results Export API (p. 80).

3. Link the C program into the CFX code.

   For details, see Linking Code with the Mesh and Results Export API (p. 80).

4. Use the program.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For details, see Using a Customized Export Program.

Numerous keywords are required for development and use of custom export files. For details, see cfx5export Arguments.

An example source routine can be used as the basis of a customized program; one is given in the next section.

3.1.1. An Example of an Export Program

The following is an annotated listing of the C source code for a reasonably simple example of a customized Export program. The full source code is available for use as a template and is located in `CFX/examples/ExportTemplate.c`, where CFX is the directory in which CFX is installed.

Note:

The following documentation is representative only. For the latest version of the source code, refer to file `ExportTemplate.c`.

The example program is a reasonably simple example of an export program, which opens a CFX results file, writes a geometry file (ignoring pyramid elements) and several files containing results. After the program listing, a sample of the output produced is shown.

3.1.1.1. File Header

The file header uses several #include entries. The first set includes standard header files.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <io.h>
```

The second set includes cfx5export header files.

```
#include "cfxExport.h"
#include "getargs.h"
```

Obtaining CFX-Mesh and Results Export API header files is described in more detail. For details, see Linking Code with the Mesh and Results Export API (p. 80).

3.1.1.2. Allowed Arguments

The definition of allowed arguments appears as:

```
static char options[] = "u:d:t:cif";
```

The following piece of code simply defines the message that is printed if you enter incorrect options to the program.

```
static char *usgmsg[] = {
   "usage: ExportTemplate [options] res_file [basename]",
   " options are:",
   " -u<level>     = user level of interest",
   " -d<domain>    = domain of interest (default is 0 - all the domains",
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
   "                 are combined into a single domain)",
   " -t<timestep>  = timestep of interest (if set to -1, all timesteps",
   "                 are exported)"
   " -c            = use corrected boundary node data",
   " -i            = include boundary node only data",
   " -f            = get info on the res_file (No output is created)",
   " <basename> is the base filename for Template file output.",
   "If not specified, it defaults to 'res_file'. The Template",
   "geometry file will be written to <basename>.geom, the",
   "results file to <basename>.res, and the variables to",
   "<basename>.s## or <basename>.v## where ## is the variable",
   "number and s indicates a scalar and v a vector.",
    NULL
};
```

3.1.1.3. Main Program Initialization

As is standard, the variables `argc` and `argv` are the number of arguments and a pointer to the argument list. The variable `cfxCNT_SIZE` and the types `cfxNode` and `cfxElement` are defined in the header file `cfxExport.h` as are all variables and functions starting with the letters `cfx`. For details, see Mesh and Results Export API (p. 67). The variables `level`, `zone`, `alias`, `bndfix` and `bnddat` are used for setting the default values for the various parameters that can be set on the command line of the program.

```
void main (int argc, char *argv[])
{
   char *pptr;
   char baseFileName[256], fileName[256], errmsg[256];
   int i, n, counts[cfxCNT_SIZE], dim, length, namelen;
   int nnodes, nelems, nscalars, nvectors, nvalues;
   int level = 1, zone = 0, alias = 1, bndfix = 0, bnddat = 0;
   int timestep = -1, infoOnly = 0;
   int ts, t, t1, t2;
   int nTimeDig = 1;     /* number of digits in transient file suffix */
   char zoneExt[256];    /* zone extension added to the base filename */
   int isTimestep = 0;
   float timeVal = 0.0;  /* time value in the single timestep mode */
   char *wildcard = { "******" }; /* used in transient file specification */
   FILE *fp;
   cfxNode *nodes;
   cfxElement *elems;
   float *var;
```

The variable `cfxCNT_SIZE` and the types `cfxNode` and `cfxElement` are defined in the header file `cfxExport.h` as are all variables and functions starting with the letters `cfx`. For details, see Mesh and Results Export API (p. 67). The variables `level`, `zone`, `alias`, `bndfix` and `bnddat` are used for setting the default values for the various parameters that can be set on the command line of the program.

The following line prints an error message if there are not enough arguments to proceed.

```
if (argc < 2)
   cfxUsage (usgmsg, NULL);
```

The following piece of code reads the specified options and assigns values to certain variables accordingly. If an invalid or incomplete option is specified, then `getargs` prints an error message and the export program stops.

```
while ((n = getargs (argc, argv, options)) > 0) {
   switch (n) {
      case 'u':
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
         level = atoi (argarg);
         break;
      case ‘d’:
         zone = atoi (argarg);
         break;
      case ‘t’:
         timestep = atoi (argarg);
         isTimestep = 1;
         break;
      case ‘c’:
         bndfix = 1;
         break;
      case ‘i’:
         bnddat = 1;
         break;
      case ‘f’:
         infoOnly = 1;
         break;
   }
}
```

After this, the `level` variable contains the user level specified. All results are output if they are of this user level or below it. The `zone` variable contains the domain number that you specified. The variable `alias` determines whether the variables are referred to by their long names or short names. The default here is for short names to be used because some post-processors need variable names to contain no spaces, but you are encouraged to use long variable names wherever possible. The variable `bndfix` determines whether the variables are exported with corrected boundary node values - if `bndfix` is set to 1, then corrected values are used. Finally, `bnddat` determines whether variables that contain meaningful values only on the boundary (such as Yplus) are exported or not; if `bnddat` is set to 1, then these variables are exported.

3.1.1.4. Checking File Names

The following code checks to make sure that a CFX results file has been specified, and that it can be read by the export program. If this is not the case, the export program exits.

```
/* CFX-5 results file */
   if (argind >= argc)
       cfxUsage (usgmsg, "CFX-5 results file not specified");
   if (access (argv[argind], 0)) {
      fprintf (stderr, "result file <%s> does not exist\n", argv[argind]);
      exit (1);
   }
```

The following code writes the basename specified to the character array `baseFileName`. If one was not specified, then it defaults to the name of the results file specified. A basename name may be specified in another directory (for example, "../template/output"). However, later in the code this basename without the preceding directory information is required (in this example "output"); and so the pointer `pptr` is assigned to point to the first character of this name.

```
/* base file name */
   if (argind + 1 < argc)
      strcpy (baseFileName, argv[argind+1]);
   else
      strcpy (baseFileName, argv[argind]);
if (NULL != (pptr = strrchr (baseFileName, ‘/’)))
      pptr++;
   else if (NULL != (pptr = strrchr (baseFileName, ‘\\’)))
      pptr++;
   else
      pptr = baseFileName;
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following code checks that the results file that will be produced by the export program will not overwrite an existing results file.

```
/* don't overwrite results file */
sprintf (fileName, "%s.res", baseFileName);
if (0 == strcmp (argv[argind], fileName)) {
fprintf (stderr, "Template res file would overwrite CFX results file\n");
fprintf (stderr, "Need to select new Template output base file name\n");
exit (1);
   }
```

3.1.1.5. Opening the CFX Results File

The following code prints a message to the screen telling you that the program is reading the results file. It then calls `cfxExportInit`, which must always be called before any of the other export routines. The variable `n` is set to equal the number of zones in the results file. If the `-f` option has been selected, information about the results file will be displayed. The number of domains to be exported is also determined so that the format of the exported file includes the appropriate suffix. Finally, a check is made to make sure that the zone (if any) that you specified in the program options is a valid zone for this results file.

```
/* open CFX-5 results file */
printf ("\nreading CFX results from <%s>\n", argv[argind]);
   n = cfxExportInit (argv[argind], NULL);
   if (infoOnly) {
      int nt;
      printf("\n%d domains:\n", n);
      for(i = 1; i <= n; i++)
         printf("  %d   %s\n", i, cfxExportZoneName(i));
      nt = cfxExportTimestepCount();
      printf("%d timesteps:\n", nt);
      if(nt) {
         for(i = 1; i <= nt; i++)
            printf("  %d\n", cfxExportTimestepNumGet(i));
      }
      cfxExportDone();
      exit (0);
   }
   /* determine the zone suffix for the export files */
   strcpy(zoneExt, "");
   if(zone == 0) {
      printf ("processing all domains\n");
      cfxExportSetVarParams(bndfix, level);
   }
   else {
      printf ("processing domain %d\n", zone);
if(n != 1) {
         float f;
         int nZoneDig = 0;
/* count number of digits needed to fit any zone number */
         f = (float) n;
         while((f /= 10) >= 1) nZoneDig++;
         sprintf(zoneExt, "_d%*.*d", nZoneDig, nZoneDig, zone);
      }
   }
if (cfxExportZoneSet (zone, counts) < 0)
         cfxExportFatal ("invalid zone number");
```

The following code is ignoring any pyramid elements (elements with 5 nodes) and decreases `nelems` by the number of pyramid elements. It then checks to make sure that neither the number of nodes nor the number of elements is zero; if so, the program exits with return code -1.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The first two lines focus on the number of nodes in the zone and the number of elements in the zone.

```
nnodes = cfxExportNodeCount();
nelems = cfxExportElementCount();
if (counts[cfxCNT_PYR]) {
   printf ("%d pyramid elements found - they are being ignored\n",
      counts[cfxCNT_PYR]);
   nelems -= counts[cfxCNT_PYR];
}
if (!nnodes || !nelems)
   cfxExportFatal ("no nodes and/or elements");
```

3.1.1.6. Timestep Setup

The following code determines whether all of the timesteps, a specific timestep or the final timestep (steady-state) have been selected for export.

```
if(isTimestep && timestep == -1 && !cfxExportTimestepCount()) {
      isTimestep = 0;
   }
   if(isTimestep) {
      int i;
      float f;
      if(timestep == -1) {
         printf("processing all timesteps\n");
         t1 = 1;
         t2 = cfxExportTimestepCount() + 1;
      }
      else {
         int isFound = 0;
         printf("processing timestep %d\n", timestep);
         for(i = 1; i <= cfxExportTimestepCount() + 1; i++)
            if(cfxExportTimestepNumGet(i) == timestep) {
               timeVal = cfxExportTimestepTimeGet(i);
               t1 = t2 = i;
               isFound = 1;
               break;
            }
         if(!isFound) {
            sprintf(errmsg, "\nTimestep %d not found. "
               "Use -f to see the list of valid timesteps.\n", timestep);
            cfxExportFatal (errmsg);
         }
      }
      /* count number of digits needed to fit any timestep number */
      f = (float) cfxExportTimestepCount();
      while((f /= 10) >= 1) nTimeDig++;
   }
   else {
      timeVal = cfxExportTimestepTimeGet(cfxExportTimestepCount() + 1);
      timestep = cfxExportTimestepNumGet(cfxExportTimestepCount() + 1);
      t1 = t2 = cfxExportTimestepCount() + 1;
   }
```

3.1.1.7. Geometry File Output

The following code opens the geometry file `basename.geom`, printing an error if it cannot be opened for any reason. A message is then displayed informing you that the application is writing the geometry file.

```
/* Template geometry output */
   sprintf (fileName, "%s.geom", baseFileName);
   if (NULL == (fp = fopen (fileName, "w+"))) {
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
      sprintf (errmsg, "can't open <%s> for output", fileName);
      cfxExportFatal (errmsg);
   }
   printf ("writing Template Geometry file to <%s>\n", fileName);
```

The header of this file is shown after the program listing.

```
/* write header */
   fprintf( fp, "Template Geometry file exported from CFX\n");
   fprintf( fp, " \n");
   fprintf( fp, "node id given\n");
   fprintf( fp, "element id off\n");
```

The following code writes first the word "coordinates" and the number of nodes that will be written. The pointer nodes is initialized to point at the data for the first node and the node data is written into the geometry file. For each node, a node number is written, followed by the three coordinates of that node. Note that `n` ranges between 0 and `nnodes-1`. This program adds 1 to each node number so that the nodes in the geometry file are numbered between 1 and `nnodes`. When it has finished, the `cfxExportNodeFree` routine frees the memory that was used to store the node data, and finally the word "done" is printed on the screen to alert you that it has finished writing the node data.

```
/* write nodes */
   fprintf( fp, "coordinates\n");
   fprintf( fp, "%8d\n", nnodes );
   nodes = cfxExportNodeList();
   printf ("  writing %d nodes ...", nnodes);
   fflush (stdout);
   for (n = 0; n < nnodes; n++, nodes++) {
fprintf( fp, "%8d %12.5e %12.5e %12.5e\n", n + 1, nodes->x,
   nodes->y, nodes->z );
}
   cfxExportNodeFree();
printf (" done\n");
```

Next, the data for each element must be written.

Firstly, some general information is written. Then the data for each element type is written in turn.

```
/* write elements */
   fprintf( fp, "part 1\n" );
   fprintf( fp, "volume elements\n");
   printf ("  writing %d elements...", nelems);
   fflush (stdout);
```

For tetrahedral elements, the word "tetra4" is written to the file, followed by the number of tetrahedral elements written.

```
/* tets */
   fprintf( fp, "tetra4\n");
   fprintf( fp, "%8d\n", counts[cfxCNT_TET] );
```

The following code is executed only if the number of tetrahedral elements is non-zero. Assuming this, `elems` is set to point to the list of elements stored in the results file. The index `n` loops over all the elements. For each element, the following step is carried out: "If the element is a tetrahedron, then loop over its four vertices and write their node numbers to the geometry file, then start a new line (ready for the next set of data)." The output produced can be seen in the examples of the exported files in the next section.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
if (counts[cfxCNT_TET]) {
   elems = cfxExportElementList();
   for (n = 0; n < nelems; n++, elems++) {
      if (cfxELEM_TET == elems->type) {
         for (i = 0; i < elems->type; i++)
            fprintf (fp, "%8d", elems->nodeid[i]);
         putc ('\n', fp);
      }
   }
}
```

For wedges (triangular prisms) and hexahedral elements, the same procedure is followed. However, there is a slight difference in the way that the `fprintf` line is written for hexahedral elements. This is because the order that the element nodes are written to the geometry file is different to the order in which they were read from the results file. This may need to be done if a post-processor has a different convention for node order than the one that the cfx5export node routines have. The order the nodes are written in will affect which node is connected to which. The node ordering for exported elements is illustrated in cfxExportElementList (p. 89).

```
/* wedges */
   fprintf( fp, "penta6\n");
   fprintf( fp, "%8d\n", counts[cfxCNT_WDG] );
   if (counts[cfxCNT_WDG]) {
      elems = cfxExportElementList();
      for (n = 0; n < nelems; n++, elems++) {
         if (cfxELEM_WDG == elems->type) {
            for (i = 0; i < elems->type; i++)
               fprintf (fp, "%8d", elems->nodeid[i]);
         }
         putc ('\n', fp);
      }
   }
/* hexes */
   fprintf( fp, "hexa8\n");
   fprintf( fp, "%8d\n", counts[cfxCNT_HEX] );
   if (counts[cfxCNT_HEX]) {
      elems = cfxExportElementList();
      for (n = 0; n < nelems; n++, elems++) {
         if (cfxELEM_HEX == elems->type)
            fprintf (fp, "%8d%8d%8d%8d%8d%8d%8d%8d\n",
               elems->nodeid[0], elems->nodeid[1],
               elems->nodeid[3], elems->nodeid[2],
               elems->nodeid[4], elems->nodeid[5],
               elems->nodeid[7], elems->nodeid[6]);
      }
   }
```

Then the geometry file is closed and the memory occupied by the element data is freed.

```
printf (" done\n");
   fclose (fp);
   cfxExportElementFree();
```

3.1.1.8. Template Results File

Despite its name, the Template results file does not contain any actual values of results. It simply contains information about how many variables there are and in which file each is stored.

The first job is to make sure that there are some results for export. First, the code checks that there is a nonzero number of variables that have the specified user level. Then it counts the number of scalar and vector variables that will be exported. To be exported, a variable must:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1. Have a dimension of 1 (scalar variable) or 3 (vector variable) and,

2. Either be a variable with useful values everywhere in the zone or be a variable that has values only on the boundaries (in which case it will be exported only if you asked to "include boundary node only data" by specifying the option `-i` when starting the export program, which translated to setting `bnddat = 1` when the arguments were processed).

Review the `cfxExportVariableSize` routine if this logic is unclear. For details, see cfxExportVariableSize (p. 96).

Once results are identified, the code calculates the variable `namelen`, which is the length of the longest variable name to be exported (the `alias` variable was set when processing the arguments passed to the export program, and depends upon whether you wanted to use long names or short names). If there are no vector or scalar variables to be exported, the export program exits.

```
/* output results file */
   nscalars = nvectors = namelen = 0;
   if ((nvalues = cfxExportVariableCount(level)) > 0) {
      for (n = 1; n <= nvalues; n++) {
         cfxExportVariableSize (n, &dim, &length, &i);
         if ((1 != dim && 3 != dim) ||
            (length != nnodes && length != bnddat))
            continue;
         if (1 == dim)
            nscalars++;
         else
            nvectors++;
         i = strlen (cfxExportVariableName (n, alias));
         if (namelen < i)
            namelen = i;
      }
   }
   if (0 == (nscalars + nvectors)) {
      cfxExportDone ();
      exit (0);
   }
```

The following code checks that the results file can be opened for writing to, and exits if not. The number of scalar and vector variables are written to the file, followed by some numbers (which EnSight, for example, requires) that are always the same for any export of this kind.

```
sprintf (fileName, "%s.res", baseFileName);
   if (NULL == (fp = fopen (fileName, "w+"))) {
      sprintf (errmsg, "can't open <%s> for writing", fileName);
      cfxExportFatal (errmsg);
   }
   printf ("writing Template results file to <%s>\n", fileName);
   fflush (stdout);
   fprintf( fp, "%d %d 0\n", nscalars, nvectors );
   fprintf( fp, "%d\n", t2 - t1 + 1 );
   for(i = t1; i <= t2; i++) {
      fprintf( fp, "%13.4e", cfxExportTimestepTimeGet(i));
      if(!(i % 6)) fprintf( fp, "\n");
   }
   fprintf( fp, "\n");
   if(isTimestep && t1 != t2)
      fprintf( fp, "0 1\n");
```

Next, for each scalar variable, a line is written that contains the filename where the scalar will be written, and then the name of the variable. Note that the filename is not the basename, but the basename with all the directory structure (if any) stripped off the front. For details, see Checking

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

File Names (p. 70). This is done because these file will be written in the same directory as this Template results file, so there is no need for directory information.

```
if ( nscalars ) {
   for (n = 1; n <= nvalues; n++) {
      cfxExportVariableSize (n, &dim, &length, &i);
      if (1 == dim && (length == nnodes || length == bnddat))
         if(!isTimestep)
            fprintf (fp, "%s%s.s%2.2d %s\n", pptr, zoneExt,
               n, cfxExportVariableName(n, alias));
         else if(t1 == t2)
            fprintf (fp, "%s%s_t%d.s%2.2d %s\n", pptr, zoneExt,
               cfxExportTimestepNumGet(t1), n,
               cfxExportVariableName(n, alias));
         else
            fprintf (fp, "%s%s_t%*.*s.s%2.2d %s\n", pptr, zoneExt,
               nTimeDig, nTimeDig, wildcard, n,
               cfxExportVariableName(n, alias));
      }
   }
```

The same information is then written for each vector variable and the Template results file is closed.

```
if ( nvectors ) {
   for (n = 1; n <= nvalues; n++) {
      cfxExportVariableSize (n, &dim, &length, &i);
      if (3 == dim && (length == nnodes || length == bnddat))
         if(!isTimestep)
            fprintf (fp, "%s%s.v%2.2d %s\n", pptr, zoneExt,
               n, cfxExportVariableName(n, alias));
         else if(t1 == t2)
            fprintf (fp, "%s%s_t%d.v%2.2d %s\n", pptr, zoneExt,
               cfxExportTimestepNumGet(t1), n,
               cfxExportVariableName(n, alias));
         else
            fprintf (fp, "%s%s_t%*.*s.v%2.2d %s\n", pptr, zoneExt,
               nTimeDig, nTimeDig, wildcard, n,
               cfxExportVariableName(n, alias));
      }
   }
fclose( fp );
```

3.1.1.9. Creating Files with Results for Each Variable

The results for each variable are written to separate files, called <*basename*>.s01, <*basename*>.s02, <*basename*>.v03, for example. Each file with an extension containing a letter "s" contains a scalar variable, and each with a "v" contains a vector variable. Which variable is written to each file is tabulated in the Template results file that has just been written.

The following code reads the information for each variable, after you decide that it should be exported - the logic is very similar to that used when counting the relevant variables when creating the Template results file. The marked `if` loop executes if the variable needs to be exported. It checks to make sure that the variable information can be read, and (assuming it can) then builds the filename and checks to see if it can be opened. Continuing, it writes to the screen where it is putting the variable, and then loops through all the values, writing them to the file, inserting a new line every six values. After each variable, the memory used to store that variable is restored.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

After all the variable files have been written, the program calls the `cfxExportDone` routine, which close the CFX results file, and frees up any remaining memory. This routine must be the last call to any of the API routines. The program then exits.

Note:

This program makes no use of any of the region routines, which enable access to boundary condition data, nor the volume routines that enable access to the subdomains that are defined for a problem.

- Region Routines (p. 90)
- Volume Routines (p. 93)

```
/* output each timestep to a different file */
   for(t = t1; t <= t2; t++) {
      ts = cfxExportTimestepNumGet(t);
      if(cfxExportTimestepSet(ts) < 0) {
         continue;
      }
   /* build file name and open file */
   if(!isTimestep)
      sprintf( fileName, "%s%s.%c%2.2d", baseFileName, zoneExt,
         1 == dim ? 's' : 'v', n);
      else if(t1 == t2)
         sprintf( fileName, "%s%s_t%d.%c%2.2d", baseFileName, zoneExt,
            ts, 1 == dim ? 's' : 'v', n);
      else
         sprintf( fileName, "%s%s_t%*.*d.%c%2.2d", baseFileName, zoneExt,
            nTimeDig, nTimeDig, t-1, 1 == dim ? 's' : 'v', n);
      if (NULL == (fp = fopen (fileName, "w+"))) {
         sprintf (errmsg, "can't open <%s> for writing\n", fileName);
         cfxExportFatal (errmsg);
      }
      printf ("  %-*s -> %s ...", namelen,
         cfxExportVariableName(n, alias), fileName);
      fflush (stdout);
      fprintf( fp, "%s\n", cfxExportVariableName(n, alias));
      length = nnodes * dim;
      for ( i = 0; i < length; i++, var++ ) {
         fprintf( fp, "%12.5e ", *var );
         if ( i && 5 == (i % 6) )
            putc ('\n', fp);
      }
      if ( 0 != ( nvalues % 6 ) )
         putc( '\n', fp );
      fclose( fp );
      cfxExportVariableFree (n);
      printf (" done\n");
      }
   }
   } /* loop for each timestep */
      cfxExportDone();
      exit (0);
}
```

3.1.2. Example of Output Produced

If the export program is correctly compiled and run, the following output is obtained. For details, see Using a Customized Export Program.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In this example, the CFX results file contains three variables at user level 1: pressure, temperature and velocity. This is in a file named file.res. No timesteps or domains were specified, and the basename was specified as an example.

The following is displayed on screen:

```
reading CFX results from <file.res>
processing all domains
writing Template Geometry file to <example.geom>
   writing 2365 nodes ... done
   writing 11435 elements... done
writing Template results file to <example.res>
writing variable output files
   Pressure    -> example.s01 ... done
   Temperature -> example.s02 ... done
   Velocity    -> example.v03 ... done
```

Five files are produced: the geometry file `example.geom`, the Template results file `example.res`, and three variable files called `example.s01`, `example.s02` and `example.v03`, which contain the results for pressure, temperature and velocity, respectively. For details, see:

- example.geom (p. 78)
- example.res (p. 79)
- example.s01 (p. 79)

3.1.2.1. example.geom

The content of this file appears as:

```
Template Geometry file exported from CFX
node id given
element id off
coordinates
   2365
      1 2.00000e+00 0.00000e+00 0.00000e+00
      2-2.00000e+00-6.51683e-07 0.00000e+00
      3 2.00000e+00 0.00000e+00 2.00000e+00
      4-2.00000e+00-6.51683e-07 2.00000e+00
      5 3.00000e+00 1.00000e+00 5.00000e-01
      ....
      ....
      ....
      2362-1.13337e+00 2.18877e-01 4.02491e-01
      2363-1.12115e+00-3.66598e-01 2.22610e-01
      2364 1.36924e+00 4.78359e-01 1.22588e-01
      2365-3.30703e-01 1.38487e+00 2.23515e+00
part 1
volume elements
tetra4
   11435
      754     230      12     145
      755     216       8     122
      756     212     125     215
      ....
      ....
      ....
      2365   496      475     474
penta6
   0
hexa8
   0
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.1.2.2. example.res

The content of this file appears as:

```
2 1 0
1
0.0
0 1
example.s01 Pressure
example.s02 Temperature
example.v03 Velocity
```

3.1.2.3. example.s01

The content of this file appears as:

```
Pressure
   1.42748e+04 1.42621e+04 1.43425e+04 1.43350e+04 1.44118e+04 1.44777e+04
   1.38639e+04 1.37352e+04 1.44130e+04 1.44755e+04 1.37733e+04 1.37626e+04
   ....
   ....
   ....
   1.39092e+04 1.40699e+04 1.24139e+04 1.34786e+04 1.34859e+04 1.37959e+04
```

3.1.3. Source Code for getargs.c

The following code is the C code that defines the functions `cfxUsage` and `getargs`, both of which are called by the example listing above. You do not need to include this code with your custom export program (it is automatically linked in if you use the compiler as described in the next section).

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include "getargs.h"
/*---------- usage ---------------------------------------------------
 * display usage message and exit
 *--------------------------------------------------------------------*/
void cfxUsage (
#ifdef PROTOTYPE
   char **usgmsg, char *errmsg)
#else
   usgmsg, errmsg)
char **usgmsg, *errmsg;
#endif
{
   int n;
   if (NULL != errmsg)
      fprintf (stderr, "ERROR: %s\n", errmsg);
   for (n = 0; NULL != usgmsg[n]; n++)
      fprintf (stderr, "%s\n", usgmsg[n]);
   exit (NULL != errmsg);
}
/*---------- getargs -------------------------------------------------
 * get option letter from argument vector or terminates on error
 * this is similar to getopt()
 *--------------------------------------------------------------------*/
int argind = 0;  /* index into argv array */
char *argarg;    /* pointer to argument string */
int getargs (
#ifdef PROTOTYPE
   int argc, char **argv, char *ostr)
#else
   argc, argv, ostr)
int argc;
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
char **argv, *ostr;
#endif
{
   int argopt;
   char *oli;
   static char *place;
   static int nextarg;
   /* initialisation */
   if (!argind)
      nextarg = 1;
   if (nextarg) {  /* update scanning pointer */
      nextarg = 0;
      /* end of arguments */
      if (++argind >= argc || '-' != argv[argind][0])
         return (0);
      place = argarg = &argv[argind][1];
   }
   /* check for valid option */
   if ((argopt = *place++) == ':' ||
      (oli = strchr (ostr, argopt)) == NULL) {
      fprintf (stderr, "invalid command line option `%c'\n", argopt);
      exit (1);
   }
   /* check for an argument */
   if (*++oli != ':') {        /* don't need argument */
      argarg = NULL;
      if (!*place)
         nextarg = 1;
   }
   else {                      /* need an argument */
      if (!*place) {
         if (++argind >= argc) {
            fprintf (stderr, "missing argument for option `%c'\n", argopt);
            exit (1);
         }
         place = argv[argind];
      }
      argarg = place;
      nextarg = 1;
   }
   return (argopt);    /* return option letter */
}
```

3.2. Compiling Code with the Mesh and Results Export API

Compilation of a customized executable must be performed using an appropriate compiler and compiler flags. The supported compilers are listed in Compiler Requirements for All Windows Versions in the *Ansys, Inc. Installation Guides* and Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

The customized executable must be linked with the provided Mesh and Results Export API library and the provided I/O library as detailed in Linking Code with the Mesh and Results Export API (p. 80).

3.3. Linking Code with the Mesh and Results Export API

In order to build a customized export utility, it must be linked with the applicable libraries listed below.

Libraries located in <*CFXROOT*>`/lib/`<*os*>`/`:

- `libmeshexport.lib` (on Windows), or `libmeshexport.a` (on Linux)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- `libratlas_api.lib` (on Windows), or `libratlas_api.a` (on Linux)
- `libratlas.lib` (on Windows), or `libratlas.a` (on Linux)
- `libpgtapi.lib` (on Windows), or `libpgtapi.a` (on Linux)
- `libunits.lib` (on Windows), or `libunits.a` (on Linux)
- `libcclapilt.lib` (on Windows), or `libcclapilt.a` (on Linux)
- `libio.lib` (on Windows), or `libio.a` (on Linux)

Libraries located in *<AWPROOT>*`/tp/zlib/1.2.13/` (where *<AWPROOT>* is the directory one level higher than the directory in which CFX is installed, *<CFXROOT>*, and where the version-specific directory under `zlib` is subject to change):

- `zlib1.lib` (on Windows), or `libz.so` (on Linux)

3.3.1. Linking a Customized Mesh and Results Export Executable on a Windows Platform

The following topics are discussed:

3.3.1.1. Linking a Customized C Mesh and Results Export Executable on a Windows Platform
3.3.1.2. Linking a Customized Fortran Mesh and Results Export Executable on a Windows Platform

3.3.1.1. Linking a Customized C Mesh and Results Export Executable on a Windows Platform

You can build the executables on Windows systems. An example command line follows:

```
cl .\ExportTemplate.c /FeExportTemplate.exe /MD -openmp /I "C:\Program
Files\Ansys Inc\v251\CFX\include" /link "/libpath:C:\Program
Files\Ansys Inc\v251\CFX\lib\winnt-amd64" libmeshexport.lib
libpgtapi.lib libcclapilt.lib libunits.lib libio.lib libratlas.lib
"/libpath:C:\Program Files\Ansys Inc\v251\tp\zlib\1.2.13\winx64"
zlib1.lib
```

Note:

The version-specific directory under `zlib` is subject to change.

3.3.1.2. Linking a Customized Fortran Mesh and Results Export Executable on a Windows Platform

You can write the export program in Fortran and then compile and link it. An example command line follows:

```
ifort /MD /I "C:\Program Files\Ansys Inc\v251\CFX\include" /threads
/iface:mixed_str_len_arg ExportTemplate.F /exe:ExportTemplate.exe
/libs:dll /link "/libpath:C:\Program Files\Ansys
Inc\v251\CFX\lib\winnt-amd64" libcclapilt.lib libio.lib
libmeshexport.lib libunits.lib libpgtapi.lib libratlas_api.lib
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
libratlas.lib "/libpath:C:\Program Files\Ansys
Inc\v251\tp\zlib\1.2.13\winx64" zlib1.lib
```

Note:

The version-specific directory under `zlib` is subject to change.

The supported compilers are listed in Compiler Requirements for All Windows Versions in the *Ansys, Inc. Installation Guides*.

3.3.2. Linking a Customized Mesh and Results Export Executable on a Linux Platform

The following topics are discussed:

3.3.2.1. Linking a Customized C Mesh and Results Export Executable on a Linux Platform

3.3.2.2. Linking a Customized Fortran Mesh and Results Export Executable on a Linux Platform

3.3.2.1. Linking a Customized C Mesh and Results Export Executable on a Linux Platform

On most Linux systems you should be able to build the executable with the command:

```
gcc -g export.c -o export.exe -fopenmp -I<CFXROOT>/include/ -m64 \
-L<CFXROOT>/lib/linux-amd64 -lmeshexport -lratlas_api -lratlas -lpgtapi -lunits \
-lcclapilt -lio -L<AWPROOT>/tp/zlib/1.2.13/linx64/lib -lz -lm -lc
```

where *<CFXROOT>* is the directory in which CFX is installed, *<AWPROOT>* is the directory one level higher, and the version-specific directory under `zlib` is subject to change.

Here, `-lmeshexport`, `-lratlas_api`, `-lratlas`, `-lpgtapi`, `-lunits`, `-lcclapilt`, and `-lio` indicate the libraries mentioned above, while `-lm`, `-lc` and `-fopenmp` are system libraries.

In this example, your own export program is named `export.c` and the executable file will be called `export.exe`. You should ensure that the libraries to which you are linking (which are in the path given after `-L`) appear on the command line after the source file (or object file if you are just linking to an existing object).

The compiler flags and required libraries may vary, depending on the compiler and the custom program.

The supported compilers are listed in Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

3.3.2.2. Linking a Customized Fortran Mesh and Results Export Executable on a Linux Platform

The following is an example of how to build the executable on Linux, when the source code for the executable is written in Fortran:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
ifort myexport.F -L<CFXROOT>/lib/linux-amd64 -lmeshexport -lratlas_api -lratlas \
  -lpgtapi -lunits -lcclapilt -lio -qopenmp -o myexport.exe \
  -L<AWPROOT>/tp/zlib/1.2.13/linx64/lib -lz -lm
```

where <*CFXROOT*> is the directory in which CFX is installed, <*AWPROOT*> is the directory one level higher, and the version-specific directory under `zlib` is subject to change.

The supported compilers are listed in Compiler Requirements for All Linux Versions in the *Ansys, Inc. Installation Guides*.

3.4. Details of the Mesh Export API

The full list of constants, data structures, types and functions available to the programmer are given in the following sections:

3.4.1. Defined Constants and Structures
3.4.2. Initialization and Error Routines
3.4.3. Zone Routines
3.4.4. Node Routines
3.4.5. Element Routines
3.4.6. Region Routines
3.4.7. Face Routines
3.4.8. Volume Routines
3.4.9. Boundary Condition Routines
3.4.10. Variable Routines
3.4.11. Timestep Routines

3.4.1. Defined Constants and Structures

The following constants and data structures are defined in the header file `cfxExport.h`, which should be included in the export program.

3.4.1.1. Element Types

CFX can use 4 types of element, which are identified by the number of nodes: tetrahedrons (4 nodes), pyramids (5 nodes), prisms/wedges (6 nodes), and hexahedrons (8 nodes). The element types are identified in the Export API by the following constants:

```
#define cfxELEM_TET 4
#define cfxELEM_PYR 5
#define cfxELEM_WDG 6
#define cfxELEM_HEX 8
```

3.4.1.2. Volume List Types

The Export API contains functions that enable you to query how volumes are defined in the results file. It is possible to request how a volume is defined in terms of nodes or elements. (For details, see Volume Routines (p. 93).)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following constants are defined in the header file and should be used as arguments to the Volume routines:

```
#define cfxVOL_NODES 0
#define cfxVOL_ELEMS 1
```

3.4.1.3. Region List Types

The Export API contains functions that enable you to query how regions are defined in the results file. It is possible to request how a region is defined in terms of nodes or faces. (For details, see Region Routines (p. 90).)

The following constants are defined in the header file and should be used as arguments to the Region routines:

```
#define cfxREG_NODES 0
#define cfxREG_FACES 1
```

In the case of nodes, the global node number is returned, while in the case of faces, the returned value is a combination of the global element number and local face number of the element. The following macros are available to enable you to extract the element and face number from the combined value:

```
#define cfxFACENUM(face) ((face) & 7)
#define cfxELEMNUM(face) ((face) >> 3)
```

3.4.1.4. Count Entries

Two routines exist for initializing the Export API (see cfxExportInit (p. 85)) and requesting the totals of certain quantities in a zone (see cfxExportZoneSet (p. 86)). The array returned from both of these routines requires the following constants to be used by the calling program to reference the correct quantities.

```
enum cfxCounts {
cfxCNT_NODE = 0, /* number of nodes */
cfxCNT_ELEMENT, /* number of elements */
cfxCNT_VOLUME,   /* number of volumes */
cfxCNT_REGION, /* number of regions */
cfxCNT_VARIABLE, /* number of variables */
cfxCNT_TET, /* number of tetrahedral elements */
cfxCNT_PYR, /* number of pyramid elements */
cfxCNT_WDG, /* number of wedge elements */
cfxCNT_HEX, /* number of hexahedral elements */
cfxCNT_SIZE /* size of count array */
};
```

3.4.1.5. Node Data Structure

Nodes are represented in the Export API using the following structure (note the change in data type of x, y and z):

```
typedef struct cfxNode {
float x, y, z;
} cfxNode;
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where x, y, and z are the coordinates of the node. A pointer to an array of these structures is returned by cfxExportNodeList. For details, see cfxExportNodeList (p. 88).

3.4.1.6. Element Data Structure

Elements are represented by the Export API using the following structure:

```
typedef struct cfxElement {
int type;
int *nodeid;
} cfxElement;
```

where type is the element type and nodeid is an array of node numbers that define the topology of the element. A pointer to an array of these structures is returned by cfxExportElementList. For details, see Element Types (p. 83) and cfxExportElementList (p. 89).

3.4.2. Initialization and Error Routines

The following routines open and close the CFX results file, initialize the Export API, and handle fatal error processing. The first call to any of the API routines must be cfxExportInit and the last call should be cfxExportDone. For details, see:

- cfxExportInit (p. 85)
- cfxExportDone (p. 85).

3.4.2.1. cfxExportInit

```
int cfxExportInit (char *resfile, int counts[cfxCNT_SIZE])
```

Opens the CFX results file named resfile and initializes the Export API. This should be the first call made to the API.

The routine returns the total number of zones. If the array counts is supplied to the routine (that is, it is not NULL), the array is filled with values representing the total number of nodes, elements, volumes, regions and variables for all the zones are returned in this array.

3.4.2.2. cfxExportDone

```
void cfxExportDone ()
```

Closes the CFX results file and destroys any internal storage used by the API. This should be the final call made to the Export API.

3.4.2.3. cfxExportError

```
void cfxExportError (void (*callback) (char *errmsg))
```

Specify a callback function that will be executed when a fatal error is generated by a call to cfxImportFatal (see cfxImportFatal (p. 47)). The argument, callback, is the function that will be called, it should take an argument that is the error message passed to cfxImportFatal. It is the responsibility of the function to terminate the application if required.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.2.4. cfxExportFatal

```
void cfxExportFatal (char *errmsg)
```

Generate a fatal error message (`errmsg`) and close the Ansys CFX results file. This routine also calls a callback function, if one has been specified by `cfxExportError` (see cfxExportError (p. 85)). If no callback function has been specified the function also terminates the application. There is no return from this call.

3.4.3. Zone Routines

A zone is defined as groups of nodes or faces that are located on the external boundaries of the domain. The following routines provide functionality for returning the number of zones in the open CFX results file specifying and requesting the current zone, and destroying any internal storage associated with a zone. All other routines in the Export API refer to quantities in the current zone being accessed by the API. By default the current zone is the global zone (a combination of all zones in the Ansys CFX results file), but this can be the current zone can be altered by making a call to `cfxExportZoneSet` (see cfxExportZoneSet (p. 86)). Once this call has been made, any other function returns information about this zone until a subsequent call is made.

3.4.3.1. cfxExportZoneCount

```
int cfxExportZoneCount ()
```

Return the number of zones in the CFX results file.

3.4.3.2. cfxExportZoneSet

```
int cfxExportZoneSet (int zone, int counts[cfxCNT_SIZE])
```

Set the current zone being accessed by the Export API.

The value of zone should be between 1 and the value returned by `cfxExportZoneCount` (see cfxExportZoneCount (p. 86)) or 0 if the global zone is to be accessed.

The function returns 0 if the value of zone is invalid or the value zone if setting of the zone was successful.

The argument counts can be passed as a NULL pointer. In this case no information is returned to the calling function other than the return value mentioned above. If counts is specified it must be at least cfxCNT_SIZE in size, not specifying an array large enough can result in errors. In the case when counts is supplied correctly the total number of nodes, elements, volumes, regions and variables will be returned.

3.4.3.3. cfxExportZoneGet

```
int cfxExportZoneGet ()
```

Returns the current zone number.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.3.4. cfxExportZoneFree

```
void cfxExportZoneFree ()
```

While a zone is being accessed, internal storage is allocated, this storage should be deallocated when no longer required. This can be done by calling `cfxExportZoneFree` or by calling `cfxExportNodeFree`, `cfxExportElementFree`, `cfxExportVolumeFree`, `cfxExportRegionFree` and `cfxExportVariableFree`. Details on each of these routines is available; see:

- cfxExportNodeFree (p. 88)
- cfxExportElementFree (p. 90)
- cfxExportVolumeFree (p. 94)
- cfxExportRegionFree (p. 91)
- cfxExportVariableFree (p. 97).

3.4.3.5. cfxExportZoneIsRotating

```
int cfxExportZoneIsRotating(double rotationAxis[2][3], double *angularVelocity)
```

Query whether the current zone is rotating and describe axis and angular velocity of the rotation if applicable. Returns 1 if the current zone is rotating and 0 if it is not; for the combined zone the return value is always -1. If successful the rotation axis is returned in `rotationAxis` and the velocity in `angularVelocity` in radians/second.

3.4.3.6. cfxExportZoneMotionAction

```
int cfxExportZoneMotionAction(const int zone, const int flag)
```

Specify whether grid coordinates and variables should have the appropriate rotation applied to them if the zone is rotating so that grid coordinates appear in their correct locations and velocities (for examples) take this rotation into consideration. If `cfxExportZoneList` and `cfxExportVariableList` should return rotated values, flag should be set to `cfxMOTION_USE`. The default behavior for a particular zone will be used if `cfxMOTION_IGNORE` is specified or this function is not called. If zone is not valid or flag is not `cfxMOTION_USE`, `cfxMOTION_IGNORE` the return value will be -1 otherwise 0 is returned.

3.4.4. Node Routines

Accessing nodes within the current zone (see cfxExportZoneSet (p. 86)) is performed by making calls to the following functions.

It should be noted that the nodes for a zone are not loaded into the Export API until either `cfxExportNodeList` (see cfxExportNodeList (p. 88)) or `cfxExportNodeGet` (see cfxExportNodeGet (p. 88)) are called. This reduces memory overheads in the API by not allocating space until required.

When access to nodes in the current zone is no longer required, a call to `cfxExportNodeFree` (see cfxExportNodeFree (p. 88)) should be made to deallocate any internal storage.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.4.1. cfxExportNodeCount

```
int cfxExportNodeCount ()
```

Query the number of nodes defined in the current zone.

3.4.4.2. cfxExportNodeList

```
cfxNode *cfxExportNodeList ()
```

Return a pointer to an array of cfxNode elements (see cfxnode (p. 55)) containing the coordinate values of each node in the current zone. The first node in the zone is the first element of the array, the second is the second and so on.

The memory allocated to represent this information should be deallocated using `cfxExportNodeFree` (see cfxExportNodeFree (p. 88)) when no longer required.

3.4.4.3. cfxExportNodeGet

```
int cfxExportNodeGet (int nodeid, double *x, double *y, double *z)
```

Query the coordinates of a specific node in the current zone.

The index (nodeid) is specified between 1 and the number of nodes returned by `cfxExportNodeCount` (see cfxExportNodeCount (p. 88)). If the value of nodeid is out of range the return value is 0 otherwise it is nodeid.

3.4.4.4. cfxExportNodeFree

```
void cfxExportNodeFree ()
```

Deallocate any internal storage allocated by the Export API after calls to `cfxExportNodeList` (see cfxExportNodeList (p. 88)) and `cfxExportNodeGet` (see cfxExportNodeGet (p. 88)) have been made in the current zone.

3.4.4.5. cfxExportNodeUnits

```
const char* cfxExportNodeUnits()
```

Return a character string that represents the units associated with the mesh, for example, `m`.

3.4.5. Element Routines

Accessing elements within the current zone (see cfxExportZoneSet (p. 86)) is performed by making calls to the following functions. It should be noted that the elements for a zone are not loaded into the Export API until either `cfxExportElementList` (see cfxExportElementList (p. 89)) or `cfxExportElementGet` (see cfxExportElementGet (p. 89)) are called. This reduces memory overheads in the API by not allocating space until required.

When access to elements in the current zone is no longer required a call to `cfxExportElementFree` (see cfxExportElementFree (p. 90)) should be made to deallocate any internal storage.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.5.1. cfxExportElementCount

```
int cfxExportElementCount ()
```

Query the number of elements defined in the current zone.

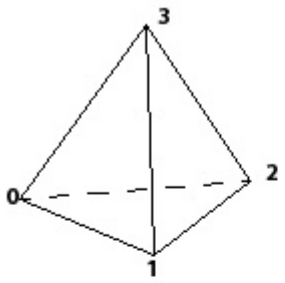
3.4.5.2. cfxExportElementList

```
cfxElement *cfxExportElementList ()
```

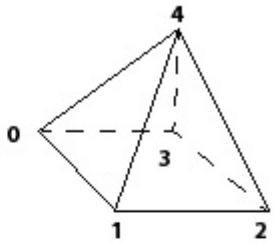
Return a pointer to an array of cfxElement elements (see cfxelem (p. 55)) containing the type and vertices of each element in the current zone. The first element in the zone is the first element of the array, the second the second and so on.

The memory allocated to represent this information should be deallocated using `cfxExportElementFree` (see cfxExportElementFree (p. 90)) when no longer required.

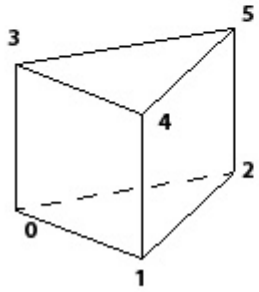
The following diagrams show the order of the nodes and connections that Ansys CFX uses for exporting elements:



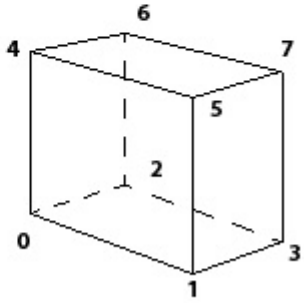
Tetrahedral element



Pyramid element



"Wedge" (prism) element



Hexahedral element

Note:

The vertex ordering for the import API is different. For details, see cfxImportElement (p. 48).

3.4.5.3. cfxExportElementGet

```
int cfxExportElementGet (int elemid, int elemtype, int *nodelist)
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Query the type and vertices of a specific element in the current zone.

The index (elemid) is specified between 1 and the number of elements returned by `cfxExportElementCount` (see cfxExportElementCount (p. 89)). If the value of elemid is out of range the return value is 0 otherwise it is elemid.

The type of the element is returned in elemtype and the vertices defining the element in nodelist. Note that nodelist must be large enough to hold the element number of vertices in the element (normally an array of 8 integers is used as this allows space enough for all element types to be handled).

3.4.5.4. cfxExportElementFree

```
void cfxExportElementFree ()
```

Deallocates any internal storage allocated by making calls to `cfxExportElementList` (see cfxExportElementList (p. 89)) or `cfxExportElementGet` (see cfxExportElementGet (p. 89)).

3.4.6. Region Routines

Regions are groups of faces in an Ansys CFX results file. Accessing regions within the current zone (see cfxExportZoneSet (p. 86)) is performed by making calls to the following functions. It should be noted that the region information is not loaded into the Export API until either `cfxExportRegionList` (see cfxExportRegionList (p. 91)) or `cfxExportRegionGet` (see cfxExportRegionGet (p. 91)) are called. This reduces memory overheads in the API by not allocating space until required.

When access to region in the current zone is no longer required a call to `cfxExportRegionFree` (see cfxExportRegionFree (p. 91)) should be made to deallocate any internal storage.

3.4.6.1. cfxExportRegionCount

```
int cfxExportRegionCount ()
```

Query the number of regions defined in the current zone.

3.4.6.2. cfxExportRegionSize

```
int cfxExportRegionSize (int regnum, int type)
```

Query the number of faces (if type is `cfxREG_FACES`) or nodes (if type is `cfxREG_NODES`) defined in the region identified by `regnum` in the current zone.

The function returns the number of faces or nodes in the current zone or 0 if either `regnum` is out of range or type is invalid.

3.4.6.3. cfxExportRegionName

```
char *cfxExportRegionName (int regnum)
```

Query the name of the region in the current zone identifies by regnum.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The function returns the name of the region or NULL if the region number supplied is out of range. The pointer returned points to static storage, which will be overwritten by the next call to `cfxExportRegionName`.

3.4.6.4. cfxExportRegionList

```
int *cfxExportRegionList (int regnum, int type)
```

Query the nodes (type is `cfxREG_NODES`) or faces (`cfxREG_FACES`) that define a region. This function returns a pointer to an array of node ids or face ids that define the region identified by regnum or NULL if the region number is out of range or the type is not recognized. If type is specified as `cfxREG_FACES`, the returned ids will represent faces. The element number and local element face number may be extracted from each face id returned by using the macros `cfxELEMNUM` and `cfxFACENUM`. The node numbers for the face may be obtained by calling `cfxExportFaceNodes`. For details, see cfxExportFaceNodes (p. 91).

3.4.6.5. cfxExportRegionGet

```
int cfxExportRegionGet (int regnum, int type, int index, int *id)
```

Query the index'th element (type is `cfxREG_ELEM`) or index'th node (type is `cfxREG_NODE`) that defines a region regnum in the current zone.

If regnum is out of range or type is not recognized or index is out of range, 0 is returned.

Otherwise id will contain the id of the appropriate node or face defining the region and the function will return index.

If type is specified as `cfxREG_FACES`, the returned id will represent the identity of a face. The element number and local element face number may be extracted from the id by using the macros `cfxELEMNUM` and `cfxFACENUM`.

3.4.6.6. cfxExportRegionFree

```
void cfxExportRegionFree (int regnum)
```

Deallocate any internal data storage associated with the region defined by `regnum`.

3.4.7. Face Routines

Faces are 2 dimensional (2D) units of mesh. Each global face ID is returned from `cfxExportBoundaryList` (see cfxExportBoundaryList (p. 95)) or cfxExportRegionList (see cfxExportRegionList (p. 91)).

Within CFX faces are either represented as Triangles (three vertices) or Quadrilaterals (two vertices). Each face in a CFX .res file will be parented by a single 3D element. The parent element of a face can be returned by the `cfxELEMNUM` macro with the global face ID, and the local face of that element can be determined by calling `cfxFACENUM` with the same global face ID

3.4.7.1. cfxExportFaceNodes

```
int cfxExportFaceNodes (int faceid, int *nodes)
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Requests the vertices for the face identified by `faceid`. The argument faceid should be constructed from the element number and local face number using the following formula:

```
(element_number << 3) & local_face_number
```

Values returned from `cfxExportRegionGet` and `cfxExportRegionList` can be supplied directly to this function.

The number of vertices defining the face are returned if `faceid` is valid, otherwise 0 is returned. The node numbers are returned in the array `nodes`, which should be dimensioned to a minimum size of 4 in the calling routine.

The face numbers and associated node indices are tabulated here:

Element Type	**Face**	**Nodes**			
tetrahedron	1	0	1	2	
	2	0	3	1	
	3	1	3	2	
	4	0	2	3	
pyramid	1	0	3	4	
	2	1	4	2	
	3	0	4	1	
	4	2	4	3	
	5	0	1	2	3
prism	1	0	2	5	3
	2	0	3	4	1
	3	1	4	5	2
	4	0	1	2	
	5	3	5	4	
hexahedron	1	0	2	6	4
	2	1	5	7	3
	3	0	4	5	1
	4	2	3	7	6
	5	0	1	3	2
	6	4	6	7	5

Note:

The face numbers and associated node indices are different when importing elements. For details, see cfxImportGetFace (p. 50).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.8. Volume Routines

Volumes are groups of elements in a CFX results file. Accessing volumes within the current zone (see cfxExportZoneSet (p. 86)) is performed by making calls to the following functions. It should be noted that the volume definitions for a zone are not loaded into the Export API until either `cfxExportVolumeList` (see cfxExportVolumeList (p. 93)) or `cfxExportVolumeGet` (see cfxExportVolumeGet (p. 93)) are called. This reduces memory overheads in the API by not allocating space until required.

When access to volume information in the current zone is no longer required a call to `cfxExportVolumeFree` (see cfxExportVolumeFree (p. 94)) should be made to deallocate any internal storage.

3.4.8.1. cfxExportVolumeCount

```
int cfxExportVolumeCount ()
```

Query the number of volumes defined in the current zone.

3.4.8.2. cfxExportVolumeSize

```
int cfxExportVolumeSize (int volnum, int type)
```

Query the number of nodes (if type is `cfxVOL_NODES`) or number of elements (if type is `cfxVOL_ELEMS`) defining the volume indexed by `volnum` in the current zone. The return value will be 0 if `volnum` is out of range or type is invalid.

3.4.8.3. cfxExportVolumeName

```
char *cfxExportVolumeName (int volnum)
```

Query the name of the volume in the current zone indexed by `volnum`. Returns `NULL` if the `volnum` is out of range.

Note:

The returned pointer points to internal storage, which will be overwritten by the next call to `cfxExportVolumeName`.

3.4.8.4. cfxExportVolumeList

```
int *cfxExportVolumeList (int volnum, int type)
```

Query the nodes (type is `cfxVOL_NODES`) or elements (`cfxVOL_ELEMS`) that define a volume.

This function returns a pointer to an array of node ids or element ids that define the volume identified by `volnum` or `NULL` if the volume number is out of range or the type is not recognized.

3.4.8.5. cfxExportVolumeGet

```
int cfxExportVolumeGet (int volnum, int type, int index, int *id)
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Query the [index]th element (type is `cfxVOL_ELEM`) or [index]th node (type is `cfxVOL_NODE`) that defines a volume `volnum` in the current zone.

If volnum is out of range or type is not recognized or index is out of range, 0 is returned.

Otherwise `id` will contain the id of the appropriate node or element in defining the volume and the function will return `index`.

3.4.8.6. cfxExportVolumeFree

```
void cfxExportVolumeFree (int volnum)
```

Deallocate any internal data storage associated with the volume defined by `volnum`.

3.4.9. Boundary Condition Routines

Boundary condition are located on groups of faces in a CFX results file. Accessing boundary condition locations within the current zone (see cfxExportZoneSet (p. 86)) is performed by making calls to the following functions. It should be noted that the boundary condition location information is not loaded into the Export API until either `cfxExportBoundaryList` (see cfxExportBoundaryList (p. 95)) or `cfxExportBoundaryGet` (see cfxExportBoundaryGet (p. 95)) are called. This reduces memory overheads in the API by not allocating space until required. When access to regions in the current zone are no longer required a call to `cfxExportBoundaryFree` (see cfxExportBoundaryFree (p. 96)) should be made to deallocate any internal storage.

3.4.9.1. cfxExportBoundaryCount

```
int cfxExportBoundaryCount ()
```

Query the number of boundary conditions defined in the current zone.

The function returns the number of boundary conditions in the current zone.

3.4.9.2. cfxExportBoundaryName

```
const char *cfxExportBoundaryName (const int bcidx)
```

Query the name of the boundary condition in the current zone identified by `bcidx`.

The function returns the name of the boundary condition or `NULL` if the `bcidx` supplied is out of range.

The pointer returned points to static storage, which will be overwritten by the next call to `cfxExportBoundaryName`.

Note:

The following routines use `bcidx`, which must lie between 1 and `cfxExportBoundaryCount()` and use index which must lie between 1 and `cfxExportBoundarySize` (bcidx, type).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

3.4.9.3. cfxExportBoundaryType

```
const char *cfxExportBoundaryType (const int bcidx)
```

Query the type (for example, Inlet, Outlet, and so on) of the boundary condition in the current zone identified by `bcidx`.

The function returns the type of the boundary condition or `NULL` if the `bcidx` supplied is out of range.

The pointer returned points to static storage, which will be overwritten by the next call to `cfxExportBoundaryType`.

3.4.9.4. cfxExportBoundarySize

```
int cfxExportBoundarySize (const int bcidx, const int type)
```

Query the number of faces (if type is `cfxREG_FACES`) or nodes (if type is `cfxREG_NODES`) defined in the boundary condition identified by `bcidx` in the current zone.

The function returns the number of faces or nodes or 0 if either `bcidx` is out of range or `type` is invalid.

3.4.9.5. cfxExportBoundaryList

```
int *cfxExportBoundaryList (const int bcidx, const int type)
```

Query the faces (if type is `cfxREG_FACES`) or nodes (if type is `cfxREG_NODES`) that define a boundary condition.

This function returns a pointer to an array of node ids or face ids that define the location of the boundary condition identified by `bcidx` or `NULL` if `bcidx` is out of range or the type is not recognized. If type is specified as `cfxREG_FACES`, the returned ids will represent faces. The element number and local element face number may be extracted from each face id returned by using the macros `cfxELEMNUM` and `cfxFACENUM` respectively. The node numbers for the face may be obtained by calling `cfxExportFaceNodes`. For details, see cfxExportFaceNodes (p. 91).

The returned pointer points to static data that should be destroyed using `cfxExportBoundaryFree`. Subsequent calls to `cfxExportBoundaryList` will overwrite the array.

3.4.9.6. cfxExportBoundaryGet

```
int cfxExportBoundaryGet (const int bcidx, const int type, const int index, int *id)
```

Query the index'th face (type is `cfxREG_FACES`) or index'th node (type is `cfxREG_NODES`) that defines the boundary condition location indexed by `bcidx` in the current zone. If `bcidx` is out of range or type is not recognized or `index` is out of range (not between 1 and `cfxExportBoundarySize`), 0 is returned. Otherwise `id` will contain the identifier of the appropriate node or face defining the boundary condition location and the function will return `index`.If type is specified as `cfxREG_FACES`, the returned `id` will represent the identity of a face. The element number and

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

local element face number may be extracted from the `id` by using the macros `cfxELEMNUM` and `cfxFACENUM` respectively.

3.4.9.7. cfxExportBoundaryFree

```
void cfxExportBoundaryFree (const int bcidx)
```

Deallocate any internal data storage associated with the boundary condition defined by `bcidx`.

3.4.10. Variable Routines

These routines access the variable data defined on the current zone as defined by `cfxExportZoneSet`. For details, see cfxExportZoneSet (p. 86). The variable data arrays are not loaded into memory until either `cfxExportVariableList` or `cfxExportVariableGet` are called, and remain in memory until `cfxExportVariableFree` is called. For details, see:

- cfxExportVariableList (p. 97)
- cfxExportVariableGet (p. 97)
- cfxExportVariableFree (p. 97).

3.4.10.1. cfxExportVariableCount

```
int cfxExportVariableCount(int usr_level)
```

Query the number of variables at interest level `usr_level` or below. If `usr_level` is 0, then the total number of variables is returned.

3.4.10.2. cfxExportVariableSize

```
int cfxExportVariableSize (int varnum, int *dimension, int *length, int *bdnflag)
```

Query the dimension, `dimension`, and length, `length`, for the variable identified by `varnum`, which should be from 1 to the number of variables, returned by `cfxExportVariableCount` (cfxExportVariableCount (p. 96)). The length, `length`, will either be 1 or the same as the number of nodes returned by `cfxExportNodeCount` (see cfxExportNodeCount (p. 88)). If 1, then the variable has meaningful values only at the boundary nodes, with a constant value in the interior.

The function also returns `bdnflag`, which indicates if the variable contains corrected boundary node values (`1`) or not (`0`).

The function returns `varnum` if successful, or 0 if the variable number is out of range.

3.4.10.3. cfxExportVariableName

```
char *cfxExportVariableName (int varnum, int alias)
```

Query the name of the variable identified by `varnum`.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The return value of the function is NULL if the variable number is out of range or the name of the variable.

The pointer returned points to static storage, which will be overwritten by the next call to cfxExportVariableName.

The argument alias indicates whether the short name (alias=0) or long name (alias=1) should be returned. For example, the short and long names for the total temperature variable are TEMPTOT and Total Temperature, respectively.

3.4.10.4. cfxExportVariableList

```
float *cfxExportVariableList (int varnum, int correct)
```

Query the results data for a variable identified by varnum.

Returns NULL if the variable number is out of range or the variable data if successful.

The flag correct indicates whether to correct boundary node data (correct=1) or not (correct=0), assuming that it exists.

The data is in the same order as the nodes returned from cfxExportNodeList (see cfxExportNodeList (p. 88)).

For multidimensional variables, the data is stored with dimension consecutive values for each node.

The storage for the data is created by the Export API when this function is called. When the data is no longer required a call to cfxExportVariableFree (see cfxExportVariableFree (p. 97)) should be made by the calling function.

3.4.10.5. cfxExportVariableGet

```
int cfxExportVariableGet (int varnum, int correct, int index, float *value)
```

Request the values of the variable identified by varnum at the location given by index, which should be from 1 to the length of the variable, inclusively.

The flag correct indicates whether to correct boundary node data (correct=1) or not (correct=0), assuming that it exists.

The function returns index, or 0 if the location is out of range.

3.4.10.6. cfxExportVariableFree

```
void cfxExportVariableFree (int varnum)
```

Deallocate the internal data storage for the variable identified by varnum for the current zone.

3.4.10.7. cfxExportVariableQuantityDimensions

```
const float* cfxExportVariableQuantityDimensions(const int varnum)
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Take an argument that is the index of the variable in `cfxExportVariableList`, and return an array of floats of size `cfxNUM_QUANTITIES`. Each entry will be 0 if that indexed quantity does not apply to this variable, or will otherwise be the power to which the unit associated with the indexed quantity is raised.

3.4.10.8. cfxExportVariableUnitsString

```
const char* cfxExportVariableUnitsString(const int varnum)
```

Take an argument that is the index of the variable in `cfxExportVariableList` and return a combined units string. You can use this function to avoid having to process the dimensions using the `cfxExportVariableQuantityDimensions` function.

3.4.11. Timestep Routines

3.4.11.1. cfxExportTimestepCount

```
int cfxExportTimestepCount ()
```

Query the number of timesteps defined in the CFX results file.

If the case is not transient, the function returns 0.

3.4.11.2. cfxExportTimestepTimeGet

```
float cfxExportTimestepTimeGet( const int tsIndex )
```

Queries the time represented by the timestep index `tsIndex`.

The timestep index `tsIndex` must be between 1 and the return value of `cfxExportTimestepCount()`.

If `tsIndex` is invalid, the function returns -1.0.

3.4.11.3. cfxExportTimestepNumGet

```
int cfxExportTimestepNumGet ( const int tsIndex )
```

Query the timestep number for the timestep index supplied.

The timestep index `tsIndex` must be between 1 and the return value of `cfxExportTimestepCount()`.

If `tsIndex` is invalid, the function returns -1.

3.4.11.4. cfxExportTimestepSet

```
int cfxExportTimestepSet ( const int tsNumber )
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Sets the current timestep being accessed by the Export API.

The value of `tsNumber` should be obtained via `cfxExportTimestepNumGet()`.

Note:

`tsNumber` is not necessarily the same as the associated timestep index as is supplied to the other routines in this section.

If `tsNumber` is invalid, the function returns -1.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

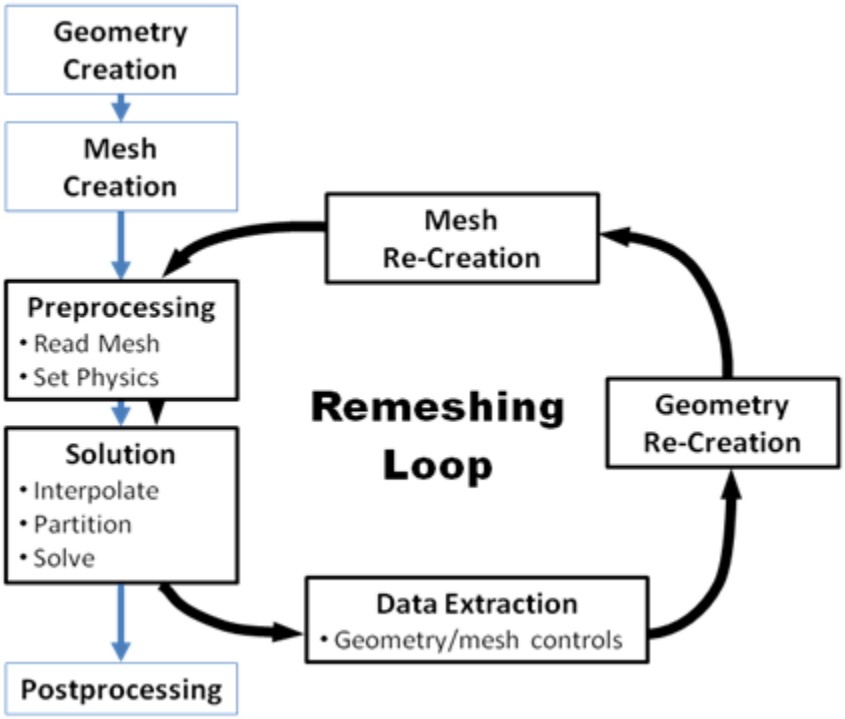
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 4: Remeshing Guide

Periodic remeshing is an important part of running analyses that involve significant mesh deformation. Remeshing is often required simply to maintain acceptable mesh quality, as described in Discretization Errors in the *CFX-Solver Theory Guide* and Measures of Mesh Quality in the *CFX-Solver Modeling Guide*.

A schematic illustrating the integration of remeshing into the general simulation workflow is shown in the figure below.

Figure 4.1: Integration of a Remeshing Loop into the General Simulation Workflow



As shown in Figure 4.1: Integration of a Remeshing Loop into the General Simulation Workflow (p. 101), in addition to the **Preprocessing** and **Solution** steps of the standard simulation workflow, the remeshing loop includes three additional steps:

- **Data Extraction**
- **Geometry Modification**
- **Mesh Recreation**.

In the context of remeshing, those steps are responsible for completing the following sub-steps;

- **Data Extraction**: Extract any data needed to guide geometry modifications and mesh re-creation from the most recent analysis results and monitor point values.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

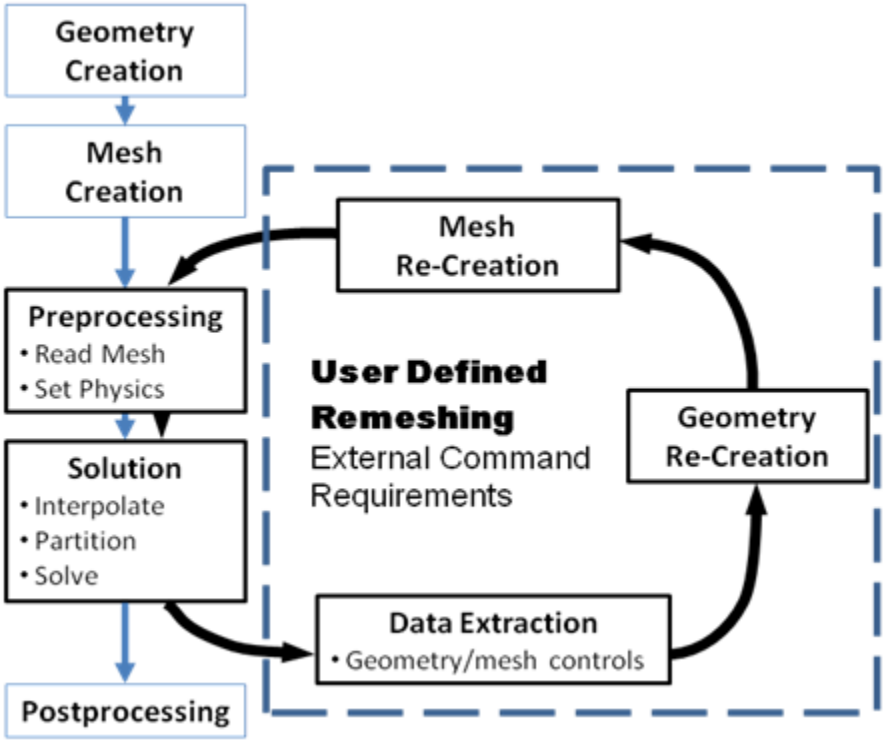
- **Geometry Re-Creation**: Update the analysis' geometry so that it conforms to that of the most recent analysis results (that is, account for mesh deformation).
- **Mesh Re-Creation**: Generate new mesh(es) that correspond to the updated geometry.
- **Preprocessing**: Insert the new mesh(es) into the analysis definition, and generate an updated CFX-Solver Input File.
- **Solution**: Interpolate the previously generated analysis results onto the new mesh, re-partition the mesh if a parallel run mode is selected, and continue the solution process.

As described in Remeshing Tab in the *CFX-Pre User's Guide*, there are two options available for remeshing: **User Defined** and **ICEM CFD Replay**. As outlined in the discussions that follow, the **Preprocessing** and **Solution** steps (and their respective sub-steps) are automatically executed for both remeshing options. Although the remaining steps are automatically executed for the **ICEM CFD Replay** remeshing option, they become the responsibility of a user defined external command for the **User Defined** remeshing option.

4.1. User Defined Remeshing

User Defined remeshing offers the greatest flexibility to customize the remeshing process. This comes at the expense of requiring that the data extraction, and geometry and mesh re-creation steps are executed by a user-specified external command, as illustrated in the figure below where the dashed line identifies steps that must be executed by the user-specified **External Command**.

Figure 4.2: Schematic for User Defined remeshing



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

This remeshing option is ideally suited for users who have previously completed an 'in-house' remeshing solution involving scripts or varying degrees of manual user-intervention. When this option is used, the following steps are automatically executed:

- Run the specified external command to generate a new mesh(es)
- Insert the new mesh(es) into the analysis definition, and generate an updated CFX-Solver Input file
- Interpolate the previously generated analysis results onto the new mesh, re-partition the mesh if a parallel run mode is selected, and continue the solution process.

The following examples outline the use of the **User Defined** remeshing option:

- Remeshing with Key-Frame Meshes (p. 103)
- Remeshing with Automatic Geometry Extraction (p. 104).

4.1.1. Remeshing with Key-Frame Meshes

In some analyses involving mesh deformation, the motion of various boundaries and sub-domains is known beforehand. Thus, the iteration or time step number at which unacceptable mesh quality will occur due to mesh motion is also known. A sequence of 'key-frame' meshes (of any mesh file type) corresponding to these instances of poor mesh quality can consequently be generated and applied during the analysis.

Once the sequence of key-frame meshes has been generated, they should be placed in a location that will be accessible during the analysis' execution. The analysis definition is then modified to include one or more control conditions that will interrupt the solver at the iteration or time step at which a key-frame mesh should be inserted. A configuration is subsequently defined (unless this has already been done), and a remeshing definition is created with the following settings:

- Set **Option** to **User Defined**.
- Set the **Activation Condition**(s) to the previously created interrupt control condition(s).
- Set the **Location** to the mesh region that will be replaced.
- Set the **External Command** to the command that will be used to generate the replacement mesh file.
- Set the **Replacement File** to the name of the file that will be generated by the external command.

The **External Command** is typically a shell script or batch file that completes the following tasks:

- Determine which key-frame mesh to use. This will require parsing the run's CFX-Solver Output file for the iteration or time step number, or the actual simulation time. Output generated from the `cfx5mondata` executable can also be parsed instead of the run's CFX-Solver Output file. For details, see Exporting Monitor Data from the Command Line in the *CFX-Solver Manager User's Guide*.
- Copy the key-frame mesh to the path the specified by the **Replacement File** setting.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

4.1.2. Remeshing with Automatic Geometry Extraction

In some analyses involving mesh deformation, the motion of various boundaries and sub-domains is not known beforehand and the key-frame remeshing strategy presented above is not applicable. In these analyses, geometrical information must be extracted from the most recent analysis results and applied in the remeshing process.

In such cases, the analysis definition is modified to include one or more control conditions that will interrupt the solver when, for example, mesh quality deteriorates significantly. A configuration is subsequently defined (unless this has already been done), and a remeshing definition is created with the following settings:

- Set **Option** to **User Defined**.
- Set the **Activation Condition**(s) to the previously created interrupt control condition(s).
- Set the **Location** to the mesh region that will be replaced.
- Set the **External Command** to the command that will be used to generate the replacement mesh file.
- Set the **Replacement File** to the name of the file that will be generated by the external command.

The **External Command** is typically a shell script or batch file that completes the following tasks:

- Extract geometry data from the most recent solution of the analysis, and either update or replace the original geometry. This may be done using mesh-to-geometry conversion tools available in software such as Ansys ICEM CFD, or by extracting monitor point data values (for example, the Total Centroid Displacement variable) using the `cfx5mondata` executable. For details, see Exporting Monitor Data from the Command Line in the *CFX-Solver Manager User's Guide*.
- Create a replacement mesh file using the updated or newly generated geometry. This may be done in any suitable mesh generation application.

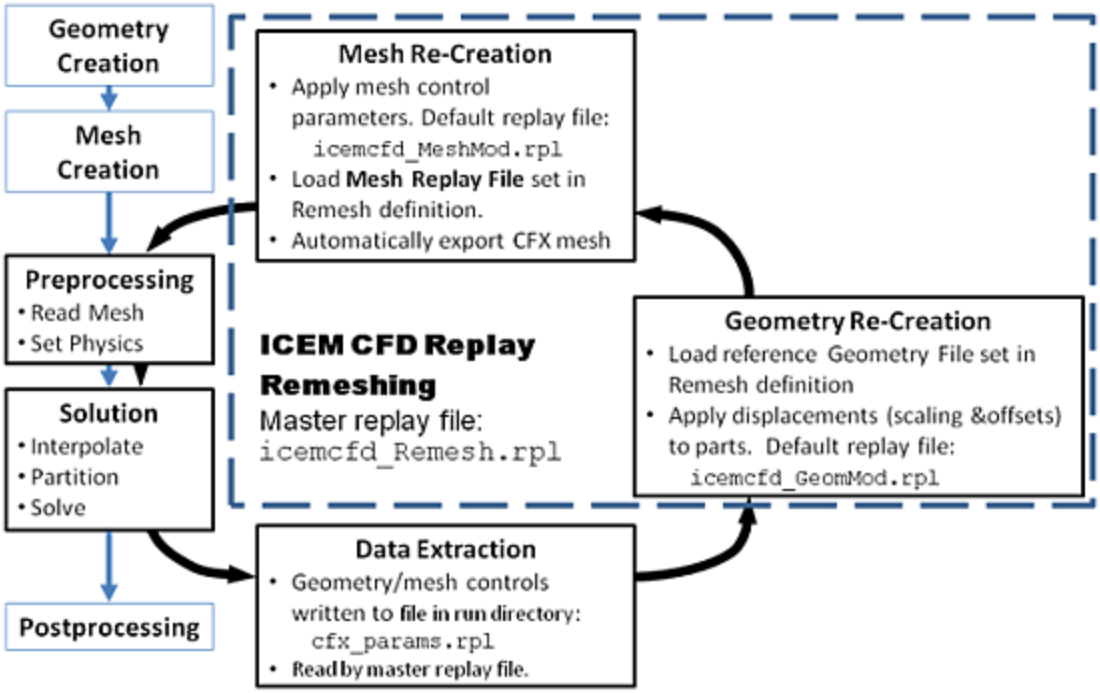
Note that some mesh-to-geometry conversion tools are unable to extract the latest mesh coordinates from the most recent CFX-Solver Results file. If this is the case, then introduce a call to CFX-Pre (within the **External Command**) that executes a session file that simply loads the latest CFX-Solver Results file and writes a new CFX-Solver Input file. That CFX-Solver Input file will contain the required, latest mesh coordinates.

4.2. ICEM CFD Replay Remeshing

ICEM CFD Replay remeshing provides a highly automated remeshing process that is ideally suited for users of the Ansys ICEM CFD mesh generation software and cases that involve translational mesh motion only (that is, no rotation or general deformation). When this option is used, a master replay file is assembled from other task-oriented replay files and submitted to the Ansys ICEM CFD mesh generator for batch execution. These replay files are illustrated in the figure below, along with the general process flow for this remeshing option. The dashed line in the figure highlights components of the master replay file and identifies files and steps that you can modify. Unless otherwise noted, files are contained in the `<CFXROOT>/etc/Remeshing` directory.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 4.3: Schematic for ICEM CFD Replay remeshing



When this option is used, the following steps are automatically executed:

- Extract geometry and mesh control data and write them to the `cfx_params.rpl` replay file in the run directory. The data includes:
  - Centroid displacements for boundaries that are included in Ansys ICEM CFD Part Maps
  - Mesh control parameters (for example, `ehgt` and `emax`)
  - Scalar parameters.
- Run (in batch) the Ansys ICEM CFD mesh generation program using the master replay file. This master replay file executes the following tasks:
  - Read the `cfx_params.rpl` file.
  - Load the reference geometry from the **Geometry File** identified in the **Remesh** definition.
  - Apply displacements (including scaling and any offsets) corresponding to all Ansys ICEM CFD Part Map definitions contained in the **Remesh** definition. This is done using the default geometry replay file provided, or using the user defined replay file if specified in the ICEM CFD Geometry Control setting.
  - Apply **ICEM CFD Mesh Controls** defined in the **Remesh** definition. This is done using the provided controls, or using the user-defined replay file if specified in the **ICEM CFD Mesh Control** setting.
  - Load your **Mesh Replay File**, specified in the **Remesh** definition.
  - Export a new mesh for Ansys CFX.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Insert the new mesh(es) into the analysis definition, and generate an updated CFX-Solver Input file.

- Interpolate the previously generated analysis results onto the new mesh, re-partition the mesh if a parallel run mode is selected, and continue the solution process.

You must create the reference **Geometry File** and the **Mesh Replay File** because these are specific to each case. However, the generic default replay files (`icemcfd_Remesh.rpl`, `icemcfd_GeomMod.rpl`, and `icemcfd_MeshMod.rpl`) used by this option are provided in the *<CFXROOT>*`/etc/Remeshing` directory. These files may be edited to provide installation-wide changes to the ICEM CFD Replay remeshing behavior. Alternatively, the geometry and mesh modification files may be copied and edited to provide case-specific changes.

Note:

As indicated previously, only translational mesh motion is automatically handled by the ICEM CFD Replay remeshing option. This is accomplished by applying the displacements of centroids of boundaries in the Ansys CFX analysis definition to parts in the Ansys ICEM CFD geometry. All other mesh motion (such as rotation about the centroid or another point, or general deformation) will not be applied, and an inconsistency in the analysis geometry before and after remeshing will be introduced.

4.2.1. Steps to Set Up a Simulation Using ICEM CFD Replay Remeshing

The following discussion presents the three general steps required to set up a simulation using the ICEM CFD Replay remeshing option.

The first step involves creating the reference **Geometry File** within the Ansys ICEM CFD environment. If the geometry was not created within that environment, use one of the **File** > **Import Geometry** options in the Ansys ICEM CFD environment. At this point, ensure that all required Parts (or Families) are defined and named so that they can be referenced when completing the ICEM CFD Replay remeshing definition later in CFX-Pre. Finally, store the geometry in the ICEM CFD native geometry file format (namely, a `.tin` file).

The second step involves generating the **Mesh Replay File**, again, from within the Ansys ICEM CFD environment. Start with the previously created geometry loaded, and work sequentially through the mesh generation process until acceptable mesh controls have been specified. This may require fine tuning, which will involve the regeneration of your mesh after moving the geometry through its expected range of motion. Once you are satisfied with the mesh control settings, purge the last mesh using **File** > **Mesh** > **Close Mesh**, and reload the original reference geometry. Complete the following tasks to generate the required **Mesh Replay File**:

1. Use **File** > **Replay Scripts** > **Replay Control** to begin recording the commands for the **Mesh Replay File**. The **Replay Control** dialog box is displayed.

2. Revisit all of the mesh related tabs and settings used to generate the mesh, clicking either the **Apply** or **OK** to commit the settings into the **Replay Control** panel.

3. Generate the mesh.

4. In the **Replay Control** panel, clear the **Record (after current)** toggle and select **Save** to write the settings to replay file.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

You may also want to export the mesh that was (re)generated for use in the simulation definition (as in the next step).

The third step involves defining the simulation within CFX-Pre. Complete the following tasks to prepare the simulation:

1. Start a new simulation and import the (previously generated) mesh.
2. Define expressions for the motion of the geometry.

   Note:

   See also the discussion in Mesh Re-Initialization During Remeshing (p. 108).

3. Define the flow analysis including the definition of one or more solver interrupt controls, as described in Interrupt Control in the *CFX-Pre User's Guide*, to identify the condition(s) under which solver execution will be interrupted.
4. Define a configuration and complete the ICEM CFD Replay remeshing setup as described in Ansys ICEM CFD Replay Remeshing in the *CFX-Pre User's Guide*. The **Geometry File** and **Mesh Replay File** created above are referenced here. Note also, that references to one or more of the previously defined solver interrupt control conditions are required to activate remeshing.
5. Complete any execution controls for the simulation and either start the solver or write the CFX-Solver Input file for later use.

4.3. Directory Structure and Files Used During Remeshing

CFX-Solver runs that include remeshing will have a slightly non-standard directory structure during execution. For example, using a CFX-Solver input file named `case.def`, a directory structure similar to the following will exist just after solution execution is interrupted and the second instance of remeshing begins:

```
case.def
case_001/
      1_full.trn
      0_full.trn
      2_full.trn
      3_oldmesh.res
      3_remesh.out
 case_001.dir/
      3_full.trn
      4_full.trn res mon
```

The first instance of remeshing occurred when the solver was interrupted after the third time step. Following this instance of remeshing, all CFX-Solver Results files (such as transient, backup, and remeshing) contained in the run directory, `case_001.dir`, were moved into the final solution directory, `case_001`. The results file written when the solver was interrupted before remeshing was renamed to `3_oldmesh.res`. Any text output to the console window during remeshing was redirected to the file named `3_remesh.out`, which is also placed in the final solution directory.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The second, and currently running, instance of remeshing began when the solver was interrupted after the fifth time step. The results file written by the solver still has the generic name, `res`, and monitor data (contained in the `mon` file) has not yet been inserted into the results file.

Just after inserting the new mesh(es) into the analysis definition, the files contained in the final solution and run directories change slightly. The CFX-Solver Results and CFX-Solver Output files are renamed (to `5_oldmesh.res` and `5_remesh.out`, respectively) and moved from the run directory into the final solution directory. An automatically generated session file, `meshUpdate.pre`, is used by CFX-Pre to generate the updated the solver input file, `5_newmesh.def`, and each of these files are present in the run directory. These files are, however, replaced or removed during the next instance of remeshing or when the analysis ends and the run directory is deleted.

4.4. Additional Considerations

This section discusses additional considerations for remeshing.

4.4.1. Mesh Re-Initialization During Remeshing

4.4.2. Software License Handling

4.4.3. Results File Option

4.4.1. Mesh Re-Initialization During Remeshing

The following points are important to note during remeshing:

- The total mesh displacement variable is relative to a specific mesh topology. Because the mesh topology changes, this variable is reset each time remeshing occurs.
- The new variable called `total centroid displacement` tracks the displacement of each boundary's centroid since the beginning of the analysis (that is, relative to the original mesh).
- The specified displacement based mesh motion is relative to the initial mesh and must therefore include an offset to account for mesh re-initialization. The `Mesh Initialisation Time` variable corresponds to the time at which mesh re-initialization last occurred. This can be used to evaluate the required offset for time varying mesh displacement.

Note:

`Total Centroid Displacement` is the sum of total mesh displacement and an offset vector. The value of the offset is determined by the displacement of the boundary's centroid from its original position to its initial position. When using `Total Centroid Displacement` in a non-remeshing case, the original and the initial position are the same, resulting in an offset value of zero. In this case, there seems to be no difference between the variables, `Total Centroid Displacement` and `Total Mesh Displacement`. However, when using `Total Centroid Displacement` in a remeshing case, the original position is defined at the start of the simulation, while the initial position is defined after the last remesh; this results in an offset contribution that is non-zero.

An example of the expressions used to evaluate an applied displacement that includes the required offset to account for mesh re-initialization is given below. In this example, the applied displacement

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

is evaluated as the desired displacement minus the value of the desired displacement at the `Mesh Initialisation Time`.

```
Disp Desired = 1[m]*0.5*(1-cos(2.[s^-1]*pi*t))
Disp Mesh ReInit = 1[m]*0.5*(1-cos(2.[s^-1]*pi*Mesh Initialisation Time ))
Disp Applied = Disp Desired - Disp Mesh ReInit
```

4.4.2. Software License Handling

Several software components (for example, CFX-Pre, the CFX-Solver, Ansys ICEM CFD, and so on) are used while executing steps in the overall remeshing process. Rather than holding all of these licenses for the entire duration of the analysis, they are only 'checked out' as required. Although this frees up the licenses for other users when remeshing is not executing, it also introduces the possibility that required licenses are not available when they are needed for remeshing.

This model for software license handling may cause problems in multi-user environments, but work is underway to provide a broader range of handling options for future releases.

4.4.3. Results File Option

In order for remeshing to proceed, results files must be of type `Standard`, `Essential`, or `Refiner`.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 5: Reference Guide for Mesh Deformation and Fluid-Structure Interaction

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with little or moderate experience using CFX for applications involving Mesh Deformation and/or Fluid Structure Interaction.

This guide describes:

5.1. Mesh Deformation

5.2. Fluid Structure Interaction

5.1. Mesh Deformation

Mesh deformation is an important part of executing simulations with changing domain geometry. In CFX, this capability is available in fluid and solid domains. Motion can be specified on selected regions via CEL or an external solver coupling, or on all nodes in an entire domain via a user-Fortran junction box routine.

5.1.1. Mesh Folding: Negative Sector and Element Volumes

It is not uncommon for the mesh to become folded (or tangled) during the mesh deformation process. When this occurs, a message indicating the existence and location of either negative sector volumes or negative (that is, topologically invalid) elements is written to the CFX-Solver Output file. Notification of negative sector volumes highlights the existence of non-convex mesh elements that still have a positive volume. Although the existence of negative sector volumes is not a fatal condition, it does indicate that:

- Mesh elements are only barely positive
- Further mesh deformation is likely to yield elements with negative volumes, which is a fatal condition

Some of the most common causes for mesh folding during deformation are identified in the following sections.

5.1.2. Applying Large Displacements Gradually

In many simulations that require mesh deformation, the motion is known a priori. In these cases, the motion can be applied gradually, by relating it to the iteration or timestep counters, to reduce the likelihood of mesh folding. Mesh folding is often avoided with this strategy because the mesh displacement equations are assembled using the updated meshes from each deformation step (that is, outer iteration or timestep). In general, the desired total mesh deformation should be split up so that regions where motion is specified move through less than approximately 5 adjacent elements per step.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5.1.3. Consistency of Mesh Motion Specifications

Mesh motion options such as `Specified Displacement` may be applied on multiple boundary and subdomain regions. Because the specified motion is applied directly to mesh nodes, rather than control volume integration points, care is required to ensure that motion specified on adjacent regions is self-consistent. For example, the motion specified on one moving wall should be reduced to zero for any nodes that are shared with another stationary wall. If this is not done, then the motion applied to the shared nodes will be either the moving or stationary condition, depending on which was applied last during the equation assembly process.

Folded meshes often result from the application of inconsistent motion specifications.

5.1.4. Solving the Mesh Displacement Equations and Updating Mesh Coordinates

During each outer iteration or timestep, the mesh displacement equations are solved to the specified convergence level and the resulting displacements are applied to update the mesh coordinates. This occurs before proceeding to solve the general transport (for example, hydrodynamics, turbulence, and so on) equations.

Unlike other equation classes, the convergence level (that is, controls and criteria) applied to mesh displacement equations is unaffected by changes made to the basic settings for all other equations. The default convergence controls and criteria for the mesh displacement equation are tabulated below, and are changed by visiting the **Mesh Displacement** entry in the **Equation Class Settings** tab under **Solver Control**.

Setting	Value
Maximum Number of Coefficient Loops	5
Minimum Number of Coefficient Loops	1
Residual Type	RMS
Residual Target	1.0E-4

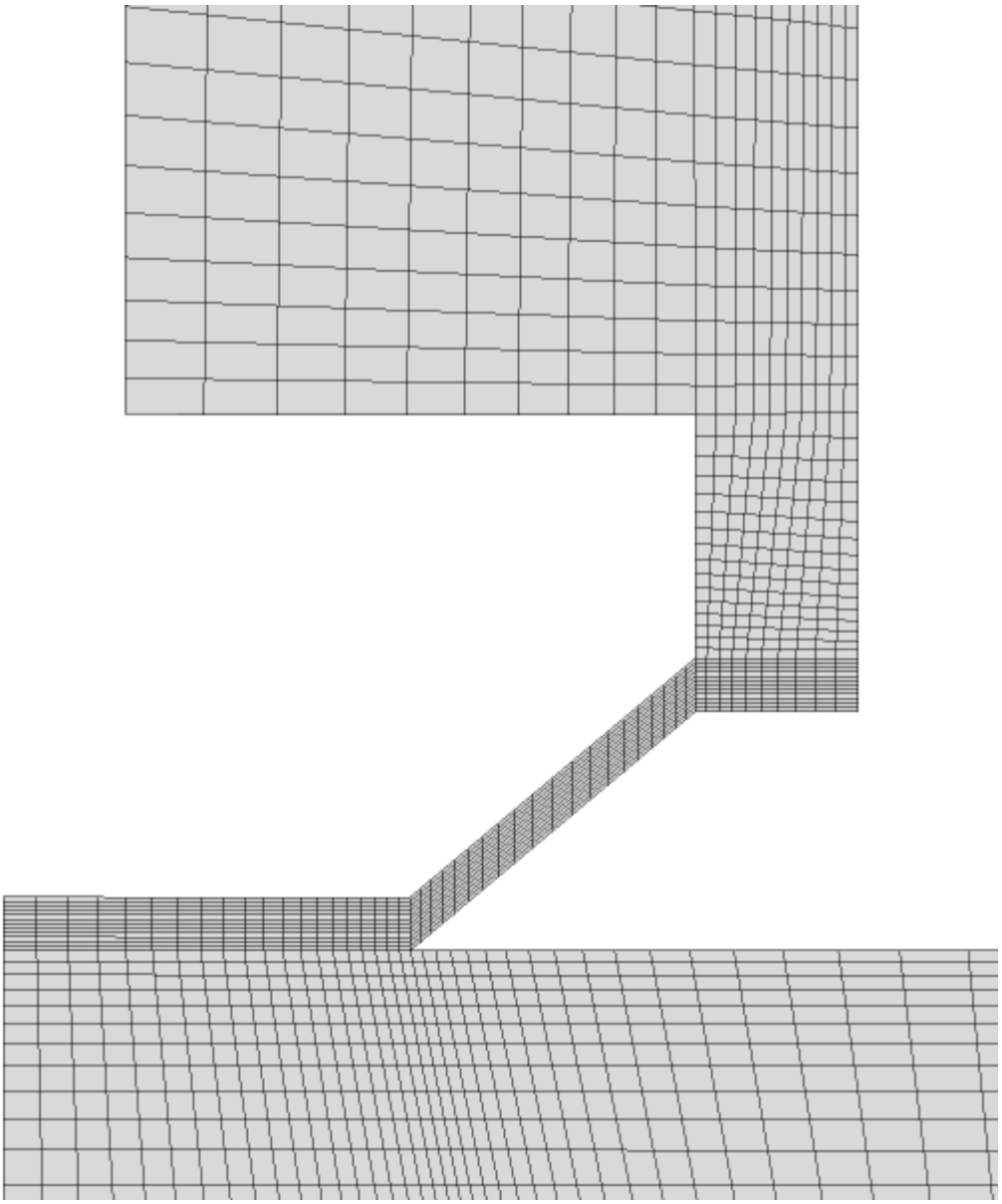
Mesh folding occurs and is detected when the displacements are used to update the mesh coordinates. Folded meshes can occur if the displacement equations are incompletely solved. In this case, the unconverged displacement solution field does not vary smoothly enough to ensure that adjacent mesh nodes move by similar amounts.

5.1.5. Mesh Displacement Diffusion Scheme

A number of numerical schemes are available for the solution of the mesh displacement diffusion equation. In many cases the default scheme is appropriate, but in some situations mesh folding can be avoided and improved mesh quality can be obtained by using a different scheme. The expert parameter `meshdisp diffusion scheme` controls which numerical scheme is used. For details, see Discretization Parameters. Setting `meshdisp diffusion scheme` to 3 can be helpful if the mesh folds unexpectedly at sharp corners or if uniform mesh deformation is expected but non-uniform deformation is observed. The figures below show an example of using different mesh displacement diffusion schemes.

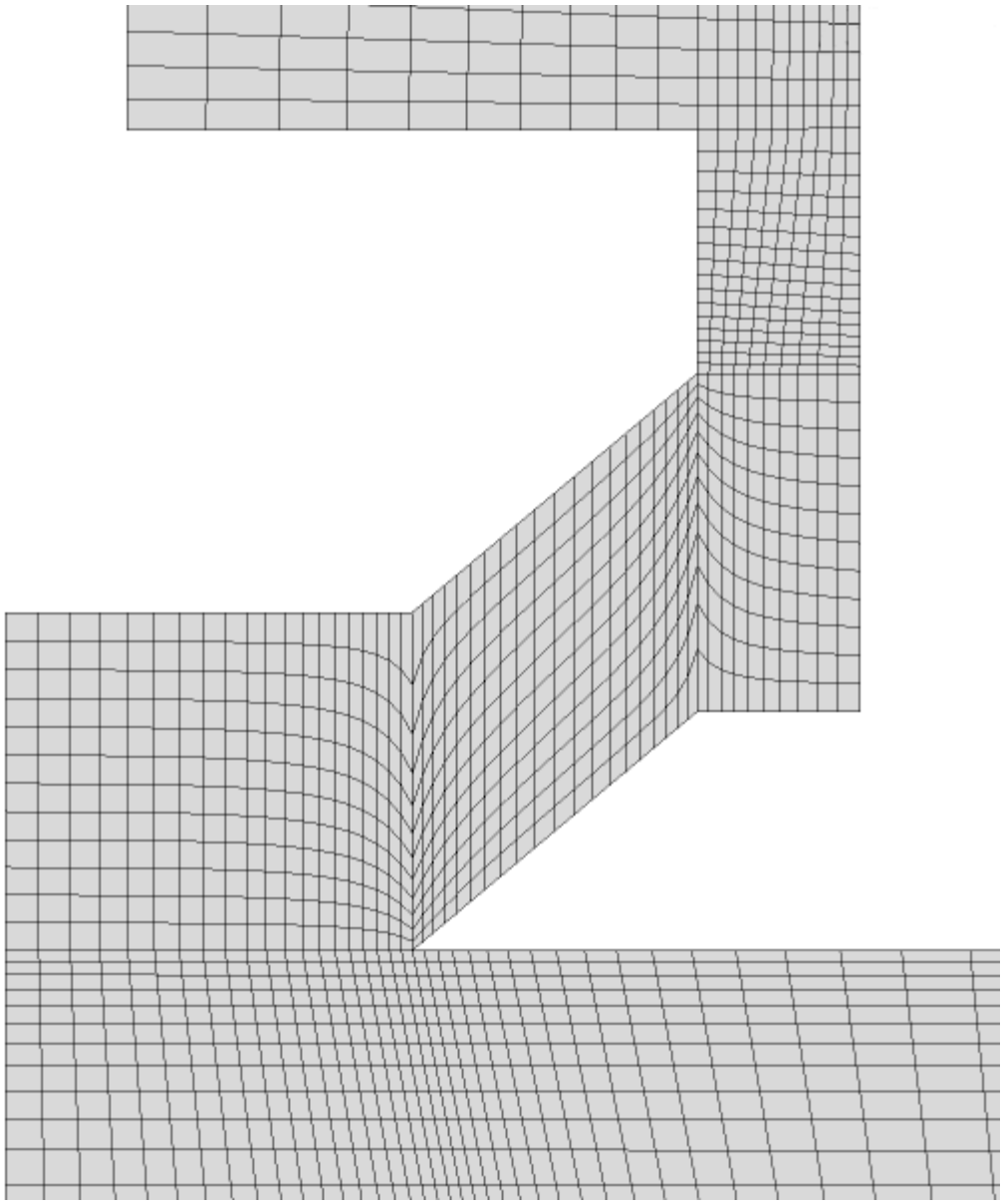
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 5.1: Original Undeformed Mesh



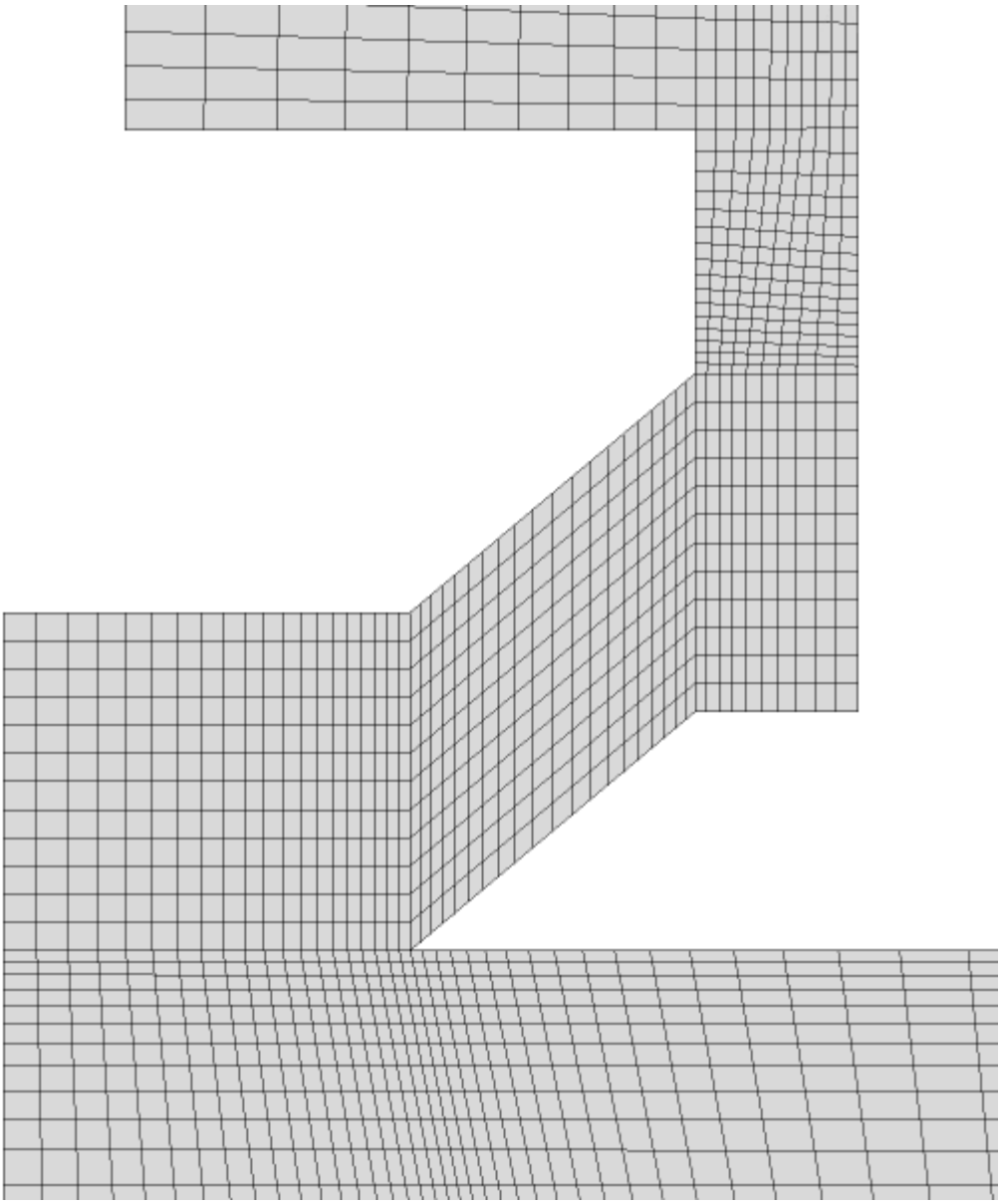
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 5.2: Deformed Mesh with the Default Diffusion Scheme



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 5.3: Deformed Mesh with "meshdisp diffusion scheme = 3"



5.1.6. Mesh Displacement vs. Total Mesh Displacement

A number of new variables become available when executing simulations with mesh deformation. Two of these variables are `Mesh Displacement` and `Total Mesh Displacement`.

`Mesh Displacement` is the principal variable that is solved for by the mesh motion model (see Mesh Deformation in the *CFX-Solver Modeling Guide*). This variable represents the displacement relative to the previous mesh locations. Conversely, `Total Mesh Displacement` is a derived quantity that represents the displacement relative to the initial mesh.

5.1.7. Simulation Restart Behavior

The following table summarizes the behavior that occurs when simulations with (or without) mesh deformation are restarted with (or without) mesh deformation. With only the exception noted, the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

simulation type (that is, steady-state or transient) used for the initial or restart run does not affect behavior.

Initial Simulation	Restart Simulation	Restart Behavior
No Deformation	Deformation	Mesh from initial run serves as initial mesh for restart run
Deformation	No Deformation	Final mesh from initial run serves as mesh for restart run
Deformation	Deformation	Initial mesh from initial run serves as initial mesh for restart run[a]

[a] If the restart is a transient run with the initial time set to `Value`, then the final mesh from the initial run will serve as the initial mesh for the restart simulation.

5.2. Fluid Structure Interaction

CFX provides the ability to solve, or take part in the solution of cases that involve the coupling of solution fields in fluid and solid domains. This coupling is commonly referred to as Fluid Structure Interaction (FSI). One example of FSI is the simulation of an internal combustion engine, which involves the solution of fluid flow, conjugate heat transfer and combustion problems on deforming meshes.

In the discussion that follows, examples are presented to demonstrate the FSI capabilities using CFX by itself or with other CAE packages like Ansys Mechanical. These examples are grouped according to the degree of coupling that must be maintained during the simulation in order to ensure that accurate results are obtained.

You can couple CFX to an external solver using System Coupling. For more information about simulations involving System Coupling and CFX, refer to Coupling CFX to an External Solver: System Coupling Simulations in the *CFX-Solver Modeling Guide*.

5.2.1. Unidirectional (One-Way) FSI

In many FSI simulations, the coupling between the solution fields is predominantly unidirectional; a given field may strongly affect, but not be affected by other fields. In CFX, there are a variety of strategies to efficiently execute such simulations. These strategies are identified in the following examples.

5.2.1.1. Using CFX Only

One of the most useful examples of unidirectional FSI within CFX involves prescribed mesh deformation of fluid or solid domains. This is possible using the CEL to specify the motion of sub-domains or domain boundaries, or by reading a sequence of pre-defined meshes.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5.2.1.2. Using CFX and the Mechanical Application

In many FSI simulations, the capabilities of additional solvers are required to compliment those of CFX. In these circumstances, CFX provides tools to facilitate the import and export of solution data in a variety of formats.

5.2.1.2.1. Importing Data from the Mechanical Application Solver

The recommended method for importing boundary condition data from the Mechanical application into CFX is via boundary profile data. For information about the creation and use of profile data files, refer to Unidirectional Load-Transfer for CFD Applications in the *Coupled-Field Analysis Guide* and Use Profile Data in the *CFX-Pre User's Guide*.

5.2.1.2.2. Mechanical Import/Export Example: One-Way FSI Data Transfer

You can perform one-way FSI operations manually (by exporting CDB files from the Mechanical APDL application, importing the surface in CFD-Post, and exporting the SFE commands).

To create a Mechanical load file using CFD-Post to transfer FSI data:

1. Load the fluids results file, from which you want to transfer results, into CFD-Post

2. Select **File** > **Import** > **Import Mechanical CDB Surface**. The **Import Mechanical CDB Surface** dialog box appears.

3. In the **Import Mechanical CDB Surface** dialog box, either:

   - Select the CDB file that specifies the surface mesh of the solid object to which to transfer data. Also select the **Associated Boundary** for the surface to map onto, and make other selections as appropriate.

   - Select the XML document that provides all transfer information. Click **OK**, and the surface data is loaded.

4. Select **File** > **Export** > **Export Mechanical Load File**. The **Export Mechanical Load File** dialog box appears.

5. In the **Export Mechanical Load File** dialog box, select a filename to which to save the data. For the **Location** parameter value, select the imported Ansys mesh object. Under **File Format** select **Ansys Load Commands (FSE or D)**. (Alternatively, you can select **WB Simulation Input (XML)** to get XML output.) Also select the appropriate data to export: Normal Stress Vector, Tangential Stress Vector, Stress Vector, Heat Transfer Coefficient, Heat Flux, or Temperature. Click **Save**, and the data file is created.

The one-way FSI data transfer described above is performed automatically when using the **FSI: Fluid Flow (CFX) > Static Structural** custom system in Ansys Workbench. For details, see the FSI: Fluid Flow (Ansys CFX) > Static Structural in the *Workbench User's Guide* section in the Ansys documentation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

5.2.1.3. Using CFX and Other CAE Software

Solution data can be exported from CFX in a variety of general formats during or after execution of the CFX-Solver. For information about the export of data in CGNS format during the execution of the solver, refer to Export Tab in the *CFX-Pre User's Guide*. For information about the extraction and export of CGNS, MSC Patran, FIELDVIEW, EnSight and custom data from CFX results files, refer to Generic Export Options in the *CFX-Solver Manager User's Guide*.

5.2.2. Bidirectional (Two-Way) FSI

In some simulations, there is a strong and potentially nonlinear relationship between the fields that are coupled in the Fluid Structure Interaction. Under these conditions, the ability to reach a converged solution will likely require the use of bidirectional FSI. As for unidirectional interaction, examples are provided below that demonstrate the variety of strategies to execute such simulations.

5.2.2.1. Using CFX Only

Conjugate heat transfer is an example of bidirectional interaction that can be solved using the CFX-Solver only.

5.2.2.2. Using CFX and Other CAE Software

Third-party code-coupling software or proprietary interfaces provided by the CAE software vendors can also be used in conjunction with CFX. Contact those software providers and your CFX service representative for more information.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 6: CFX Best Practices Guide for Numerical Accuracy

This guide provides best practice guidelines for Computational Fluid Dynamics (CFD) simulation and documentation of the verification, validation, and demonstration test cases. It describes:

- An Approach to Error Identification, Estimation and Validation (p. 119)
- Definition of Errors in CFD Simulations (p. 120)
- General Best Practice Guidelines (p. 129)
- Selection and Evaluation of Experimental Data (p. 140)

This guide is aimed at users who have moderate or little experience using Ansys CFX. It is part of a series that provides advice for using Ansys CFX in specific engineering application areas. The current guidelines are adapted from Best Practice Guidelines developed for the nuclear reactor safety applications [143 (p. 598)].

6.1. An Approach to Error Identification, Estimation and Validation

An evaluation of CFD capabilities has to ensure that the different types of errors are identified and, as far as possible, treated separately. It is known from single-phase studies that the quantification and documentation of modeling errors (as in turbulence models, for example) can be achieved only if the other major sources of errors are reduced below an "acceptable" level. In an ideal world, this would mean, among other demands, that solutions are provided for grids and with timesteps that are fine enough so that numerical errors can be neglected. This is not a trivial task and the separation of errors cannot always be achieved. These difficulties will be greatly increased by the inclusion of multi-phase physics and unsteady effects. Nevertheless, the worst strategy would be to avoid the subject and to provide solutions on a single grid, with a single timestep, and with other uncertainties in initial conditions and boundary conditions not evaluated. This would result in solutions that would be of little use for the validation goals.

An essential quantity in the quality assurance procedure is the definition of target variables. They will mainly be scalar (integral) quantities (for instance, forces, heat transfer rates, and maximum temperature) or one-dimensional distributions, such as the wall heat transfer along a certain line. Convergence studies can be based on these variables without a reference to the grid used in the simulation. They can also be used for an asymptotic evaluation of convergence on unstructured meshes. Even more important, these quantities are of immediate meaning to engineers and enable them to understand the uncertainty from a physical standpoint. A danger of integral or local scalar quantities is that they might not be sensitive enough to detect local changes in the solutions under grid refinement. This should be kept in mind during the analysis.

In order to tackle the problem, it is necessary to first define the different type of errors that can impact a CFD simulation. It is then required that you list the most promising strategies in order to reduce or

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

avoid these errors. Based on these strategies, procedures have to be defined that can be used for the test case simulations.

It might be not possible to rigorously perform the error estimation and reduction procedures described in the following sections for the complex demonstration cases. However, the best attempt should be made to follow the principal ideas and to avoid single grid solutions without sensitivity studies. For these cases, it is even more important to follow a stringent documentation procedure and to list the possible deficiencies and uncertainties in the simulations.

The strategies for the reduction and evaluation of numerical errors have been developed for single-phase flows. There is no principal difference between the single- and multi-phase flow formulations. They are both based on (ensemble) averaged equations, and are mathematically similar. From a physical standpoint, there are however significant additional challenges due to the presence of the different phases, besides the obviously higher demands on model formulation. One of the additional complication lies in the presence of sharp interfaces between the phases, which require a higher degree of grid resolution than usually necessary for single-phase flows. In addition, multi-phase flows have a higher affinity to physical instabilities that might be suppressed on coarse grids, but appear under grid refinement. (This effect is sometimes also observed in single-phase flows. An example is the blunt trailing edge of an airfoil, where extreme grid refinement will eventually capture the vortex shedding of the mixing layer). It is to be kept in mind that the brute application of procedures might not lead to the desired results. Also in these cases, the spirit behind the guidelines should be followed and carried as far as possible.

Validation studies have to be based on experimental data. These data can introduce significant errors into the comparison. It is therefore required to select the project test cases with attention to potential error sources and experimental uncertainties. Definitions on the different types of test cases as well as on the requirements for the project are given in Selection and Evaluation of Experimental Data (p. 140).

6.2. Definition of Errors in CFD Simulations

CFD simulations have the following potential sources for errors or uncertainties:

- Numerical Errors (p. 121)

  Numerical errors result from the differences between the exact equations and the discretized equations solved by the CFD code. For consistent discretization schemes, these errors can be reduced by an increased spatial grid density and/or by smaller timesteps.

- Modeling Errors (p. 127)

  Modeling errors result from the necessity to describe flow phenomena such as turbulence, combustion, and multi-phase flows by empirical models. For turbulent flows, the necessity for using empirical models derives from the excessive computational effort to solve the exact equations[1] with a *Direct Numerical Simulation* (DNS) approach. Turbulence models are therefore required to bridge the gap between the real flow and the statistically averaged equations. Other examples are combustion models and models for interpenetrating continua, for example, two-fluid models for two-phase flows.

- User Errors (p. 128)

[1] The Navier-Stokes equations for single-phase, Newtonian fluids

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

User errors result from incorrect use of CFD software and are usually a result of insufficient expertise by the CFD user. Errors can be reduced or avoided by additional training and experience in combination with high-quality project management and by provision and use of Best Practice Guidelines and associated checklists.

- Application Uncertainties (p. 128)

  Application uncertainties are related to insufficient information to define a CFD simulation. A typical example is insufficient information on the boundary conditions.

- Software Errors (p. 129).

  Software errors are the result of an inconsistency between the documented equations and the actual implementation in the CFD software. They are usually a result of programming errors.

A more detailed definition of the different errors follows.

6.2.1. Numerical Errors

Numerical Errors are of the following types:

- Solution Errors (p. 121)
- Spatial Discretization Errors (p. 122)
- Time Discretization Errors (p. 123)
- Iteration Errors (p. 124)
- Round-off Error (p. 124)
- Solution Error Estimation (p. 124)

6.2.1.1. Solution Errors

The most relevant errors from a practical standpoint are solution errors[2]. They are the difference between the exact solution of the model equations and the numerical solution. The relative solution error can be formally defined as:

$$E_s = \frac{f_{\text{exact}} - f_{\text{numeric}}}{f_{\text{exact}}} \tag{6.1}$$

Equation 6.1 (p. 121) is valid for every grid point for which the numerical solution exists. A global number can be defined by applying suitable norms, as:

$$\hat{E_s} = \frac{\|f_{\text{exact}} - f_{\text{numeric}}\|}{\|f_{\text{exact}}\|} \tag{6.2}$$

The goal of a numerical simulation is to reduce this error below an acceptable limit.

[2] Sometimes also called 'discretization errors'

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Obviously, this is not a straightforward task, as the exact solution is not known and the error can therefore not be computed. Exceptions are simple test cases for code verification where an analytical solution is available.

Given a grid spacing Δ, and the truncation error order of a consistent discretization scheme, p, a Taylor series can be written to express the exact solution as:

$$f_{\text{exact}} = f_{\text{numeric}} + c\,\Delta^{p} + \text{HOT} \tag{6.3}$$

In other words, the numerical solution converges towards the exact solution with the p^{th} power of the grid spacing. Analogous definitions are available for time discretization errors.

6.2.1.2. Spatial Discretization Errors

Spatial discretization errors are the result of replacing the analytical derivatives or integrals in the exact equations by numerical approximations that have a certain truncation error. The truncation error can be obtained by inserting a Taylor series expansion of the numerical solution into the different terms of the discretized equations:

$$f_{\text{numerical}} = f_{\text{exact}} + \sum_{i=1}^{\infty} c_i\, f^{(i)}\, \Delta^{i} \tag{6.4}$$

where $f^{(i)}$ is the i^{th} derivative of the exact solution at a given location. An example is a central difference for a spatial derivative:

$$\begin{aligned}
&\frac{\partial f}{\partial x} \approx \frac{f_{i+1} - f_{i-1}}{x_{i+1} - x_{i-1}} \\
&= \left(f_{\text{exact}} + f^{(1)}\Delta x + c_2 f^{(2)}\Delta x^2 + c_3 f^{(3)}\Delta x^3 + \text{HOT}\right)/(2\Delta x) \\
&- \left(f_{\text{exact}} - f^{(1)}\Delta x + c_2 f^{(2)}\Delta x^2 - c_3 f^{(3)}\Delta x^3 + \text{HOT}\right)/(2\Delta x) \\
&= f^{(1)} + o(\Delta x^2)
\end{aligned} \tag{6.5}$$

This formulation has a truncation error of order 2 and is therefore second-order accurate. The overall truncation error order of the spatial discretization scheme is determined by the lowest order truncation error after all terms have been discretized.

In the $o\left(\Delta x^2\right)$ term of Equation 6.5 (p. 122), the leading term is proportional to $f^{(3)}\,\Delta x^2$. First-order upwind differencing of the convective terms yields truncation errors $o\left(\Delta x\right)$ with leading term proportional to $f^{(2)}\,\Delta x$. This term then contributes to the diffusion term (numerical/false diffusion), which is most dangerous in 3D problems with grid lines not aligned to the flow direction. These schemes enhance the dissipation property of the numerical algorithm (see for example, Ferziger and Peric [141 (p. 598)]) and are not desirable in high-quality CFD simulations.

From a practical standpoint, it is important to understand that for a first-order method, the error is reduced to 50% by a doubling of the grid resolution in each spatial direction. For a second-order method, it is reduced to 25% for the same grid refinement.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6.2.1.3. Time Discretization Errors

Time adds another dimension to a CFD simulation. The definition of time discretization errors is therefore similar to the definition of the spatial discretization errors. The spatial discretization usually results in a system of nonlinear algebraic equations of the form:

$$\frac{\partial \varphi}{\partial t} = g\left(\varphi\right) \tag{6.6}$$

The error in the time discretization can again be obtained by a Taylor series expansion of the numerical formulation of this equation. With the example of a backward Euler integration:

$$\frac{\varphi^{n+1} - \varphi^{n}}{\Delta t} = g\left(\varphi^{n+1}\right) \tag{6.7}$$

the discretization error is:

$$\begin{aligned} &\frac{\varphi^{n+1} - \left(\varphi^{n+1} - \varphi^{(1)n+1}\Delta t + c_2 \varphi^{(2)n+1}\Delta t^2 + \mathrm{HOT}\right)}{\Delta t} \\ &= \frac{\partial \varphi^{n+1}}{\partial t} + c_2\, \varphi^{(2)n+1}\, \Delta t + \mathrm{HOT} \end{aligned} \tag{6.8}$$

The error is therefore first-order for the time derivative.

An additional complication for implicit methods comes from the inclusion of the unknown φ^{n+1} in the right hand side of Equation 6.7 (p. 123). In order to benefit from an implicit method, a linearization of g has to be included:

$$g\left(\varphi^{n+1}\right) = g\left(\varphi^{n}\right) + G\frac{\Delta \varphi}{\Delta t}\,\Delta t + o\left(\Delta t^{2}\right); \qquad \text{with} \qquad G = \frac{\partial g}{\partial \varphi} \tag{6.9}$$

The resulting discretized equation is therefore:

$$\left[\frac{1}{\Delta t} - G\right]\left(\varphi^{n+1} - \varphi^{n}\right) = g\left(\varphi^{n}\right) \tag{6.10}$$

This constitutes an implicit formulation with first-order accuracy. A second-order time differencing is not compatible with this linearization of the right hand side, as the linearization introduced a first-order error in Δt. In order to be able to satisfy the implicit dependency of the right hand side on the time level $n+1$ more closely, inner iterations (or coefficient loops) are frequently introduced:

$$\begin{aligned} &\frac{\varphi^{n+1,m+1} - \varphi^{n}}{\Delta t} = g\left(\varphi^{n+1,m+1}\right) \\ &= g\left(\varphi^{n+1,m}\right) + G\left(\varphi^{n+1,m+1} - \varphi^{n+1,m}\right) + o\left(\varphi^{n+1,m+1} - \varphi^{n+1,m}\right)^{2} \end{aligned} \tag{6.11}$$

where an additional iteration over the index m is carried out. This equation can be reformulated as:

$$\left[\frac{1}{\Delta t} - G\right]\left(\varphi^{n+1,m+1} - \varphi^{n+1,m}\right) = g\left(\varphi^{n+1,m}\right) - \frac{\varphi^{n+1,m} - \varphi^{n}}{\Delta t} \tag{6.12}$$

This equation can be converged completely (left hand side goes to zero) in m in order to solve the original exact implicit formulation given by Equation 6.7 (p. 123). It is obvious that it is not necessary to converge the coefficient loop to zero, while the right hand side has a finite (first-order) error in Δt. It can be shown that for a first-order time integration, one coefficient loop is consistent with the accuracy of the method. In a case where a second-order accurate scheme is used in the time derivative, two coefficient loops will ensure overall second-order accuracy of the method. Note, however, that this is correct only if the coefficient loops are not under-relaxed in any way.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For explicit methods, no coefficient loops are required and the time discretization error is defined solely from a Taylor series expansion.

6.2.1.4. Iteration Errors

The iteration error is similar to the coefficient loop error described above. It occurs in a case where a steady-state solution is sought from an iterative method. In most CFD codes, the iteration is carried out via a (pseudo-) timestepping scheme as given in this example, which also appears above:

$$\left[\frac{1}{\Delta t} - G\right]\left(\varphi^{n+1} - \varphi^{n}\right) = g\left(\varphi^{n}\right) \tag{6.13}$$

Zero iteration error would mean that the left hand side is converged to zero, leading to the converged solution $g\left(\varphi\right)=0$. However, in practical situations, the iterative process is stopped at a certain level, in order to reduce the numerical effort. The difference between this solution and the fully converged solution defines the iteration error.

The iteration error is usually quantified in terms of a residual or a residual norm. This can be the maximum absolute value of the right hand side, $g\left(\varphi\right)$, for all grid points, or a root mean square of this quantity. In most CFD methods, the residual is nondimensionalized to enable a comparison between different applications with different scaling. However, the non-dimensionalization is different for different CFD codes, making general statements as to the required absolute level of residuals impractical. Typically, the quality of a solution is measured by the overall reduction in the residual, compared to the level at the start of the simulation.

The iteration error should be controlled with the use of the target variables. The value of the target variable can be plotted as a function of the convergence level. In case of iterative convergence, the target variable should remain constant with the convergence level. It is desirable to display the target variable in the solver monitor during the simulation.

6.2.1.5. Round-off Error

Another numerical error is the round-off error. It results from the fact that a computer only solves the equations with a finite number of digits (around 8 for single-precision and around 16 for double-precision). Due to the limited number of digits, the computer cannot differentiate between numbers that are different by an amount below the available accuracy. For flow simulations with large-scale differences (for instance, extent of the domain vs. cell size), this can be a problem for single-precision simulations. Round-off errors are often characterized by a random behavior of the numerical solution.

6.2.1.6. Solution Error Estimation

The most practical method to obtain estimates for the solution error is systematic grid refinement or timestep reduction. In the following, the equations for error estimation are given for grid refinement. The same process can be used for timestep refinement.

If the asymptotic range of the convergence properties of the numerical method is reached, the difference between solutions on successively refined grids can be used as an error estimator. This allows the application of Richardson extrapolation to the solutions on the different grids (Roache [139 (p. 597)]). In the asymptotic limit, the solution can be written as follows:

$$f_{\text{exact}} = f_i + g_1 h_i + g_2 h_i^2 + \ldots \tag{6.14}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In this formulation, h is the grid spacing (or a linear measure of it) and the g_i are functions independent of the grid spacing. The subscript, i, refers to the current level of grid resolution. Solutions on different grids are represented by different subscripts.

The assumption for the derivation of an error estimate is that the order of the numerical discretization is known. This is usually the case. Assuming a second-order accurate method, the above expansion can be written for two different grids:

$$f_{\text{exact}} = f_1 + g_2 h_1^2 + \ldots \tag{6.15}$$

$$f_{\text{exact}} = f_2 + g_2 h_2^2 + \ldots$$

Neglecting higher-order terms, the unknown function g_2 can be eliminated from this equation. An estimate for the exact solution is therefore:

$$f_{\text{exact}} = \frac{h_2^2 f_1 - h_1^2 f_2}{h_2^2 - h_1^2} + \text{HOT} \tag{6.16}$$

The difference between the fine grid solution and the exact solution (defining the error) is therefore:

$$E = f_{\text{exact}} - f_1 = \frac{f_1 - f_2}{r^2 - 1} + \text{HOT} ; \qquad \text{with} \qquad r = \frac{h_2}{h_1} \tag{6.17}$$

For an arbitrary order of accuracy, p, of the underlying numerical scheme, the error is given by:

$$E = f_{\text{exact}} - f_1 = \frac{f_1 - f_2}{r^p - 1} + \text{HOT} \tag{6.18}$$

In order to build the difference between the solutions f_1 and f_2, it is required that the coarse and the fine grid solution is available at the same location. In the case of a doubling of the grid density without a movement of the coarse grid nodes, all information is available on the coarse grid nodes. The application of the correction to the fine-grid solution requires an interpolation of the correction to the fine grid nodes (Roache [139 (p. 597)]). In the case of a general grid refinement, the solutions are not available on the same physical locations. An interpolation of the solution between the different grids is then required for a direct error estimate. It has to be ensured that the interpolation error is lower than the solution error in order to avoid a contamination of the estimate.

Richardson interpolation can also be applied to integral quantities (target variables), such as lift or drag coefficients. In this case, no interpolation of the solution between grids is required.

Note that the above derivation is valid only if the underlying method has the same order of accuracy everywhere in the domain and if the coarse grid is already in the asymptotic range (the error decreases with the order of the numerical method). In addition, the method magnifies round-off and iteration errors.

The intention of the Richardson interpolation was originally to improve the solution on the fine grid. This requires an interpolation of the correction to the fine grid and introduces additional inaccuracies into the extrapolated solution, such as errors in the conservation properties of the solution. A more practical use of the Richardson extrapolation is the determination of the relative solution error, A_1:

$$A_1 = \frac{f_1 - f_{\text{exact}}}{f_{\text{exact}}} \tag{6.19}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

An estimate, E_1, of this quantity can be derived from Equation 6.16 (p. 125):

$$A_1 = \frac{f_2 - f_1}{f_1} \frac{1}{r^p - 1} \tag{6.20}$$

It can be shown (Roache [139 (p. 597)]) that the exact relative error and the approximation are related by:

$$A_1 = E_1 + O\left(h^{p+1}\right) \tag{6.21}$$

Equation 6.20 (p. 126) can also be divided by the range of f_1 or another suitable quantity in order to prevent the error to become infinite as f_1 goes to zero.

In order to arrive at a practical error estimator, the following definitions are proposed:

Field error:

$$A_f = \frac{|f_2 - f_1|}{\text{range}\left(f_1\right)} \frac{1}{\left(r^p - 1\right)} \tag{6.22}$$

Maximum error:

$$A_{\max} = \frac{\max\left(f_2 - f_1\right)}{\text{range}\left(f_1\right)} \frac{1}{\left(r^p - 1\right)} \tag{6.23}$$

RMS error:

$$A_{\text{rms}} = \frac{\text{rms}\left(f_2 - f_1\right)}{\text{range}\left(f_1\right)} \frac{1}{\left(r^p - 1\right)} \tag{6.24}$$

Target variable error:

$$A_{\text{rms}} = \frac{|\Theta_1 - \Theta_2|}{|\Theta_1|} \frac{1}{\left(r^p - 1\right)} \tag{6.25}$$

where Θ is the defined target variable (list, drag, heat transfer coefficient, maximum temperature, mass flow, and so on).

Similar error measures can be defined for derived variables, which can be specified for each test case. Typical examples would be the total mass flow, the pressure drop, or the overall heat transfer. This will be the recommended strategy, as it avoids the interpolation of solutions between the coarse and the fine grid.

For unstructured meshes, the above considerations are valid only in cases of a global refinement of the mesh. Otherwise, the solution error will not be reduced continuously across the domain. For unstructured refinement the refinement level, r, can be defined as follows:

$$r_{\text{effective}} = \left(\frac{N_1}{N_2}\right)^{1/D} \tag{6.26}$$

where N_i is the number of grid points and D is the dimension of the problem.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

It must be emphasized that these definitions do not impose an upper limit on the real error, but are estimates for the evaluation of the quality of the numerical results. Limitations of the above error estimates are:

- The solution has to be smooth
- The truncation error order of the method has to be known
- The solution has to be sufficiently converged in the iteration domain
- The coarse grid solution has to be in the asymptotic range.

For three-dimensional simulations, the demand that the coarse grid solution be in the asymptotic range is often hard to ensure. It is therefore required to compute the error for three different grid levels, to avoid fortuitous results. If the solution is in the asymptotic range, the following indicator should be close to constant:

$$E_h = \frac{\text{error}}{h^p} \tag{6.27}$$

6.2.2. Modeling Errors

In industrial CFD methods, numerous physical and chemical models are incorporated. Models are usually applied to avoid the resolution of a large range of scales, which would result in excessive computing requirements.

The classical model used in almost all industrial CFD applications is a turbulence model. It is based on time or ensemble averaging of the equations resulting in the so-called *Reynolds Averaged Navier-Stokes* (RANS) equations. Due to the averaging procedure, information from the full Navier-Stokes equations is lost. It is supplied back into the code by the turbulence model. The most widely used industrial models are two-equation models, such as the $\mathrm{k}-\varepsilon$ or $\mathrm{k}-\omega$ models.

The statistical model approach reduces the resolution requirements in time and space by many orders of magnitude, but requires the calibration of model coefficients for certain classes of flows against experimental data. There is a wide variety of models that are introduced to reduce the resolution requirements for CFD simulations, including:

- Turbulence models
- Multi-phase models
- Combustion models
- Radiation models.

In combustion models, the reduction can be both in terms of the chemical species and in terms of the turbulence-combustion interaction. In radiation, the reduction is typically in terms of the wavelength and/or the directional information. For multi-phase flows, it is usually not possible to resolve a large number of individual bubbles or droplets. In this case, the equations are averaged over the different phases to produce continuous distributions for each phase in space and time.

As all of these models are based on a reduction of the 'real' physics to a reduced 'resolution', information has to be introduced from outside the original equations. This is usually achieved by experimental calibration, or by available DNS (p. 120) results.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Once a model has been selected, the accuracy of the simulation cannot be increased beyond the capabilities of the model. This is the largest factor of uncertainty in CFD methods, as modeling errors can be of the order of 100% or more. These large errors occur in cases where the CFD solution is very sensitive to the model assumptions and where a model is applied outside its range of calibration.

Because of the complexity of industrial simulations, it cannot be ensured that the models available in a given CFD code are suitable for a new application. While in most industrial codes a number of different models are available, there is no *a priori* criterion as to the selection of the most appropriate one. Successful model selection is largely based on the expertise and the knowledge of the CFD user.

6.2.3. User Errors

User errors result from the inadequate use of the resources available for a CFD simulation. The resources are given by:

- Problem description
- Computing power
- CFD software
- Physical models in the software
- Project time frame.

According to the *ERCOFTAC Best Practice Guidelines* [140 (p. 598)], some of the sources for user errors are:

- Lack of experience
- Lack of attention to detail or other mistakes.

Often, user errors are related to management errors when insufficient resources are assigned to a project, or inexperienced users are given a too complex application. Typical user errors are:

- Oversimplification of a given problem; for example, geometry, equation system, and so on
- Poor geometry and grid generation
- Use of incorrect boundary conditions
- Selection of non-optimal physical models
- Incorrect or inadequate solver parameters; for example, timestep, and so on
- Acceptance of non-converged solutions
- Postprocessing errors.

6.2.4. Application Uncertainties

Application uncertainties result from insufficient knowledge to carry out the simulation. This is in most cases a lack of information on the boundary conditions or of the details of the geometry. A

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

typical example is the lack of detailed information at the inlet. A complete set of inlet boundary conditions is composed of inflow profiles for all transported variables (momentum, energy, turbulence intensity, turbulence length scale, volume fractions, and so on). This information can be supplied from experiments or from a CFD simulation of the upstream flow. In most industrial applications, this information is not known and bulk values are given instead. In some cases, the detailed information can be obtained from a separate CFD simulation (for instance a fully developed pipe inlet flow). In other cases, the boundaries can be moved far enough away from the area of interest to minimize the influence of the required assumptions for the complete specification of the boundary conditions.

Typical application uncertainties are:

- Lack of boundary condition information
- Insufficient information on the geometry
- Uncertainty in experimental data for solution evaluation.

6.2.5. Software Errors

Software errors are defined as any inconsistency in the software package. This includes the code, its documentation, and the technical service support. Software errors occur when the information you have on the equations to be solved by the software is different from the actual equations solved by the code. This difference can be a result of:

- Coding errors (bugs)
- Errors in the graphical user interface (GUI)
- Documentation errors
- Incorrect support information.

6.3. General Best Practice Guidelines

In order to reduce the numerical errors, it is necessary to have procedures for the estimation of the different errors described in Definition of Errors in CFD Simulations (p. 120). The main goal is to reduce the solution error to a minimum with given computer resources.

6.3.1. Avoiding User Errors

User errors are directly related to the expertise, the thoroughness, and the experience of the user. For a given user, these errors can only be minimized by good project management and thorough interaction with others. In case of inexperienced users, day-to-day interaction with a CFD expert/manager is required to avoid major quality problems. A structured work plan with intermediate results is important for intermediate and long-term projects.

A careful study of the CFD code documentation and other literature on the numerical methods as well as the physical models is highly recommended. Furthermore, benchmark studies are recommended to enable you to understand the capabilities and limitations of CFD methods. A comparison of different CFD methods is desirable, but not always possible.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6.3.2. Geometry Generation

Before the grid generation can start, the geometry has to be created or imported from CAD-data. In both cases, attention should be given to:

- The use of a correct coordinate system
- The use of the correct units
- The use of geometrical simplification, for example, symmetry planes
- Local details. In general, geometrical features with dimensions below the local mesh size (for example, wall roughness or porous elements) are not included in the geometrical model. These should be incorporated through a suitable model.

In the case that the geometry is imported from CAD-data, the data should be checked beforehand. Frequently, after the import of CAD-data, the CAD-data has to be adapted (cleaned) before it can be used for mesh generation. It is essential for mesh generation to have closed volumes. The various CAD-data formats do not always contain these closed volumes. Therefore, the CAD-data has to be altered in order to create the closed volumes. It has to be ensured that these changes do not influence the flow to be computed.

6.3.3. Grid Generation

In a CFD analysis, the flow domain is subdivided in a large number of computational cells. All these computational cells together form the so-called mesh or grid. The number of cells in the mesh should be taken sufficiently large, such that an adequate resolution is obtained for the representation of the geometry of the flow domain and the expected flow phenomena in this domain.

A good mesh quality is essential for performing a good CFD analysis. Therefore, assessment of the mesh quality before performing a large and complex CFD analysis is very important. Most of the mesh generators and CFD solvers offer the possibility of checking the mesh on several cells or mesh parameters, such as aspect ratio, internal angle, face warpage, right handiness, negative volumes, cracks, and tetrahedral quality. The reader is referred to the user's guides of the various mesh generators and CFD solvers for more information on these cells and mesh parameters.

Recommendations for grid generation are:

- Avoid high grid stretching ratios.
  - Aspect ratios should not be larger than 20 to 50 in regions away from the boundary.
  - Aspect ratios may be larger in unimportant regions.
  - Aspect ratios may, and should, be larger in the boundary layers. For well resolved boundary layers at high Re numbers, the near-wall aspect ratios can be of the order of 10^3 for single-precision runs or 10^4 for double-precision runs.
- Avoid jumps in grid density.
  - Growth factors should be smaller than 1.3.
- Avoid poor grid angles.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Avoid non-scalable grid topologies. Non-scalable topologies can occur in block-structured grids and are characterized by a deterioration of grid quality under grid refinement.
- Avoid non-orthogonal, for example, unstructured tetrahedral meshes, in (thin) boundary layers.
- Use a finer and more regular grid in critical regions, for example, regions with high gradients or large changes such as shocks.
- Avoid the presence of arbitrary grid interfaces, mesh refinements, or changes in element types in critical regions. An arbitrary grid interface occurs when there is no one-to-one correspondence between the cell faces on both sides of a common interface, between adjacent mesh parts.

If possible, determine the size of the cells adjacent to wall boundaries where turbulence models are used, before grid generation has started.

Numerical diffusion is high when computational cells are created that are not orthogonal to the fluid flow. If possible, avoid computational cells that are not orthogonal to the fluid flow.

Judge the mesh quality by using the possibilities offered by the mesh generator. Most mesh generators offer checks on mesh parameters, such as aspect ratio, internal angle, face warpage, right handiness, negative volumes, cracks, and tetrahedral quality.

It should be demonstrated that the final result of the calculations is independent of the grid that is used. This is usually done by comparison of the results of calculations on grids with different grid sizes.

Some CFD methods enable the application of grid adaptation procedures. In these methods, the grid is refined in critical regions (high truncation errors, large solution gradients, and so on). In these methods, the selection of appropriate indicator functions for the adaptation is essential for the success of the simulations. They should be based on the most important flow features to be computed.

As a general rule, any important shear layer in the flow (boundary layer, mixing layer, free jets, wakes, and so on) should be resolved with at least 10 nodes normal to the layer. This is a very challenging requirement that often requires the use of grids that are aligned with the shear layers.

6.3.4. Model Selection and Application

Modeling errors are the most difficult errors to avoid, as they cannot be reduced systematically. The most important factor for the reduction of modeling errors is the quality of the models available in the CFD package and the experience of the user. There is also a strong interaction between modeling errors and the time and space resolution of the grid. The resolution has to be sufficient for the model selected for the application.

In principle, modeling errors can only be estimated in cases where the validation of the model is 'close' to the intended application. Model validation is essential for the level of confidence you can have in a CFD simulation. It is therefore required that you gather all available information on the validation of the selected model, both from the open literature and from the code developers (vendors). In case that CFD is to be applied to a new field, it is recommended that you carry out additional validation studies, in order to gain confidence that the physical models are adequate for the intended simulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

If several modeling options are available in the code (as is usually the case for turbulence, combustion and multi-phase flow models), it is recommended that you carry out the simulation with different models in order to test the sensitivity of the application with respect to the model selection.

In case you have personal access to a modeling expert in the required area, it is recommended that you interact with the model developer or expert to ensure the optimal selection and use of the model.

6.3.4.1. Turbulence Models

There are different methods for the treatment of turbulent flows. The need for a model results from the inability of CFD simulations to fully resolve all time and length scales of a turbulent motion. In classical CFD methods, the Navier-Stokes equations are usually time- or ensemble-averaged, reducing the resolution requirements by many orders of magnitude. The resulting equations are the RANS (p. 127) equations. Due to the averaging procedure, information is lost, which is then fed back into the equations by a turbulence model.

The amount of information that has to be provided by the turbulence model can be reduced if the large time and length scales of the turbulent motion are resolved. The equations for this so-called *Large Eddy Simulation* (LES) method are usually filtered over the grid size of the computational cells. All scales smaller than the resolution of the mesh are modeled and all scales larger than the cells are computed. This approach is several orders of magnitude more expensive than a RANS (p. 127) simulation and is therefore not used routinely in industrial flow simulations. It is most appropriate for free shear flows, as the length scales near the solid walls are usually very small and require small cells even for the LES (p. 132) method.

RANS (p. 127) methods are the most widely used approach for CFD simulations of industrial flows. Early methods, using algebraic formulations, have been largely replaced by more general transport equation models, for both implementation and accuracy considerations. The use of algebraic models is not recommended for general flow simulations, due to their limitations in generality and their geometric restrictions. The lowest level of turbulence models that offer sufficient generality and flexibility are two-equation models. They are based on the description of the dominant length and time scale by two independent variables. Models that are more complex have been developed and offer more general platforms for the inclusion of physical effects. The most complex RANS (p. 127) model used in industrial CFD applications are *Second Moment Closure* (SMC) models. Instead of two equations for the two main turbulent scales, this approach requires the solution of seven transport equations for the independent Reynolds stresses and one length (or related) scale.

The challenge for the user of a CFD method is to select the optimal model for the application at hand from the models available in the CFD method. In most cases, it cannot be specified beforehand which model will offer the highest accuracy. However, there are indications as to the range of applicability of different turbulence closures. This information can be obtained from validation studies carried out with the model.

In addition to the accuracy of the model, consideration has to be given to its numerical properties and the required computer power. It is often observed that more complex models are less robust and require many times more computing power than the additional number of equations would indicate. Frequently, the complex models cannot be converged at all, or, in the worst case, the code becomes unstable and the solution is lost.

It is not trivial to provide general rules and recommendations for the selection and use of turbulence models for complex applications. Different CFD groups have given preference to different models

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

for historical reasons or personal experiences. Even turbulence experts cannot always agree as to which model offers the best cost-performance ratio for a new application.

6.3.4.1.1. One-equation Models

A number of one-equation turbulence models based on an equation for the eddy viscosity have been developed over the last years. Typical applications are:

- Airplane- and wing flows
- External automobile aerodynamics
- Flow around ships.

These models have typically been optimized for aerodynamic flows and are not recommended as general-purpose models.

6.3.4.1.2. Two-equation Models

The two-equation models are the main-stand of industrial CFD simulations. They offer a good compromise between complexity, accuracy and robustness. The most popular models are the standard model and different versions of the $k-\omega$ model, see Wilcox [30 (p. 581)]. The standard $k-\omega$ model of Wilcox is the most well known of the $k-\omega$ based models, but shows a severe free-stream dependency. It is therefore not recommended for general industrial flow simulations, as the results are strongly dependent on the user input. Alternative formulations are available, see for example, the **Shear Stress Transport** (SST) model, Menter [9 (p. 578)].

An important weakness of standard two-equation models is that they are insensitive to streamline curvature and system rotation. Particularly for swirling flows, this can lead to an over-prediction of turbulent mixing and to a strong decay of the core vortex. There are curvature correction models available, but they have not been generally validated for complex flows.

The standard two-equation models can also exhibit a strong build-up of turbulence in stagnation regions, due to their modeling of the production terms. Several modifications are available to reduce this effect, for instance by Kato and Launder [128 (p. 596)]. They should be used for flows around rods, blades, airfoils, and so on.

6.3.4.1.3. Second Moment Closure (SMC) Models

SMC (p. 132) models are based on the solution of a transport equation for each of the independent Reynolds stresses in combination with the ε- or the ω-equation. These models offer generally a wider modeling platform and account for certain effects due to their exact form of the turbulent production terms. Some of these models show the proper sensitivity to swirl and system rotation, which have to be modeled explicitly in a two-equation framework. SMC (p. 132) models are also superior for flows in stagnation regions, where no additional modifications are required.

One of the weak points of the SMC (p. 132) closure is that the same scale equations are used as in the two-equation framework. As the scale equation is typically one of the main sources of uncertainty, it is found that SMC (p. 132) models do not consistently produce superior results compared to the simpler models. In addition, experience has shown that SMC (p. 132) models are often much harder to handle numerically. The model can introduce a strong nonlinearity into the CFD method, leading to numerical problems in many applications.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

SMC (p. 132) models are usually not started from a pre-specified initial condition, but from an already available solution from a two-equation (or simpler) model. This reduces some of the numerical problems of the SMC (p. 132) approach. In addition, it offers an important sensitivity study, as it allows quantifying the influence of the turbulence model on the solution. It is therefore recommended that you fully converge the two-equation model solution and save it for a comparison with the SMC (p. 132) model solution. The difference between the solutions is a measure of the influence of the turbulence model and therefore an indication of the modeling uncertainty. This is possible only in steady-state simulations. For unsteady flows, the models usually have to be started from the initial condition.

6.3.4.1.4. Large Eddy Simulation Models

LES (p. 132) models are based on the numerical resolution of the large turbulence scales and the modeling of the small scales. LES (p. 132) is not yet a widely used industrial approach, due to the large cost of the required unsteady simulations. For certain classes of applications, LES (p. 132) will be applicable in the near future. The most appropriate area will be free shear flows, where the large scales are of the order of the solution domain (or only an order of magnitude smaller). For boundary layer flows, the resolution requirements are much higher, as the near-wall turbulent length scales become much smaller. The internal flows (pipe flows, channel flows) are in between, as they have a restricted domain in the wall normal direction, but small scales have to be resolved in the other two directions.

LES (p. 132) simulations do not easily lend themselves to the application of grid refinement studies both in the time and the space domain. The main reason is that the turbulence model adjusts itself to the resolution of the grid. Two simulations on different grids are therefore not comparable by asymptotic expansion, as they are based on different levels of the eddy viscosity and therefore on a different resolution of the turbulent scales. From a theoretical standpoint, the problem can be avoided, if the LES (p. 132) model is not based on the grid spacing, but on a pre-specified filter-width. This would enable reaching grid-independent LES (p. 132) solutions above the DNS (p. 120) limit. However, LES (p. 132) is a very expensive method and systematic grid and timestep studies are prohibitive even for a pre-specified filter. It is one of the disturbing facts that LES (p. 132) does not lend itself naturally to quality assurance using classical methods. This property of the LES (p. 132) also indicates that (nonlinear) multigrid methods of convergence acceleration are not suitable in this application.

On a more global level, the grid convergence can be tested using averaged quantities resulting from the LES (p. 132) simulation. The averaged LES (p. 132) results can be analyzed in a similar way as RANS (p. 127) solutions (at least qualitatively). Again, it is expensive to perform several LES (p. 132) simulations and grid refinement will therefore be more the exception than the rule.

Due to the high computing requirements of LES (p. 132), modern developments in the turbulence models focus on a combination of RANS (p. 127) and LES (p. 132) models. The goal is to cover the wall boundary layers with RANS (p. 127) and to enable unsteady (LES (p. 132)-like) solutions in largely separated and unsteady flow regions (for example, flow behind a building, or other blunt bodies). There are two alternatives of such methods available in Ansys CFX.

The first alternative is called *Scale-Adaptive-Simulation* (SAS) model (Menter and Egorov [130 (p. 596)], [131 (p. 596)], [144 (p. 598)], [145 (p. 598)]). It is essentially an improved *Unsteady* RANS (URANS) method that develops LES (p. 132)-like solutions in unstable flow regimes.

The second alternative is called *Detached Eddy Simulation* (DES) (Spalart [146 (p. 599)]), implemented in the version of Strelets [58 (p. 586)]. The current recommendation is to use the SAS (p. 134)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

model, as it has less grid sensitivity than the DES (p. 134) formulation. In case that SAS (p. 134) does not provide an unsteady solution, the DES (p. 134) model should be applied. It should be noted that both model formulations require small timesteps with a Courant number of CFL<1. You are encouraged to read the original references before applying these models.

6.3.4.1.5. Wall Boundary Conditions

There are generally three types of boundary conditions that can be applied to a RANS (p. 127) simulation:

- Wall Function Boundary Conditions (p. 135)
- Integration to the wall (low-Reynolds number formulation) (p. 135)
- Mixed formulation (automatic near-wall treatment) (p. 136).

6.3.4.1.5.1. Wall Function Boundary Conditions

Standard wall functions are based on the assumption that the first grid point off the wall (or the first integration point) is located in the universal law-of-the-wall or logarithmic region. Wall functions eliminate the need to resolve the very thin viscous sublayer, leading to a reduction in the number of cells and to a more moderate (and desirable) aspect ratio of the cells (ratio of the longest to the smallest side in a structured grid). High aspect ratios can result in numerical problems due to round-off errors.

On the other hand, standard wall function formulations are difficult to handle, because you have to ensure that the grid resolution near the wall satisfies the wall function requirements. If the grid becomes too coarse, the resolution of the boundary layer is no longer ensured. If the resolution becomes too fine, the first grid spacing can be too small to bridge the viscous sublayer. In this case, the logarithmic profile assumptions are no longer satisfied. You have to ensure that both limits are not overstepped in the grid generation phase.

The lower limit on the grid resolution for standard wall functions is a severe detriment to a systematic grid refinement process, as required by the best practice approach. That is, instead of an improved accuracy of the solution with grid refinement, the solution will deteriorate from a certain level on, leading eventually to a singularity of the numerical method. Standard wall functions are therefore not recommended for systematic grid refinement studies. Recently, alternative formulations (scalable wall functions) have become available, Menter and Esch [142 (p. 598)], which enable a systematic grid refinement when using wall functions.

6.3.4.1.5.2. Integration to the wall (low-Reynolds number formulation)

The use of low-Reynolds (low-Re) number formulations of turbulence models for the integration of the equations through the viscous sublayer is generally more accurate, as no additional assumptions are required concerning the variation of the variables near the wall. On the downside, most low-Re extensions of turbulence models are quite complex and can reduce the numerical performance or even destabilize the numerical method. In addition, classical low-Re models require a very fine near-wall resolution of $y^+ \sim 1$ at all wall nodes. This is very hard to ensure for all walls of a complex industrial application. In the case that significantly coarser grids are used, the wall shear stress and the wall heat transfer can be reduced significantly below their correct values.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6.3.4.1.5.3. Mixed formulation (automatic near-wall treatment)

In Ansys CFX, hybrid methods are available for all ω-equation based turbulence models (automatic near-wall treatment), which automatically switch from a low-Re formulation to wall functions based on the grid spacing you provide. These formulations provide the optimal boundary condition for a given grid. From a best practice standpoint, they are the most desirable, as they enable an accurate near-wall treatment over a wide range of grid spacings. However, accurate boundary layer simulations do not depend only on the y^+ near-wall spacing, but also require a minimum of at least 10 grid nodes inside the boundary layer.

6.3.4.1.5.4. Recommendations for Model Selection

- Avoid the use of classical wall functions, as they are inconsistent with grid refinement.
- Avoid strict low-Re number formulations, unless it is ensured that all near-wall cells are within the resolution requirements of the formulation.
- In combination with the $\mathrm{k}-\varepsilon$ model, use scalable wall functions. They can be applied to a range of grids without immediate deterioration of the solution (default in Ansys CFX).
- For more accurate simulations, use an automatic wall treatment in combination with SST (p. 133) turbulence model (default in Ansys CFX).

6.3.4.2. Heat Transfer Models

The heat transfer formulation is strongly linked to the underlying turbulence model. For eddy viscosity models, the heat transfer simulation is generally based on the analogy between heat and momentum transfer. Given the eddy viscosity of the two-equation model, the heat transfer prediction is based on the introduction of a molecular and a turbulent Prandtl number. The treatment of the energy equation is therefore similar to the treatment of the momentum equations. No additional transport equations are required for the turbulent heat transfer prediction. The boundary conditions are the same as for the momentum equations and follow the same recommendations.

For SMC (p. 132) models, three additional transport equations must be solved for the turbulent heat transfer vector in order to be consistent with the overall closure level. Only a few CFD methods offer this option. In most cases, the heat transfer is computed from an eddy diffusivity with a constant turbulent Prandtl number.

6.3.4.3. Multi-Phase Models

Multi-phase models are required in cases where more than one phase is involved in the simulation (phases can also be non-miscible fluids). There is a wide variety of multi-phase flow scenarios, with the two extremes of small-scale mixing of the phases or a total separation of the phases by a sharp interface.

Depending on the flow simulation, different types of models are available. The main distinction of the models is given below.

Lagrange models solve a separate equation for individual particles, bubbles, or droplets in a surrounding fluid. The method integrates the three-dimensional trajectories of the particles based on the forces acting on them from the surrounding fluid and other sources. Turbulence is usually accounted for by a random motion, superimposed on the trajectory.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Lagrange models are usually applied to flows with low particle (bubble) densities. In these flows, the interaction between the particles can usually be neglected, thereby reducing the complexity of the problem significantly.

The Euler-Euler formulation is based on the assumption of interpenetrating continua. A separate set of mass, momentum, and energy conservation equations is solved for each phase. Interphase transfer terms have to be modeled to account for the interaction of the phases. Euler-Euler methods can be applied to separated and dispersed flows by changing the interface transfer model.

Additional models are required for flows with mass transfer between the phases (condensation, evaporation, boiling). These models can be applied in the form of correlations for a large number of particles (bubbles) in a given control volume, or directly at the interface between the resolved phase boundary.

6.3.5. Reduction of Application Uncertainties

Application uncertainties cannot always be avoided because the missing information can frequently not be recovered. The uncertainty can be minimized by interaction with the supplier of the test case. The potential uncertainties have to be documented before the start of the CFD application.

In the case that the assumptions have to be made concerning any input to a CFD analysis, they have to be communicated to the partners in the project. Alternative assumptions have to be proposed and the sensitivity of the solution to these assumptions has to be evaluated by case studies (alteration of inflow profiles, different locations for arbitrary boundary conditions, and so on).

Recommendations are:

- Identify all uncertainties in the numerical setup:
  - Geometry reduction
  - Boundary condition assumptions
  - Arbitrary modeling assumptions, for example, bubble diameter, and so on.
- Perform a sensitivity analysis with at least two settings for each arbitrary parameter.
- Document the sensitivity of the solution on the assumptions.

6.3.6. CFD Simulation

This section provides recommendations concerning the optimal application of a CFD method, once the grids are available and the basic physical models have been defined.

6.3.6.1. Target Variables

In order to monitor numerical errors, it is recommended that you define target variables. The convergence of the numerical scheme can then be checked more easily and without interpolation between different grids. You should select target variables that:

1. Are representative of the goals of the simulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2. Are sensitive to numerical treatment and resolution.

   This criteria should help to avoid the use of measures that are insensitive to the resolution, such as pressure-based variables in boundary layer simulations.

3. Can be computed with existing postprocessing tools.

4. Can be computed inside the solver and displayed during run time (optimal).

It is optimal if the variable can be computed during run time and displayed as part of the convergence history. This enables you to follow the development of the target variable during the iterative process.

6.3.6.2. Minimizing Iteration Errors

A first indication of the convergence of the solution to steady-state is the reduction in the residuals. Experience shows, however, that different types of flows require different levels of residual reduction. For example, it is found regularly that swirling flows can exhibit significant changes even if the residuals are reduced by more than 5 - 6 orders of magnitude. Other flows are well converged with a reduction of only 3 - 4 orders.

In addition to the residual reduction, it is therefore required to monitor the solution during convergence and to plot the pre-defined target quantities of the simulation as a function of the residual (or the iteration number). A visual observation of the solution at different levels of convergence is recommended.

It is also recommended that you monitor the global balances of conserved variables, such as mass, momentum and energy, vs. the iteration number.

Convergence is therefore monitored and ensured by the following steps:

- Reduce residuals by a pre-specified level and provide residual plots.
- Plot evolution of r.m.s. and maximum residual with iteration number.
- Report global mass balance with iteration number.
- Plot target variables as a function of iteration number or residual level.
- Report target variables as a function of r.m.s. residual (table).

It is desirable to have the target variable written out at every timestep in order to display it during the simulation run.

Depending on the numerical scheme, the recommendations may also be relevant to the iterative convergence within the timestep loop for transient simulations.

6.3.6.3. Minimizing Spatial Discretization Errors

Spatial discretization errors result from the numerical order of accuracy of the discretization scheme and from the grid spacing. It is well known that only second- and higher-order space discretization methods are able to produce high quality solutions on realistic grids. First-order methods should therefore be avoided for high quality CFD simulations.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

As the order of the scheme is usually given (mostly second-order), spatial discretization errors can be influenced only by the provision of an optimal grid. It is important for the quality of the solution and the applicability of the error estimation procedures defined in Solution Error Estimation (p. 124), that the coarse grid already resolves the main features of the flow. This requires that the grid points are concentrated in areas of large solution variation. For the reduction of spatial discretization errors, it is also important to provide a high-quality numerical grid.

For grid convergence tests, the simulations are carried out for a minimum of three grids. The target quantities will be given as a function of the grid density. In addition, an error estimate based on the definition given in Solution Error Estimation (p. 124) (Equation 6.25 (p. 126)) will be carried out. It is also recommended that you compute the quantity given by Equation 6.27 (p. 127) to test the assumption of asymptotic convergence.

It is further recommended that the graphical comparison between the experiments and the simulations show the grid influence for some selected examples. The following information should be provided:

- Define target variable as given in Target Variables (p. 137).
- Provide three (or more) grids using the same topology (or for unstructured meshes, a uniform refinement over all cells).
- Compute solution on these grids:
  - Ensure convergence of the target variable in the time- or iteration domain. See Iteration Errors (p. 124) and Minimizing Iteration Errors (p. 138).
  - Compute target variables for these solutions.
- Compute and report error measure for target variable(s) based on Equation 6.25 (p. 126).
- Plot selected variables for the different grids in one picture.
- Check if the solution is in the asymptotic range using Equation 6.27 (p. 127).

6.3.6.4. Minimizing Time Discretization Errors

In order to reduce time integration errors for unsteady-state simulations, it is recommended that you use at least a second-order-accurate time-discretization scheme. Usually, the relevant frequencies can be estimated beforehand and the timestep can be adjusted to provide at least 10 - 20 steps for each period of the highest relevant frequency. In case of unsteadiness due to a moving front, the timestep should be chosen as a fraction of:

$$\Delta t \sim \frac{\Delta x}{U} \tag{6.28}$$

with the grid spacing Δx and the front speed U.

It should be noted that under strong grid and timestep refinement, sometimes flow features are resolved that are not relevant for the simulation. An example is the (undesirable) resolution of the vortex shedding at the trailing edge of an airfoil or a turbine blade in a RANS (p. 127) simulation for very fine grids and timesteps. Another example is the gradual switch to a DNS (p. 120) for the simulation of free surface flows with a **Volume of Fluid** (VOF) method (for example, drop formation, wave excitation for free surfaces, and so on). This is a difficult situation, as it usually means that no

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

grid/timestep converged solution exists below the DNS (p. 120) range, which can usually not be achieved.

In principle, the time dependency of the solution can be treated as another dimension of the problem with the definitions in Solution Error Estimation (p. 124). However, a four-dimensional grid study would be very demanding. It is therefore more practical to carry out the error estimation in the time domain separately from the space discretization. Under the assumption that a sufficiently fine space discretization is available, the error estimation in the time domain can be performed as a one-dimensional study.

Studies should be carried out with at least two and if possible three different timesteps for one given spatial resolution. Again, the error estimators given in Solution Error Estimation (p. 124) (Equation 6.25 (p. 126)) can be used, if Δ is replaced by the timestep. The following information should be provided:

- Unsteady target variables as a function of timestep (graphical representation)
- Error estimate based on Equation 6.25 (p. 126) for (time averaged) target variables
- Comparison with experimental data for different timesteps.

6.3.6.5. Avoiding Round-Off Errors

Round-off errors are usually not a significant problem. They can occur for high-Reynolds number flows where the boundary layer resolution can lead to very small cells near the wall. The number of digits of a single precision simulation can be insufficient for such cases. The only way to avoid round-off errors with a given CFD code is the use of a double precision version. In case of an erratic behavior of the CFD method, the use of a double precision version is recommended.

6.3.7. Handling Software Errors

Software errors can be detected by verification studies. They are based on a systematic comparison of CFD results with verified solutions (in the optimal case analytical solutions). It is the task of the software developer to ensure the functionality of the software by systematic testing.

In most cases, pre-existing software will be used. It is assumed that all CFD packages have been sufficiently tested to ensure that no software verification studies have to be carried out in the project (except for newly developed modules). In case that two CFD packages give different results for the same application, using the same physical models, the sources for these differences will have to be evaluated. In case of code errors, they will be reported to the code developers and if possible removed.

6.4. Selection and Evaluation of Experimental Data

Because of the necessity to model many of the unresolved details of technical flows, it is necessary to assess the accuracy of the CFD method with the help of experimental data. Experiments are required for the following tasks and purposes:

- Verification of model implementation
- Validation and calibration of statistical models

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Demonstration of model capabilities.

There is no philosophical difference between the different types of test cases. The same test case can be used for the different phases of model development, implementation, validation, and application, depending on the status of the model and the suitability of the data.

6.4.1. Verification Experiments

The purpose of verification tests is to ensure the correct implementation of all numerical and physical models in a CFD method. The best verification data would be the analytical solutions for simple cases that enable the testing of all relevant implementation aspects of a CFD code and the models implemented. As analytical solutions are not always available, simple experimental test cases are often used instead.

6.4.1.1. Description

For CFD code verification, convergence can be tested against exact analytical solutions like:

- Convection of disturbances by a given flow
- Laminar Couette flow
- Laminar channel flow.

For the verification of newly implemented models, verification can only in limited cases be based on analytical solutions. An example is the terminal rise velocity of a spherical bubble in a calm fluid.

In most other cases, simple experiments are used for the verification. It is recommended that you compute the test cases given by the model developer in the original publication of the model, or other trustworthy publications. Quite often experimental correlations can be applied, without the need for comparison with one specific experiment. For instance for turbulence model verification, the most frequently used correlations are those for flat plate boundary layers.

6.4.1.2. Requirements

The only requirement for verification data is that they enable a judgement of the correct implementation of the code and/or the models. This requires information from other sources concerning the performance of the model for the test case. Strictly speaking, it is not required that the simulations are in good agreement with the data, but that the differences between the simulations and the data are as expected.

The test suite for model verification must be diverse enough to check all aspects of the implementation. As an example, a fully developed channel flow does not enable a test of the correct implementation of the convective terms of a transport equation. The test suite should also enable testing the correct interaction of the new model with other existing features of the software.

Software verification for physical models should be carried out in the same environment that the end-user has available. Testing of the new features in an expert environment might miss some of error sources, such as the GUI.

Verification cases should be selected before the model is implemented. They must be considered an integral part of the model implementation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

6.4.2. Validation Experiments

The purpose of validation tests is to check the quality of a statistical model for a given flow situation. Validation tests are the only method to ensure that a new model is applicable with confidence to certain types of flows. The more validation tests a model passes with acceptable accuracy, the more generally it can be applied to industrial flows. The goal of validation tests is to minimize and quantify modeling errors. Validation cases are often called building block experiments, as they test different aspects of a CFD code and its physical models. The successful simulation of these building blocks is a prerequisite for a complex industrial flow simulation.

6.4.2.1. Description

Examples of validation cases are flows with a high degree of information required to test the different aspects of the model formulation. In an ideal case, a validation test case should be sufficiently complete to enable an improvement of the physical models it was designed to evaluate. Increasingly, validation data are obtained from DNS (p. 120) studies. The main limitation here is in the low-Reynolds number and the limited physical complexity of DNS (p. 120) data. Typically, validation cases are geometrically simple and often based on two-dimensional or axisymmetric geometries.

6.4.2.2. Requirements

Validation cases are selected to be as close as possible to the intended application of the model. As an example, the validation of a turbulence model for a flat plate boundary layer does not ensure the applicability of the model to flows with separation (as is known from the $k-\varepsilon$ model). It is well accepted by the CFD community and by model developers that no model (turbulence, multi-phase or other) will be able to cover all applications with sufficient accuracy. This is the reason why there are always multiple models for each application. The validation cases enable the CFD user to select the most appropriate model for the intended type of application.

Test case selection requires that the main features of the CFD models that are to be tested be clearly identified. They must then be dominant in the validation case. Validation cases are often 'single physics' cases, but it will be more and more necessary to validate CFD methods for combined effects.

The requirements for validation cases are that there should be sufficient detail to be able to compute the flow unambiguously and to evaluate the performance of the CFD method for a given target application.

Completeness of information is one of the most important requirements for a validation test case. This includes all information required to run the simulation, like:

- Geometry
- Boundary conditions
- Initial conditions (for unsteady flows)
- Physical effects involved.

While the first three demands are clearly necessary to be able to set up and run the simulation, the knowledge of all physical effects taking place in the experiment is not always considered. However,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

it is crucial to have a clear understanding of the overall flow in order to be able to judge the quality of a test case. Typical questions are:

- Is the flow steady-state or does it have a large-scale unsteadiness?
- Is the flow two-dimensional (axisymmetric, for example)?
- Are all the relevant physical effects known (multi-phase, transition, and so on)?
- Have any corrections been applied to the data and are they appropriate?
- Was there any measurement/wind or water tunnel interference?

Completeness of information is also essential for the comparison of the simulation results with the experimental data. A validation case should have sufficient detail to identify the sources for the discrepancies between the simulations and the data. This is a vague statement and cannot always be ensured, but a validation experiment should provide more information than isolated point measurements. Profiles and distributions of variables at least in one space dimension should be available (possibly at different locations). More desirable is the availability of field data in two-dimensional measuring planes including flow visualizations.

Completeness also relates to the non-dimensionalization of the data. Frequently the information provided is not sufficient to reconstruct the data in the form required by the validation exercise.

In case that the data provided are not sufficient, the impact of the missing information has to be assessed. Most crucial is the completeness of the data required to set up the simulation. In case of missing information, the influence of this information deficit has to be assessed. Typical examples are incomplete inlet boundary conditions. While the mean flow quantities are often provided, other information required by the method, as profiles for turbulent length scales and volume fractions is frequently missing. The importance of this deficit can be estimated by experience with similar flows and by sensitivity studies during the validation exercise.

Next to the completeness of the data, their quality is of primary importance for a successful validation exercise. The quality of the data is mainly evaluated by error bounds provided by the experimentalists. Unfortunately, most experiments still do not provide this information. Moreover, even if error estimates are available, they cannot exclude systematic errors by the experimentalist.

In addition to error bounds, it is therefore desirable to have an overlap of experimental data that enable testing the consistency of the measurements. Examples are the same data from different experimental techniques. It is also a quality criterion when different experimental groups in different facilities have carried out the same experiment. Consistency can also be judged from total balances, like mass, momentum and energy conservation. Quality and consistency can frequently be checked if validation exercises have already been carried out by other CFD groups, even if they used different models.

The availability of the data has to be considered before any CFD validation is carried out. This includes questions of ownership. For most CFD code developers, data that cannot be shown publicly are much less valuable than freely available experimental results.

6.4.3. Demonstration Experiments

The purpose of a demonstration exercise is to build confidence in the ability of a CFD method to simulate complex flows. While validation studies have shown for a number of building block experi-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ments that the physical models can cover the basic aspects of the target application, the demonstration cases test the ability of a method to predict combined effects, including geometrical complexity.

6.4.3.1. Description

For an aerodynamic study, a typical hierarchy would be:

- Verification - Flat plate
- Validation - Airfoil or wing
- Demonstration - Complete aircraft.

Similar hierarchies can be established for other industrial areas.

6.4.3.2. Requirements

Typically, the detail of the experimental data is much lower than for verification or validation cases.

Completeness of information to set up the test case is of similar importance as for validation cases and involves the same aspects as listed below:

- Geometry
- Boundary conditions
- Initial conditions (for unsteady flows)
- Physical effects involved.

Typically, the level of completeness of the data for demonstration cases is much lower than for validation cases. It is therefore even more essential to identify the missing information and to carry out sensitivity studies with respect to these data.

In terms of postprocessing, demonstration cases often do not provide a high degree of detail. They are usually not appropriate to identify specific weaknesses in the physical models or the CFD codes. Typically, only the point data or global parameters, as efficiencies, are provided.

Even though the density of data is usually lower, the quality should satisfy the same criteria as for validation cases. Error estimates are desirable and so are independent measurements.

Due to the limited amount of data available, the information is usually not sufficient to carry out consistency checks.

The requirements in terms of availability/openness are usually lower than for validation cases, as the demonstration applies usually to a smaller audience. A demonstration case might be carried out for a single customer or one specific industrial sector. It has to be ensured, as in all cases, that the data can be shown to the target audience of the simulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 7: CFX Best Practices Guide for Cavitation

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with moderate or little experience using CFX for applications involving cavitation.

Cavitation is the formation of vapor bubbles within a liquid where flow dynamics cause the local static pressure to drop below the vapor pressure. The bubbles of vapor usually last a short time, collapsing when they encounter higher pressure. Cavitation should not be confused with boiling. When a liquid boils, thermal energy drives the local temperature to exceed the saturation temperature.

Cavitation is a concern in several application areas, including pumps, inducers, marine propellers, water turbines, and fuel injectors. One of the major problems caused by cavitation is a loss of pressure rise across a pump. Other problems include noise, vibration, and damage to metal components.

This guide describes:

7.1. Approaches to Modeling Cavitation

7.2. Liquid Pumps

7.1. Approaches to Modeling Cavitation

In order to model cavitation, your simulation must have definitions for the liquid and vapor state of a material, as well as the saturation temperature.

Two approaches to modeling cavitation are:

- Define an Eulerian multiphase simulation that uses one fluid for the liquid state and one fluid for the vapor state, then use an interphase mass transfer cavitation model (see Cavitation Model in the *CFX-Solver Modeling Guide*) to transfer mass between the two fluids.
- If thermodynamic equilibrium can be assumed, you can model cavitation (also flashing) by defining an equilibrium phase change simulation using a homogeneous binary mixture for both states (liquid and vapor). For details, see Setting up an Equilibrium Phase Change Simulation in the *CFX-Solver Modeling Guide*.

Recommendations

- Initially establish the flow without cavitation, for example, by using a high outlet pressure or by turning off interphase mass transfer, depending on the case.
- In an equilibrium phase change simulation, initially maintain a large enough time step to avoid instabilities when cavitation starts.
- You may need to reduce the time step, especially if the cavitation boundary moves.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7.2. Liquid Pumps

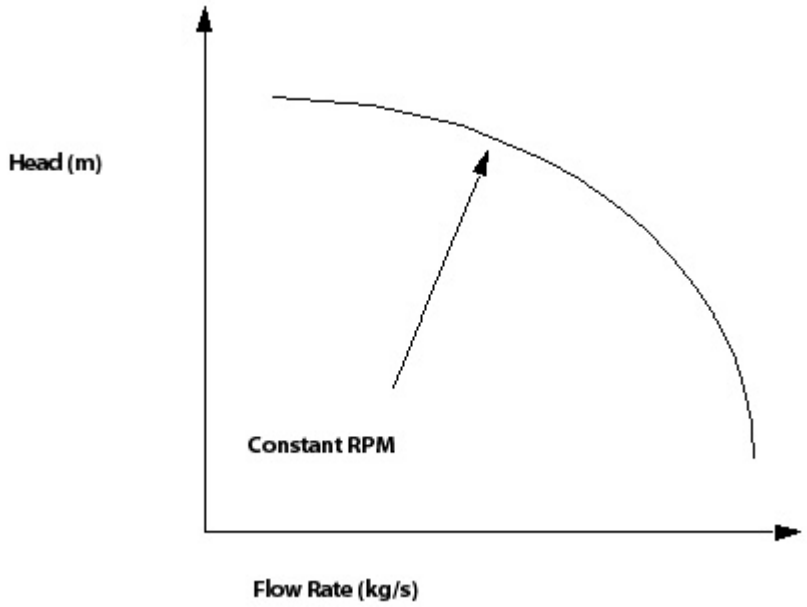
Water pumps must take in water and deliver it at a higher total pressure with an acceptable flow rate. Under certain conditions, cavitation may occur on the low pressure side of the pump, causing a loss of pressure rise and/or flow rate.

Both pump performance without cavitation and the affects of cavitation on performance will be discussed.

7.2.1. Pump Performance without Cavitation

As long as the static pressure remains sufficiently high everywhere in the system, cavitation will not occur. In this case, for a given pump RPM, the pressure rise and flow rate are directly coupled, and can be plotted in a pump performance diagram, as shown below.

Figure 7.1: Flow Rate vs Pressure Rise for a Liquid Pump

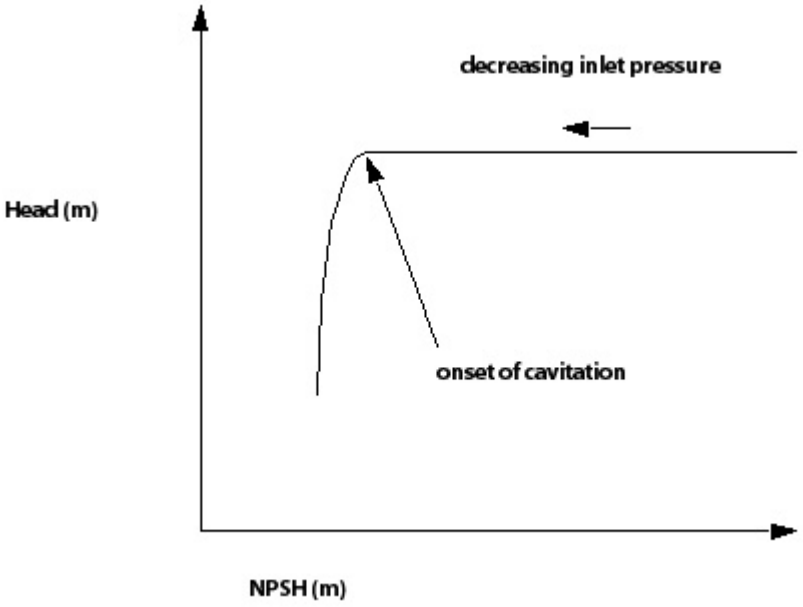


7.2.2. Pump Performance with Cavitation

If the inlet total pressure is below the critical value for a particular flow rate and RPM, cavitation will occur causing the pressure rise to diminish. The following performance diagram shows the effect of cavitation on pressure rise.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 7.2: Cavitation Performance at Constant RPM and Flow Rate



NPSH is the Net Positive Suction Head, a quantity directly related to the inlet total pressure by the relation:

$$NPSH = \left(\frac{p_{T,\text{inlet}} - p_v}{\rho g} \right) \tag{7.1}$$

where $p_{T,\text{inlet}}$ is the inlet total pressure, p_v is the vapor pressure, ρ is density, and g is the acceleration due to gravity. As the inlet total pressure drops, so does the NPSH value, and the amount of cavitation increases.

To generate this diagram, RPM and flow rate are fixed, and the pressure rise is measured at progressively lower inlet total pressures. For the part of the test where the inlet total pressure is sufficiently high to prevent cavitation, the pressure rise across the pump is constant, equal to the amount predicted by the first performance diagram. This results in a horizontal trend in the performance curve as the inlet total pressure is dropped. Because the pressure rise remains constant, the total pressure at the outlet drops by the same amount as at the inlet. Using CFX software, a mass flow outlet boundary condition can be specified to fix the flow rate while the inlet total pressure is varied.

When the inlet total pressure reaches a sufficiently low value, cavitation occurs. A further reduction in inlet total pressure causes more cavitation, which almost always causes a large loss of pressure rise. In rare cases, pressure rise can actually increase slightly with small amounts of cavitation. Even in such cases, however, a further increase in cavitation causes a sudden loss of pressure rise. In the lab, the pressure rise will eventually become insufficient to maintain the required flow rate. Using CFX software, the solution will eventually fail to converge. Before this point, data should be collected with a sufficient resolution (sufficiently small changes in inlet pressure) to resolve the part of the performance curve where the pressure starts to drop. The point of cavitation is often marked by the NPSH at which the pressure rise has fallen by a few percent.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

7.2.3. Procedure for Plotting Performance Curve

1. Set up a simulation with cavitation turned on and pressure levels set high enough to avoid levels of cavitation that significantly affect pressure rise.

   If you have trouble getting a converged solution, try running a simulation with cavitation turned off, then use the result as an initial guess for a simulation with cavitation turned on.

2. Run the solver to obtain a solution.

3. Calculate the pressure rise across the pump and the NPSH value, then plot a point in the performance diagram.

4. Lower the pressure boundary condition by about 5% to 10%.

5. Repeat starting from step 2, using the previous solution as the initial guess, until cavitation has caused a significant loss of pump head.

7.2.4. Setup

To facilitate setting up typical domain settings for the cavitation of water, you may load a single-domain mesh, then run the template `.ccl` file:

`CFX/etc/model-templates/cavitating_water.ccl`

This file should be examined in a text editor before using it so that you understand which settings it specifies.

For the domain fluids list, specify both a liquid and a vapor for the same material. In most cases, it is sufficient to use a constant-density vapor.

Under **Fluid Models** for the domain, it is strongly recommended that you select **Homogeneous Model** under **Multiphase Options**. You do not need to select **Free Surface Model** for the purpose of simulating cavitation.

Under **Fluid Pairs** for the domain, select the fluid pair from the list and, for **Mass Transfer**, set **Option** to `Cavitation`. Select the **Rayleigh Plesset** cavitation model or a **User Defined** Cavitation Model. For the **Rayleigh Plesset** model, the **Mean Diameter** for cavitation bubbles represents the mean nucleation site diameter and must be specified. The default value of 2e-06 m is a reasonable value in most cases. The **Saturation Pressure** must be defined unless the materials you have selected are the components of a homogeneous binary mixture. In the latter case, the saturation properties will already be defined in the mixture definition, but you may still choose to override the **Saturation Pressure** by specifying it on the **Fluid Pairs** tab.

When initializing the domain, set the volume fraction of vapor to zero and the volume fraction of liquid to unity. These settings (represented by the `Automatic` setting for **Volume Fraction**) are used by default in CFX.

Set up the problem with one of the following boundary condition combinations:

1. Inlet total pressure and outlet mass flow (recommended)

2. Inlet velocity profile and outlet static pressure

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The inlet boundary condition should specify that the volume fraction of vapor is zero.

Turbulence models should be chosen as usual (for example, k-epsilon or SST). For turbulence induced cavitation, consider using the DES model.

For advection scheme, use high resolution, or a specified blend factor of unity.

If editing a material, remember that the vapor pressure is on an absolute scale; it does not depend on the specified reference pressure.

Cavitation models cannot be combined with other types of interphase mass transfer, such as thermal phase changes.

7.2.5. Convergence Tips

If performing a single solution, initially turn off cavitation, then turn on cavitation and use the first set of results as an initial guess.

If performing a series of simulations using one solution to initialize the next, solve the cases in order of decreasing pressure (for example, approaching cavitation).

7.2.6. Postprocessing

A contour plot of volume fraction for the vapor can show where cavitation bubbles exist.

To calculate the inlet and outlet pressures, use the function calculator.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 8: CFX Best Practices Guide for Combustion

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with moderate or little experience using CFX for applications involving combustion.

This guide describes:

- Gas Turbine Combustors (p. 151)
- Combustion Modeling in HVAC Cases (p. 153)

8.1. Gas Turbine Combustors

Gas turbines are widely used in stationary and aircraft applications. The combustor receives the working fluid in a compressed state, burns fuel to increase its temperature, and passes it to the turbine. One of the key design goals for the combustor is to achieve a stable combustion process. Another key design goal is to minimize the emission of pollutants, particularly oxides of nitrogen.

8.1.1. Setup

8.1.1.1. Steady-state vs. Transient

Most simulations are steady-state, particularly for stationary gas turbines that operate at a constant load.

8.1.1.2. Turbulence Model

The k–ε turbulence model is used in many applications, but the SST model should be considered for flows with separated boundary layers, and the Reynolds stress model is the best choice for highly swirling flows.

8.1.1.3. Reference Pressure

Because of the high inlet pressure, a reference pressure between 4 and 20 atmospheres is common, and depends upon the type of simulation you are running.

8.1.1.4. Combustion Model

The choice of combustion model depends of whether the fuel/oxidant combination is premixed. The following table outlines some of the differences.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Premixed Combustion	Non-Premixed Combustion
Commonly used for recent stationary gas turbines in power generation.	Typically used for flight engines because it is easier to control variable operating conditions.
Combustion Models: EDM with product limiter and/or extinction submodels, FRC/EDM combined, Partially Premixed (Turbulent Flame Speed Closure (TFC)) Note that the EDM model usually needs adjusting for premixed combustion (for example, extinction by temperature or by mixing/chemical time scales).	Combustion Models: EDM, FRC/EDM combined, Flamelet (and, for some cases, also the Premixed model).

For the preliminary analysis of high speed turbulent flow, the Eddy Dissipation combustion model is a sensible choice, but cannot simulate burning velocities or flame position.

The Laminar Flamelet model is applicable for turbulent flow with non-premixed combustion, and provides a robust solution at a low computational expense for multi-step reactions. The Flamelet model uses a chemistry library, meaning that only two additional transport equations are solved to calculate multiple species. As a result, the Flamelet model is a very good choice for modeling the formation of various pollutants during the combustion process. The Flamelet model predicts incomplete combustion to some extent (CO in exhaust gas), which helps to predict reduction in temperature (unlike EDM).

8.1.2. Reactions

During the initial analysis of a combustor, the highest values of temperature and outgoing heat flux are likely to be of primary concern. For this purpose, a single-step Eddy Dissipation reaction can be used. Such a reaction is likely to overpredict the temperature, and will not predict emissions correctly, but can provide a conservative indicator of the expected temperature levels.

Other reaction steps might then be added to the simulation to account for the formation of combustion byproducts. Each reaction step has its own separate time scale. As a result, convergence can become very difficult when a multi-step reaction contains more than about 5 steps.

8.1.3. Convergence Tips

The Equation Class Settings tab in CFX-Pre can be used to set different advection schemes and time scales for each class of equation you are solving. For multi-step Eddy Dissipation reactions, convergence can be improved by temporarily increasing the mass fraction time scale by a factor of about 5-10.

For the Eddy Dissipation Model, multistep convergence can be aided by first running a simplified single-step simulation and using the results from the run as an initial values file for a multi-step run.

You may restart a Flamelet model from a cold solution. You should avoid restarting with the Flamelet model from an EDM solution. You may restart an EDM case from a Flamelet model solution.

The High Resolution advection scheme is always recommended for combustion simulations because it is bounded and prevents over/undershoots. Care must be taken, however, to provide a mesh of

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sufficient quality to resolve most of the flow features. A very poor mesh will result in the scheme using a blend factor close to zero (therefore not providing a solution as accurate as expected).

For simulations that include Finite Rate Chemistry, small temperature variations can result in large changes in reaction rate. When a solution is converging, temperature values may change sufficiently to make the solution unstable. To aid convergence, add a `TEMPERATURE DAMPING` CCL structure within a `SOLVER CONTROL` block, as follows:

```
FLOW:
  SOLVER CONTROL:
    TEMPERATURE DAMPING:
       Option = Temperature Damping
       Temperature Damping Limit = <Real number>
       Under Relaxation Factor = <Real number>
    END
  END
END
```

Depending on the location of the SOLVER CONTROL block, the temperature damping may be applied to a particular domain or phase. Set the `Temperature Damping Limit` to 0 so that positive damping is always applied. The `Under Relaxation Factor` can be set to multiply changes in temperature by a value between 0 and 1. You should try a factor of 0.2 if you are having trouble converging a solution.

8.1.4. Postprocessing

Some of the most common plots to create in CFD-Post include:

- Mass Fractions: fuel, O_2, products, intermediate species (CO), pollutants (NO)
- Turbulent Mixing Time Scale (Eddy Dissipation / Turbulent Kinetic Energy)
- Reaction Rates

  The variable "<my reaction>.Molar Reaction Rate" is available for every "Single Step" reaction (EDM, FRC or combined model).
- Plots of the turbulent Damköhler number (the ratio of the turbulent time scale to the chemical time scale)

8.2. Combustion Modeling in HVAC Cases

This section deals with the setup of combustion cases for HVAC simulations, where it is important to accurately model the combustion process. Such processes are known as "combusting fire" simulations, as opposed to "inert fire" simulations.

Using a combusting fire simulation is the most accurate way to model fires in all HVAC cases. It is particularly important in cases when the fire is under-ventilated, or when the ventilation cannot be easily predicted. The drawback is the additional computational expense involved in solving a full combustion model as part of the main solution.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

8.2.1. Set Up

Most simulations are set up as transient. The choice of timestep is generally model dependent, but will usually fall into the range 0.5 s to 2 s. The Total Energy heat transfer model should be selected to fully model buoyancy.

Note:

When modeling buoyancy, it is very important to correctly specify the buoyancy reference density when opening boundary conditions are used.

The RNG k–ε turbulence model is a good choice for combusting flows, with either no buoyancy terms in equations or buoyancy terms in both the equations (with C3=1). The SST model is also reasonable, but may not converge well for natural convection. The SSG model is accurate, but convergence may be very slow.

The most common fuels used are hydrocarbons such as methane, diesel and petroleum. Cellulosic materials, plastics and wood are also used. The simulation will dictate the type of materials to use as a fuel.

The Eddy Dissipation Model is widely used, combined with an Additional Variable for toxins. The flamelet model is more rigorous, and is a better choice when the fire is under ventilated.

8.2.2. Convergence Tips

Convergence can be slowed if care is not taken in the setup of buoyancy and openings.

The presence of an instantaneous fuel supply is sometimes not physical, and can slow convergence. In many transient cases, the amount of fuel available can be controlled by using time-dependent expressions.

8.2.3. Postprocessing

The most common parameters of interest in a combusting fire model are simulation-dependent, but will usually include one of more of the following:

- Temperature
- Products (including carbon monoxide and other toxins)
- Visibility
- Wall Temperature
- Wall convective and radiative heat fluxes

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 9: CFX Best Practices Guide for HVAC

This guide discusses best practices for setting up, solving, and postprocessing an HVAC simulation:

9.1. HVAC Simulations

9.2. Convergence Tips

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with moderate or little experience using CFX for HVAC[1] applications.

9.1. HVAC Simulations

HVAC studies range in scale from single components (such as a radiator) to large and complicated systems (such as a climate control system for a large building).

Physical processes that are commonly modeled in HVAC simulations include:

- Buoyancy
- Thermal radiation
- Conjugate heat transfer (CHT) between fluids and solids.

Typical HVAC systems include the following components:

- Heating/cooling units such as furnaces, heaters, air conditioners, and radiators
- Fans/pumps
- Thermostats.

9.1.1. Setting Up HVAC Simulations

This section discusses how to set up various physical processes, CFD features, and components involved in HVAC simulations.

9.1.1.1. Buoyancy

Most HVAC cases involve flow that is affected by buoyancy. Buoyancy can be activated on the **Basic Settings** tab of the **Domain** details view.

Two buoyancy models are available: Full and Boussinesq. These models are automatically selected according to the properties of the selected fluid(s).

[1] HVAC is a reference to Heating, Ventilation (or Ventilating), and Air Conditioning. Often, it is also used as a reference to refrigeration.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The Full buoyancy model is used if fluid density is a function of temperature and/or pressure (which includes all ideal gases and real fluids). In this case, a Buoyancy Reference Density must be set as the expected average density of the domain.
- The Boussinesq model is used if fluid density is not a function of temperature or pressure. In this case, a Buoyancy Reference Temperature must be set as the expected average temperature of the domain.

When fluid properties are functions of pressure, the absolute pressure is used to evaluate them. The calculation of absolute pressure requires a Buoyancy Reference Location to be defined, preferably by specifying coordinates.

When modeling fire, it is recommended that you choose a compressible fluid because density variations will be significant. An incompressible fluid should be chosen only if density variations are small (a few percent or less).

9.1.1.2. Thermal Radiation

To set the radiation model for a fluid domain, visit the **Fluid Models** panel for that domain and set the following:

9.1.1.2.1. Thermal Radiation Model

For HVAC studies, select either **Monte Carlo** or **Discrete Transfer**. If directed radiation is to be modeled, Monte Carlo must be used.

9.1.1.2.2. Spectral Model

Select either **Gray** or **Multiband**. Spectral bands are used to discretize the spectrum and should therefore be able to adequately resolve all radiation quantities that depend on wavelength (or frequency or wave number). For HVAC, two bands will usually suffice.

9.1.1.2.3. Scattering Model

A scattering model should not be used if you are modeling clear air. The isotropic scattering model should be used if you are modeling air that contains dust or fog.

To set up radiation for a solid domain, visit the **Solid Models** panel for that domain (each solid domain must be set up separately). The only radiation model available for solid domains is Monte Carlo.

Note:

If any solid domain uses the Monte Carlo radiation model (that is, if it uses radiation at all), then all fluid domains using a radiation model must use the Monte Carlo model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The material used in a domain that transmits radiation has radiation properties that specify **Absorption Coefficient**, **Scattering Coefficient**, and the **Refractive Index**. These properties may be edited in the **Materials** details view.

Note:

Radiation modeling cannot be used with Eulerian multiphase simulations.

Thermal radiation properties are specified on the **Boundary Details** panel for each boundary of a domain that transmits radiation. For opaque surfaces, the properties that must be specified are: **Emissivity** and **Diffuse Fraction**. For inlets, outlets, and openings, you may specify either the **Local Temperature** or an **External Blackbody Temperature**.

The Monte Carlo and Discrete Transfer models allow radiation sources to be specified on the **Sources** panel for any subdomain or wall boundary. For subdomains, radiation sources per unit volume are specified; for boundaries, radiation fluxes are specified. Radiation sources may be directed or isotropic. Multiple isotropic sources and up to one directed source may be specified for any given wall boundary or subdomain.

Material properties related to radiation, thermal radiation properties for boundaries, and source strengths can be specified as expressions that depend on one or more of the built-in variables: **Wavelength in Vacuum** (or **wavelo**), **Frequency** (or **freq**), **Wavenumber in Vacuum** (or **waveno**).

A domain representing an opaque solid should not have a radiation model set. The boundaries of radiation-transmitting domains that interface with such a solid domain should be specified as opaque.

External windows of a room can be modeled as solid domains which interface with the room (air) domain; they may also be modeled as an external boundary of the room domain. In either case, the exterior boundary must be modeled as an opaque wall. A diffuse and a directed radiation source emitted from the opaque surface can be used to simulate sunlight. In order to simulate the motion of the sun, the direction vector for directed radiation can be specified by CEL expressions that depend on time (**t**). Radiation escaping through a window can be modeled by specifying a non-zero emissivity (to cause radiation absorption) and either:

- Specifying a heat transfer coefficient via a CEL expression that accounts for the thermal energy lost
- Specifying a fixed wall temperature.

When using solid domains that transmit radiation, a spectral radiation model is recommended. If a simulation contains no solid domains that transmit radiation, a gray radiation model can be used for rough calculations but a spectral model should be used for more detailed modeling.

9.1.1.3. CHT (Conjugate Heat Transfer) Domains

CHT domains are solid domains that model heat transfer. In CFX, all solid domains must model heat transfer, and are therefore CHT domains. If you do not want to model heat transfer in a particular region, do not assign the mesh for that region to any domain.

Boundaries between domains that model heat transfer have temperatures and thermal fluxes calculated automatically, and should not have thermal boundary conditions specified. External

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

boundaries (which can represent solids that are not explicitly modeled) require the specification of a thermal boundary condition.

Boundary conditions other than thermal boundary conditions (for example, wall roughness) may be specified on the boundaries of a fluid domain that interface with a solid domain.

Sources of thermal energy and/or radiation can be added to a subdomain of a CHT domain.

9.1.1.4. Mesh Quality

Ensure that wall boundary layers have adequate mesh resolution. This is important regardless of the type of wall heat transfer: adiabatic, specified temperature, specified heat flux, or heat transfer coefficient.

The mesh resolution in a boundary layer affects the prediction of convective heat transfer and the temperature gradient near the wall. For walls without a specified temperature, the temperature gradient near the wall affects the calculated wall temperature and, consequently, the amount of radiation emitted (provided that the emissivity of the wall is non-zero).

9.1.1.5. Fans

Fans should be represented by momentum sources if they are embedded in the domain. Fans can also be represented by an inlet or outlet boundary condition or both.

9.1.1.6. Thermostats

A Thermostat can be defined using a User Fortran routine. Refer to the HVAC tutorial for details.

9.1.1.7. Collections of Objects

If your HVAC simulation models a large number of people/equipment/items, consider volumetric sources of heat, CO_2, and resistances.

9.2. Convergence Tips

Buoyancy and coupling between the relevant equations often make convergence difficult. Smaller timesteps may help convergence.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 10: CFX Best Practices Guide for Multiphase

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with moderate or little experience using CFX for applications involving multiphase flows.

In the context of CFX, a multiphase flow is a flow composed of more than one fluid. Each fluid may possess its own flow field, or all fluids may share a common flow field. Unlike multicomponent flow[1], the fluids are not mixed on a microscopic scale; rather, they are mixed on a macroscopic scale, with a discernible interface between the fluids. CFX includes a variety of multiphase models to allow the simulation of multiple fluid streams, bubbles, droplets, and free surface flows.

This guide describes:

- Bubble Columns (p. 159)
- Mixing Vessels (p. 160)
- Free Surface Applications (p. 161)
- Multiphase Flow with Turbulence Dispersion Force (p. 162)

10.1. Bubble Columns

Bubble columns are tall gas-liquid contacting vessels and are often used in processes where gas absorption is important (for example, bioreactors to dissolve oxygen in broths) and to limit the exposure of micro-organisms to excessive shear imparted by mechanically driven mixers. There are two types of bubble columns in general use: airlift reactors that use a baffle to separate the riser and downcomer regions, and other columns that do not use a baffle.

10.1.1. Setup

The choice of a steady-state or transient simulation depends on the type of simulation you want to analyze. For example, an analysis using a steady-state simulation is often satisfactory for monitoring global quantities. A transient simulation can be used to observe transient effects, such as recirculation zones.

Most bubble columns use two fluids: one continuous fluid and one dispersed fluid. The $k-\varepsilon$ model is typically used in the continuous fluid, and the dispersed phase zero equation is used for the dispersed phase.

Non-drag forces become less significant with increasing size of the bubble column. For smaller columns, non-drag forces may be significant.

[1] Note that a fluid in a multiphase flow may be a multicomponent mixture.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The Grace drag model is recommended, especially for modeling air/water.

A degassing boundary condition is generally employed at the top of the bubble column. The degassing boundary behaves as an outlet boundary to the dispersed phase, but as a wall to the continuous phase.

A reasonable estimate of the time scale is given by a factor of the length of the domain divided by the velocity scale (for example, 0.5 * L/U).

10.1.2. Convergence Tips

Sometimes, physical instabilities (such as recirculation zones) can result in slow or stalled convergence. In these cases, you can still obtain an indicator of convergence for a global quantity by creating a monitor point at some point in the domain. As a result, you can determine whether values for selected global quantities (such as gas hold-up) are meaningful.

10.1.3. Postprocessing

The main design objective for bubble columns is efficient mixing, which is strongly influenced by the dispersed phase. Mixing efficiency can be measured in a number of ways. One example is to measure the gas hold-up in the riser as a function of the superficial gas velocity. This would require solving for the gas volume fraction for a number of simulations, each with a different mass flow rate of the dispersed phase at the sparger. Another option would be to use the same input parameters, this time measuring the liquid velocity in the downcomer.

10.2. Mixing Vessels

Mixing vessels are widely used in the chemical industry to optimize mixing and/or heat transfer between fluids. Mixing must be efficient, precise and repeatable to ensure optimum product quality. Quantities of interest may include mixing times, gas hold-up, power draw, local shear and strain rates, and solids distribution. The application of Computational Fluid Dynamics to address these needs results in faster and lower cost design through reduced experimentation, more reliable scale-up, and better understanding of the processes, leading to higher yields and reduced waste.

10.2.1. Setup

Mixing vessels generally use two domains. The impeller domain is a small, rotating domain that encloses the impeller. The rest of the tank is represented by a stationary domain. Different types of domain interfaces are available for the connection between the stationary and the rotating domains. The recommended types are:

- Frozen Rotor: faster but cruder
- Transient: slower (transient analysis) but much more accurate

The choice of a steady-state or transient simulation is dependent on the type of interface that exists between the two domains. Where a Frozen Rotor interface is used, a steady-state simulation is usually carried out. Performing a transient simulation allows you to use the transient rotor/stator frame change model to account for transient effects.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The initial guess for velocity can be set to zero in the relative frame for each domain.

10.3. Free Surface Applications

Free Surface flow refers to a multiphase situation where the fluids (commonly water and air) are separated by a distinct resolvable interface. Such flows occur, for example, around the hull of a ship, or in a breaking wave.

10.3.1. Setup

The choice of using a steady-state or transient simulation is problem-dependent. There are two models available for free surface flow: homogeneous and inhomogeneous. The homogeneous model can be used when the interface between the two phases remains well defined and none of the dispersed phase becomes entrained in the continuous phase. An example of homogeneous free surface flow is flow in an open channel. A breaking wave is one example of an inhomogeneous flow case.

The same choice of turbulence model as for single phase simulations is appropriate. When using the inhomogeneous model, you should use the homogeneous turbulence option in CFX-Pre. The Buoyancy Reference Density should be set to the density of the least dense fluid.

When setting boundary conditions, the volume fractions can be set up using step functions to set the liquid height at each boundary. An outlet boundary having supercritical flow should use the **Supercritical** option for **Mass And Momentum**. This requires that you set the relative pressure of the gas above the free surface at the outlet.

For most free surface applications, the initial conditions can use step functions to set the volume fractions of the phases as a function of height. The initial condition for pressure should be set to hydrostatic conditions for the specified volume fraction initialization and the buoyancy reference density.

The timestep for free surface flows should be based on a L/U (Length/Velocity) scale. The length scale should be a geometric length scale. The velocity scale should be the maximum of a representative flow velocity and a buoyant velocity, which is given by:

$$U_{\text{buoyant}}=\sqrt{g\,L}$$

In addition, it is sometimes helpful to reduce the timestep for the volume fraction equations by an order of magnitude below that of the other equations.

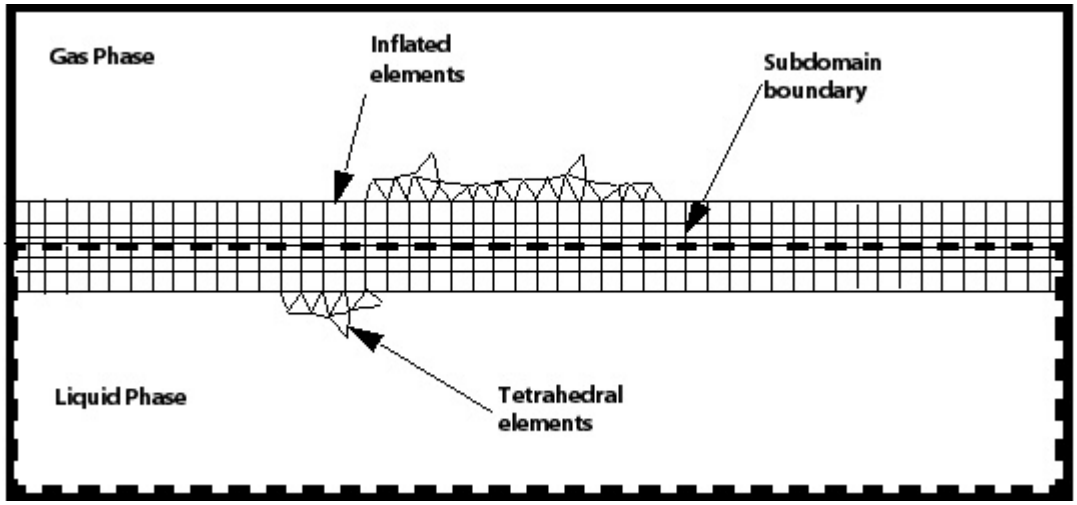
10.3.2. Convergence Tips

The interface between the liquid and gas phase can sometimes become blurry. This could be due to physical properties (such as a breaking wave or sloshing in a vessel). Where the dispersed phase becomes entrained in the continuous phase, the inhomogeneous model is a better choice.

A technique to increase the mesh density in the region of a liquid-gas interface is to create a subdomain that occupies the same region as the liquid (or gas) phase, and inflate the mesh in both directions from the edge of the subdomain, as shown in Figure 10.1: An exaggerated view of three inflation layers on each side of the uppermost subdomain boundary surface. (p. 162). The inflation layers can increase the resolution in the region of the interface and enhance the results.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 10.1: An exaggerated view of three inflation layers on each side of the uppermost subdomain boundary surface.



10.4. Multiphase Flow with Turbulence Dispersion Force

In order to reduce convergence difficulties encountered in some multiphase flow problems, the value of the expert parameter `ggi ap relaxation` is multiplied internally by 0.75. This occurs in the following situation only:

- Multiphase flow
- Nontrivial turbulence dispersion force included
- Coupled volume fraction solution algorithm.

In this situation, the default value of 1.0 is converted internally to 0.75. If you override the default with a smaller value, the new value is also multiplied internally by 0.75.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 11: CFX Best Practices Guide for Turbomachinery

Turbomachinery applications can generally be divided into three main categories: gas compressors and turbines, liquid pumps and turbines, and fans and blowers. Each category is discussed in a separate section below.

This guide describes best practices for setting up simulations involving:

11.1. Gas Compressors and Turbines

11.2. Liquid Pumps and Turbines

11.3. Fans and Blowers

11.4. Frame Change Models

11.5. Domain Interface Setup

11.6. Transient Blade Row

This guide is part of a series that provides advice for using CFX in specific engineering application areas. It is aimed at users with moderate or little experience using CFX for applications involving turbomachinery.

11.1. Gas Compressors and Turbines

This section describes:

- Setup for Simulations of Gas Compressors and Turbines (p. 163)
- Convergence Tips (p. 164)
- Computing Speedlines for a Machine (p. 164)
- Postprocessing (p. 165)

11.1.1. Setup for Simulations of Gas Compressors and Turbines

Heat transfer and viscous work are involved, and can be modeled by using the Total Energy heat transfer model and enabling the Viscous Work Term option in CFX-Pre.

The industry-standard Shear Stress Transport model is the recommended choice for these cases. When using the Shear Stress Transport model, ensure a resolution of the boundary layer of more than 10 points. For details, see The k-omega and SST Models in the *CFX-Solver Modeling Guide*.

A common boundary condition configuration is to specify the total pressure and total temperature at the inlet and either the mass flow, exit corrected mass flow, or static pressure at the outlet, depending on the flow condition. Other configurations are also commonly used.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A good estimate of the timestep is within the region $0.1/\omega$ to $1/\omega$, where ω is the angular velocity of the rotating domain in radians per second. Selecting an automatic timestep will result in a timestep of $0.1/\omega$.

The second-order high-resolution advection scheme is generally recommended.

11.1.2. Convergence Tips

For high speed compressors and turbines, where the machine mass flow is choked, it is generally not possible to specify the specified mass flow at the outlet to match the numerical choked mass flow value. For choked flow conditions, you can either specify static pressure or exit corrected mass flow at the outlet. Both these conditions allow the mass flow through the machine to adjust to the numerical choked mass flow value.

If you have trouble converging a simulation involving real gases, try to obtain a solution first using an ideal gas. Ideal gases are available in the real gas library.

If you have trouble converging a problem with many stages, you may find that solving for a reduced number of stages can give you a better understanding of the physics, and may allow you to restart a multi-stage problem with a better initial guess. You can also try ramping up boundary conditions and the RPM.

Low pressure ratio Gas compressors (1.1 or less) can be treated more like liquid pumps. For details, see Liquid Pumps and Turbines (p. 166).

11.1.3. Computing Speedlines for a Machine

Most turbomachinery simulations examine the machine performance across a range of operating points. A *speedline* involves varying the mass flow rate from choke to stall at a fixed rotational speed. This is often accomplished by varying the exit mass flow rate or exit pressure while maintaining a constant, stationary frame total temperature and pressure at the inlet. If total conditions are not maintained at the inlet, the resulting inlet mass flow rate must be corrected to a standard reference condition.

Performance along a speedline is typically measured in terms of a pressure ratio and an efficiency. A full *performance map* is obtained by examining the performance across a range of rotation rates; that is, multiple speedlines. A plot containing multiple speedlines is called the *performance map* of the machine. Computing speedlines and performance maps requires many individual computations (one per mass flow), therefore it is desirable to automate the process.

Depending on the machine type, you can use the following boundary conditions to effectively compute speedlines:

- *For low speed compressors and turbines (subsonic), liquid pumps*: Specify **Total Pressure** and **Temperature** at the inlet and a **Mass Flow Rate** condition or a **Exit Corrected Mass Flow Rate** condition at the outlet. This enables you to easily vary the mass flow across the full speedline while maintaining stationary inlet total conditions. For compressors, the specification of the mass flow rate at the outlet is essential towards the low flow end of a speedline where the machine pressure ratio is insensitive to the mass flow rate. For this reason, you should not specify a **Static Pressure** condition at the outlet boundary for these machines.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- *For high speed turbines and compressors (transonic)*: In these machines, the mass flow through the system is limited by a choked condition. If the specified exit mass flow exceeds the choke flow rate, the net mass downstream of the choke condition will decrease until it reaches zero density, causing the solver to fail. It is therefore important to avoid an exit mass flow near choke.

  Similarly, at lower mass flow rates, close to stall, the pressure ratio is either constant or decreasing. A pressure specified outlet is generally unstable in this regime because there are multiple mass flow rates that may satisfy the outlet condition. Therefore a pressure boundary condition should be avoided in this range.

  This presents a problem with running speedlines using pressure and mass flow conditions, because neither boundary condition can be used across the entire speedline.

  While the exit pressure and mass flow rate may be asymptotic over portions of a speedline, the exit corrected mass flow remains non-asymptotic and is therefore applicable across the entire operating range. Thus, it is recommended that you use the **Exit Corrected Mass Flow Rate** outlet boundary condition for running speedlines.

Some advantages of using the **Exit Corrected Mass Flow Rate** outlet boundary condition are:

- It allows you to specify the exit corrected mass flow, which is a function of the outlet mass flow rate, total conditions, and a user-supplied reference condition.
- It makes it possible to automate the speedline analysis by enabling you to specify one boundary condition type that functions well across the entire speedline.
- It improves stability when a run is started with poor initial conditions, allowing for the use of a much larger timescale factor, so that all runs can be started from a simple or automatic initial guess. This allows points along the speedline to be computed independently rather than using each existing solution to initialize the next.
- Typically in compressor operations, a fixed value of **Exit Corrected Mass Flow Rate** represents a fixed running or "op" line. As a result, you could keep the **Exit Corrected Mass Flow Rate** boundary condition fixed and vary the rotational speed to generate the compressor performance along an op line. This is useful when comparing designs.

For more details on the mathematical treatment of exit corrected mass flow rate as an outlet boundary condition, see Exit Corrected Mass Flow Rate in the *CFX-Solver Theory Guide*.

11.1.4. Postprocessing

CFD-Post offers a powerful array of postprocessing tools for turbomachinery applications, including turbo-specific plots and performance calculation macros. To use many of the Turbo Post tools, you must first initialize each domain by specifying the locations of turbo regions and instancing information.

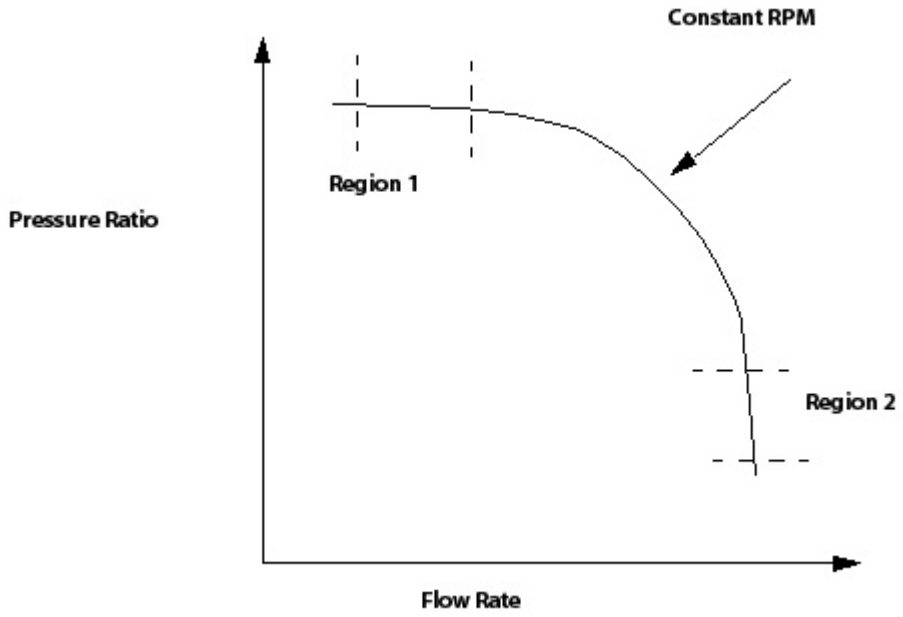
The Turbo Calculator from the Turbo menu in CFD-Post allows you to perform calculations on the type of application you are modeling. The macro prints a report in HTML showing a number of calculated variables, including torque, head and flow coefficients, blade loading, and efficiency.

You can also create your own macros to customize postprocessing using Power Syntax, which is based on the Perl language.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The optimal performance characteristics can be determined by creating a curve of pressure ratio versus flow rate.

Figure 11.1: Flow Rate vs Pressure Rise for a Gas Compressor



In Figure 11.1: Flow Rate vs Pressure Rise for a Gas Compressor (p. 166), Region 1 shows an area where a large change in mass flow rate represents a small change in pressure rise. When modeling flow in this region, a mass flow rate specified outlet is better than a pressure specified outlet. Region 2 shows an area where a small change in flow rate represents a large pressure variation. This region is close to "choking", and a pressure-specified or exit corrected mass flow rate outlet is the best choice. For details, see Convergence Tips (p. 164).

CFD-Post provides a performance macro for gas compressors and turbines.

11.2. Liquid Pumps and Turbines

This section describes:

- Setup for Simulations of Liquid Pumps and Turbines (p. 166)
- Convergence Tips (p. 167)
- Postprocessing (p. 167)

11.2.1. Setup for Simulations of Liquid Pumps and Turbines

Heat transfer is not significant in most cases, so the heat transfer option can be set to None in CFX-Pre.

The k–ε and Shear Stress Transport models are appropriate choices for modeling turbulence. When using the Shear Stress Transport model, ensure a resolution of the boundary layer of more than 10 points. For details, see The k-epsilon Model in the *CFX-Solver Modeling Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When setting boundary conditions, a total pressure specified inlet and a mass flow outlet are a recommended practice. The total pressure inlet condition is often more appropriate than the uniform velocity or mass flow inlet condition for cases that assume that the machine is drawing fluid directly from a static reservoir.

As with gas compressors, a good estimate of the timestep is within the region $0.1/\omega$ to $1/\omega$, where ω is the angular velocity of the rotating domain in radians per second. Selecting an automatic timestep will result in a timestep of $0.2/\omega$.

The high-resolution advection scheme is recommended.

This document deals with obtaining solutions for cases without cavitation, but cavitation may be present. For advice on how to deal with cavitation, see CFX Best Practices Guide for Cavitation (p. 145).

11.2.2. Convergence Tips

When only a poor initial guess is available, it may be helpful to first run with a specified mass flow inlet and a static pressure outlet. The outlet pressure in this case is fairly arbitrary and is usually set at, or close to zero to reduce round-off error. The specification of a mass flow inlet may be more robust. However, a mass flow inlet assumes a uniform inlet velocity—which may not be appropriate. Once the overall flow is established, the boundary conditions may then be changed to total pressure at the inlet and mass flow at the outlet.

11.2.3. Postprocessing

If a total pressure inlet boundary condition is used (recommended where possible), it will also provide a useful starting point for streamlines that are colored by total pressure during postprocessing. The uniform total pressure distribution means lines will begin with a uniform color. It may be harder to visually resolve these pressure values if an inlet velocity profile is used.

CFD-Post provides a performance macro for liquid pumps and turbines.

11.3. Fans and Blowers

This section describes:

- Setup for Simulations of Fans and Blowers (p. 167)
- Convergence Tips (p. 168)
- Postprocessing (p. 168)

11.3.1. Setup for Simulations of Fans and Blowers

Fans and blowers behave like liquid pumps, and require a similar model setup. The flow is generally modeled as incompressible and isothermal. The fluid is typically air as a general fluid or as an ideal gas at a specified temperature. The k–ε or SST model is used to model turbulence.

Boundary conditions, turbulence models and choice of timestep are the same as for liquid pumps and turbines.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

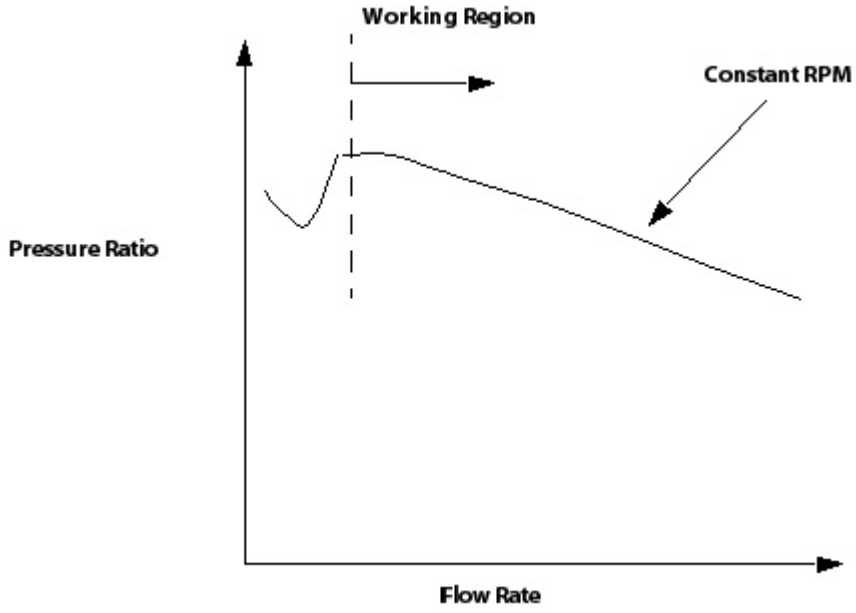
11.3.2. Convergence Tips

The use of the alternate rotation model is an important consideration when modeling fans and blowers. Where long axisymmetric inlets exist, the absolute frame velocity has less swirl than the relative frame velocity. Because the alternate rotation model solves for the absolute frame velocity, it can reduce numerical error in such inlet sections. The model may introduce errors in the exit stream if the flow is highly swirling. Hence, the length of the inlet and exit sections can be an important factor when choosing whether to implement the model. The alternate rotation model is generally recommended, especially for axial fans. In most realistic flow situations, this model reduces (or at least will not increase) numerical errors.

Air foil drag is significant and boundary layer friction is an important modeling issue for fans and blowers. A good resolution of the boundary layer, requiring a high concentration of nodes close to the blade surfaces, is therefore important. The Shear Stress Transport model can provide relatively accurate results where the boundary layer is sufficiently resolved by the mesh.

11.3.3. Postprocessing

A similar postprocessing approach to pumps and turbines is also useful for fans and blowers. For details, see Postprocessing (p. 167). See the following figure for a plot of flow rate vs pressure rise for a blower.



11.4. Frame Change Models

When specifying domain interfaces in CFX-Pre, you must select the type of analysis that will be carried out in the solver. The choices are:

- Frozen Rotor (p. 169)
- Stage (Mixing-Plane) (p. 169)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Transient Rotor-Stator (p. 169)

11.4.1. Frozen Rotor

The Frozen Rotor model treats the flow from one component to the next by changing the frame of reference while maintaining the relative position of the components. Usually, periodicity is used to reduce the number of components to a subset that has approximately the same pitch ratio as the full geometry. To account for differences in pitch ratio between the subset and the full geometry, the flow passing through the interface is scaled according to the net pitch ratio of the subsets.

The Frozen Rotor model must be used for non-axisymmetric flow domains, such as impeller/volute, turbine/draft tube, propeller/ship and scroll/volute cases. It can also be used for axial compressors and turbines. The Frozen Rotor model has the advantages of being robust, using less computer resources than the other frame change models, and being well suited for high blade counts. The drawbacks of the model include inadequate prediction of physics for local flow values and sensitivity of the results to the relative position of the rotor and stator for tightly coupled components.

11.4.2. Stage (Mixing-Plane)

The Stage (Mixing-Plane) model circumferentially averages the fluxes in bands and transmits the average fluxes to the downstream component. Possible applications include axial turbines, compressors and pumps, as well as fans and torque converters. The model is useful for large pitch ratios and still takes a relatively short time to solve. The model is not suitable for applications with tight coupling of components and/or significant wake interaction effects and may not accurately predict loading.

11.4.3. Transient Rotor-Stator

The Transient Rotor-Stator model takes into account all of the transient flow characteristics. A sliding interface is used to allow a smooth rotation between components. As with the Frozen Rotor model, the Transient Rotor-Stator model scales the flow from one component to the next in order to account for a non-unity net pitch ratio. This model is robust and yields high accuracy predictions of loading. The drawbacks include high computational cost and large amounts of storage required to hold the transient data.

Note:

The dynamic re-intersection of the interface at the start of each time step may result in a different interface topology, which in turn may require more or less memory. Unlike the static interfaces (Frozen Rotor, Stage), which are only intersected once (first time step of a serial run or partitioner in a parallel run), the initial memory estimate might not be sufficient for the whole run. To avoid potential memory problems, it might be necessary to start the simulation with more conservative (larger) memory factors.

11.5. Domain Interface Setup

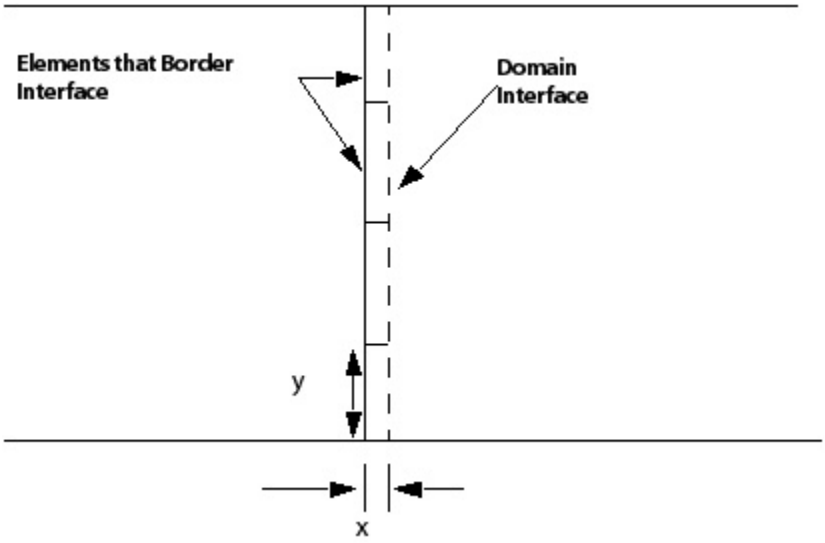
The setup of domain interfaces is an important consideration when defining a problem. The following section outlines some approved practices for use in turbomachinery applications.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11.5.1. General Considerations

- Domain interfaces should typically be placed midway between the rotor and stator for turbomachinery cases.
- To avoid numerical errors, the aspect ratio of elements on the domain interface should be between 0.1:1 and 10:1, as measured by x/y in Figure 11.2: Element Aspect Ratio at Domain Interface (p. 170).
- Where circular domain interfaces exist, they must be axisymmetric in shape as shown in Figure 11.3: Impeller/Volute (p. 171).

Figure 11.2: Element Aspect Ratio at Domain Interface

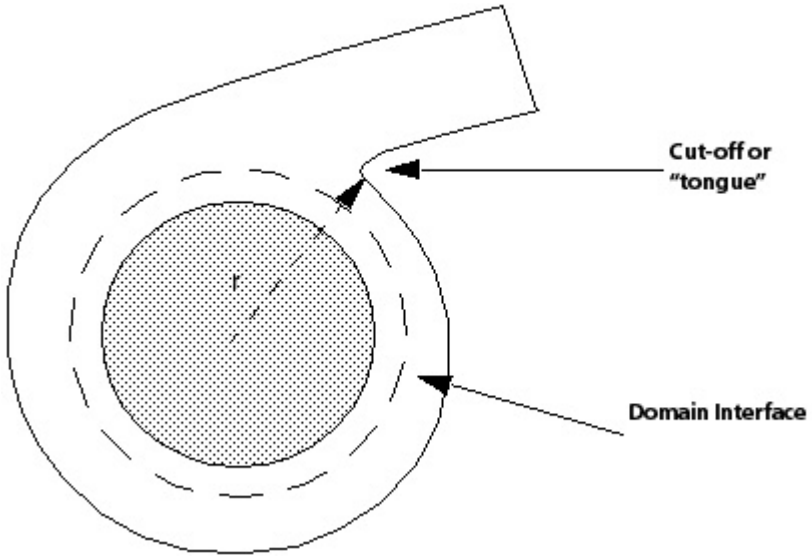


11.5.2. Case 1: Impeller/Volute

A basic impeller/volute sketch is shown in Figure 11.3: Impeller/Volute (p. 171). The edge of the inner circle shows the maximum extent of the impeller blades. A good practice here is to create the domain interface halfway across the narrowest gap between the blade and volute wall. This usually occurs around the cut-off or "tongue" illustrated in the diagram.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

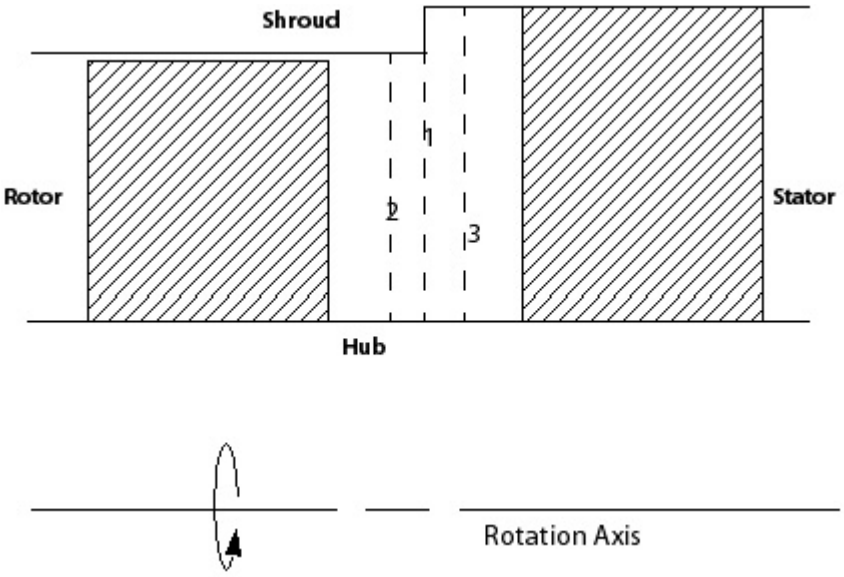
Figure 11.3: Impeller/Volute



11.5.3. Case 2: Step Change Between Rotor and Stator

For the case shown, there is a step change in the passage height between the rotor and stator. A common choice for placement of the interface would be choice 1. However, take care with this setup because the non-overlap regions of the interface should be specified as walls. A better alternative may be to use a domain interface upstream or downstream of the step change, at position 2 or position 3.

Figure 11.4: Possible Domain Interface Positions with Step Change in Passage Height



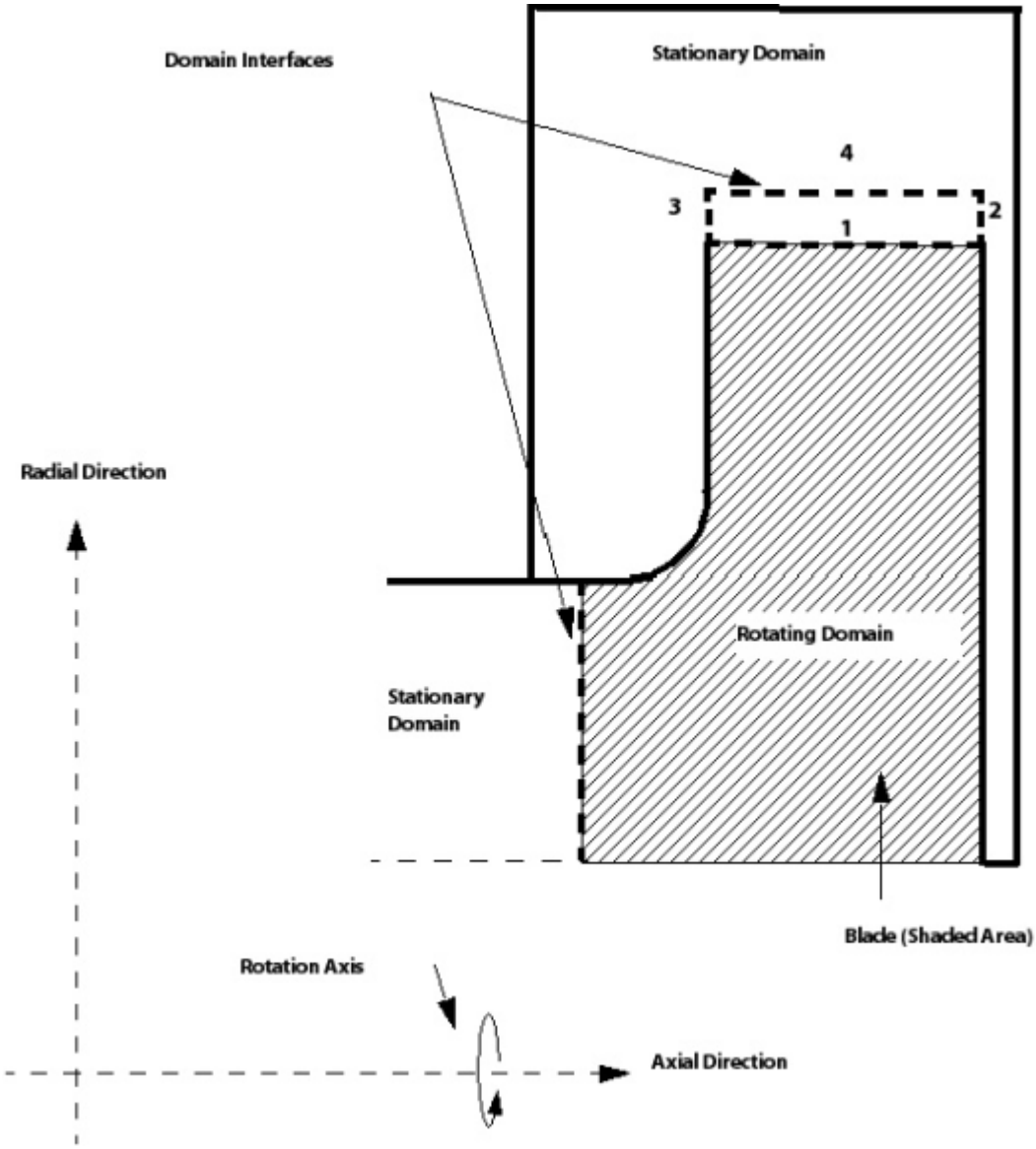
11.5.4. Case 3: Blade Passage at or Close to the Edge of a Domain

Figure 11.5: Radial Compressor (p. 172) shows a blade that extends to the edge of the rotating domain. Although it is convenient to place a domain interface at the blade edge (1), this can result in unrealistic results (The area of the interface would be reduced on one side where the interface is displaced

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

by the blade edge, resulting in an inaccurate pitch change calculation. Also, in the case of a stage interface, the wake would be mixed out at the trailing edge.) A better arrangement is to extend the rotating domain away from the blade edge. Domain Interfaces can then be created at (2), (3), and (4).

Figure 11.5: Radial Compressor

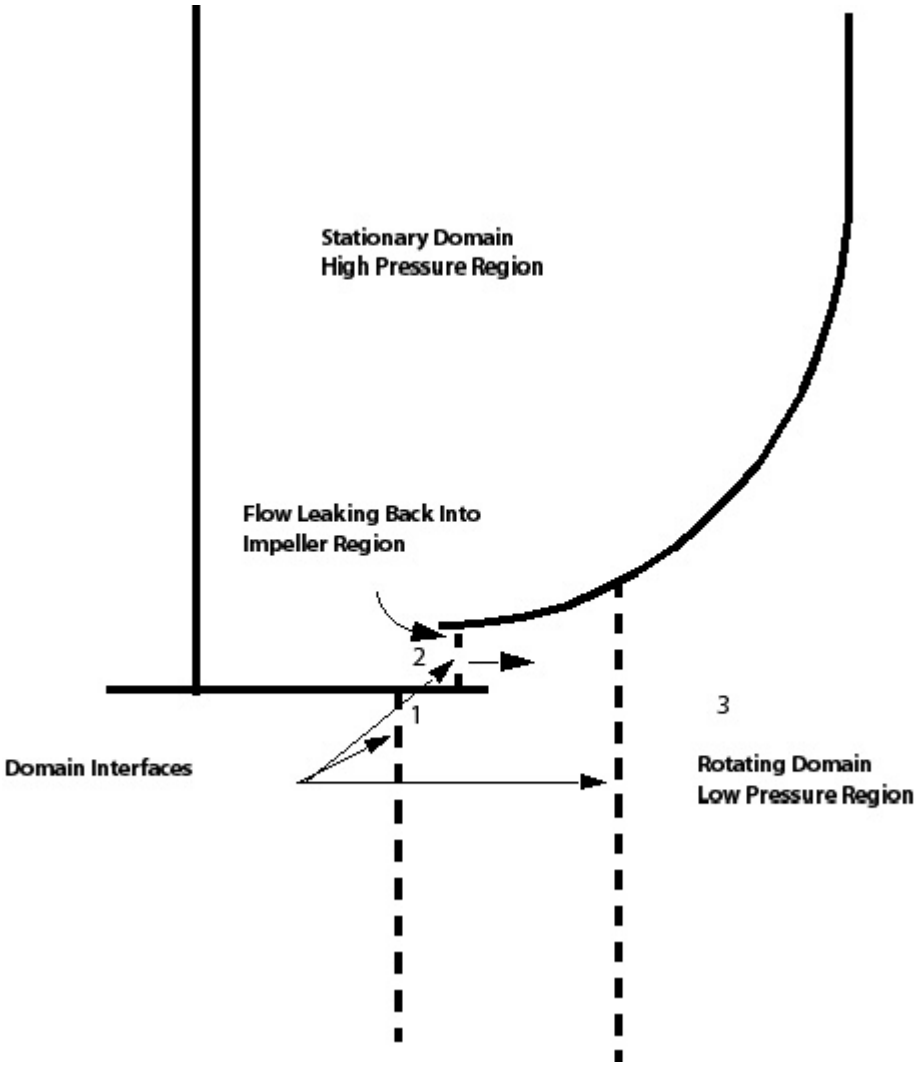


11.5.5. Case 4: Impeller Leakage

A close-up view of part of Figure 11.5: Radial Compressor (p. 172), which models flow leaking from a volute back into the impeller region. To model the feature, you can use two domain interfaces (at positions 1 and 2), or a single domain interface downstream of the leak (position 3).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 11.6: Flow Leakage Through Gap Near Impeller Inlet



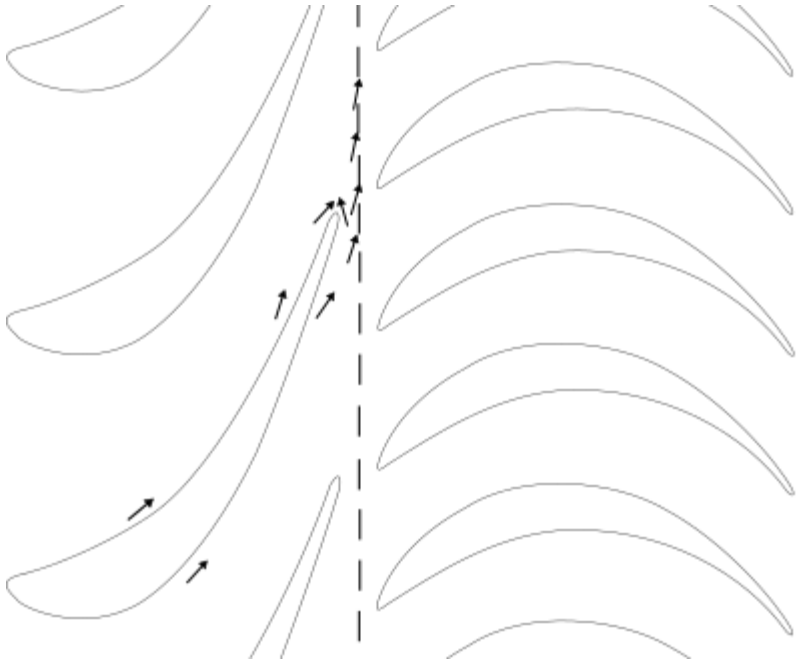
11.5.6. Case 5: Domain Interface Near Zone of Reversed Flow

Be wary of flow moving backwards across stage or frozen rotor interfaces. Because of the approximations implied by these interfaces, flow moving upstream and downstream on the same interface will lead to unphysical results. Try relocating the interface to prevent this from occurring.

As an example, Figure 11.7: Domain Interface Between Blade Rows in an Axial Machine (p. 174) shows two blade rows of an axial machine with a frozen rotor interface between them. The flow moves from left to right everywhere except in a small region just downstream of the trailing edge of the first row of blades. In this case, the domain interface, shown as a dashed line, should be located to the right of this region, as shown.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 11.7: Domain Interface Between Blade Rows in an Axial Machine



11.6. Transient Blade Row

This section describes the following topics:

11.6.1. Steady versus Transient Blade Row Analysis
11.6.2. Full Model Simulation versus Reduced Geometry Simulation (Pitch Change Models)
11.6.3. Selecting an Appropriate Transient Blade Row Model with Pitch Change
11.6.4. Convergence and Solution Monitoring of Transient Blade Row Flow Problems
11.6.5. Boundary Conditions in Blade Row Simulation
11.6.6. Transient versus Harmonic Solution Method

11.6.1. Steady versus Transient Blade Row Analysis

For many turbomachinery problems, steady-state stage simulations involving frame change/mixing models such as Stage or Frozen Rotor, are sufficient to obtain machine performance and analyze flow details. However, when turbomachine component interactions are strong due to close proximity or high speed flow, transient blade row simulation become necessary to improve the prediction of turbomachine aerodynamic performance.

A transient blade row simulation is also needed for aeromechanical (for example, flutter and forced response), aerothermodynamic (for example, hot streak analysis) and aeroacoustics analysis.

In general, a transient blade row analysis is more demanding on computer resources than a steady-state analysis.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11.6.2. Full Model Simulation versus Reduced Geometry Simulation (Pitch Change Models)

To reduce costs (CPU & memory) of a transient blade row simulation, you can model a turbomachinery flow on a small sector of the machine and apply a pitch change model (such as Profile Transformation, Time Transformation or Fourier Transformation) to account for the difference in pitch between blade rows.

In general, the transient blade row simulation becomes more efficient on the small sector model with respect to the full wheel model as the number of blade counts in the original machine increases. For example, in some cases it is better to do a transient blade row simulation on a full wheel rather than a reduced geometry with pitch change models if the blade count is very small. Axial machines can have a large number of blades per row (for example, 100 blades or more) so the cost savings with transient blade row methods can be very large (of the order of 100x for such a machine). For radial machines, the blade count is typically much smaller (for example, 5-20 blades) resulting in reduced potential cost savings for the pitch change models.

11.6.3. Selecting an Appropriate Transient Blade Row Model with Pitch Change

Ansys CFX provides a variety of transient blade row pitch change models to be used on a diverse range of turbomachinery flow problems:

11.6.3.1. Profile Transformation

11.6.3.2. Time Transformation

11.6.3.3. Fourier Transformation

11.6.3.1. Profile Transformation

The Profile Transformation method should be used to predict performance and improve on steady-state stage simulations. It can be used to capture the strong interactions between components.

The Profile Transformation method can be used for single-stage or multistage machines for all flow physics including liquids and gases, at any range of Mach number.

The Profile Transformation method can be used on small to moderate pitch-ratio configurations. There is no formal limit on the pitch ratio for the Profile Transformation method, but the model error grows proportionally with the pitch ratio between components. For large pitch-ratio modeling, the error can be minimized by adding more than one blade passage per row to reduce the pitch ratio of the ensemble.

The Profile Transformation method can also be mixed freely with stage interfaces, as well as with the Time Transformation method, making it a highly useful and flexible approximation.

11.6.3.2. Time Transformation

The Time Transformation method should be used to predict both performance and blade passing frequency.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For small to moderate pitch ratios (0.75-1.4), usually one blade passage per row is needed. However, this pitch-ratio range can be substantially reduced for a very low rotation machine. If stability of the method is reached, then adding a second blade passage may be necessary to regain stability.

The Time Transformation method usually reaches periodically established flow regimes after just a few blade passing periods.

The Time Transformation method does not make any prior assumptions about the main frequencies involved in the simulation. However, this pitch-change method is only applicable to compressible flows.

For multistage modeling, the Time Transformation TRS interface can be combined with Profile Transformation TRS. Sometimes the accuracy provided by the Time Transformation method can impact performance predictions. For subsonic compressors, the Time Transformation method gives a more accurate resolution of the surge point, but perhaps gives a similar performance prediction as Profile Transformation and even steady-state analysis away from stall. For transonic compressors, the Time Transformation method can give an overall improvement in the accuracy of the performance map due to the strong interactions between components. This is caused by the shocks interacting between components. The choked mass flow and the surge point may both change noticeably for the Time Transformation method compared to Profile Transformation method and steady-state method.

Time Transformation TRS can be combined with stage interfaces to model multistage turbomachines.

11.6.3.3. Fourier Transformation

The Fourier Transformation method is similar to the Time Transformation method however, it can be used on very large pitch ratio configurations (for example, fan inlet distortion problems or fan bypass regions where the pitch ratio between the fan blade and the downstream stator row is very large).

The Fourier Transformation method is capable of modeling both compressible and incompressible flows. For optimum computational efficiency, it is recommended that you use the Fourier Transformation method when the pitch ratio is too large to be modeled sensibly with the Time Transformation method.

The Fourier Transformation method is the best method to use for blade flutter analysis. The Fourier Transformation method allows you to specify the nodal diameter (phase shift) for fluttering blade rows.

A transient blade row simulation with the Fourier Transformation method is more efficient with respect to the reference solution when the machine has a large number of blades per row. Therefore, it may not be useful to perform transient blade row simulation with the Fourier Transformation method on turbomachines with a low blade count.

The Fourier Transformation TRS interface can be combined with stage interfaces to model multistage turbomachines but can not be combined with Profile Transformation TRS or Time Transformation TRS interfaces.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

11.6.4. Convergence and Solution Monitoring of Transient Blade Row Flow Problems

Due to the periodic nature of the flow in blade row configurations, ensure that the solution is monitored sufficiently in order to determine when a transient periodic state has been reached. The following monitoring options can be used:

- Use flow field monitors to check pressure, temperature and velocity variation. The solution is usually deemed converged when the monitor repeats the same pattern over a common period. Sometimes local flow field monitors do not show perfect repeatability within a common period due to local flow instabilities. The **Frequency Filtering** option was added to the Fourier Transformation model to avoid instabilities. The setting for enabling frequency filtering is described in Frequency Filtering in the *CFX-Pre User's Guide*.

- Use monitors of integrated quantities such as pressure surface loads, stage pressure ratio, or stage efficiency. Integrated quantity monitors tend to be less influenced by local instabilities and can give a better sense of the overall solution convergence than local flow field monitors. Monitoring of instantaneous integrated quantities is not recommended for Time Transformation. Instead, you should monitor integrated quantities using monitor averaging.

- Use monitors on inlet and outlet flow rates.

- Use monitor averaging to assess the convergence of the periodic monitors. By selecting an appropriate averaging range, you can monitor the average of the variation to determine if the solution has reached a transient periodic state.

Ensure that a transient solution is converging properly at each time step:

- Check that the residuals of all equations are converging sufficiently at every time step.

- For time accurate solutions, convergence must be achieved without reaching the maximum number of coefficient loops. If convergence is not achieved, a mesh issue could be the problem preventing the solution from converging, requiring a re-examination of the mesh quality.

Also see Using Interrupt Control in Cases with Transient Convergence Behavior in the *CFX-Solver Modeling Guide*.

11.6.5. Boundary Conditions in Blade Row Simulation

The following sections provide advice on applying boundary conditions:

11.6.5.1. Steady-state Analysis

11.6.5.2. Transient Analysis

11.6.5.1. Steady-state Analysis

To obtain the speed line performance of a turbomachine stage in steady-state analysis you can use one of the following approaches:

- At the exit use a mass flow boundary from a stall point toward a near-choke point and pressure boundary from a near-choke point to a deep-choke point.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Use the exit corrected mass flow rate boundary condition, which enables you to traverse the speed line without changing the boundary condition type.

11.6.5.2. Transient Analysis

In a transient simulation it is recommended that you use a pressure boundary to traverse the speed line. This is particularly true when you are modeling a blade row flow problem on small sector of the wheel and using one of the pitch-change models.

It is very important to note that a mass flow boundary at the exit should not be used with an incompressible flow setting. For a more accurate representation of the pressure field at the outlet of an axial machine, use a radially distributed pressure profile. If you want to compare a transient solution obtained from reduced geometry having pitch-change models (for example, Time Transformation or Fourier Transformation) with a solution obtained from a full domain model, then it is best to select a boundary condition setup that minimizes the differences between the two simulations. Therefore, on the exit boundary it is recommended that you use a pressure profile without circumferential variation (thus, set `Pressure Profile Blend = 1`). The pressure profile can vary radially.

11.6.6. Transient versus Harmonic Solution Method

As mentioned in the theory guide, the turbomachinery flow is usually transient and periodic. Therefore, the harmonic solution method can be applied to converge the flow solution to a steady-periodic state faster than marching the solution in time until a steady-periodic state is reached.

True transient analyses differ from harmonic analyses:

- The flow solution obtained via a time-marching method typically has large frequency content and captures most of the flow characteristics, particularly when not using any pitch-change approximation. The amount of frequency content and flow details captured by transient flow are controlled by the true time-step size or the number of time steps per period.

- For a Harmonic Analysis, the solution contains only the frequency associated with targeted fundamental frequencies and retained harmonics (see Harmonic Analysis in the *CFX-Solver Theory Guide*). The targeted fundamental frequency, such as the blade passing frequency, is usually known in advance. Frequencies that are not associated with the blade passing frequency are not known before obtaining the HA solution and therefore will not be captured; they will be filtered out and not be part of the solution.

Additional factors that affect the accuracy and efficiency of the harmonic solution method:

- Number of harmonics retained (M): The more complex the flow features in the transient flow (for example, sharp discontinuities), the greater the number of harmonics that must be retained in order to resolve the flow features. Typically, if the flow contains no discontinuity or translating sharp wakes then a single harmonic is sufficient to resolve the flow. Flow that contains moving discontinuities and/or wakes requires three to five harmonics to be retained. Of course, the more harmonics that are retained, the more expensive the Harmonic Analysis becomes, reducing efficiency with respect to a true transient simulation.

- Number of pseudo time steps per period: In order to obtain a harmonic solution quickly, the number of pseudo-time steps should be minimized. This can be done by increasing the pseudo-time step size or by lowering the number of pseudo-time steps per period, within limits: an excess-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ively large pseudo-time step and/or small number of pseudo-time steps per period can adversely affect solution stability. The number of pseudo-time steps per period should normally be in the range of 15 to 30.

Note:

To properly measure the accuracy and efficiency of the harmonic solution method with respect to the transient time-marching solution method, it is of paramount importance that you run the time-marching solution with a sufficient number of time steps per period (to achieve a time-step-independent solution) and a sufficient number of blade passings or cycles to reach a fully converged steady-periodic state.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 12: CFX Best Practices Guide for Turbulence

The following topics are discussed:

12.1. Scale-Resolving Simulations in Ansys CFD

12.2. RANS Turbulence Modeling in Ansys CFD

12.3. Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD

12.1. Scale-Resolving Simulations in Ansys CFD

While today's CFD simulations are mainly based on Reynolds-Averaged Navier-Stokes (RANS) turbulence models, it is becoming increasingly clear that certain classes of flows are better covered by models in which all or a part of the turbulence spectrum is resolved in at least a portion of the numerical domain. Such methods are termed Scale-Resolving Simulation (SRS) models in this document.

There are two main motivations for using SRS models in favor of RANS formulations. The first reason for using SRS models is the need for additional information that cannot be obtained from the RANS simulation. Examples are acoustics simulations where the turbulence generates noise sources, which cannot be extracted with accuracy from RANS simulations. Other examples are unsteady heat loading in unsteady mixing zones of flow streams at different temperatures, which can lead to material failure, or multi-physics effects like vortex cavitation, where the unsteady turbulence pressure field is the cause of cavitation. In such situations, the need for SRS can exist even in cases where the RANS model would in principle be capable of computing the correct time-averaged flow field.

The second reason for using SRS models is related to accuracy. It is known that RANS models have their limitations in accuracy in certain flow situations. RANS models have shown their strength essentially for wall-bounded flows, where the calibration according to the law-of-the-wall provides a sound foundation for further refinement. For free shear flows, the performance of RANS models is much less uniform. There is a wide variety of such flows, ranging from simple self-similar flows such as jets, mixing layers, and wakes to impinging flows, flows with strong swirl, massively separated flows, and many more. Considering that RANS models typically already have limitations covering the most basic self-similar free shear flows with one set of constants, there is little hope that even the most advanced Reynolds Stress Models (RSM) will eventually be able to provide a reliable foundation for all such flows. (For an overview of RANS modeling, see Durbin, Pettersson and Reif, 2003 5 (p. 269); Wilcox, 2006 39 (p. 274); or Hanjalic and Launder, 2011 13 (p. 270).)

For free shear flows, it is typically much easier to resolve the largest turbulence scales, as they are of the order of the shear layer thickness. In contrast, in wall boundary layers the turbulence length scale near the wall becomes very small relative to the boundary layer thickness (increasingly so at higher Re numbers). This poses severe limitations for Large Eddy Simulation (LES) as the computational effort required is still far from the computing power available to industry (Spalart, 1997 30 (p. 273)). (For an overview of LES modeling, see Geurts, 2004 12 (p. 270), and Wagner et al., 2007 36 (p. 274).) For this reason, hybrid models are under development where large eddies are resolved only away from walls and the wall boundary layers are covered by a RANS model. Examples of such global hybrid models are Detached Eddy Simulation (DES, see Spalart, 2000 31 (p. 273)) or Scale-Adaptive Simulation (SAS, see

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Menter and Egorov, 2010 18 (p. 271)). More recent developments include the Shielded Detached Eddy Simulation (SDES) and the Stress-Blended Eddy Simulation (SBES).

A further step is to apply a RANS model only in the innermost part of the wall boundary layer and then to switch to a LES model for the main part of the boundary layer. Such models are termed Wall-Modeled LES (WMLES) (for example, Shur et al., 2008 27 (p. 273)). Finally, for large domains, it is frequently necessary to cover only a small portion with SRS models, while the majority of the flow can be computed in RANS mode. In such situations, zonal or embedded LES methods are attractive because they enable you to specify ahead of time the region where LES is required. Such methods are typically not new models in the strict sense, but enable the combination of existing models/technologies in a flexible way in different portions of the flowfield. Important elements of zonal models are interface conditions, which convert turbulence from RANS mode to resolved mode at pre-defined locations. In most cases, this is achieved by introducing synthetic turbulence based on the length and time scales from the RANS model.

There are many hybrid RANS-LES models, often with somewhat confusing naming conventions, that vary in the range of turbulence eddies they can resolve. For a general overview of SRS modeling concepts, see Fröhlich and von Terzi, 2008 8 (p. 270), Sagaut et al, 2006 25 (p. 272).

SRS models are very challenging in their proper application to industrial flows. The models typically require special attention to various details such as:

- Model selection
- Grid generation
- Numerical settings
- Solution interpretation
- Postprocessing
- Quality assurance

Unfortunately, there is no unique model covering all industrial flows, and each individual model poses its own set of challenges. In general, when using a CFD code, you must understand the intricacies of the SRS model formulation in order to be able to select the optimal model and to use it efficiently. This document is intended to support you in the basic understanding of such models and to provide best practice guidelines for their usage. The discussion is focused on the models available in the Ansys CFD software.

This document is intended as an addition to the code-specific Theory and User Documentation available for both Ansys Fluent and Ansys CFX . The Theory and User Documentation describes in detail how to select and activate these models, so that information is not repeated here. This document is intended to provide a general understanding of the underlying principles and the associated limitations of each of the described modeling concepts. It also covers the types of flows for which the models are suitable as well as flows where they will likely not work well. Finally, the impact of numerical settings on model performance is discussed.

In accordance with the intention of providing recommendations for day-to-day work, several appendices can be found at the end of this document for quick reference of the most important points.

The following topics are discussed:

12.1.1. Scale-Resolving Simulation (SRS) Models – Basic Formulations

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.1.2. Generic Flow Types and Basic Model Selection
12.1.3. Numerical Settings for SRS
12.1.4. Initial and Boundary Conditions
12.1.5. Postprocessing and Averaging
12.1.6. Summary
12.1.7. References for Scale-Resolving Simulations

12.1.1. Scale-Resolving Simulation (SRS) Models – Basic Formulations

In the Ansys CFD codes the following SRS models are available:

1. Scale-Adaptive Simulation (SAS) models
   - SAS-SST model (Fluent, CFX)
2. Detached Eddy Simulation (DES) models
   - DES-SA (DDES) model (Fluent)
   - DES-SST model (Fluent (also DDES), CFX)
   - Realizable k–ε DES model (Fluent)
3. Shielded Detached Eddy Simulation (SDES)

   All ω-equation based 2-equation models in Fluent.
4. Stress-Blended Eddy Simulation (SBES)

   All ω-equation based 2-equation models in Fluent and CFX.
5. Large Eddy Simulation (LES)
   - Smagorinsky-Lilly model (+dynamic) (Fluent, CFX)
   - WALE model (Fluent, CFX)
   - Kinetic energy subgrid model dynamic (Fluent)
   - Algebraic Wall Modeled LES (WMLES) (Fluent, CFX)
6. Embedded LES (ELES) mode
   - Combination of all RANS models with all non-dynamic LES models (Fluent)
   - Zonal forcing model (CFX)
7. Synthetic turbulence generator
   - Vortex method (Fluent)
   - Harmonic Turbulence Generator (HTG) (CFX)

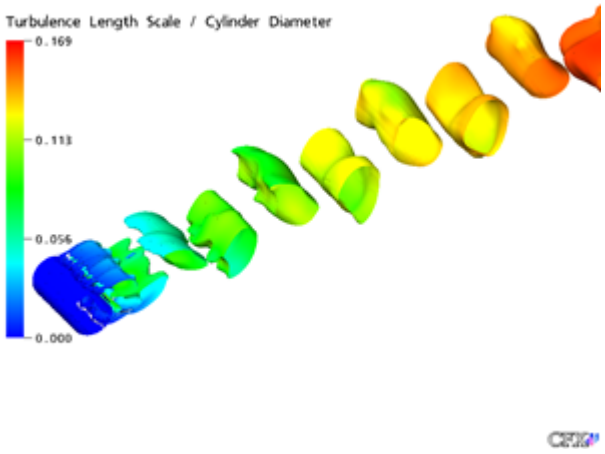
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.1.1.1. Scale-Adaptive Simulation (SAS)

In principle, all RANS models can be solved in unsteady mode (URANS). Experience shows, however, that classical URANS models do not provide any spectral content, even if the grid and time step resolution are sufficient for that purpose. It has long been argued that this behavior is a natural outcome of the RANS averaging procedure (typically time averaging), which eliminates all turbulence content from the velocity field. By that argument, it has been concluded that URANS can work only in situations of a "separation of scales," where only time variations that are of much lower frequency than turbulence are resolved. An example would be the flow over a slowly oscillating airfoil, where the turbulence is modeled entirely by the RANS model and only the slow super-imposed motion is resolved in time. A borderline case for this scenario is the flow over bluff bodies, like a cylinder in crossflow. For such flows, the URANS simulation provides unsteady solutions even without an independent external forcing. The frequency of the resulting vortex shedding is not necessarily much lower than the frequencies of the largest turbulent scales. This scenario is depicted in Figure 12.1 (p. 184), which shows that URANS models (in this case SST) produce a single mode vortex shedding even at a relatively high Re number of $\mathrm{Re}=10^6$. The vortex stream extends far into the cylinder wake, maintaining a single frequency. This result is in contradiction to experimental observations of a broadband turbulence spectrum.

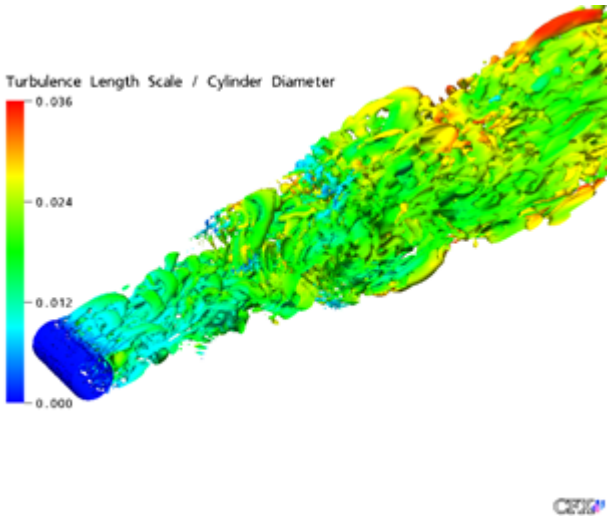
As shown in a series of publications (for example, Menter and Egorov, 2010 18 (p. 271), Egorov et al., 2010 6 (p. 269)), a class of RANS models can be derived based on a theoretical concept dating back to Rotta (see Rotta, 1972 24 (p. 272)). These models perform like standard RANS models in steady flows, but enable the formation of a broadband turbulence spectrum for certain types of unstable flows (for the types of flows, see Section 12.1.2 (p. 204)). Such models are termed Scale-Adaptive Simulation (SAS) models. This scenario is illustrated by Figure 12.2 (p. 185), which shows the same simulation as in Figure 12.1 (p. 184) but with the SAS-SST model. The behavior seen in Figure 12.1 (p. 184) is therefore not inherent to all RANS models, but only to those derived in a special fashion.

Figure 12.1: URANS computations of a flow past a circular cylinder (SST model)



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.2: SAS simulation of flow past a circular cylinder (SAS-SST model)



The SAS concept is described in much detail in the cited references and will not be repeated here. However, the basic model formulation must be provided for a discussion of the model's characteristics. The difference between standard RANS and SAS models lies in the treatment of the scale-defining equation (typically ε-, ω-, or L_t-equation). In classic RANS models, the scale equation is modeled based on an analogy with the k-equation using simple dimensional arguments. The scale equation of SAS models is based on an exact transport equation for the turbulence length scale as proposed by Rotta. This method was re-visited by Menter and Egorov, 2010 18 (p. 271) and avoids some limitations of the original Rotta model. As a result of this re-formulation, it was shown that the second derivative of the velocity field must be included in the source terms of the scale equation. The original SAS model (Menter and Egorov, 2010 18 (p. 271)) was formulated as a two-equation model, with the variable $\Phi=\sqrt{k}\,L_t$ for the scale equation:

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho \bar{U}_j k)}{\partial x_j}=P_k-c_\mu^{3/4}\rho\frac{k}{\Phi^2}+\frac{\partial}{\partial x_j}\left(\frac{\mu_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right) \tag{12.1}$$

$$\frac{\partial(\rho \Phi)}{\partial t}+\frac{\partial(\rho \bar{U}_j \Phi)}{\partial x_j}=\frac{\Phi}{k}P_k\left(\zeta_1-\zeta_2\left(\frac{L_t}{L_{vK}}\right)^2\right)-\zeta_3\rho k+\frac{\partial}{\partial x_j}\left(\frac{\mu_t}{\sigma_\Phi}\frac{\partial \Phi}{\partial x_j}\right) \tag{12.2}$$

$$\begin{aligned}
\mu_t&=c_\mu^{1/4}\rho\Phi\\
L_{vK}&=\kappa\left|\frac{\overline{U}'}{\overline{U}''}\right|\\
\overline{U}'&=S=\sqrt{2S_{ij}S_{ij}}\\
\overline{U}''&=\sqrt{\frac{\partial^2\bar{U}_i}{\partial x_j\partial x_j}\frac{\partial^2\bar{U}_i}{\partial x_k\partial x_k}}\\
S_{ij}&=\frac{1}{2}\left(\frac{\partial\bar{U}_i}{\partial x_j}+\frac{\partial\bar{U}_j}{\partial x_i}\right)\\
L_t&=\frac{\Phi}{\sqrt{k}}
\end{aligned} \tag{12.3}$$

The main new term is the one including the von Karman length scale L_{vK}, which does not appear in any standard RANS model. The second velocity derivative allows the model to adjust its length

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

scale to those structures already resolved in the flow. This functionality is not present in standard RANS models. This leads to the behavior shown in Figure 12.2 (p. 185), which agrees more closely with the experimental observations for such flows.

The L_{vK} term can be transformed and implemented into any other scale-defining equation resulting in SAS capabilities as in the case of the SAS-SST model. For the SAS-SST model, the additional term in the ω-equation resulting from the transformation has been designed to have no (or at least minimal) effect on the SST model's RANS performance for wall boundary layers. It can have a moderate effect on free shear flows (Davidson, 2006 4 (p. 269)).

The SAS model will remain in steady RANS mode for wall bounded flows, and can switch to SRS mode in flows with large and unstable separation zones (see Section 12.1.2 (p. 204)).

12.1.1.2. Detached Eddy Simulation (DES)

Detached Eddy Simulation (DES) was introduced by Spalart and co-workers (Spalart et al., 1997 30 (p. 273), 2000 31 (p. 273), Travin et al., 2000 34 (p. 274), Strelets, 2001 33 (p. 273)), to eliminate the main limitation of LES models by proposing a hybrid formulation that switches between RANS and LES based on the grid resolution provided. By this formulation, the wall boundary layers are entirely covered by the RANS model and the free shear flows away from walls are typically computed in LES mode. The formulation is mathematically relatively simple and can be built on top of any RANS turbulence model. DES has attained significant attention in the turbulence community as it was the first SRS model that allowed the inclusion of SRS capabilities into common engineering flow simulations.

Within DES models, the switch between RANS and LES is based on a criterion like the following:

$$\begin{aligned} C_{DES}\Delta_{\max} > L_t &\rightarrow \quad RANS \\ C_{DES}\Delta_{\max} \le L_t &\rightarrow \quad LES \\ \Delta_{\max} = \max(\Delta_x, \Delta_y \Delta_z) & \end{aligned} \tag{12.4}$$

where $\Delta_{\max}$ is the maximum edge length of the local computational cell. The actual formulation for a two-equation model (for example, the k-equation of the $k-\omega$ model) is:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho \bar{U}_j k)}{\partial x_j} = P_k - \rho \frac{k^{3/2}}{\min(L_t, C_{DES}\Delta_{\max})} + \frac{\partial}{\partial x_j}\left(\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right) \tag{12.5}$$

$$L_t = \frac{k^{3/2}}{\varepsilon} = \frac{\sqrt{k}}{\beta^* \omega} \tag{12.6}$$

As the grid is refined below the limit $\Delta_{\max} \le L_t$ the DES-limiter is activated and switches the model from RANS to LES mode. For wall boundary layers, this translates into the requirement that the RANS formulation is preserved as long as the following condition holds: $\Delta_{\max} > \delta$, where δ is the boundary layer thickness. The intention of the model is to run in RANS mode for attached flow regions, and to switch to LES mode in detached regions away from walls. This switch suggests that the original DES formulation, as well as its later versions, require a grid and time step resolution to be of LES quality once they switch to the grid spacing as the defining length scale. Once the limiter is activated, the models lose their RANS calibration and all relevant turbulence information must be resolved. For this reason, for example in free shear flows, the DES approach offers no computa-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

tional savings over a standard LES model. However, it allows you to avoid the high computing costs of covering the wall boundary layers in LES mode.

It is also important to note that the DES limiter can already be activated by grid refinement inside attached boundary layers. This is undesirable as it affects the RANS model by reducing the eddy viscosity; this can lead to Grid-Induced Separation (GIS), as discussed by Menter and Kuntz, 2002 19 (p. 271), where the boundary layers can separate at arbitrary locations depending on the grid spacing. In order to avoid this limitation, the DES concept has been extended to Delayed-DES (DDES) by Spalart et al., 2006 32 (p. 273), following the proposal of Menter and Kuntz, 2002 19 (p. 271) of "shielding" the boundary layer from the DES limiter. The DDES extension was also applied to the DES-SA formulation resulting in the DDES-SA model, as well as to the SST model giving the DDES-SST model.

For two-equation models, the dissipation term in the k-equation is thereby re-formulated as follows:

$$\begin{aligned} E_{DES} &= \rho \frac{k^{3/2}}{\min\left(L_t, C_{DES}\Delta\right)} \\ &= \rho \frac{k^{3/2}}{L_t \min\left(L_t, C_{DES}\frac{\Delta}{L_t}\right)} \\ &= \rho \frac{k^{3/2}}{L_t} \max\left(1; \frac{L_t}{C_{DES}\Delta}\right) \end{aligned} \tag{12.7}$$

$$E_{DDES} = \rho \frac{k^{3/2}}{L_t} \max\left(1; \frac{L_t}{C_{DES}\Delta}\left(1 - F_{DDES}\right)\right) \tag{12.8}$$

The function F_{DDES} is designed in such a way as to give $F_{DDES}=1$ inside the wall boundary layer and $F_{DDES}=0$ away from the wall. The definition of this function is intricate as it involves a balance between proper shielding and not suppressing the formation of resolved turbulence as the flow separates from the wall. As the function F_{DDES} blends over to the LES formulation near the boundary layer edge, no perfect shielding can be achieved. The limit for DDES is typically in the range of $\Delta_{\max} > 0.2 \cdot \delta$ and therefore allows for meshes where $\Delta_{\max}$ is a factor of five smaller than for DES, without negative effects on the RANS-covered boundary layer. However, even this limit is frequently reached and GIS can appear even with DDES.

There are a number of DDES models available in Ansys CFD. They follow the same principal idea with respect to switching between RANS and LES mode. The models differ therefore mostly by their RANS capabilities and should be selected accordingly.

12.1.1.3. Shielded Detached Eddy Simulation (SDES)

The SDES formulation is a member of the DDES model family, but offers alternatives for the shielding function and the definition of the grid scale. The impact on the turbulence model is, as usual, formulated as an additional sink term in the k-equation:

$$\varepsilon_{SDES} = -\beta^* \rho k \omega \cdot F_{SDES} \; with \quad F_{SDES} = \left[\max\left(\frac{L_t}{C_{SDES}\Delta_{SDES}}\left(1 - f_S\right), 1\right) - 1\right] \tag{12.9}$$

The shielding function f_S provides much stronger shielding than the corresponding F_{DDES} function above. For this reason, the natural shielding of the model based on the mesh length definition, Δ, can be reduced. The mesh length scale used in the SDES model is defined as follows:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\Delta_{SDES}=\max\left(\sqrt[3]{Vol}\,,0.2\Delta_{\max}\right) \tag{12.10}$$

The first part represents the conventional LES grid length scale definition, and the second part is again based on the maximum edge length as in the DES formulation. However, the factor 0.2 ensures that for highly stretched meshes, the grid length scale is a factor of 5 smaller than for DES/DDES. Since the grid length scale enters quadratically into the definition of the eddy viscosity in LES mode, this means a reduction by a factor of 25 in such cases. It will be shown that this drastically reduces the frequently observed problem of slow 'transition' of DES/DDES models from RANS to LES. Note that the combination of this more 'aggressive' length scale with the conventional DDES shielding function would severely reduce the shielding properties of DDES and is therefore not recommended.

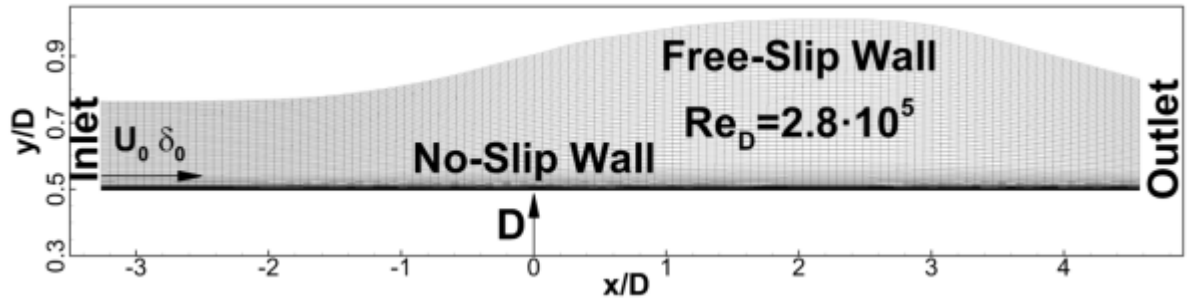
The SDES constant C_{SDES} is also different from in the DES/DDES formulation, where it is calibrated based on decaying isotropic turbulence (DIT) with the goal of matching the turbulence spectrum relative to data after certain running times. However, in engineering flows, one typically has to deal with shear flows, for which a reduced Smagorinsky constant should be used. This is achieved by setting C_{SDES}=0.4. The combination of the re-definition of the grid length scale and the modified constant leads to a reduction in the eddy viscosity by a factor of around 60 in separating shear flows on stretched grids. It will be shown later that this results in a much more rapid transition from RANS to LES.

The shielding function f_S is formulated such that it provides essentially asymptotic shielding on any grid. In flat plate tests, the limit was pushed below $\Delta_{SDES}\sim 0.01\cdot\delta$.

The following test case shows the improved shielding properties of SDES/SBES models relative to DDES. The flow is a diffuser flow in an axisymmetric geometry featuring a small separation bubble. Due to the adverse pressure gradient, the boundary layer grows strongly and shielding is difficult to achieve due to the strong increase in L_t.

The computational domain is shown in Figure 12.3: The domain and grid for the separated flow in CS0 diffuser (p. 188). The length of the domain in the streamwise direction is about $7.8\cdot D$ [m] (D is the diameter of the cylinder and $x/D=0$ corresponds to the separation point in the experiment). For this flow, a standard RANS grid is used with steps in the streamwise and circumferential directions of $\Delta x/\delta_0=0.67$ and $r\Delta\varphi/\delta_0=0.5\text{-}1.0$ respectively (the grid step in the circumferential direction is changing in the radial direction due to the axisymmetric geometry). Here $\delta_0=0.07\cdot D$ is the boundary layer thickness at the inlet section. The height of the wall cell is chosen to satisfy the condition $\Delta y^+_w<1$ in the entire domain and around 30 cells cover the boundary layer.

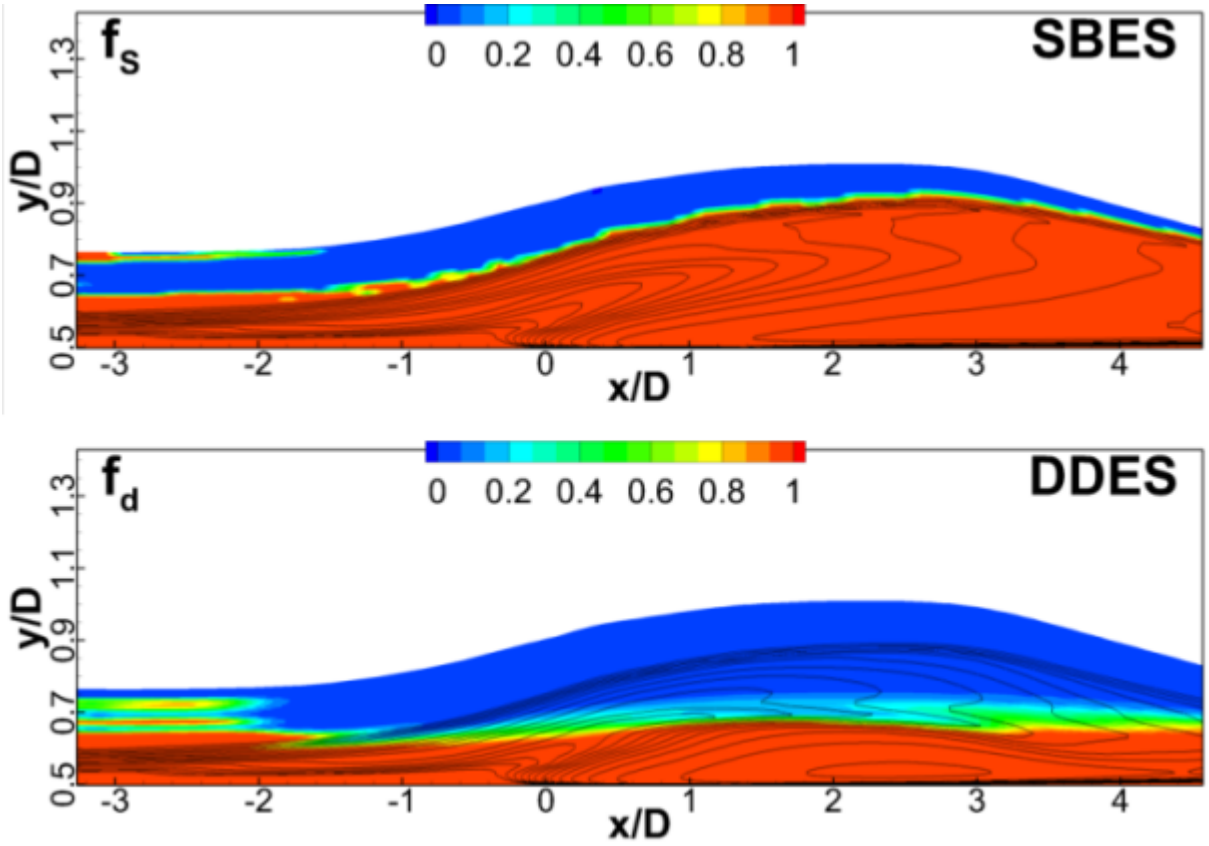
Figure 12.3: The domain and grid for the separated flow in CS0 diffuser



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

As seen from the contours of the SBES and DDES blending functions shown in Figure 12.4: Contours of blending functions overset by vorticity iso-lines for CS=0 diffuser for SBES and DDES models (p. 189), SBES covers the entire boundary layer including the rapid growth area of the boundary layer due to the separation bubble, while DDES preserves only the portion of the domain at the inlet. This means that under adverse pressure gradient conditions, the shielding properties of DDES are substantially impaired.

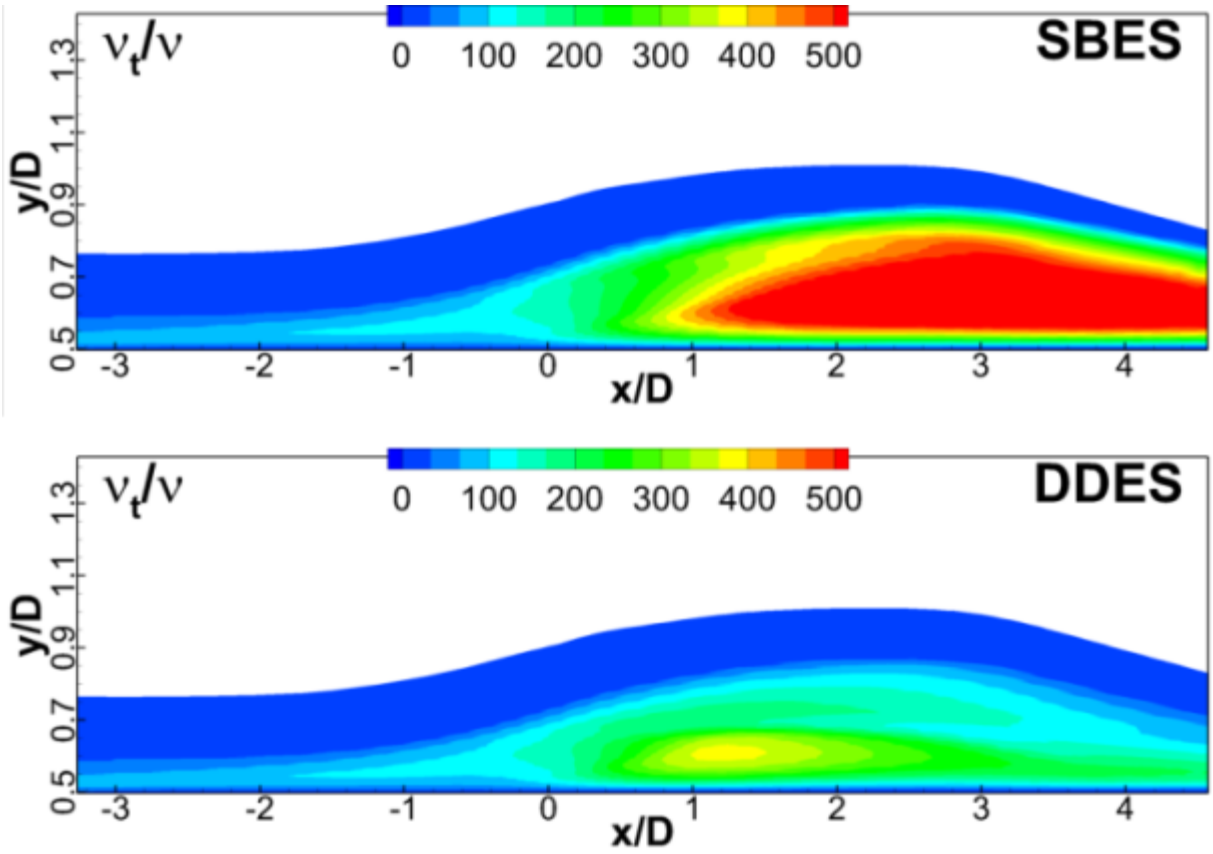
Figure 12.4: Contours of blending functions overset by vorticity iso-lines for CS=0 diffuser for SBES and DDES models



Similar observations can be made for the eddy viscosity fields shown in Figure 12.5: Contours of eddy viscosity ratio for CS0 diffuser for SBES and DDES models (p. 190). The eddy viscosity levels of SBES correspond to those of the SST model (not shown), while the DDES model produces much reduced levels in the adverse pressure gradient region.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.5: Contours of eddy viscosity ratio for CS0 diffuser for SBES and DDES models



12.1.1.4. Stress-Blended Eddy Simulation (SBES)

As stated in the introduction, the SBES model concept is built on the SDES formulation. In addition to SDES, SBES is using the shielding function f_S to explicitly switch between different turbulence model formulations in RANS and LES modes. In general terms, that means for the turbulence stress tensor:

$$\tau_{ij}^{SBES} = f_S \cdot \tau_{ij}^{RANS} + \left(1 - f_S\right) \cdot \tau_{ij}^{LES} \tag{12.11}$$

where τ_{ij}^{RANS} is the RANS part and τ_{ij}^{LES} is the LES part of the modeled stress tensor. In case both model portions are based on eddy-viscosity concepts, the formulation simplifies to:

$$\nu_t^{SBES} = f_S \cdot \nu_t^{RANS} + \left(1 - f_S\right) \cdot \nu_t^{LES} \tag{12.12}$$

Such a formulation would not be feasible without strong shielding. When using the conventional shielding functions from the DDES model, the corresponding model would not be able to maintain a zero pressure gradient RANS boundary layer on any grid.

The SBES model formulation is currently recommended relative to other global hybrid RANS-LES methods. It offers the following advantages:

- Asymptotic shielding of the RANS boundary layers
- Explicit switch to user-specified LES model in LES region

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Rapid 'transition' from RANS to LES region
- Clear visualization of RANS and LES regions based on shielding function
- Wall-modeled LES capability once in LES/WMLES mode

12.1.1.5. Large Eddy Simulation (LES)

The details of different LES models can be found in the User and Theory documentation of the corresponding solvers. As described in Section 12.1.1.5.1 (p. 191), the main purpose of LES models is to provide sufficient damping for the smallest (unresolved) scales. For this reason, it is not advisable to use complex formulations, but to stay with simple algebraic models. The most widely used LES model is the Smagorinsky, 1963 29 (p. 273) model:

$$\nu_t = \left(C_S \Delta\right)^2 S \tag{12.13}$$

The main deficiency of the Smagorinsky model is that its eddy-viscosity does not go to zero for laminar shear flows (only $\partial U / \partial y \neq 0$). For this reason, this model also requires a near-wall damping function in the viscous sublayer. It is desirable to have a LES formulation that automatically provides zero eddy-viscosity for simple laminar shear flows. This is especially important when computing flows with laminar turbulent transition, where the Smagorinsky model would negatively affect the laminar flow. The simplest model to provide this functionality is the WALE (Wall-Adapting Local Eddy-viscosity) model of Nicoud and Ducros, 1999 22 (p. 272). The same effect is also achieved by dynamic LES models, but at the cost of a somewhat higher complexity. None of the classical LES models addresses the main industrial problem of excessive computing costs for wall-bounded flows at moderate to high Reynolds numbers.

There are numerous cases at very low Reynolds numbers where LES can be an industrial option. Under such conditions, the wall boundary layers are likely laminar and turbulence forms only in separated shear layers and detached flow regions. Such situations can be identified by analyzing RANS eddy viscosity solutions for a given flow. In the case where the ratio of turbulence to molecular viscosity $R = \left(\mu_t / \mu\right)$ is smaller than $R \sim 15$ inside the boundary layer, it can be assumed that the boundary layers are laminar and no resolution of near-wall turbulence is required. Such conditions are observed for flows around valves or other small-scale devices at low Reynolds numbers.

LES can also be applied to free shear flows, where resolution requirements are much reduced relative to wall-bounded flows.

12.1.1.5.1. Limitations of Large Eddy Simulation (LES)

In order to understand the motivation for hybrid models, one has to discuss the limitations of Large Eddy Simulation (LES). LES has been the most widely used SRS model over the last decades. It is based on the concept of resolving only the large scales of turbulence and to model the small scales. The classical motivation for LES is that the large scales are problem-dependent and difficult to model, whereas the smaller scales become more and more universal and isotropic and can be modeled more easily.

LES is based on filtering the Navier-Stokes equations over a finite spatial region (typically the grid volume) and aimed at only resolving the portions of turbulence larger than the filter width. Turbulence structures smaller than the filter are then modeled, typically by a simple Eddy Viscosity model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The filtering operation is defined as:

$$\overline{\Phi}=\int_{-\infty}^{\infty}\Phi(\vec{x}')G(\vec{x}-\vec{x}')d\vec{x}$$
$$\int_{-\infty}^{\infty}G(\vec{x}-\vec{x}')d\vec{x}'=1 \tag{12.14}$$

where G is the spatial filter. Filtering the Navier-Stokes equations results in the following form (density fluctuations neglected):

$$\frac{\partial\rho\overline{U}_i}{\partial t}+\frac{\partial\rho\overline{U}_j\overline{U}_i}{\partial x_j}=-\frac{\partial P}{\partial x_i}+\frac{\partial}{\partial x_j}\left(\overline{\tau}_{ij}+\tau_{ij}^{LES}\right) \tag{12.15}$$

The equations feature an additional stress term due to the filtering operation:

$$\tau_{ij}^{LES}=\rho\overline{U}_i\overline{U}_j-\rho\overline{U_i}\overline{U}_j \tag{12.16}$$

Despite the difference in derivation, the additional sub-grid stress tensor is typically modeled as in RANS using an eddy viscosity model:

$$\tau_{ij}^{LES}=\mu_t\left(\frac{\partial\overline{U}_i}{\partial x_j}+\frac{\partial\overline{U}_j}{\partial x_i}\right) \tag{12.17}$$

The important practical implication from this modeling approach is that the modeled momentum equations for RANS and LES are identical if an eddy-viscosity model is used in both cases. The modeled Navier-Stokes equations have no knowledge of their derivation. The only information they obtain from the turbulence model is the level of the eddy viscosity. Depending on that, the equations will operate in RANS or LES mode (or in some intermediate mode). The formal identity of the filtered Navier-Stokes and the RANS equations is the basis of hybrid RANS-LES turbulence models, which can obviously be introduced into the same set of momentum equations. Only the model (and the numerics) have to be switched.

Classical LES models are of the form of the Smagorinsky, 1963 29 (p. 273) model:

$$\mu_t=\rho\left(C_S\Delta\right)^2S \tag{12.18}$$

where Δ is a measure of the grid spacing of the numerical mesh, S is the strain rate scalar and C_S is a constant. This is obviously a rather simple formulation, indicating that LES models will not provide a highly accurate representation of the smallest scales. From a practical standpoint, a very detailed modeling might not be required. A more appropriate goal for LES is not to model the impact of the unresolved scales onto the resolved ones, but to model the dissipation of the smallest resolved scales. This can be seen from Figure 12.6 (p. 194) showing the turbulence energy spectrum of a Decaying Isotropic Turbulence (DIT) test case; that is, initially stirred turbulence in a box, decaying over time (Comte-Bellot and Corrsin, 1971 3 (p. 269)). $E(\kappa)$ is the turbulence energy as a function of wave number κ. Small κ values represent large eddies and large κ values represent small eddies. Turbulence moves down the turbulence spectrum from the small wave number to the high wave numbers. In a fully resolved simulation (Direct Numerical Simulation, or DNS), the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

turbulence is dissipated into heat at the smallest scales ($k \sim 100$ in Figure 12.6 (p. 194)), by viscosity. The dissipation is achieved by:

$$\varepsilon_{DNS} = \nu \frac{\partial U_i}{\partial x_j} \frac{\partial U_i}{\partial x_j} \tag{12.19}$$

where ν is typically a very small kinematic molecular viscosity. The dissipation $\varepsilon_{\mathrm{DNS}}$ is still of finite value as the velocity gradients of the smallest scales are very large.

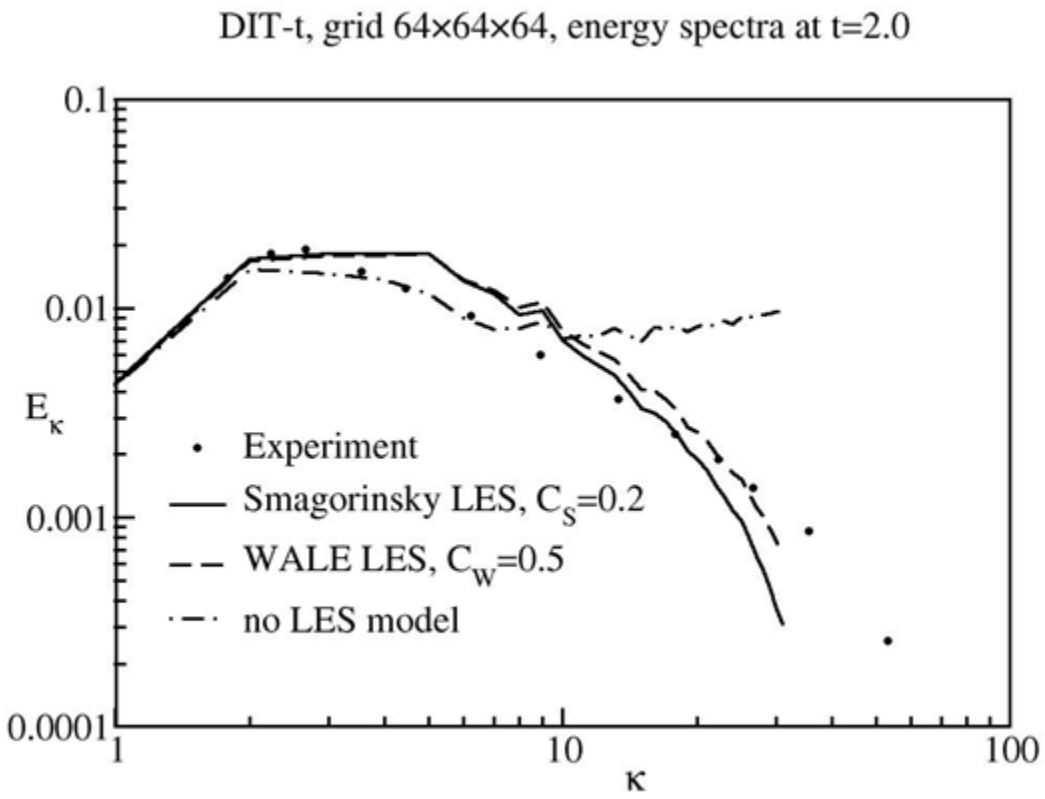
LES computations are usually performed on numerical grids that are too coarse to resolve the smallest scales. In the current example, the cut-off limit of LES (resolution limit) is at around $\kappa = 10$. The velocity gradients of the smallest resolved scales in LES are therefore much smaller than those at the DNS limit. The molecular viscosity is then not sufficient to provide the correct level of dissipation. In this case, the proper amount of dissipation can be achieved by increasing the viscosity, using an eddy-viscosity:

$$\varepsilon_{LES} = \nu_t \frac{\partial \bar{U}_i}{\partial x_j} \frac{\partial \bar{U}_i}{\partial x_j} \tag{12.20}$$

The eddy viscosity is calibrated to provide the correct amount of dissipation at the LES grid limit. The effect can be seen in Figure 12.6 (p. 194), where a LES of the DIT case is performed without a LES model and with different LES models. When the LES models are activated, the energy is dissipated and the models provide a sensible spectrum for all resolved scales. LES is not modeling the influence of unresolved small scale turbulence onto the larger, resolved scales, but the dissipation of turbulence into heat (the dissipated energy is typically very small relative to the thermal energy of the fluid and does not have to be accounted for, except for high Mach number flows).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.6: Turbulence spectrum for DIT test case after t=2. Comparison of results without Sub-Grid Scale model (no LES) with WALE and Smagorinsky LES model simulations



This discussion shows that LES is a fairly simple technology, which does not provide a reliable backbone for modeling. This is also true for more complex LES models like dynamic models. Dynamic eddy viscosity LES models (see Geurts, 2004 12 (p. 270)) are designed to estimate the required level of dissipation at the grid limit from flow conditions at larger scales (typically twice the filter width), thereby reducing the need for model calibration. Such models, however, also only provide a suitable eddy viscosity level for energy dissipation. As a result, within the LES framework, all features and effects of the flow that are of interest and relevance to engineers have to be resolved in space and time. This requirement makes LES a very CPU-expensive technology.

Even more demanding is the application of LES to wall-bounded flows, which is the typical situation in engineering flows. The turbulent length scale, L_t, of the large eddies can be expressed as:

$$L_t = \kappa y \tag{12.21}$$

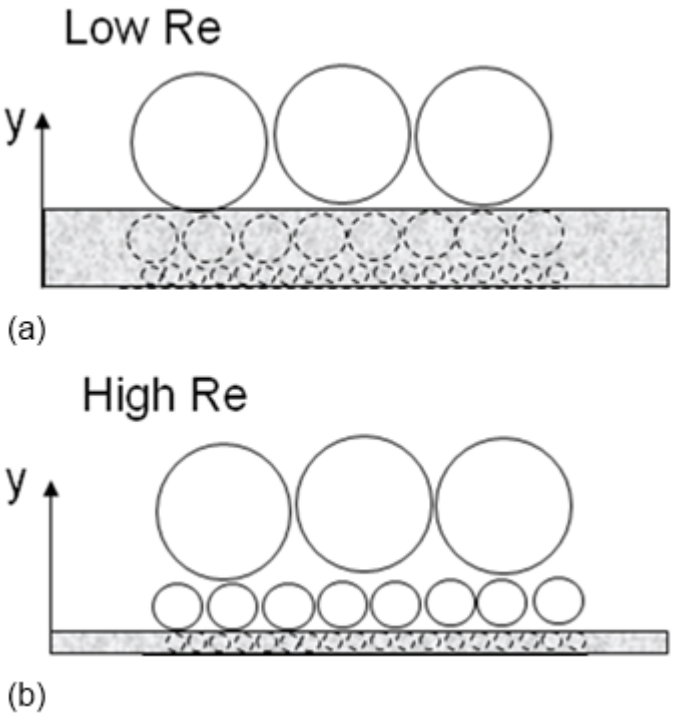
where y is the wall distance and κ a constant. Even the (locally) largest scales become very small near the wall and require a high resolution in all three space dimensions and in time.

The linear dependence of L_t on y indicates that the turbulence length scales approach zero near the wall, which would require an infinitely fine grid to resolve them. This is not the case in reality, as the molecular viscosity prevents scales smaller than the Kolmogorov limit. This is manifested by the viscous or laminar sublayer, a region very close to the wall, where turbulence is damped and does not need to be resolved. However, the viscous sublayer thickness is a function of the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Reynolds number, Re, of the flow. At higher Re numbers, the viscous sublayer becomes decreasingly thinner and thereby allows the survival of smaller and smaller eddies, which need to be resolved. This is depicted in Figure 12.7 (p. 195) showing a sketch of turbulence structures in the vicinity of the wall (for example, channel flow with flow direction normal to observer). The upper part of the picture represents a low Re number and the lower part a higher Re number situation. The gray box indicates the viscous sublayer for the two Re numbers. The structures inside the viscous sublayer (circles inside the gray box) are depicted but not present in reality due to viscous damping. Only the structures outside of the viscous sublayer (that is, above the gray box) exist and need to be resolved. Due to the reduced thickness of the viscous sublayer in the high Re case, substantially more resolution is required to resolve all active scales. Wall-resolved LES is therefore prohibitively expensive for moderate to high Reynolds numbers. This is the main reason why LES is not suitable for most engineering flows.

Figure 12.7: Sketch of turbulence structures for wall-bounded channel flow with viscous sublayer (a) Low Re number (b) High Re number (Grey area: viscous sublayer)

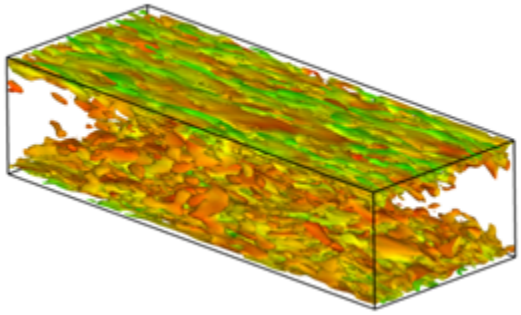


The Reynolds number dependence of wall-resolved LES can be estimated for a simple periodic channel flow as shown in Figure 12.8 (p. 196) (x-streamwise, y-wall-normal, z-spanwise, H is the channel height).

$$\begin{aligned} L_x &= 4H \\ L_y &= H = 2h \\ L_z &= 1.5H \end{aligned} \tag{12.22}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.8: Turbulence structures in a channel flow



The typical resolution requirements for LES are:

$$\begin{aligned} \Delta x^{+} &= 40 \\ \Delta z^{+} &= 20 \\ N_y &= 60-80 \end{aligned} \tag{12.23}$$

where Δx^{+} is the non-dimensional grid spacing in the streamwise direction, Δz^{+} in the spanwise and N_y the number of cells across half of the channel height. With the definitions:

$$\begin{aligned} \Delta x^{+} &= \frac{u_\tau \Delta x}{\nu} \\ \Delta z^{+} &= \frac{u_\tau \Delta z}{\nu} \end{aligned} \tag{12.24}$$

one can find the number, $N_t = N_x \times N_y \times N_z$ of cells required as a function of Re_τ, for resolving this limited domain of simple flow (see Table 12.1 (p. 196)).

$$\begin{aligned} N_x &= \frac{8h}{\Delta x} = \frac{8\mathrm{Re}_\tau}{\Delta x^{+}} \\ N_z &= \frac{3h}{\Delta z} = \frac{3\mathrm{Re}_\tau}{\Delta z^{+}} \\ &\text{with } \mathrm{Re}_\tau = \frac{u_\tau h}{\nu} \end{aligned} \tag{12.25}$$

Table 12.1: Number of cells, N_t, vs Reynolds number for channel flow

Re_τ	500	10^3	10^4	10^5
N_t	5×10^5	2×10^6	2×10^8	2×10^{10}

(For the practitioner: the Reynolds, Re, number based on the bulk velocity is around a factor of ten larger than the Reynolds number, Re_τ, based on friction velocity. Note that Re_τ is based on $h = H/2$). The number of cells increases strongly with Re number, demanding high computing resources even for very simple flows. The CPU power scales even less favorably, as the time step must also be reduced to maintain a constant CFL number ($CFL = (U\Delta t)/\Delta x$).

The Re number scaling for channel flows could be reduced by the application of wall functions with increasing y^{+} values for higher Re numbers. However, wall functions are a strong source of modeling uncertainty and can undermine the overall accuracy of simulations. Furthermore, the experience with RANS models shows that the generation of high quality wall-function grids for

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

complex geometries is a very challenging task. This task is even more challenging for LES applications, where you would have to control the resolution in all three space dimensions to conform to the LES requirements (for example, Δx^+ and Δz^+ then depend on Δy^+).

For external flows, there is an additional Re number effect resulting from the relative thickness of the boundary layer (for example, boundary layer thickness relative to chord length of an airfoil). At high Re numbers, the boundary layer becomes very thin relative to the body's dimensions. Assuming a constant resolution per boundary layer volume, Spalart et al., 1997 30 (p. 273), 2000 31 (p. 273) provided estimates of computing power requirements for high Reynolds number aerodynamic flows under the most favorable assumptions. Even then, the computing resources are excessive and will not be met even by optimistic estimates of computing power increases for several decades, except for simple flows.

While the computing requirements for high Re number flows are dominated by the relatively thin boundary layers, the situation for low Re number technical flows is often equally unfavorable, as effects such as laminar-turbulent transition dominate and need to be resolved. Based on reduced geometry simulations of turbomachinery blades (see Michelassi, 2003 20 (p. 271)), an estimate for a single turbine blade with end-walls is given in Table 12.2 (p. 197).

Table 12.2: Computing power estimate for a single turbomachinery blade with end-walls

Method	Cells	Time steps	Inner loops per time step	Ratio to RANS
RANS	~10^6	~10^2	1	1
LES	~10^8–10^9	~10^4–10^5	10	10^5–10^7

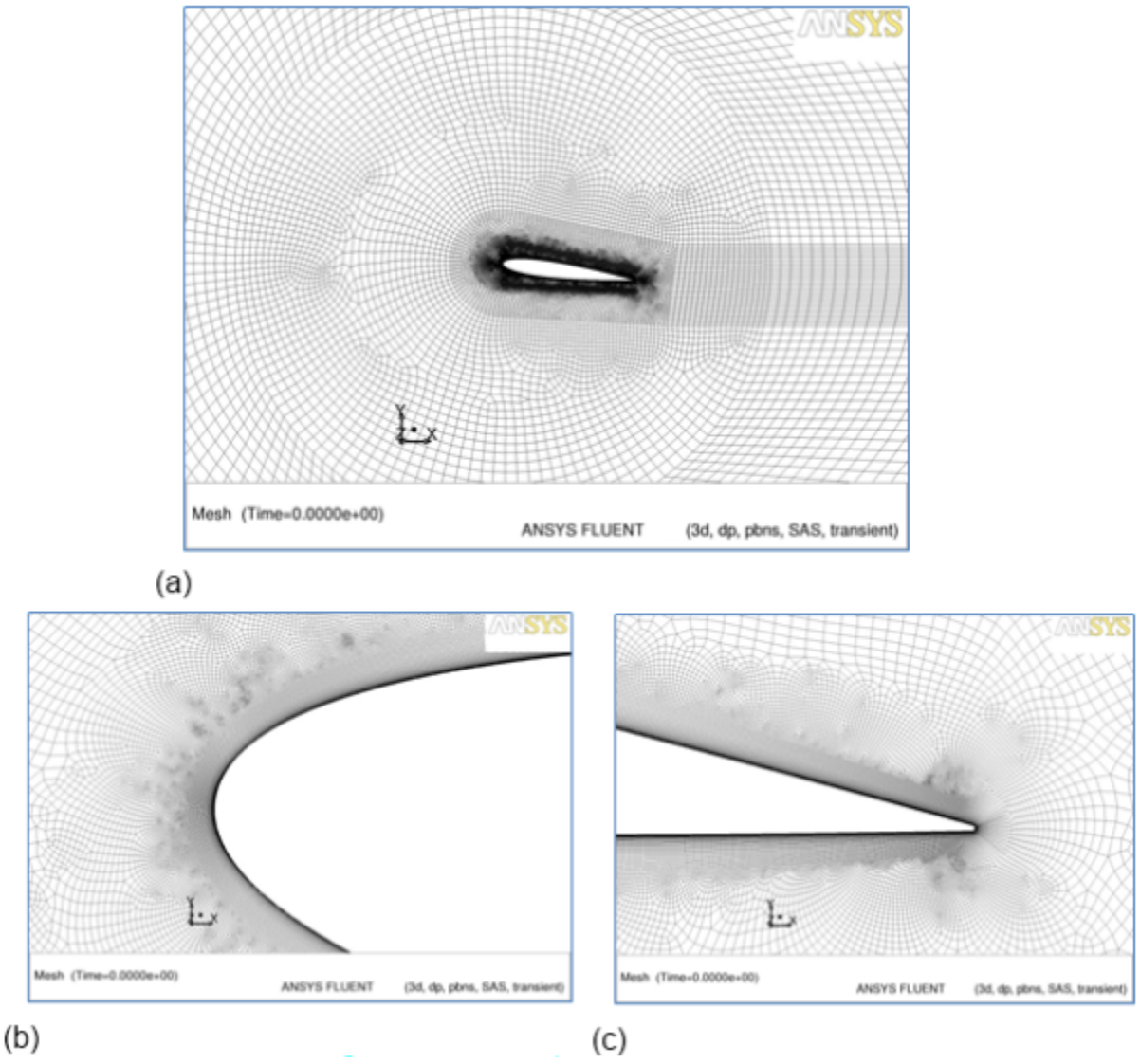
Considering that the goal of turbomachinery companies is the simulation of entire machines (or at least significant parts of them), it is unrealistic to assume that LES will become a major element of industrial CFD simulations even for such low Re number ($\mathrm{Re}\sim 10^5$) applications. However, LES can play a role in the detailed analysis of elements of such flows like cooling holes or active flow control.

All the above does not mean that LES of wall-bounded flows is not feasible at all, but just that the costs of such simulations are high. Figure 12.9 (p. 198) shows the grid used for a LES around a NACA 0012 airfoil using the WALE model. The computational domain is limited in the spanwise direction to 5% of the airfoil chord length using periodic boundary conditions in that direction. At a Reynolds number of $\mathrm{Re}=1.1\times 10^6$ a spanwise extent of 5% has been estimated as the minimum domain size that allows turbulence structures to develop without being synchronized across the span by the periodic boundary conditions. The estimate was based on the boundary layer thickness at the trailing edge as obtained from a precursor RANS computation. This boundary layer thickness is about 2% chord length. The grid had 80 cells in the spanwise direction and overall 11×10^6 cells. The simulation was carried out at an angle of attack of $\alpha=7.3^\circ$, using Ansys Fluent in incompressible mode. The chord length was set to $c=0.23\ [m]$, the freestream velocity, $U=71.3\ [m\cdot s^{-1}]$ and the fluid is air at standard conditions. The time step was set to $\Delta t=1.5\times 10^{-6}\,[s]$ giving a Courant number of $CFL\sim 0.8$ inside the boundary layer. Figure 12.10 (p. 199) shows turbulence structures near the leading edge (a) and the trailing edge (b). Near the leading edge, the laminar-turbulent transition can clearly be seen. The transition is triggered by a laminar separation bubble. Near the trailing edge, the turbulence structures are already relatively large, but still appear unsynchronized in the spanwise direction (no large scale

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

2D structures with axis orientation in the spanwise direction). The simulation was run for ~10^4 time steps before the averaging procedure was started. The time averaging was conducted for $\sim 1\times10^4$ time steps. Figure 12.11 (p. 199) shows a comparison of the wall pressure coefficient C_p and Figure 12.12 (p. 200) of the wall shear stress coefficient C_f on the suction side of the airfoil in comparison to a RANS computation using the SST model (Menter, 1994 16 (p. 271)). No detailed discussion of the simulation is intended here, but the comparison of the wall shear stress with the well-calibrated RANS model indicates that the resolution of the grid is still insufficient for capturing the near-wall details. For this reason, the wall shear stress is significantly underestimated by about 30% compared to the SST model in the leading edge area. As the trailing edge is approached, the comparison improves, mainly because the boundary layer thickness is increased whereas the wall shear stress is decreased, which produces a higher relative resolution in the LES. Based on this simulation, it is estimated that a refinement by a factor of 2, in both streamwise and spanwise directions, would be required in order to reproduce the correct wall shear stress. While such a resolution is not outside the realm of available computers, it is still far too high for day-to-day simulations of more complex configurations.

Figure 12.9: Details of grid around a NACA 0012 airfoil (a) Grid topology (b) Leading edge area (c) Trailing edge area



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.10: Turbulence structures of WALE LES computation around a NACA 0012 airfoil (a) Leading edge (b) Trailing edge (Q-criterion, color- spanwise velocity component)

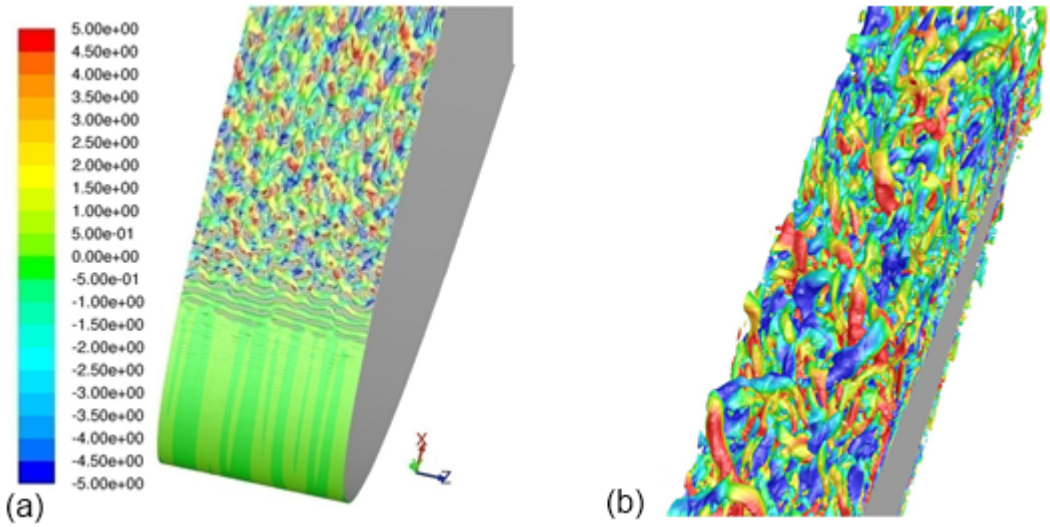
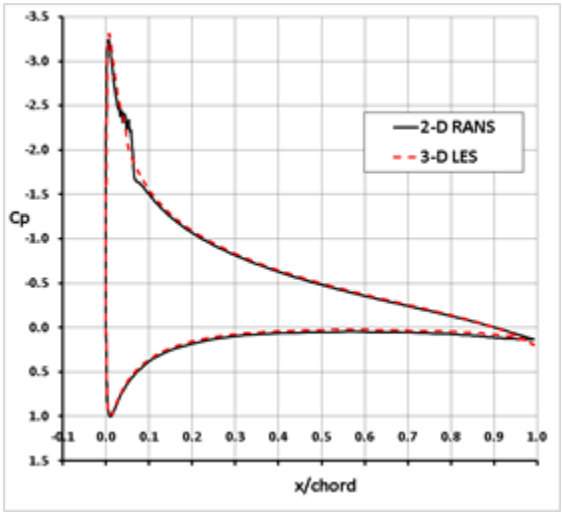
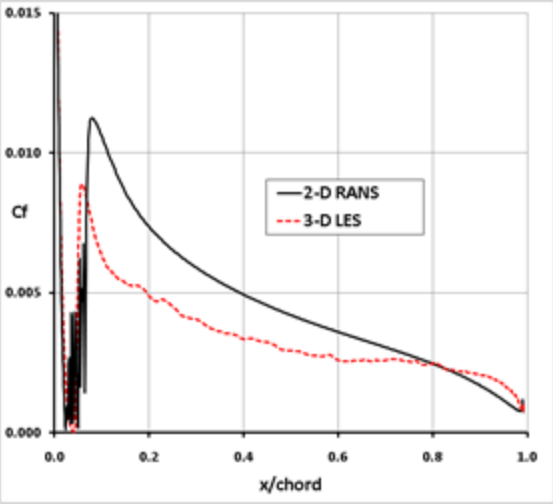


Figure 12.11: Wall pressure coefficient C_p on the suction side of a NACA 0012 airfoil: comparison of RANS-SST and LES-WALE results



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.12: Wall shear stress coefficient C_f on the suction side of a NACA 0012 airfoil: comparison of RANS-SST and LES-WALE results



Overall, LES for industrial flows will be restricted in the foreseeable future to flows not involving wall boundary layers, or wall-bounded flows in strongly reduced geometries, preferentially at low Re numbers.

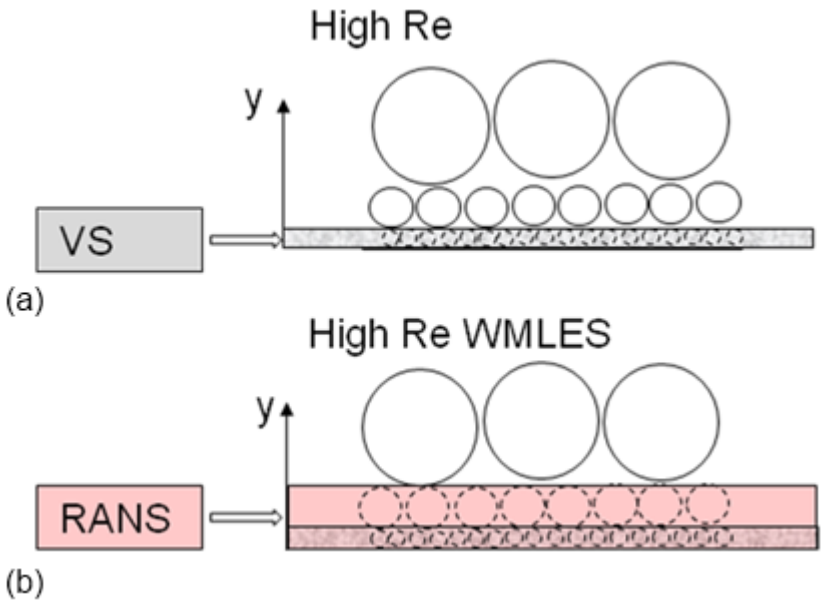
The limitations of the conventional LES approach are the driving force behind the development of hybrid RANS-LES models that are described in the later parts of this document.

12.1.1.6. Wall Modeled Large Eddy Simulation (WMLES)

Wall Modeled LES (WMLES) is an alternative to classical LES and reduces the stringent and Re number-dependent grid resolution requirements of classical wall-resolved LES (Section 12.1.1.5.1 (p. 191)) The principle idea is depicted in Figure 12.13 (p. 201). As described in Section 12.1.1.5.1 (p. 191), the near-wall turbulence length scales increase linearly with the wall distance, resulting in smaller and smaller eddies as the wall is approached. This effect is limited by molecular viscosity, which damps out eddies inside the viscous sublayer (VS). As the Re number increases, smaller and smaller eddies appear, since the viscous sublayer becomes thinner. In order to avoid the resolution of these small near-wall scales, RANS and LES models are combined such that the RANS model covers the very near-wall layer, and then switches over to the LES formulation once the grid spacing becomes sufficient to resolve the local scales. This is seen in Figure 12.13 (b) (p. 201), where the RANS layer extends outside of the VS, thus avoiding the need to resolve the inner second row of eddies depicted in the sketch.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.13: Concept of WMLES for high Re number flows (a) Wall-resolved LES. (b) WMLES



The WMLES formulation in Ansys CFD is based on the formulation of Shur et al., 2008 26 (p. 272):

$$\nu_t = f_D \min\left\{(\kappa y)^2, (C_{SMAG}\Delta)^2\right\} S \tag{12.26}$$

where y is the wall distance, κ is the von Karman constant, S is the strain rate and f_D is a near-wall damping function. This formulation was adapted to suit the needs of the Ansys general purpose CFD codes. Near the wall, the min-function selects the Prandtl mixing length model whereas away from the wall it switches over to the Smagorinsky model. Meshing requirements for the WMLES approach are given in Section 12.1.2.3.3 (p. 229).

For wall boundary layer flows, the resolution requirements of WMLES depend on the details of the model formulation. In Ansys Fluent and Ansys CFX they are (assuming for this estimate that x is the streamwise, y the wall normal and z the spanwise direction as shown in Figure 12.14 (p. 202)):

$$\begin{aligned} N_x &\approx \frac{\delta}{\Delta x} \approx 10 \\ N_y &\approx 30\text{–}40 \\ N_z &\approx \frac{\delta}{\Delta z} \approx 20 \end{aligned} \tag{12.27}$$

where N_x, N_y, and N_z are the numbers of cells per boundary layer thickness, δ, in the streamwise, wall normal, and spanwise directions respectively, (see Figure 12.14 (p. 202)). About 6000-8000 cells are needed to cover one boundary layer volume $\delta \times \delta \times \delta$. This is also the minimal resolution for classical LES models at low Reynolds numbers. Actually, for low Reynolds numbers, WMLES turns essentially into classical LES. The advantage of WMLES is that the resolution requirements relative to the boundary layer thickness remain independent of the Reynolds number.

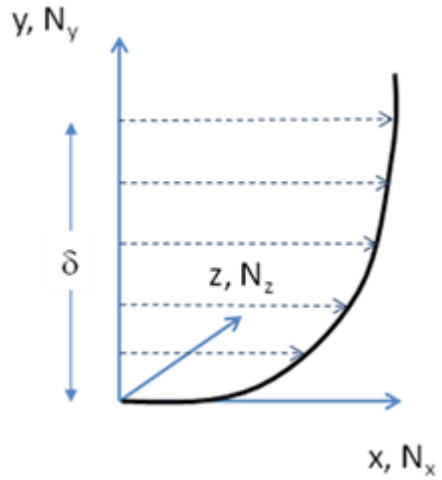
While WMLES is largely Reynolds number-independent for channel and pipe flows (where the boundary layer thickness must be replaced by half of the channel height) there remains a Reynolds number sensitivity for aerodynamic boundary layer flows, where, the ratio of the boundary layer thickness, δ, to a characteristic body dimension, L, is decreasing with increasing Reynolds number.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In aerodynamic boundary layer flows, there are more boundary layer volumes to consider at increased Reynolds numbers. It should also be noted that despite the large cost savings of WMLES compared to wall-resolved LES, the cost increase relative to RANS models is still substantial. Typical RANS computations feature only one cell per boundary layer thickness in streamwise and spanwise directions ($N_x \sim N_z \sim 1$). In addition, RANS steady-state simulations can be converged in the order of ~10^2–10^3 iterations, whereas unsteady simulations typically require ~10^4–10^5.

For wall-normal resolution in WMLES, it is recommended that you use grids with $\Delta y^+ = \sim 1$ at the wall. If this cannot be achieved, the WMLES model is formulated to tolerate coarser Δy^+ values (Δy^+-insensitive formulation) as well.

Figure 12.14: Sketch of boundary layer profile with thickness δ, x-streamwise direction, y-normal direction, and z-spanwise direction



For channel and pipe flows, the above resolution requirements for the boundary layer should be applied, only replacing the boundary layer thickness, δ, with half the channel height, or with the pipe radius in the grid estimation. This estimate would result in a minimum of ~120 cells in the circumferential direction (360°) for a fully developed pipe flow.

It should be noted that reductions in grid resolution similar to WMLES can be achieved with classical LES models when using LES wall functions. However, the generation of suitable grids for LES wall functions is very challenging as the grid spacing normal to the wall and the wall-parallel grid resolution requirements are coupled and strongly dependent on Re number (unlike RANS where only the wall-normal resolution must be considered).

In Ansys Fluent, the WMLES formulation can be selected as one of the LES options; in Ansys CFX it is always activated inside the LES zone of the Zonal Forced LES (ZFLES) method.

12.1.1.7. Embedded/Zonal LES (ELES, ZLES)

The idea behind ELES is to predefine different zones with different treatments of turbulence in the preprocessing stage. The domain is split into a RANS and a LES portion ahead of the simulation. Between the different regions, the turbulence model is switched from RANS to LES/WMLES. In order to maintain consistency, synthetic turbulence is generally introduced at RANS-LES interfaces. ELES is actually not a new model, but an infrastructure that combines existing elements of technology

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

in a zonal fashion. The recommendations for each zone are therefore the same as those applicable to the individual models.

In Ansys Fluent, an Embedded LES formulation is available (Cokljat et al., 2009 2 (p. 269)). It allows the combination of most RANS models with all non-dynamic LES models in the predefined RANS and LES regions respectively. The conversion from modeled turbulence to resolved turbulence is achieved at the RANS-LES interface using the Vortex Method (Mathey et al., 2003 15 (p. 271)).

In CFX, a similar functionality is achieved using a method called Zonal Forced LES (ZFLES) (Menter et al., 2009 17 (p. 271)). The simulation is based on a pre-selected RANS model. In a LES zone, specified via a CEL expression, forcing terms in the momentum and turbulence equations are activated. These terms push the RANS model into a WMLES formulation. In addition, synthetic turbulence is generated at the RANS-LES interface,

There is an additional option in Ansys Fluent that involves using a global turbulence model (SAS, DDES, SDES, SBES), and activates the generation of synthetic turbulence at a pre-defined interface. The code takes care of balancing the resolved and modeled turbulence through the interface. This option can be used to force global hybrid models into unsteadiness for cases where the natural flow instability is not sufficient. Unlike ELES, the same turbulence model is used upstream and downstream of the interface. In ELES, different models are used in different zones on opposite sides of the interface.

Such forcing can also be achieved in Ansys CFX by specifying a thin LES region and using the SAS or SBES model globally. The SBES model is most suitable for this scenario.

12.1.1.8. Unsteady Inlet/Interface Turbulence

Classical LES requires providing unsteady fluctuations at turbulent inlets/interfaces (RANS-LES interface) to the LES domain. This makes LES substantially more demanding than RANS, where profiles of the mean turbulence quantities (k and ε or k and ω) are typically specified. An example is a fully turbulent channel (pipe) flow. The flow enters the domain in a fully turbulent state at the inlet, so you are therefore required to provide suitable resolved turbulence at such an inlet location through unsteady inlet velocity profiles. The inlet profiles have to be composed in such a way that their time average corresponds to the correct mean flow inlet profiles, as well as to all relevant turbulence characteristics (turbulence time and length scales, turbulence stresses, and so on). For fully turbulent channel and pipe flows, this requirement can be circumvented by the application of periodic boundary conditions in the flow direction. The flow is thereby driven by a source term in the momentum equation acting in the streamwise direction. By that trick, the turbulence leaving the domain at the outlet enters the domain again at the inlet, thereby avoiding the explicit specification of unsteady turbulence profiles. This approach can be employed only for very simple configurations. It requires a sufficient length of the domain (at least ~8–10h, see Figure 12.8 (p. 196)) in the streamwise direction to enable the formation inside the domain of turbulence structures independent of the periodic boundaries.

In most practical cases, the geometry does not enable fully periodic simulations. It can however feature fully developed profiles at the inlet (again typically pipe/channel flows). In such cases, you can perform a periodic precursor simulation on a separate periodic domain and then insert the unsteady profiles obtained at any cross-section of that simulation to the inlet of the complex CFD domain. This approach requires either a direct coupling of two separate CFD simulations or the storage of a sufficient number of unsteady profiles from the periodic simulation to be read in by the full simulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In a real situation, however, the inlet profiles might not be fully developed and no simple method exists for producing consistent inlet turbulence. In such cases, synthetic turbulence can be generated, based on given inlet profiles from RANS. These are typically obtained from a precursor RANS computation of the domain upstream of the LES inlet.

There are several methods for generating synthetic turbulence. In Ansys Fluent, the most widely used method is the Vortex Method (VM) (see Mathey et al., 2003 15 (p. 271)), where a number of discrete vortices are generated at the inlet. Their distribution, strength, and size are modeled to provide the desirable characteristics of real turbulence. The input parameters to the VM are the two scales (k and ε or k and ω) from the upstream RANS computation. CFX uses the generation of synthetic turbulence through suitable harmonic functions as an alternative to the VM (see Menter et al., 2009 17 (p. 271)).

The characteristic of high-quality synthetic turbulence in wall-bounded flows is that it recovers the time-averaged turbulent stress tensor quickly downstream of the inlet. This can be checked by plotting sensitive quantities like the time-averaged wall shear stress or heat transfer coefficient and observing their variation downstream of the inlet. It is also advisable to investigate the turbulence structures visually by using, for example, an isosurface of the Q-criterion, $Q=1/2\left(\Omega^2-S^2\right)$ (where S is the Strain rate, and Ω is the vorticity rate). This can be done even after a few hundred time steps into the simulation.

Because synthetic turbulence never coincides in all aspects with true turbulence, you should avoid putting an inlet/interface at a location with strong non-equilibrium turbulence activity. In boundary layer flows, that means that the inlet or RANS-LES interface should be located several (at least ~3-5) boundary layer thicknesses upstream of any strong non-equilibrium zone (such as a separation). The boundary layers downstream of the inlet/interface need to be resolved with a sufficiently high spatial resolution (see Section 12.1.2.3.3 (p. 229)).

12.1.2. Generic Flow Types and Basic Model Selection

There is a wide range of complex industrial turbulent flows and there is no single SRS approach to cover all of them with high efficiency. The most difficult question as you use the software is therefore how to select the optimal model combination for a given simulation. For this task, it is useful to categorize flows into different types. Although such a categorization is not always easy and by no means scientifically exact (there are many flows that do not exactly fall into any one of the proposed categories or fall into more than one) it might still help in the selection of the most appropriate SRS modeling approach.

This section discusses:

12.1.2.1. Globally Unstable Flows

12.1.2.2. Locally Unstable Flows

12.1.2.3. Stable Flows and Wall Boundary Layers

12.1.2.1. Globally Unstable Flows

12.1.2.1.1. Flow Physics

The classical example of a globally unstable flow is a flow past a bluff body. Even when computed with a classical URANS model, the simulation will typically provide an unsteady output. Figure 12.17 (p. 209) shows the flow around a triangular cylinder in crossflow as computed with both

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the SAS-SST and the DES-SST model. It is important to emphasize that the flow is computed with steady-state boundary conditions (as would be employed for a RANS simulation). Still, the flow downstream of the obstacle turns quickly into unsteady (scale-resolving) mode, even though no unsteadiness is introduced by any boundary or interface condition.

From a physical standpoint, such flows are characterized by the formation of new turbulence downstream of the body. This turbulence is independent from, and effectively overrides, the turbulence coming from the thin, attached boundary layers around the body. The turbulence in the attached boundary layers has very little effect on the turbulence in the separated zone. The attached boundary layers can, however, define the separation point/line on a smoothly curved body and so affect the size of the downstream separation zone. This effect can be tackled by a suitable underlying RANS model.

Typical members of this family of flows are given in the list below. Such flows are very common in engineering applications and are also the type of flows where RANS models can exhibit a significant deterioration of their predictive accuracy.

Examples of globally unstable flows include:

- Flows past bluff bodies include:
  - Flow past buildings
  - Landing gears of airplanes
  - Baffles in mixers
  - Side mirrors of cars
  - Stalled wings/sails
  - Re-entry vehicles
  - Trains/trucks/cars in crossflow
  - Tip gap of turbomachinery blades
  - Flows past orifices, sharp nozzles
  - Cavities
  - Flows with large separation zones (relative to attached boundary layer thickness)
- Flows with strong swirl instabilities include:
  - Flow in combustion chambers of gas turbines
  - Flows past vortex generators
  - Some tip vortex flows in adverse pressure gradients
- Flows with strong flow interaction include:
  - Impinging/colliding jets

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

– Jets in crossflow

The color scheme of the preceding points above identifies flows that are clearly within the definition of globally unstable flows (black) and those where the type of the flow depends on details of its regime/geometry (gray). Such flows fall in-between globally and locally unstable flows (see Section 12.1.2.2 (p. 215)).

12.1.2.1.2. Modeling

Of all flows where SRS modeling is required, globally unstable flows are conceptually the easiest to handle. Typically they can be captured by a global RANS-LES model such as SAS, DDES, SDES, or SBES. Such models cover the attached and mildly separated boundary layers in RANS mode, thereby avoiding the high costs of resolving wall turbulence. Due to the strong flow instability past the separation line, there is no need for specifying unsteady inlet turbulence nor to define specific LES zones. Globally unstable flows are also the most beneficial for SRS, as experience shows that RANS models can fail with significant margins of error for such flows. A large number of industrial flows fall into this category.

The safest SRS model for such flows is the SAS approach. It offers the advantage that the RANS model is not affected by the grid spacing and thereby avoids the potential negative effects of (D)DES-type models (gray zones or grid induced separation). The SAS concept reverts back to (U)RANS in case the mesh/time step is not sufficient for LES and thereby preserves a backbone of modeling that is independent of space and time resolution, though at the increased cost that is associated with any transient SRS calculation. SAS also avoids the need for shielding, which for internal flows with multiple walls can suppress turbulence formation in DDES models.

The alternatives to SAS are DDES, SDES, and SBES. If proper care is taken to ensure LES mesh quality in the detached flow regions, these models will be operating in the environment for which they were designed, typically providing high-quality solutions. DDES has shown advantages for flows at the limit of globally unstable flows (see Figure 12.52 (p. 246)) where the SAS model can produce URANS-like solutions. In cases like these, DDES still provides SRS in the separated regions. As noted, the DDES has been superseded by the SDES and SBES model family.

For globally unstable flows, the behavior of all global hybrid models is often very similar.

12.1.2.1.3. Meshing Requirements

The part of the domain where the turbulence model acts in RANS mode has to be covered by a suitable RANS grid. It is especially important that all relevant boundary layers are covered with sufficient resolution (typically a minimum of 10–15 structured cells across the boundary layer). It is assumed that you are familiar with grid requirements for RANS simulations.

The estimate for the lowest possible mesh resolution in the detached SRS region is based on the assumption that the largest relevant scales are similar in size to the width of the instability zone. For a bluff body, the width is the diameter D of the body; for a combustor, the width is the diameter of the core vortex; for a jet in crossflow, the width is the diameter of the jet; and so on. Experience shows that the minimum resolution for such flows is of the order:

$$\Delta_{\max} \leq 0.05D \tag{12.28}$$

This resolution requires more than 20 cells per characteristic diameter, D (in some applications with very strong instabilities, even 10 cells across the layer may be sufficient). As is generally the case for SRS, it is best to provide isotropic (cubic) cells, or at least to avoid large aspect ratios

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

(aspect ratios smaller than 5 would be optimal, but cannot always be achieved in complex geometries).

With the above estimate for $\Delta_{\max}$, there is a good chance of resolving the main flow instability and the resulting strong turbulent mixing processes associated with the global flow instability (an effect often missed by RANS models). For acoustics simulations, it might also be important to resolve the turbulence generated in the (often thin) shear layer that is separating from the body. Resolving this turbulence poses a much more stringent demand on grid resolution on the simulation, as the shear layer scales with the boundary layer thickness at separation and so can be much smaller than the body dimension. This situation is covered in Section 12.1.2.2 (p. 215).

12.1.2.1.4. Numerical Settings

The general numerical settings are described in Section 12.1.3 (p. 251). Globally unstable flows are relatively forgiving with respect to numerics, at least as far as the mean flow characteristics are concerned. The recommended choice for the advection terms is the Bounded Central Difference (BCD) scheme, especially for complex geometries and flows. For such flows, the classical Central Difference (CD) scheme can be unstable or produce unphysical wiggles in the solution (see Figure 12.60 (p. 252)). The BCD scheme is slightly more dissipative, but is substantially more robust and is therefore frequently the optimal choice. If a visual inspection of the flow (see Section 12.1.5.1 (p. 256)) shows that turbulence structures are not produced in agreement with the expectations for the flow, you can switch to CD. If this switch is made, it is advisable to closely monitor the solution (visually and numerically through residuals) to ensure that wiggles are not dominating the simulation. With SAS the "Least Square Cell Based" or the "Node-Based Green Gauss" gradient method should be used in Ansys Fluent. The latter allows a slightly better representation of the second derivative of the velocity field that is required for the model formulation (von Karman length scale).

In Ansys CFX, the default hybrid numerical option switches explicitly between the High Resolution Scheme (in the RANS region) and the CD scheme (in the LES region). For most applications, it appears that the use of the BCD scheme should also be favored in Ansys CFX (see also Section 12.1.3.1.1 (p. 251)).

12.1.2.1.5. Examples

The following examples have been computed before the availability of the SBES model family. They are therefore based on the SAS/DDES model formulations. Due to the strong flow instability in these flows, the choice of model formulation is marginal and all hybrid models provide fairly similar solutions.

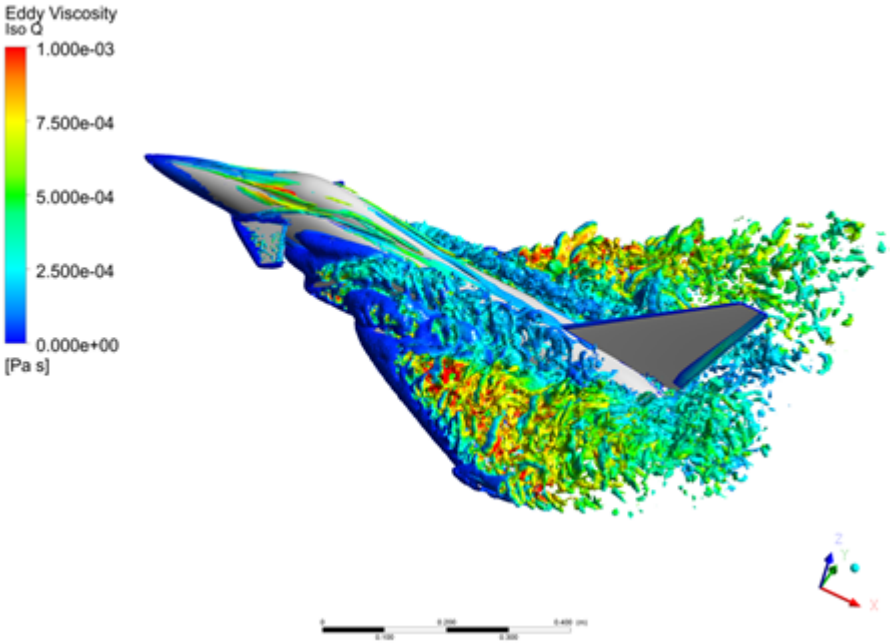
12.1.2.1.5.1. Flow around a Fighter Aircraft

Figure 12.15 (p. 208) shows a highly complex, globally unstable flow field, around a generic fighter aircraft geometry at high angle of attack as computed with the SAS-SST model. Figure 12.15 (p. 208) demonstrates the complex regional appearance of resolved turbulence around the aircraft. It is obvious that the application of global models like SAS or DDES greatly simplifies the set up for such flows compared to using ELES/ZLES, where you would have to define the LES regions and suitable interfaces between the RANS and LES regions in a preprocessing step. In contracts, when using global models, the simulation is first carried out in standard RANS mode. Starting from that RANS solution, the model is then simply switched to the SAS or DDES variant of the RANS model, the solver is set to unsteady mode, and the numeric is adjusted

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

according to Section 12.1.2.1.4 (p. 207). No further adjustment is required in order to produce the solutions shown in Figure 12.15 (p. 208).

Figure 12.15: Turbulence structures for flow around a generic fighter aircraft (Q-criterion) as computed by SAS-SST model. Courtesy - EADS Germany GmbH - Military Air Systems DESIDER Project



12.1.2.1.5.2. Flow around a Triangular Cylinder

Figure 12.16 (p. 209) shows the grid around a triangular cylinder in crossflow. The Reynolds number based on the freestream velocity (17.3 [m/s]) and the edge length is 45,500. Periodic boundary conditions have been applied in the spanwise direction. The simulations have been run with Ansys Fluent using the BCD (Bounded Central Difference) and CD (Central Difference) advection schemes and a time step of $\Delta t = 10^{-5} s$ (CFL~1 behind cylinder). The grid features 26 cells across its base. It is extended in the spanwise direction to cover 6 times the edge length of the triangle with 81 cells in that direction. Due to the strong global instability of this flow, such resolution was sufficient and has produced highly accurate solutions for mean flow and turbulence quantities (Figure 12.17 (p. 209)).

It should be noted that not all flows produce such strong instability as the triangular cylinder, and a higher grid resolution might be required for flows with less instability. Figure 12.17 (p. 209) shows that the grid does not provide resolution of the boundary layer on the walls of the triangular body. This is not a problem in the current case because the wall boundary layer has no influence on the global flow, as it separates at the corners of the triangle. In real flows, this might not always be the case and the boundary layer should be resolved with a RANS-type mesh (that is, a finer mesh in the near-wall region with higher aspect ratios being acceptable).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.16: Grid around cylinder in crossflow

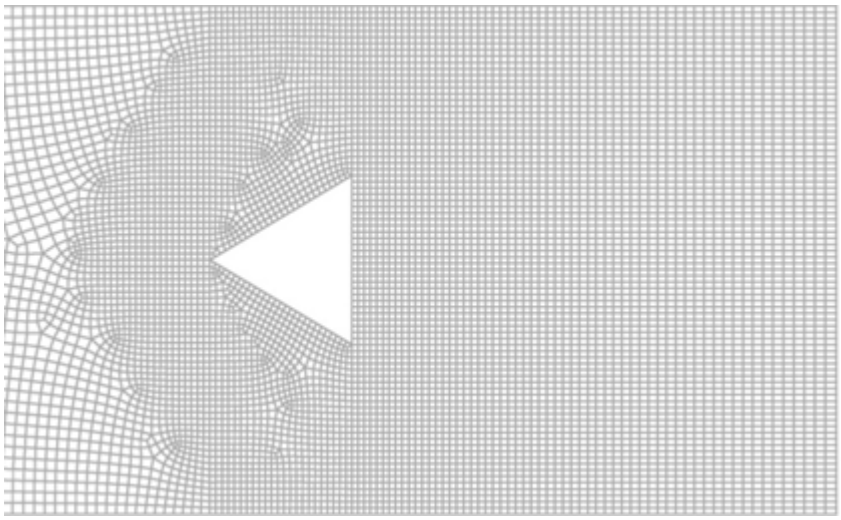
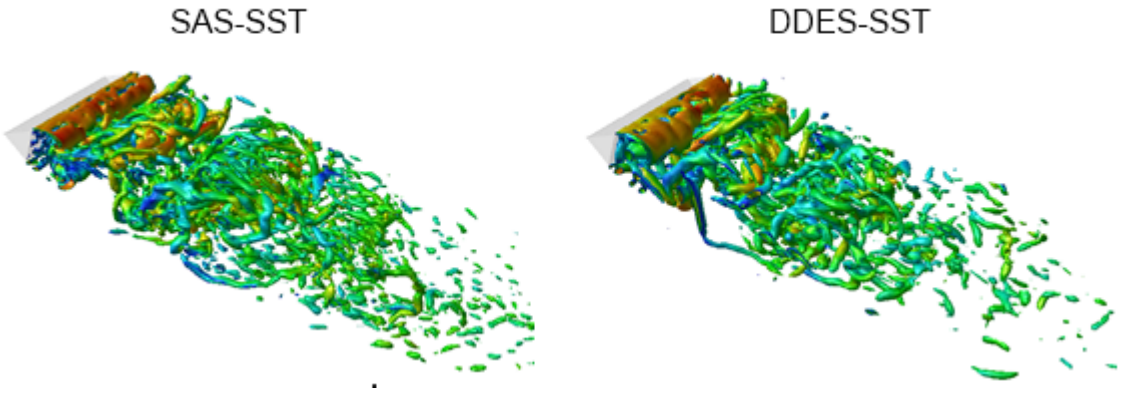


Figure 12.17 (p. 209) shows a visual representation of the flow using the DDES-SST and the SAS-SST models with the Q-criterion (see Section 12.1.5.1 (p. 256)). Both simulations have been carried out using the BCD scheme. Both models generate resolved turbulence structures in agreement with the expectation for the grid provided. Figure 12.18 (p. 210) shows a comparison with the experimental data (Sjunnesson et al., 1992 28 (p. 273)) for the wake velocity profiles as well as for turbulence characteristics.

Figure 12.17: Turbulence structures for flow around a cylinder in crossflow



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.18: Velocity profiles and turbulence RMS profiles for three different stations downstream of the triangular cylinder (x/a=0.375, x/a=1.53, x/a=3.75). Comparison of SAS-SST, DES-SST models, and experiment. (a) U-velocity, (b) u_{rms}, (c) v_{rms}, (d) u'v'

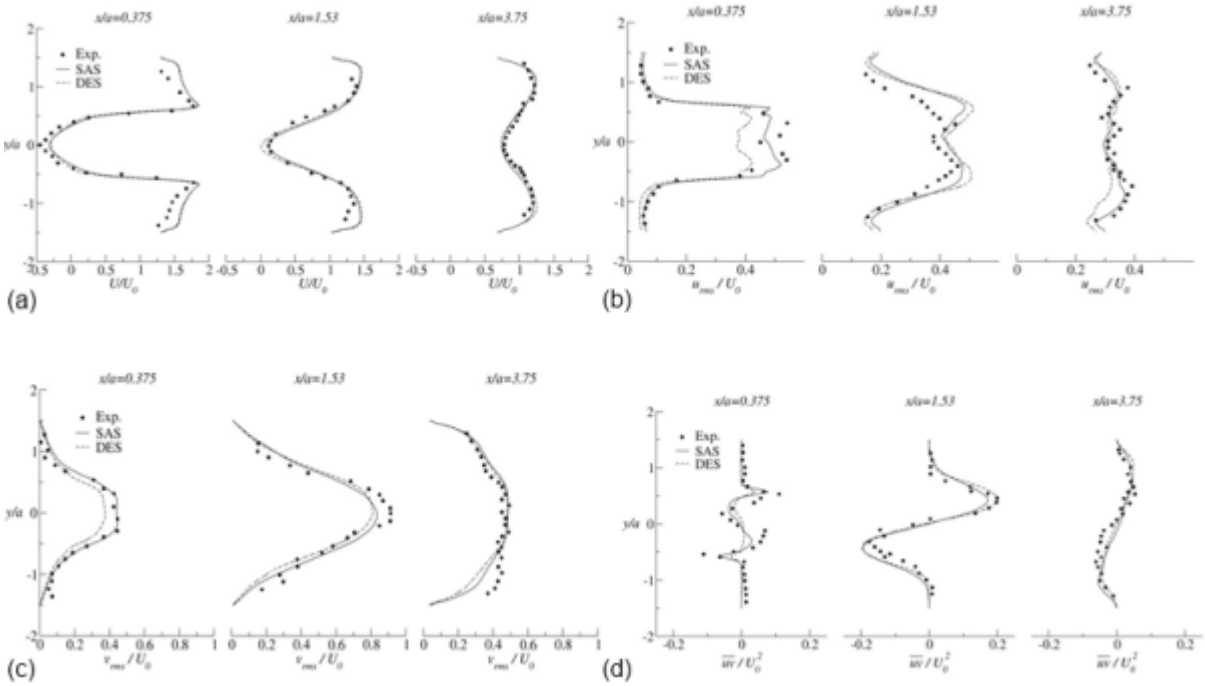
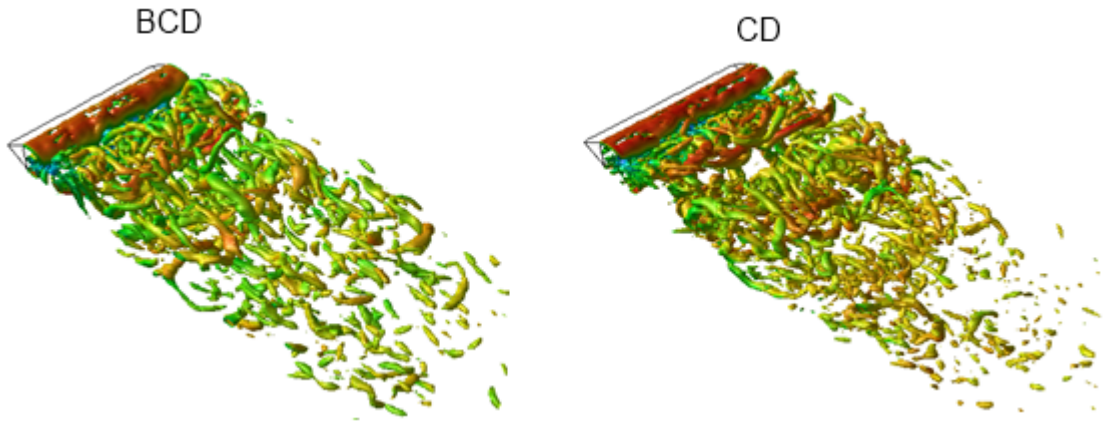


Figure 12.19 (p. 210) shows a comparison of the CD and the BCD scheme for the triangular cylinder using the SAS-SST model. The turbulence content is almost identical, except that some smaller scales are present in the CD simulation downstream of the body. A comparison with experimental data showed results that are almost identical to the ones shown in Figure 12.18 (p. 210) and independent of whether the CD or the BCD scheme was used.

Figure 12.19: SAS-SST simulation for flow around a triangular cylinder using the BCD and the CD scheme for the convective fluxes



12.1.2.1.5.3. ITS Combustion Chamber

The SAS-SST model is applied to the flow in a single swirl burner investigated experimentally by Schildmacher et al., 2000 26 (p. 272) at ITS (Institut für Thermische Strömungsmaschinen) of the University of Karlsruhe. The ITS burner is a simplified industrial gas turbine combustor. It

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

concentrates on the swirl flow in the combustion region. Similar to the triangular cylinder test case, the wall boundary layers are not important, which means that this test case is also accessible to pure LES simulations. However, in many industrial combustion chambers wall boundary layers and auxiliary pipe flows have to be considered, which makes them unsuitable for pure LES.

There are two co-axial inlet streams and both are swirling in the same direction. The swirl is generated by means of the two circumferential arrays of blades, which are not included in the current computational domain. The axisymmetric velocity profiles with the circumferential component corresponding to the given swirl number are used as the inlet boundary conditions. The swirl gives the flow a strong global instability, which can be captured well by global SRS models.

Figure 12.20 (p. 211) shows the geometry. The grid shown in Figure 12.21 (p. 212) consists of 3.6×10^{6} tetrahedral elements. As stated, the wall boundary layers are not important and are therefore not resolved on this tetrahedral mesh. The simulation was run with Ansys CFX, which internally converts the grid to a polyhedral grid with 6×10^{5} control volumes around the grid points for the node-based solver. This means that the polyhedral grid cells are larger than the visual impression from Figure 12.21 (p. 212) with ~20–30 cells covering the relevant length scale L, shown in Figure 12.21 (p. 212). The grid does not feature any near-wall boundary layer resolution. It is recommended that you provide such a boundary layer grid for industrial flows (typically more than 10 structured cells across the boundary layer), as in some geometries the separation characteristics near the burner entrance can depend on such details. The convection scheme selected was the default hybrid scheme; however, BCD should also work well.

Figure 12.20: Computational domain for the ITS swirl burner

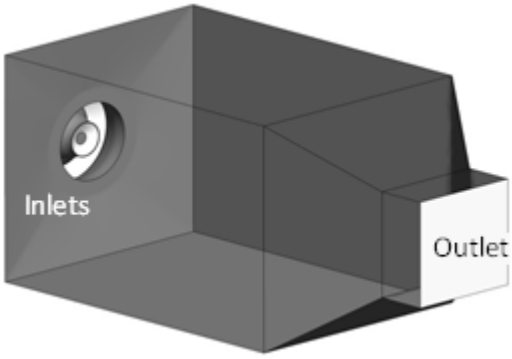
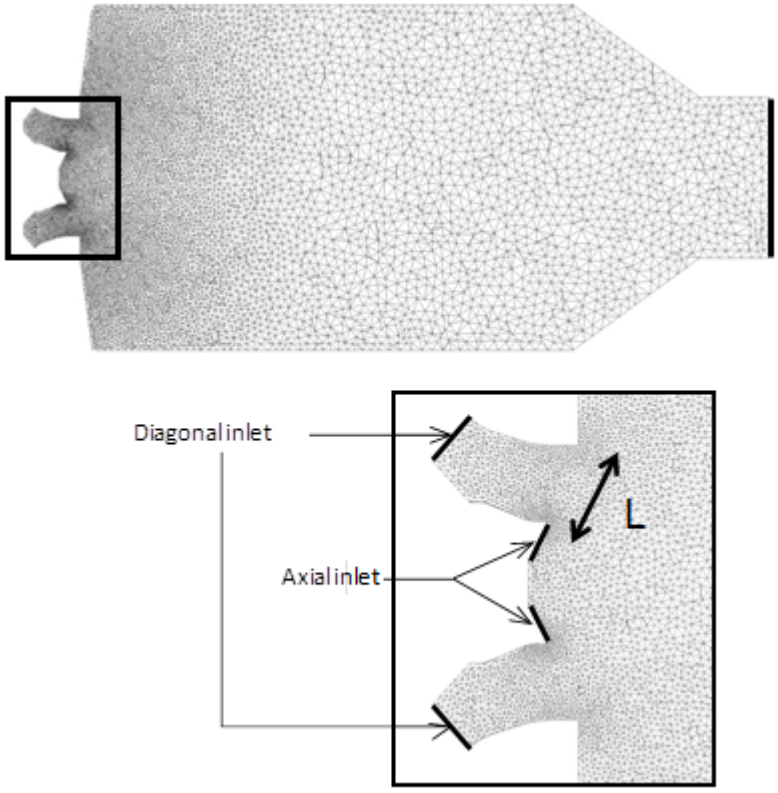


Figure 12.21: Unstructured grid on the symmetry plane and boundary locations for the ITS swirl burner and relevant length scale, L

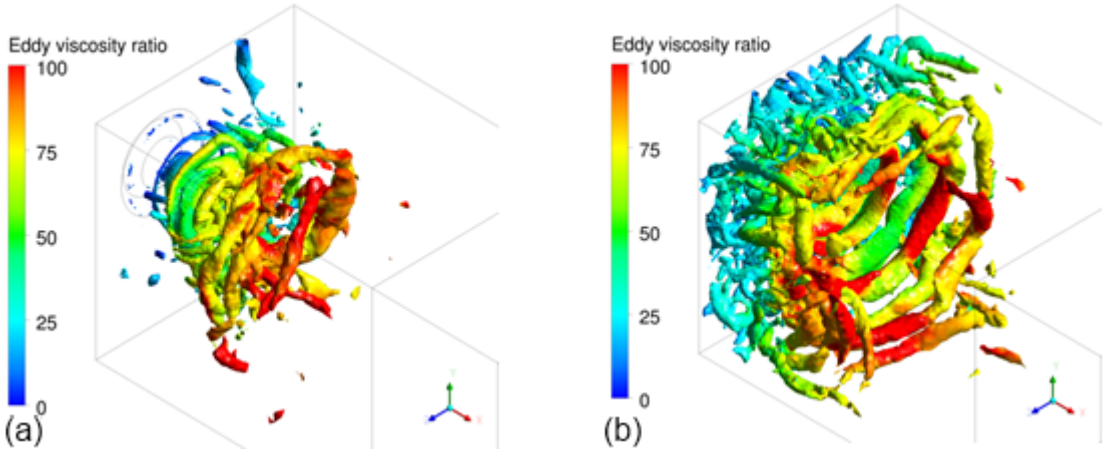


The flow structures from the SAS-SST computations of the non-reacting and the reacting flow at a given instance in time are shown in Figure 12.22 (p. 213) using the Q-criterion ($Q=2\times10^{7}\left[s^{-2}\right]$, see Section 12.1.5.1 (p. 256)). The isosurface in Figure 12.22 (p. 213) is given by $Q=1/2\left(S^{2}-\Omega^{2}\right)=2\times10^{7}\left[s^{-2}\right]$. The main turbulence structures seem to be captured well in the simulations. Clearly, small-scale turbulence cannot be resolved on such a grid. The grid resolution used here should not be considered as a recommendation for combustion chambers, but as the lowest limit for which such SRS models can be applied.

Figure 12.23 (p. 214) shows a comparison of the standard $k-\varepsilon$ RANS and SAS results at a given distance from the burner entrance. It shows the level of improvement that results from the application of SRS methods. Many more details of this simulation can be found in Egorov et al., 2010 6 (p. 269) or in a more detailed analysis of a more complex combustion chamber in Widenhorn et al., 2009 38 (p. 274).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.22: SAS solution for ITS combustion chamber (a) Non-reacting, (b) Reacting flow



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.23: Reacting flow velocity profiles at the axial distance from the inlet x=103 mm (a) Axial velocity, (b) Tangential velocity

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

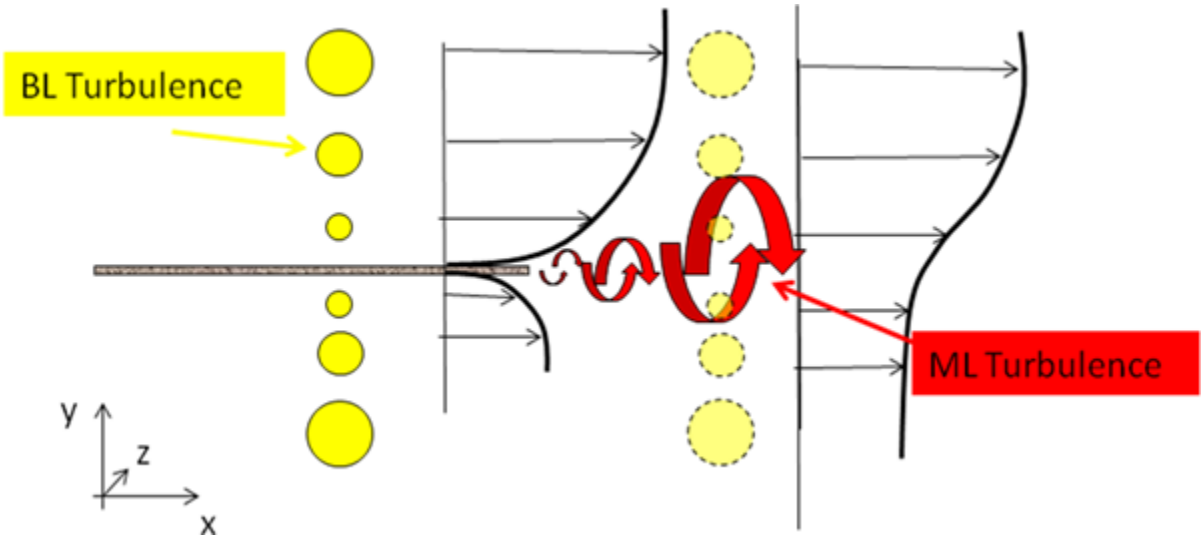
12.1.2.2. Locally Unstable Flows

12.1.2.2.1. Flow Physics

The expression "locally unstable flows" is not easily defined, because every turbulent flow is unstable by nature. It is meant to characterize flows that also produce new turbulence, typically downstream of a geometry change, but where the flow instability producing this turbulence is significantly weaker than for globally unstable flows.

Consider the computation of a mixing layer starting from two wall boundary layers with different freestream velocities in RANS mode (see Figure 12.24 (p. 215)). As the flat plate ends, the two boundary layers form a turbulent mixing layer, which quickly becomes independent of the turbulence of the two boundary layers on the flat plate (yellow circles). The mixing layer instability (red) provides for a de-coupling of the boundary layer and the mixing layer turbulence. For this reason, you can neglect the boundary layer turbulence downstream of the trailing edge (the dashed yellow boundary layer turbulence sketched in Figure 12.24 (p. 215)) and concentrate on using SRS mode to resolve the mixing layer turbulence, which will quickly dominate the flow.

Figure 12.24: Schematic of locally unstable flow: Mixing layer originating from a flat plate with two boundary layers of different freestream velocity. Full yellow circles are boundary layer turbulence. Dashed yellow circles are remains of the boundary layer turbulence. Red arrows are new mixing layer turbulence



Examples of locally unstable flows:

- All equilibrium free-shear flows emanating from walls (jets, wakes, mixing layers)
- Backward-facing step flow
- Weakly interacting equilibrium flows
- Flows with weak swirl

12.1.2.2.2. Modeling

The goal in SRS is to cover the boundary layer turbulence (solid yellow circles in Figure 12.24 (p. 215)) in RANS and the mixing layer turbulence (red in Figure 12.24 (p. 215)) in resolved

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

mode. This can only be achieved if the impact of the RANS turbulence model is significantly reduced downstream of the trailing edge; otherwise the formation of unsteady structures would be suppressed.

The SAS model will typically not switch to SRS mode in such situations, independent of the mesh provided, as the eddy-viscosity produced in the mixing layer will be too large for the flow instability at hand. From a pure turbulence modeling standpoint, this is often acceptable, as such flows are typically covered with reasonable accuracy by using RANS models (mixing layers, wakes, back step, and so on). However, in cases where unsteady information is required for other reasons (for example, acoustics), the SAS model will likely not be suitable, unless an interface is used that converts modeled turbulence energy into resolved energy (see Section 12.1.1.7 (p. 202)).

The DDES, SDES and SBES models allow SRS behavior because the shielding function is turned off past the trailing edge of the plate and, assuming a fine (LES) grid is provided downstream of the plate, the eddy-viscosity is reduced. These models then switch to LES mode in the wake, and the mixing-layer instability is strong enough to generate resolved turbulence relative quickly (within a few boundary layer thicknesses). It is important to point out that the ability of the DDES/SDES/SBES models to generate unsteady structures in the mixing layer depends on the grid provided in that area. Assuming an overly coarse grid (for example, in the spanwise direction), the grid limiter would not engage and the model would stay in RANS mode, which will not enable the formation of resolved structures. Remember that, for the DES model, the length scale is defined as:

$$L_{DES}=\min\left(L_t, C_{DES}\Delta_{\max}\right) \tag{12.29}$$

where $\Delta_{\max}$ is the largest edge length for each cell. For this case, assume that the grid in the $x-y$ plane shown in Figure 12.24 (p. 215) is very fine (of LES quality), and $\Delta_{\max}=\Delta z$ is the grid resolution in the spanwise (z) direction. Conversely, if Δz is very coarse, the DES limiter would always select the RANS length scale L_t and the model would remain in RANS mode in the wake region. No unsteady structures would develop as the RANS model will damp them out. As the grid in the z-direction is refined, the DES limiter will be activated at some location downstream of the trailing edge where $\Delta z=L_{t-\max}$ (note that L_t grows as the mixing layer becomes thicker). With further grid refinement, the location of the implicit RANS-LES interface would move closer to the trailing edge. Eventually, the entire mixing layer would be covered by LES. This behavior of (D)DES is both a disadvantage and an advantage. The disadvantage and the danger lie in the strong grid sensitivity introduced explicitly into the turbulence model. As a result, when using (D)DES (as well as SDES/SBES), you must be very careful to provide a suitable grid for a given application when using DDES. The advantage is that the model can be applied to locally unstable flows without the definition of an explicit RANS-LES interface. However, the grid sensitivity can be reduced by employing an interface which converts modeled turbulence to resolved turbulence using the DDES/SDES/SBES model upstream and downstream of the interface (see Section 12.1.1.7 (p. 202)).

The most general approach to the flows discussed here is the use of the embedded or zonal RANS-LES methods, where the boundary layers are covered by a RANS model and the mixing layer by a LES model. The models are explicitly switched from RANS to LES at a pre-defined interface upstream or at the trailing edge. In order to obtain a proper LES solution, a grid with LES resolution is required in the mixing layer. Frequently a non-conformal interface between the RANS and the LES part is used to reduce the grid resolution in the upstream RANS region. For a fully consistent simulation, one must introduce synthetic turbulence at the RANS-LES interface. By such injection of synthetic turbulence, the balance between RANS and LES turbulence across

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the interface is preserved (that is, the yellow dashed circles in Figure 12.24 (p. 215) are accounted for).

The recommendation for flows with local instabilities is to use ELES/ZLES models if the geometry and the application enable the definition of well-defined interfaces (for example, internal flows, like pipe flows). Synthetic turbulence should be introduced at these interfaces in order to preserve the balance between the RANS and LES turbulence content. Should the geometry/application be complex such that the definition of explicit RANS and LES zones is not easily possible (for example, turbomachinery flows, external flows), apply the DDES/SDES/SBES model. However, ensure careful tailoring of the grid with sufficient resolution on the LES region to avoid undefined model behavior somewhere between RANS and LES mode. It is advisable to refrain from using conventional DES in flows with extensive boundary layers, because the danger of affecting the boundary layers is too high. For such flows, the newest members of the family, namely the SBES models, are recommended.

It is very important to understand that for locally unstable flows, failure to capture the instability of the Separating Shear Layer (SSL) can have a pronounced effect on the solution downstream. The turbulence field is a result of this initial instability and missing it can severely limit the resolved content of the simulation and contaminate an expensive SRS solution. This danger is much reduced with ELES/ZLES models, (relative to DDES) because the flow enters the SSL with a prescribed synthetic turbulent content from the RANS-LES interface.

12.1.2.2.3. Meshing Requirements

In order to generalize the concepts discussed for the mixing layer example (Figure 12.24 (p. 215)), we introduce the terminology of a Separating Shear Layer (SSL). It refers to the shear layer that starts at the point of separation from the body and moves into a free shear flow (we are not considering small separation bubbles embedded within the boundary layer). In Figure 12.24 (p. 215) the SSC would be the mixing layer forming downstream of the plate. In other flows it can be a separating boundary layer from a corner. In the case of locally unstable flows, the $\Delta_{\max}$ spacing should be sufficiently small to enable resolution of the initial flow instability of the SSL. The main quantity of relevance is the ratio of RANS to grid length scale:

$$R_L = \frac{\Delta_{\max}}{L_t^{RANS}}$$
$$L_t^{RANS} = \left(\frac{k^{3/2}}{\varepsilon}\right)^{RANS} = \left(\frac{k^{1/2}}{C_\mu \omega}\right)^{RANS} \tag{12.30}$$

It is important to emphasize that this quantity should be evaluated based on a precursor RANS solution. This implies that such a solution exists and is meaningful. If the precursor solution is not available, then you can estimate the ratio based on the thickness of SSL. For equilibrium mixing layers, the following ratio is approximately correct:

$$L_t^{RANS} = 0.7 \cdot \delta^{mixing} \tag{12.31}$$

where δ^{mixing} is the thickness of the mixing layer. The value of R_L should be:

$$R_L \leq 0.2 - 0.1 \tag{12.32}$$

where 0.2 should be considered an extreme lower limit of resolution and 0.1 the desirable lower limit. Again, higher grid resolution should be used if computing power permits. The value of $R_L = 0.1$ corresponds to a resolution of ~ 15 cells across the mixing layer. This amount of cells is not a very fine grid resolution, but equal resolution should ideally be provided in all three space

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

dimensions. In addition, the SSL can be thin relative to the body dimensions, resulting in very high computational costs. The initial SSL instability is akin to a Helmholtz instability and is initially two-dimensional. A doubled grid spacing in the spanwise direction is therefore acceptable.

It is not always possible to achieve such resolution directly from the onset of the separating shear layer, especially if this layer is very thin relative to the body dimensions. This inability is not necessarily a problem as, typically, the thickness of the SSL increases strongly downstream of the separation point/line. Therefore R_L is decreasing relatively quickly and reaches sufficiently low values to provide the required resolution. It is important to note that for cases where the small scales play a significant role, such as in acoustics simulations, the delay of the initial instability can result in a loss of spectral information at high wave numbers (small scales). It is advisable to visually inspect the displayed results for the presence of the unsteady turbulent structures at the intended locations.

Of special concern are geometries with high aspect ratios, meaning a large domain size in the direction perpendicular to the SSL (long cylinders in crossflow, stalled wings of high aspect ratios, and so on). In such situations, it is not always possible to sufficiently resolve the third direction. It might then be necessary to solve only a portion of the real flow domain in SRS mode, either by using suitable boundary conditions (for example, periodicity in the spanwise direction), or by restricting the SRS to a limited portion of the domain.

12.1.2.2.4. Numerical Settings

The general numerical settings described in Section 12.1.3 (p. 251) should be applied. In addition, locally unstable flows can be very sensitive with respect to numerics. For the application of the DDES/SDES/SBES models, the recommended choice for the advection terms is the Bounded Central Difference (BCD) scheme in the entire domain.

The influence of the pressure scheme is typically small, so all schemes can be used. The PRESTO scheme may delay the initial formation of resolved structures in locally unstable flows. The PRESTO scheme should therefore only be applied if additional stabilization is required, for example, because of strong body forces. Note that the above discussion applies to 'locally unstable flows'. For flows with stronger instabilities, the choice of numerical method is not as influential, and any pressure scheme can be used.

Experience suggests that the BCD scheme is also the most suitable choice when using ELES/ZLES methods. In some applications with high demands on accuracy and where a high quality isotropic mesh can be provided in the LES region, the application of the CD scheme in the LES zone might be advantageous.

12.1.2.2.5. Examples

12.1.2.2.5.1. Mixing Layer

The single-stream mixing layer flow is experimentally investigated in Morris and Foss, 2003 21 (p. 272). The experimental section consists of a flat plate which suddenly terminates at a corner step (Figure 12.25: The domain and grid for the single stream mixing layer flow (p. 219)), which induces the formation of the mixing layer from the separation point. The experiment is carried out at a Reynolds number of Re_θ=4650 based on the momentum thickness θ_0 of the incoming boundary layer and on the free-stream velocity U_0.

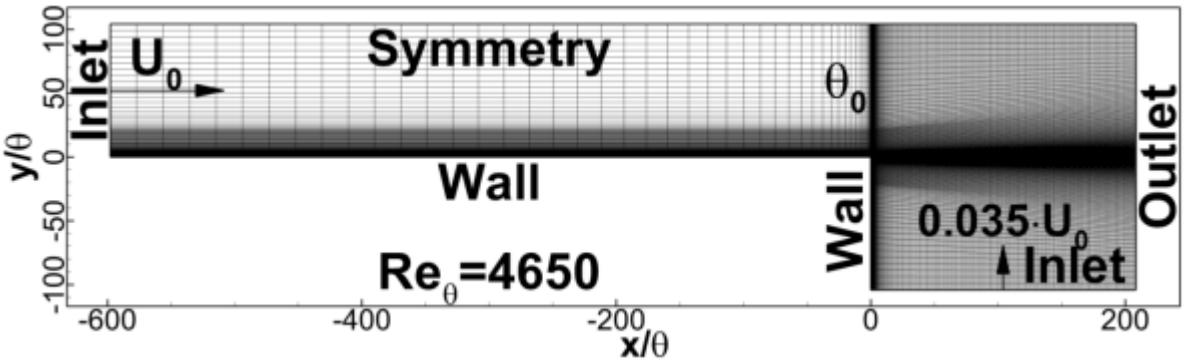
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The inlet of the computational domain shown in Figure 12.25: The domain and grid for the single stream mixing layer flow (p. 219) is located at $x/\theta_0=-597$, while the outlet boundary is located at $x/\theta_0=208$ ($x/\theta_0=0$ corresponds to the step location). The size of the domain in the wall normal direction is $104\cdot\theta_0$ upstream of the step and $208\cdot\theta_0$ downstream of it. Finally, in the spanwise direction the domain is $41.6\cdot\theta_0$ wide.

Two computational grids are considered (Figure 12.25: The domain and grid for the single stream mixing layer flow (p. 219)). Mesh 1 consists of about $2.1\cdot10^6$ hexahedral cells with the maximum grid step of $1.04\cdot\theta_0$ downstream of the step. Mesh 2 consists of $7.6\cdot10^6$ hexahedral cells and has a maximum grid step of $0.52\cdot\theta_0$ in the mixing layer region (spanwise direction). It should be noted that both meshes have the same node distribution upstream of the step with the grid steps in the wall normal direction chosen to satisfy $\Delta y_w<1$ in the entire domain.

The boundary conditions are specified as follows (Figure 12.25: The domain and grid for the single stream mixing layer flow (p. 219)). A constant velocity U_0 is specified at the flat plate inlet. Following the experimental setup, a velocity of $0.035\cdot U_0$ is set at the bottom inlet below the mixing layer to provide the correct entrainment rate for a shear layer at zero pressure gradient and to ensure that the freestream velocity gradient in the streamwise direction is approximately zero. A constant pressure is set at the outlet boundary. No-slip conditions are utilized at solid walls. Finally, periodic conditions are employed in the spanwise direction.

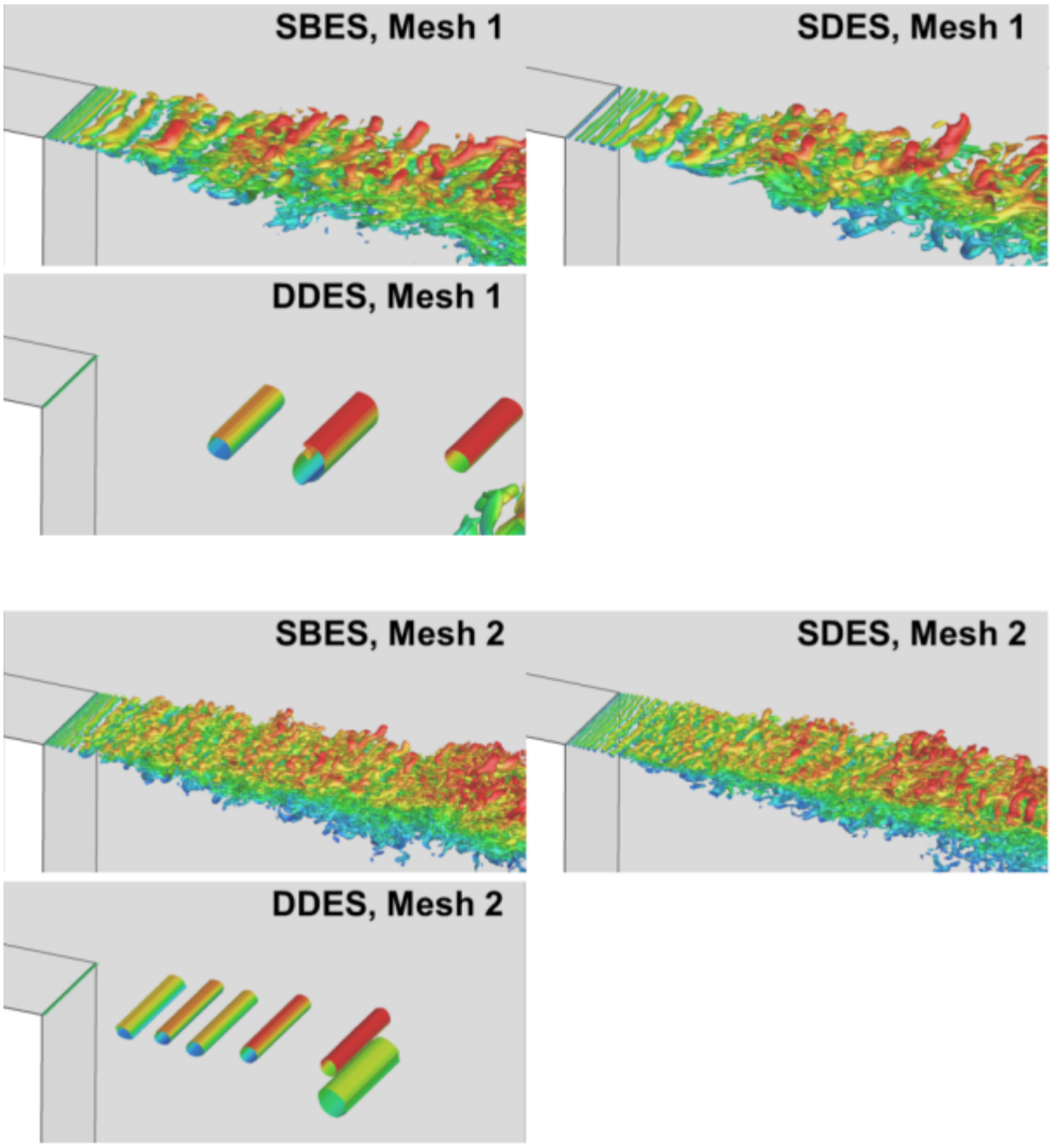
Figure 12.25: The domain and grid for the single stream mixing layer flow



As seen from the instantaneous structures visualized with the use of isosurfaces of the Q-criterion (Figure 12.26: Isosurfaces of the Q-criterion colored with the velocity magnitude (p. 220)), SBES and SDES provide a swift development of three-dimensional turbulence downstream of the step, while DDES yields almost two-dimensional structures in the entire domain regardless of the used grid. For SDES, this rapid transition relative to DDES is based on the alternative definition of the grid length scale and the reduced C_{SDES} constant. For SBES, it is based on the activation of the low eddy viscosity levels from the WALE model in the LES domain.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

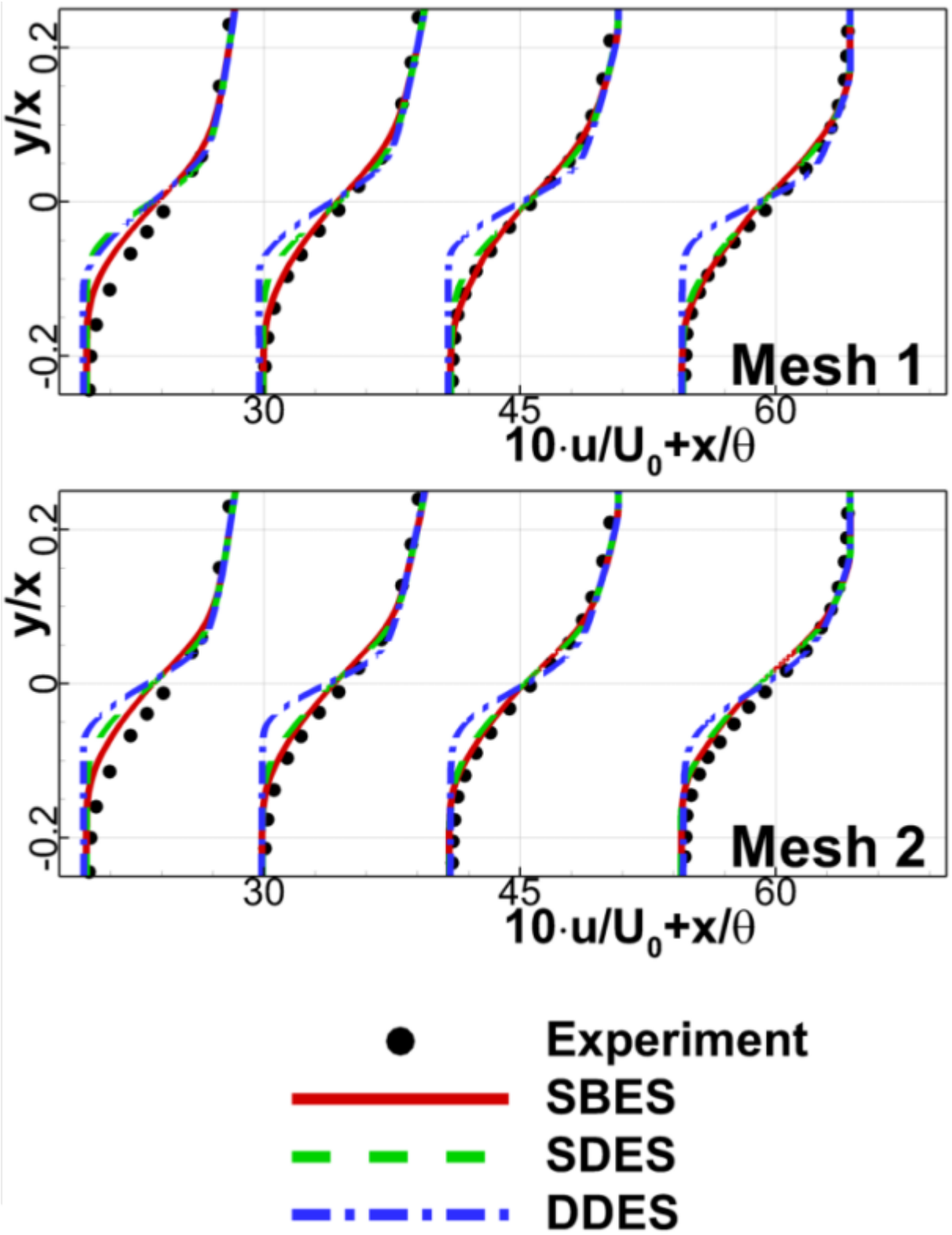
Figure 12.26: Isosurfaces of the Q-criterion colored with the velocity magnitude



Due to the much faster transition from RANS to LES, both, the SDES and the SBES model show superior agreement with the experiments in terms of the velocity profiles compared to DDES (see Figure 12.27: Profiles of the mean velocity at different sections (x/θ=19.3, 29.6, 40.6, and 54.2) for mixing layer test case (p. 221)). Similar performance could be obtained with DDES only under severe grid refinement – meaning that SDES/SBES can produce more realistic solutions on much coarser grids than required for DDES.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.27: Profiles of the mean velocity at different sections (x/θ=19.3, 29.6, 40.6, and 54.2) for mixing layer test case

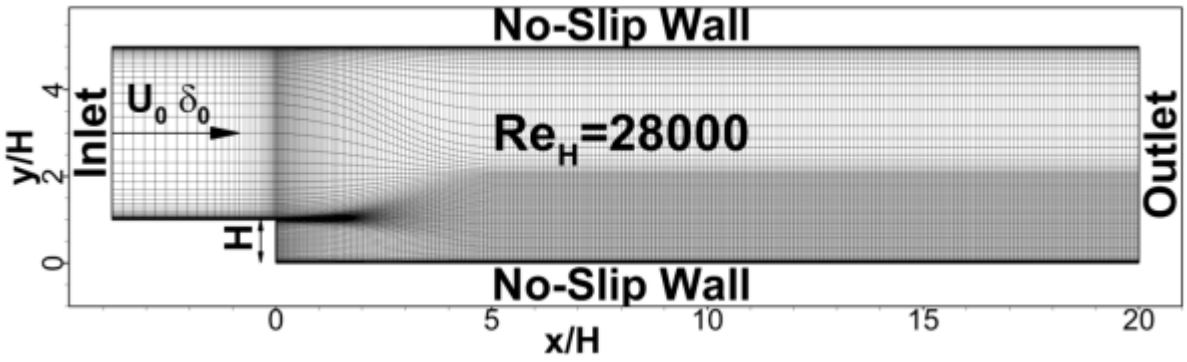


12.1.2.2.5.2. Backward-Facing Step I

This flow has been experimentally studied by Vogel and Eaton, 1985 35 (p. 274) for $\mathrm{Re}=\rho\cdot U_0\cdot H/\mu=28000$ (U_0 is the bulk velocity at the inlet section, H is the step height, ρ is the constant density, and μ is the constant dynamic viscosity). The expansion ratio relative to the upstream plane channel is 1.25.

The computational domain (Figure 12.28: The domain and grid for the backstep test case (p. 222)) extends from $x/H=-3.8$ at the inlet to $x/H=20$ at the outlet ($x/H=0$ corresponds to the step location), while in the spanwise direction it has the length of $4\cdot H$. The computational grid shown in the figure consists of $2.8\cdot 106$ hexahedral cells with maximum steps in the streamwise and spanwise directions of $0.1\cdot H$ and $0.05\cdot H$ respectively, which corresponds to $\Delta x^+=200$ and to $\Delta z^+=100$ in the wall units. The steps in the wall normal direction correspond to $\Delta y^+<1$ along all walls in the domain.

Figure 12.28: The domain and grid for the backstep test case



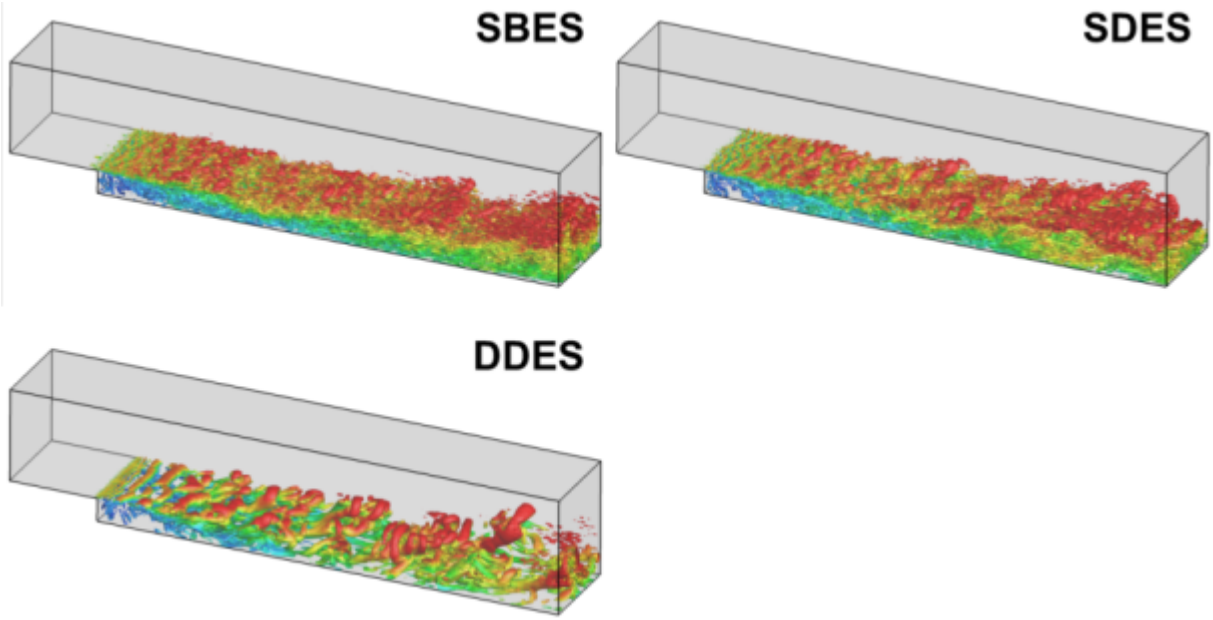
The boundary conditions are specified as shown in Figure 12.28: The domain and grid for the backstep test case (p. 222). The inflow profiles are obtained from the precursor SST-RANS simulation of the developing plane channel flow up to the experimental boundary layer thickness of $\delta_0=H$. Periodic conditions are applied in the spanwise direction. No-slip conditions are specified on the solid walls. At the outlet boundary, a constant pressure is specified, while the other transported quantities are extrapolated from the domain interior.

A non-dimensional time step of $\Delta t=0.02\cdot H/U_0$ ensures that the CFL number is less than one in the entire domain. The instantaneous flow fields are averaged over 5000 time steps in order to achieve a sufficient time sample for the unsteady statistics.

As seen from iso-surfaces of the Q-criterion shown in Figure 12.29: Isosurfaces of the Q-criterion colored with the velocity magnitude for backstep test case (p. 223), SBES and SDES provide visually similar turbulent structures downstream of the step, while relatively larger structures are observed for DDES. It can also be seen that the blending functions of SBES and SDES, relative to that of DDES, cover a smaller portion of the flow near the bottom wall (Figure 12.30: Contours of the blending function with the vorticity iso-lines for backstep test case (p. 224)), while, in the channel center, the SDES and SBES models work in the RANS mode. DDES also remains in RANS mode there, but this is not visible due to the lack of grid information in the DDES shielding function. DDES provides a larger eddy viscosity level in the mixing zone relative to those of SBES and SDES, as can be seen from Figure 12.31: Contours of the eddy viscosity ratio for backstep test case (p. 225). Again, this explains the larger turbulent structures observed in this region with DDES.

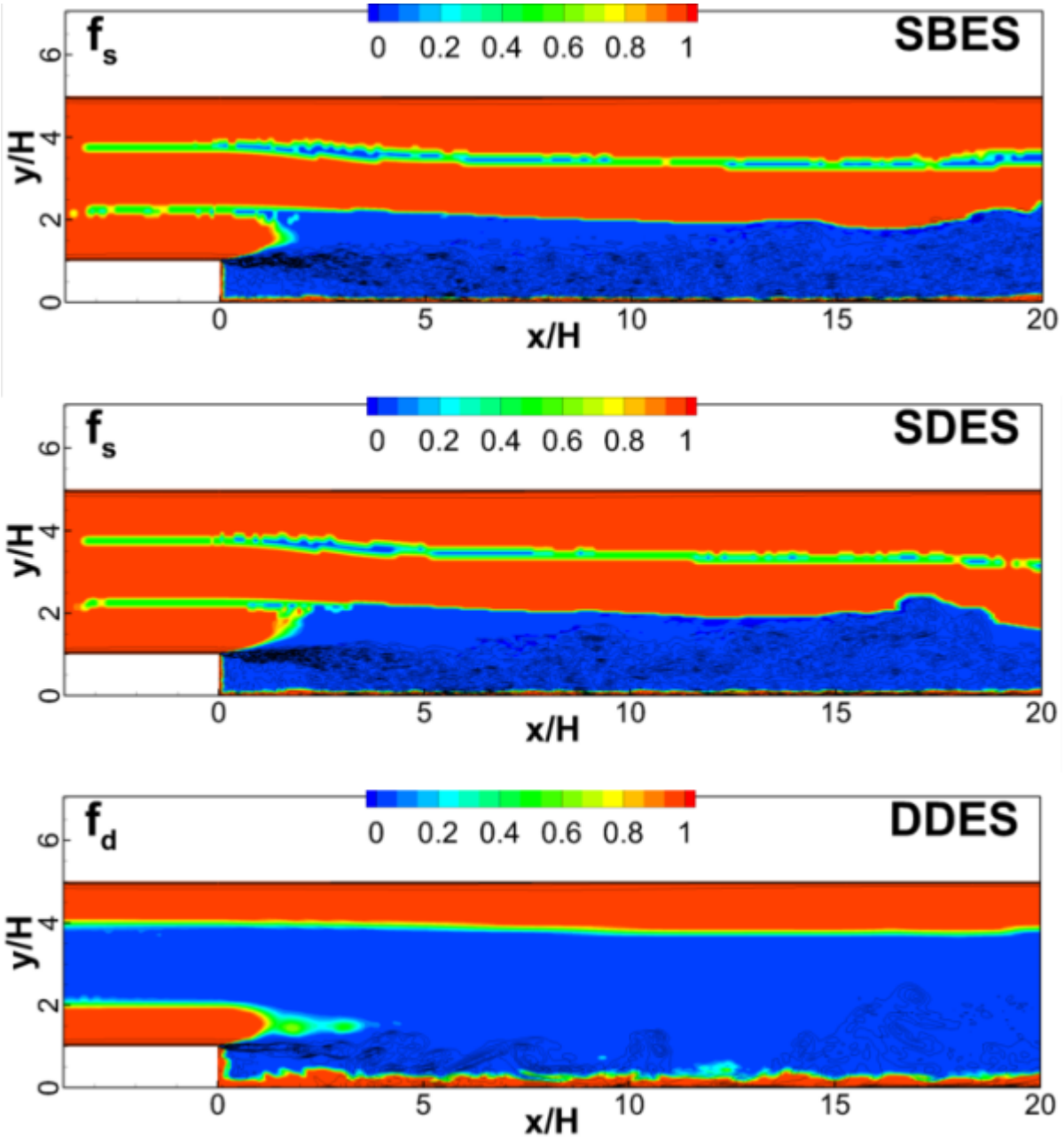
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.29: Isosurfaces of the Q-criterion colored with the velocity magnitude for backstep test case



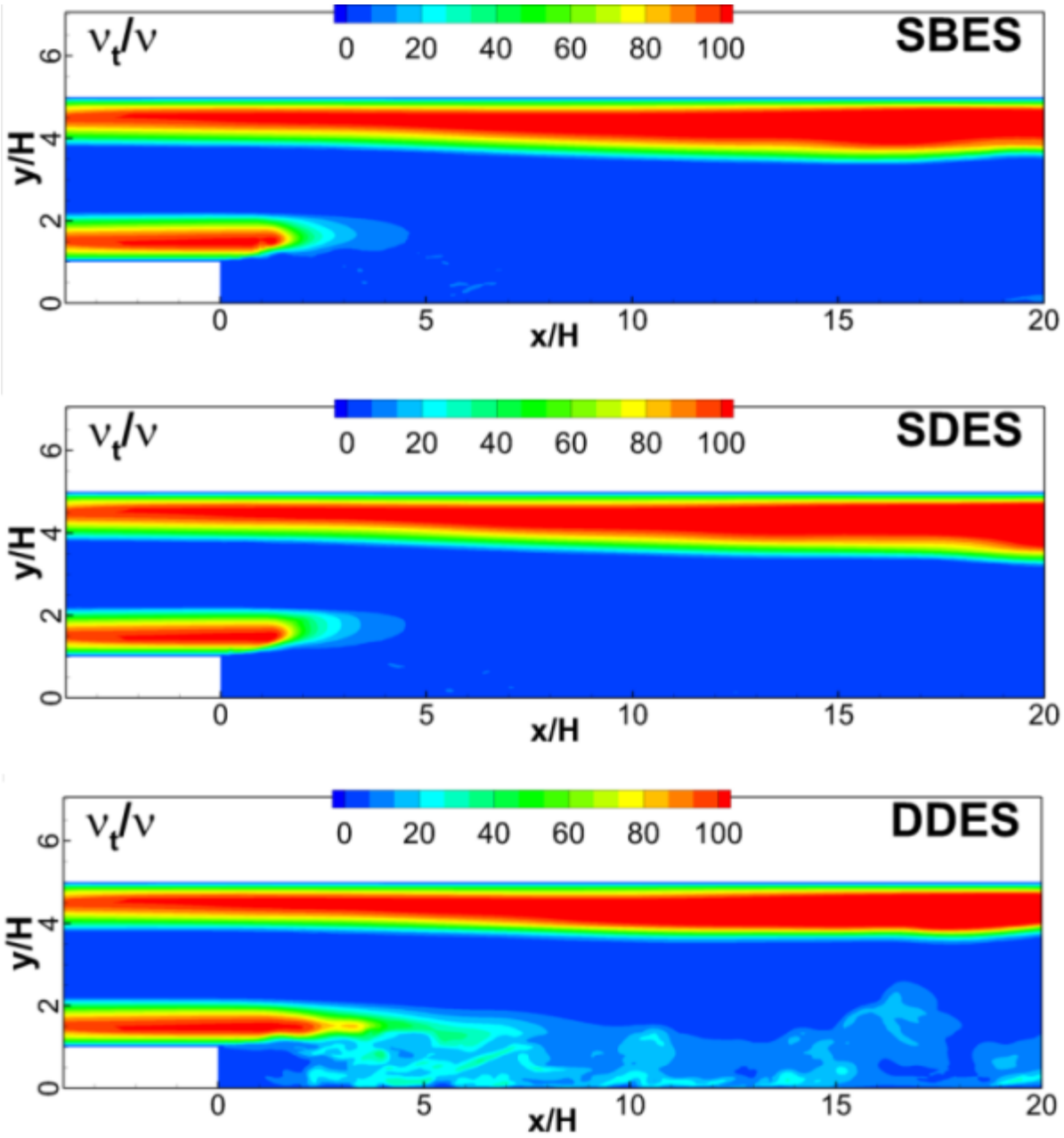
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.30: Contours of the blending function with the vorticity iso-lines for backstep test case



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

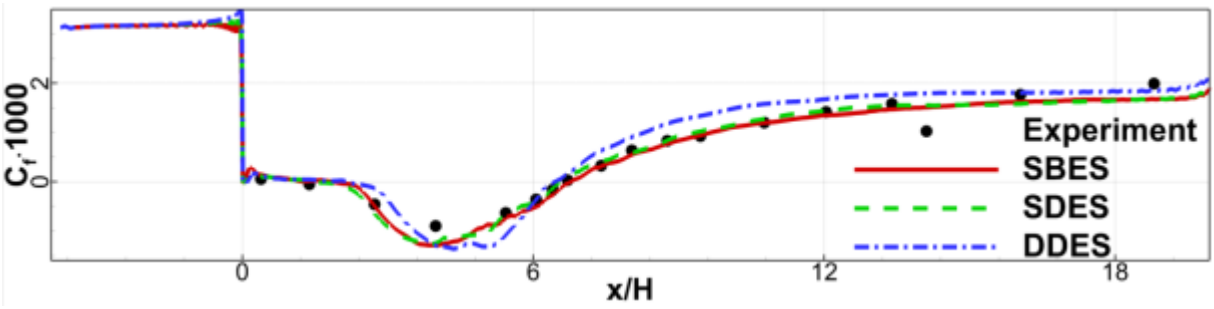
Figure 12.31: Contours of the eddy viscosity ratio for backstep test case



However, despite the above differences, the skin friction coefficient distribution depicted in Figure 12.32: Distribution of the skin friction coefficient for backstep test case (p. 226) along the bottom wall shows that the results of all the models are in good agreement with the experimental data. However, the DDES model shows a slightly overly large separation zone and a slight over-prediction of C_f-recovery downstream of the reattachment point. Finally, the mean and RMS velocity profiles for all the considered models agree fairly well with the experimental data. The better prediction for the backstep as compared to the mixing layer above is not surprising because the mesh for the backstep has been designed for the DDES model. It features lower grid spacing (relative to the upstream boundary layer thickness) in spanwise direction compared to the mixing layer case.

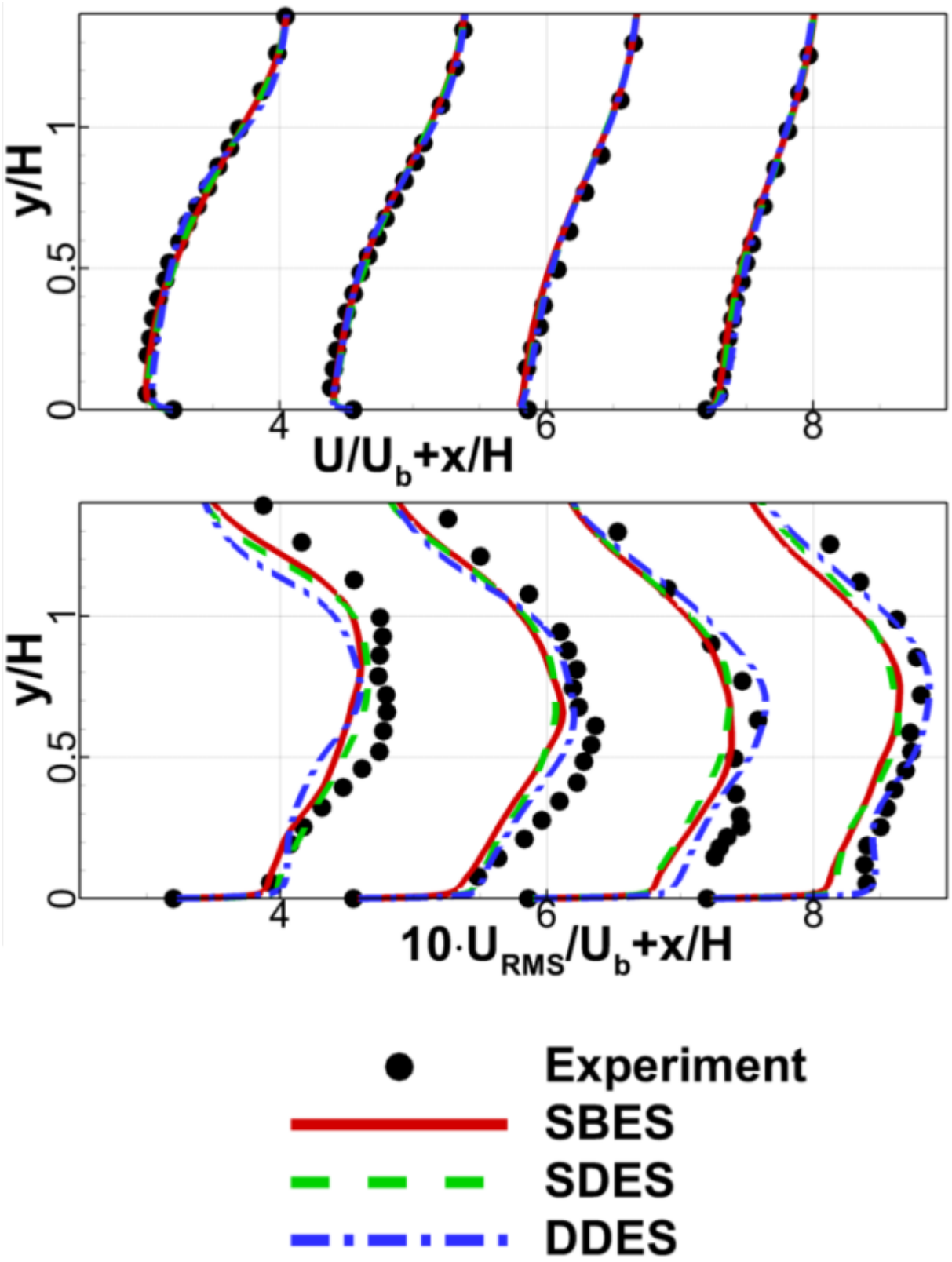
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.32: Distribution of the skin friction coefficient for backstep test case



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.33: Profiles of the mean and RMS velocity at different sections (x/H=3.2, 4.55, 5.87, and 7.2) for backstep test case



12.1.2.3. Stable Flows and Wall Boundary Layers

12.1.2.3.1. Flow Physics

Stable flows in this context are characterized by a continuous development of the turbulence field. For such flows, the turbulence at a certain location depends strongly/entirely on the turbulence upstream of it. There is no mechanism for quickly generating new turbulence and overriding the upstream turbulence field. Stable flows in the context of this discussion are essentially wall-bounded flows, which are either attached or have small separation bubbles.

- Channel and pipe flows (attached and mildly separated)
- Boundary layers (attached and mildly separated)

12.1.2.3.2. Modeling

For stable flows, the use of embedded or zonal RANS-LES methods with a well-defined interface between the RANS and the LES zone is essential. Synthetic turbulence must be introduced at the RANS-LES interface to ensure a proper balance between the modeled and the resolved content of turbulence. The introduction of resolved/synthetic turbulence allows the balance between RANS and LES turbulence across the interface to be preserved (assuming the synthetic turbulence is of sufficient quality). Neither DDES nor SAS-type models are able to switch from RANS to SRS mode in such stable situations. Even in cases where resolved turbulence is specified at the inlet (or an interface) the SAS model will typically switch back to its underlying RANS mode after some boundary layer thicknesses (for example, Davidson, 2006 4 (p. 269)). The DDES model does remain in LES mode but with a significant error in the logarithmic layer. The SBES model is able to perform in WMLES mode and provide proper velocity profiles downstream of synthetically generated turbulence.

Even an explicit switch from a RANS to a LES model (and the corresponding grid refinement in the LES zone) at the interface without an introduction of synthetic turbulence does not work well. If sufficient resolution is provided in the LES zone, the flow would eventually go through a transitional process and recover the fully turbulent state. However, such a process would require many boundary layer thicknesses, with an entirely unbalanced model formulation in-between. This method is not acceptable in most technical flows and must be avoided.

In such stable flows, the most suitable selection of hybrid RANS-LES models are Embedded- or Zonal models, where the RANS and the LES zones are user-defined and synthetic turbulence is injected at the RANS-LES interface. As mentioned previously, the RANS-LES interface should be placed in a non-critical region of the flow (equilibrium flow), since existing synthetic turbulence generators do not provide realistic turbulent fluctuations for strongly non-equilibrium flows. As a result, placing the interface in such regions results in a too-slow relaxation from synthetic to "real" turbulence (typically, several boundary layer thicknesses).

As an alternative, the RANS and LES simulations can be carried out separately. The RANS domain would include the full geometry whereas the LES solution can be carried out on a smaller portion of the original domain. This separate LES domain would be identical to the LES zone in the equivalent ELES set-up. The information from the larger RANS solution can then be mapped onto the boundaries of the LES domain. Synthetic turbulence should be introduced at the inlet of the LES domain. This approach can be used if you are confident that the physical decoupling has very little or no effect onto the overall flow topology. The advantage of the decoupled method over the ELES approach is that the RANS solution does not have to carry the burden of the ex-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

cessive temporal resolution that the LES domain would have otherwise required. However, you should be aware that some scripting is required for mapping the results from RANS to LES in the decoupled approach.

The models selected in the RANS and LES zone depend on the flow physics. In the RANS zone, a suitable model for the flow should be selected. In the LES zone, the use of a WMLES formulation is typically recommended for wall boundary layers in order to avoid the unfavorable Reynolds number scaling of classical LES models. For free shear flows, the WALE model should provide optimal performance.

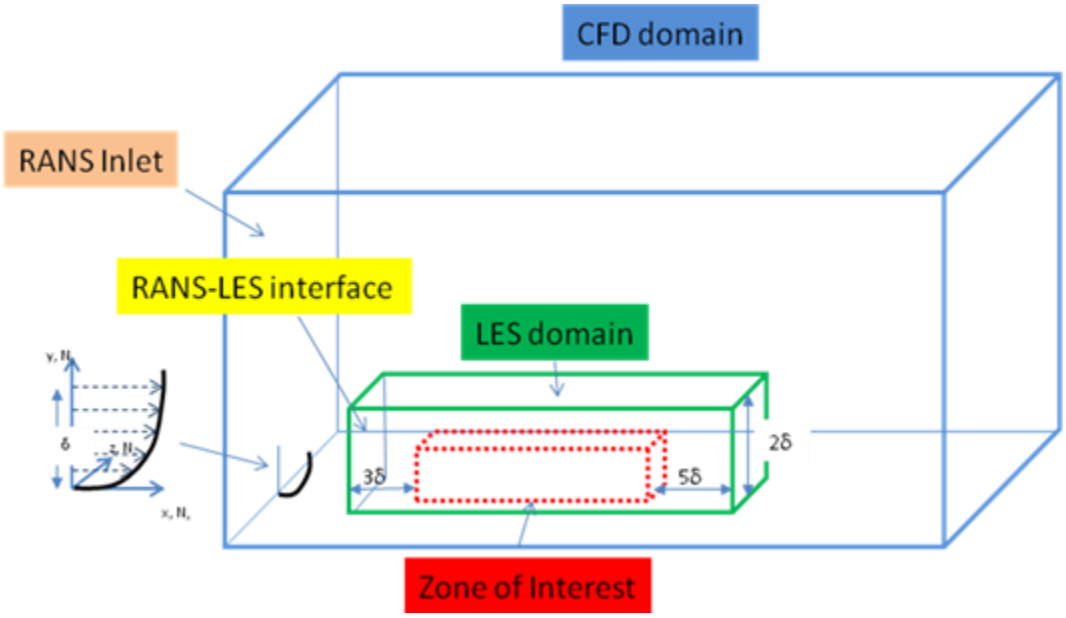
12.1.2.3.3. Meshing Requirements

Figure 12.34 (p. 229) shows the schematic of an ELES set-up. There is a central area (red), which is the domain of interest (for example, a boundary layer with a separation bubble). This area is not specifically defined in the ELES set-up, but is just used to demonstrate how such a zone would be handled. Clearly, you would not place the LES zone (green) directly at the start of the zone of interest, but extend it upstream and downstream of that region by several boundary layer thicknesses as indicated in Figure 12.34 (p. 229). For fully developed pipe/channel flow, the boundary layer thickness should be estimated as half of the pipe diameter/channel height. The LES zone is then embedded into a larger RANS zone (blue).

The meshing requirements are those of the underlying turbulence models. In the RANS zone typical RANS resolution requirements should be satisfied (20–30 cells across the wall boundary layer with possibly a $y+\sim 1$ and 15–20 cells across free shear flows).

In the LES zone, the resolution requirements depend on the details of the LES model formulation and the flow type. For free shear flows, cubic grid cells with a minimum of $\sim 15-20$ cells per shear layer thickness should be used. For wall-bounded flows, the resolution requirements are those described in Section 12.1.1.5.1 (p. 191) for classical LES and in Section 12.1.1.6 (p. 200) for WMLES.

Figure 12.34: Sketch of embedded LES (ELES) domain



For wall-bounded flows, it is clear that large domains cannot be covered in SRS mode, even when using WMLES. In most cases you would limit the domain size of the LES zone by one or more of the following concepts:

- Use only a limited spanwise domain size.
  - Apply periodic boundary conditions where appropriate. The domain size has to cover a minimum of 3–5 boundary layer thicknesses in the spanwise direction to avoid inaccuracies caused by the spanwise periodicity condition. Care must be taken that this requirement is satisfied for the entire LES domain. In case the boundary layer grows in the streamwise direction, the most downstream location is relevant for the estimate.
  - In cases where no periodicity can be applied, place the spanwise interfaces into a region of limited interest.
- Place the upstream RANS-LES interface economically to reduce the size of the LES domain. However, the interface should be located in a zone of undisturbed equilibrium flow. Place the RANS-LES interface at a minimum of ~3 boundary layer thicknesses upstream of the zone of interest (for example, a separation region). Limit the size of the RANS-LES interface to the shear layer to capture; that is, do not extend the interface far into the freestream, as the code will then generate resolved turbulence in freestream regions where no LES is required. The Vortex Method (VM) would also generate a large number of vortices if the RANS-LES interface were too large.
- Place the downstream LES-RANS interface economically to reduce the size of LES domain. However, do not place the interface immediately downstream of the zone of interest, but place it several boundary layer thicknesses farther downstream to avoid any negative influence of the downstream RANS model (this approach enables the boundary layer to recover several boundary layer thicknesses downstream of a separation before switching back to RANS)
- Limit the height of the LES zone, but allow for some space above the boundary layer. Typically the LES zone should be about twice as thick as the boundary layer.

In order to check the quality of the simulation, sensitive quantities like time-averaged wall shear stress should be plotted across the RANS-LES zones. There should be no large jump in those quantities and the unavoidable disturbance caused by the interface should be recovered before entering the zone of interest.

12.1.2.3.4. Numerical Settings

Zonal methods typically enable a separate selection of numerical settings in the RANS and LES zones. For very sensitive simulations, you can therefore select a pure Central Difference (CD) in the LES domain, while using an appropriate numerical scheme in the RANS parts. However, you can also select a global scheme, in which case the Bounded Central Difference (BCD) scheme is recommended.

12.1.2.3.5. Examples

12.1.2.3.5.1. Periodic Channel

The periodic channel flow is not an ELES, but a WMLES application. This section of the document shows that as WMLES is typically used in the LES portion of ELES/ZLES applications. The entire domain is WMLES and there are no RANS-LES interfaces. Simulations of this flow have been

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

carried out assuming incompressible fluid at several Reynolds numbers based on friction velocity μ_t and channel height $h=H/2$, Re = 395, 760, 1100, 2400, and 18000. The flow is driven by a constant pressure gradient $dp/dx=-2\cdot\rho\cdot\mu_t\mu_t/H$, where p is the pressure and ρ is the density. This pressure gradient is taken into account in the governing equations via a source term in the momentum equations, which allows imposing periodic boundary conditions not only in the spanwise direction z, but also in the streamwise direction x. Note that within such an approach, the bulk velocity of the flow is not specified and should be obtained as a part of the solution, which means that it could be different with different turbulence models. Alternatively, you can specify the mass flow and the solver will adjust the imposed pressure gradient accordingly.

The size of the computational domain shown in Figure 12.35 (p. 231) is equal to $4H$ in the streamwise direction and $1.5H$ in the spanwise direction. For all considered Reynolds numbers, the computational grid is unchanged in the streamwise and spanwise directions with a uniform grid-spacing of $0.05H$ and $0.025H$ respectively. This gives 10 cells per channel half width, $h=H/2$, (h being the relevant boundary layer thickness) in the streamwise and 20 cells per h in the spanwise direction. Different grids have been used in the wall-normal direction. This arrangement provides a sufficient resolution ($\Delta y_w^+<1$ near the wall) at different Reynolds numbers. Note, however, that all simulations could have been performed on the finest grid. The non-dimensional time step is $U\Delta t/H=0.02$, which ensures that the CFL number is $CFL<0.5$ in the entire domain. The solution was averaged in time over 5000 time steps. Table 12.3 (p. 231) gives the details of the grids used in the simulations and the resulting non-dimensional grid spacing. Note that classical wall-resolved LES would require values of $\Delta x^+<40$, $\Delta z^+<20$, demonstrating the substantial savings that can be achieved with WMLES for higher Re numbers. The y^+ range in Table 12.3 (p. 231) covers the range of y^+ values in the wall normal direction, with the largest values located at the center of the channel.

Figure 12.35: Computational domain and grid for WMLES of channel flow

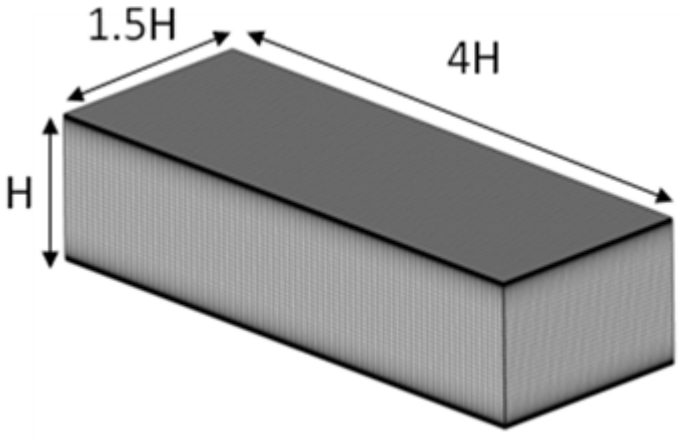


Table 12.3: Grid resolution for WMLES channel flow simulations

Re_τ	**Number of Cells**	**Number of Nodes**	Δx^+	Δy^+	Δz^+
395	384 000	$81\times81\times61$	40.0	$0.2\div30$	20.0
760	480 000	$81\times101\times61$	76.9	$0.2\div30$	38.5

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Re_τ	**Number of Cells**	**Number of Nodes**	Δx^+	Δy^+	Δz^+
1100	480 000	$81\times101\times61$	111.4	$0.2\div30$	55.7
2400	528 000	$81\times111\times61$	243.0	$0.2\div30$	121.5
18000	624 000	$81\times131\times61$	1822.7	$0.2\div30$	911.4

Figure 12.36 (p. 232) shows the turbulence structures using the Q-criterion ($Q=350[s^{-2}]$). The isosurface is colored by the streamwise velocity.

Figure 12.36: Turbulence structures for WMLES of channel flow at lowest Reynolds number (Q=350 [s^{-2}])

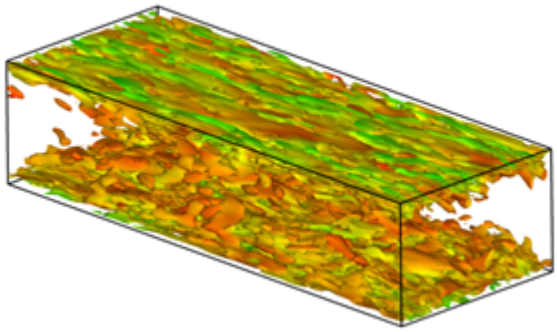
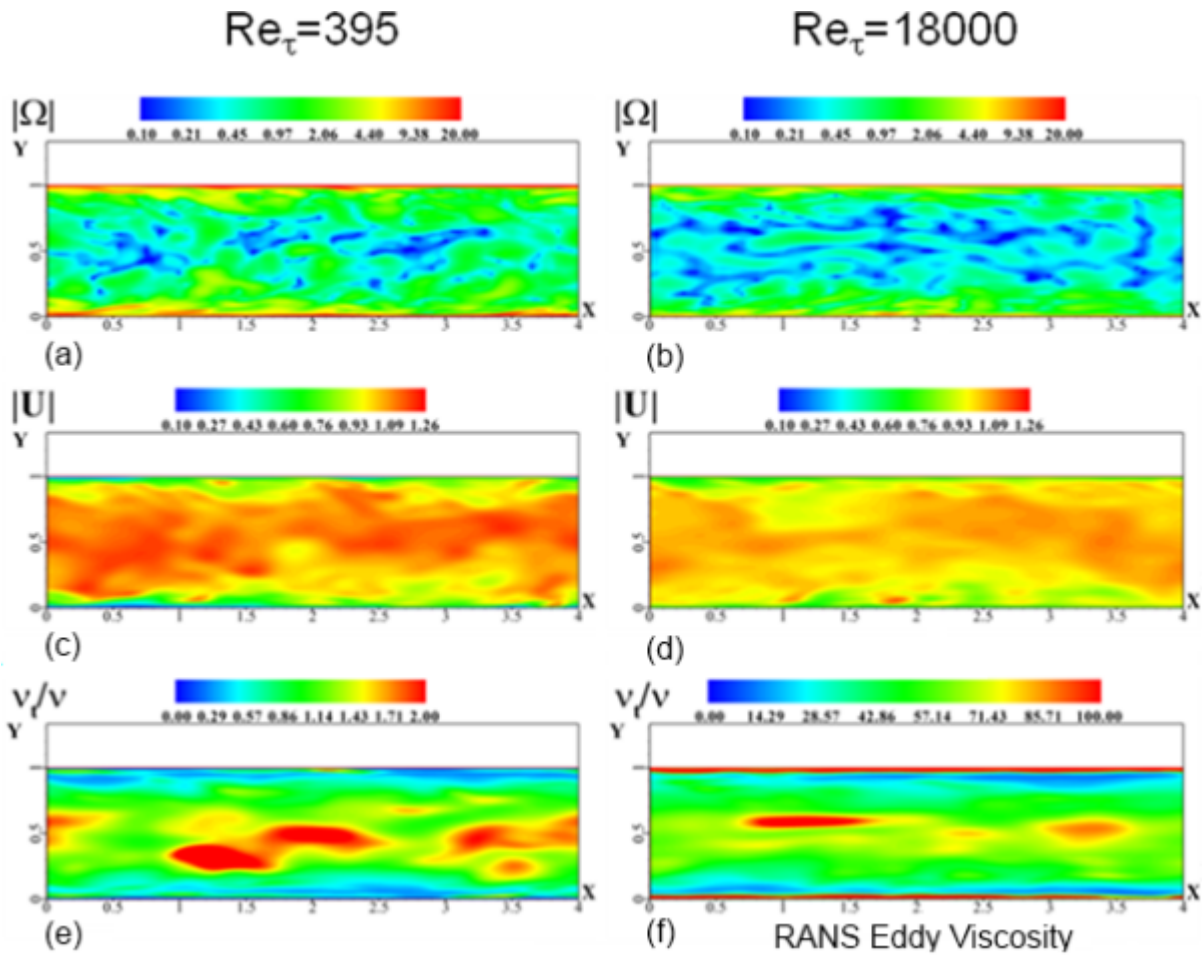


Figure 12.37 (p. 233) shows the flow in a horizontal cut through the domain for the lowest and the highest Reynolds numbers. The thin region of RANS modeling near the wall for the high Reynolds number is indicated by the high eddy-viscosity (note the different scales in the plot for the eddy-viscosity ratio for the different Reynolds numbers). RANS modeling in this context is as described in Section 12.1.1.6 (p. 200), based on the near-wall mixing length formulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.37: Flow visualization for WMLES of channel flow (a) Vorticity rate Omega, (b) Absolute value of velocity U, (c) Ratio of eddy-viscosity to molecular viscosity

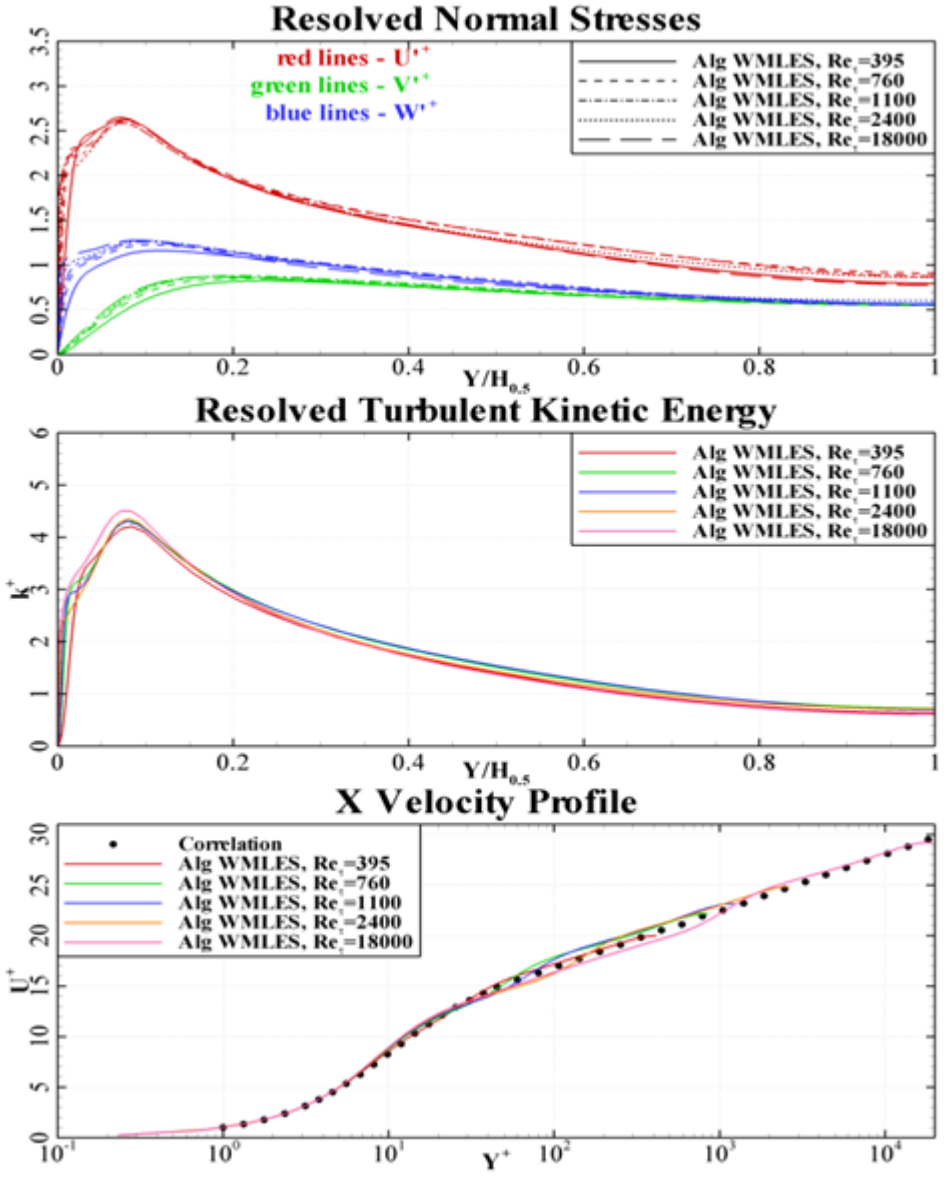


Results of the WMLES formulation and their comparison with the empirical correlation of Reichardt, 1951 23 (p. 272) are shown in Figure 12.38 (p. 234). You can see that the WMLES solutions reproduce the logarithmic layer with good accuracy. There is a slight kink at the switch from the RANS to the LES formulation, but it is moderate and does not affect global properties such as the wall shear stress.

The above simulations have been carried out with Ansys Fluent. Similar results can be obtained with Ansys CFX where WMLES is the default formulation inside the LES zone of the ZFLES method.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.38: Resolved normal stresses, turbulent kinetic energy, and mean velocity profiles for WMLES at different Reynolds numbers



The same test case has been computed with the global hybrid RANS-LES models. In order to push them into unsteady SRS mode, the simulations are started from the WMLES solution above. Please note that the DDES model is not shown because it would produce a severe shift in the logarithmic velocity profile. The IDDES model shown for comparison has been calibrated for WMLES applications; it does however have only limited shielding capabilities and is therefore not suitable for general industrial applications.

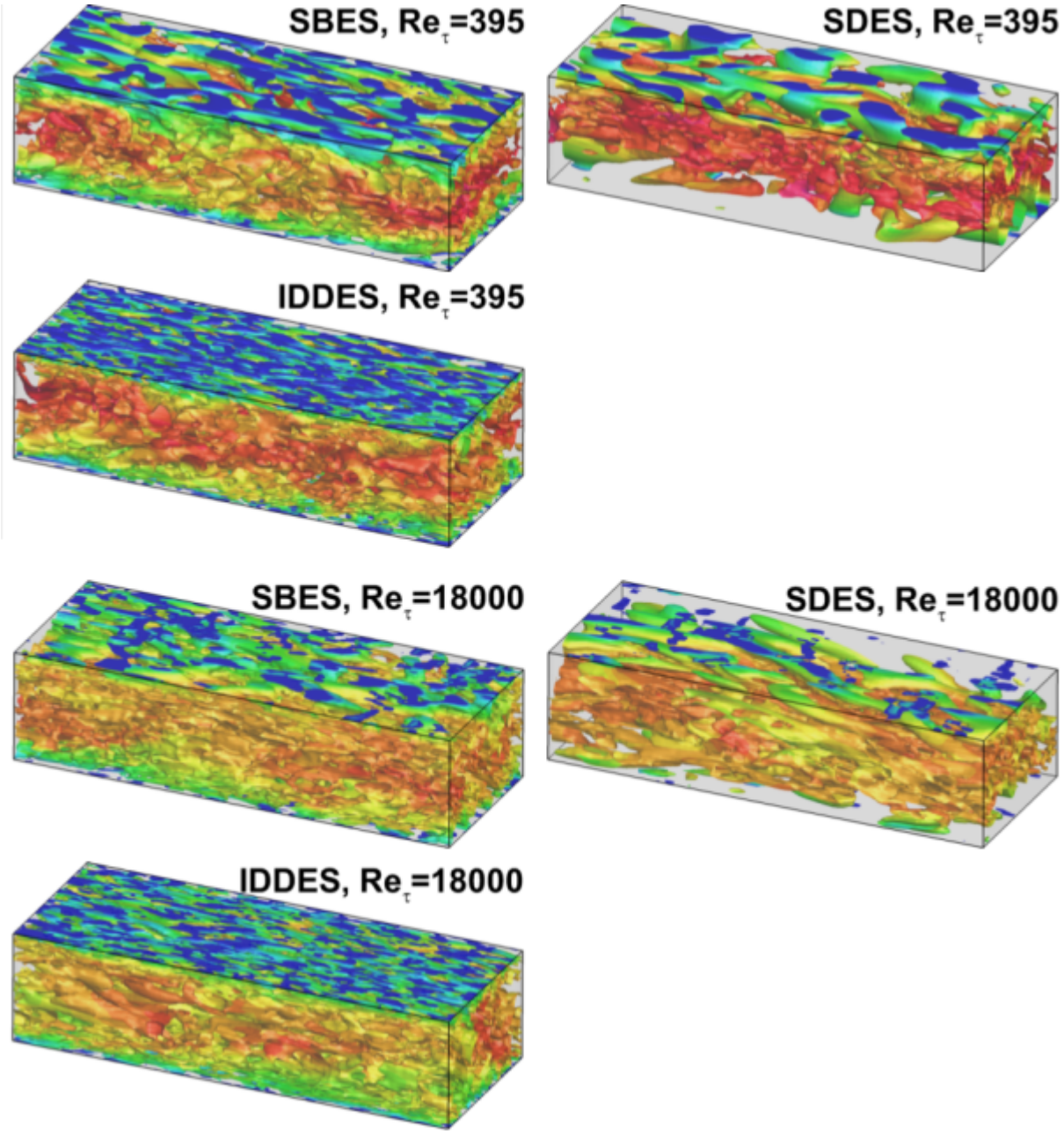
As seen from Figure 12.39: Isosurfaces of the Q-criterion colored with the velocity magnitude (p. 235), all the models provide resolved turbulent structures for both considered Reynolds numbers. The results of SBES (combination of SST and WALE) and IDDES are visually close to each other with a finer resolution provided by the IDDES model. The near wall resolved turbulence is noticeably reduced by SDES for both Reynolds numbers.

As seen from Figure 12.40: Profiles of the eddy viscosity ratio for different models (p. 236), SDES lacks resolved turbulence near the wall due to a substantially higher peak of the eddy viscosity

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

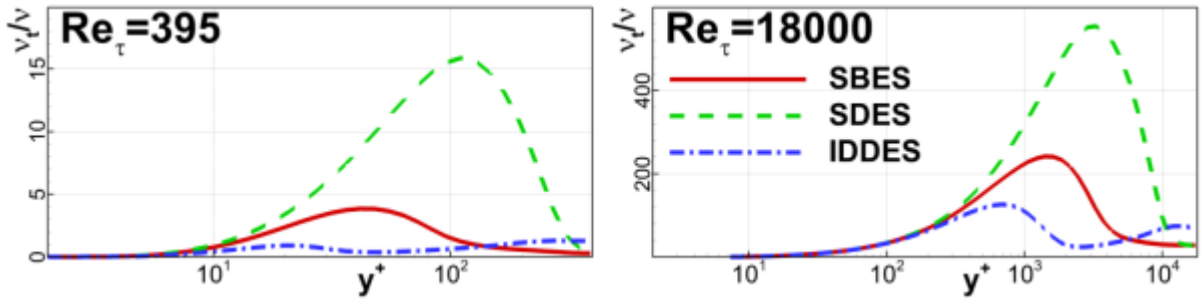
resulting from a noticeable shift of the RANS-LES interface away from the wall. SBES also yields a noticeably higher eddy viscosity peak than IDDES but much reduced relative to SDES.

Figure 12.39: Isosurfaces of the Q-criterion colored with the velocity magnitude



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.40: Profiles of the eddy viscosity ratio for different models



Interestingly, the profiles of the mean velocity as shown in Figure 12.41: Profiles of the mean velocity for different models (p. 236) are in good agreement with each other and with the empirical correlation for all models. It should be noted, that, despite SDES yielding the smallest magnitude of the LLM (Figure 12.41: Profiles of the mean velocity for different models (p. 236)), it noticeably underestimates the total turbulence kinetic energy (Figure 12.42: Profiles of the total turbulence kinetic energy for different models (p. 236)) for both Reynolds numbers. This indicates that the wall modeled LES capability of SDES is not optimal for the given meshes. At the same time, the profiles of the total kinetic energy of SBES and IDDES are relatively close to each other, which means that the models have similar wall modeled LES capabilities. The results also indicate that further improvements in terms of WMLES resolution could be obtained by moving the RANS-LES interface slightly closer to the wall.

Figure 12.41: Profiles of the mean velocity for different models

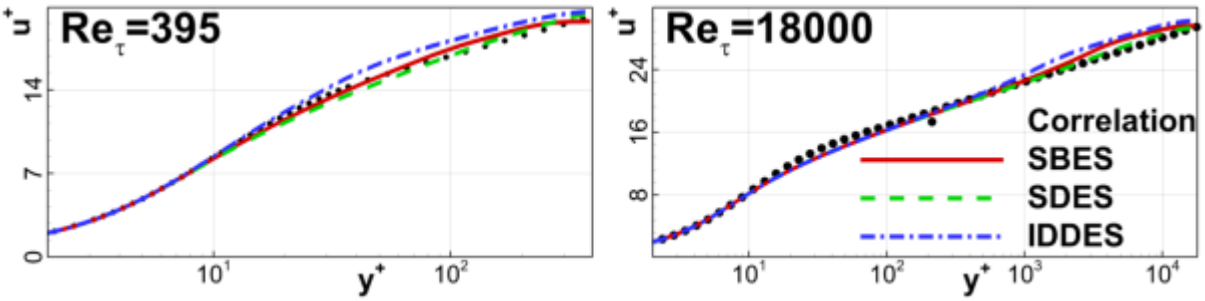
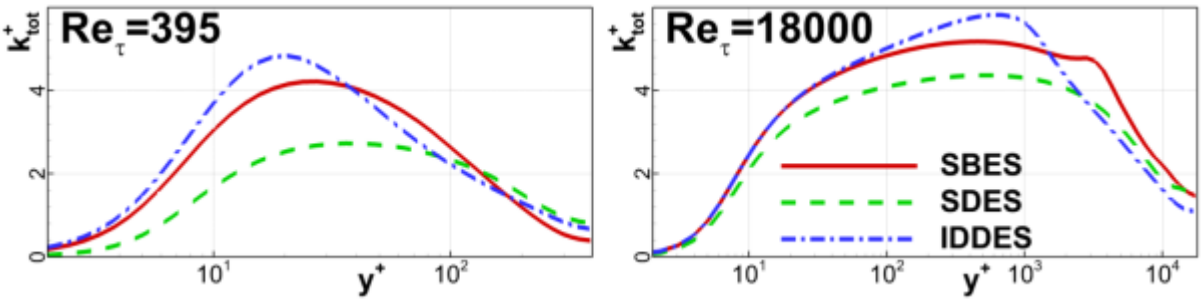


Figure 12.42: Profiles of the total turbulence kinetic energy for different models



12.1.2.3.5.2. Wall Boundary Layer

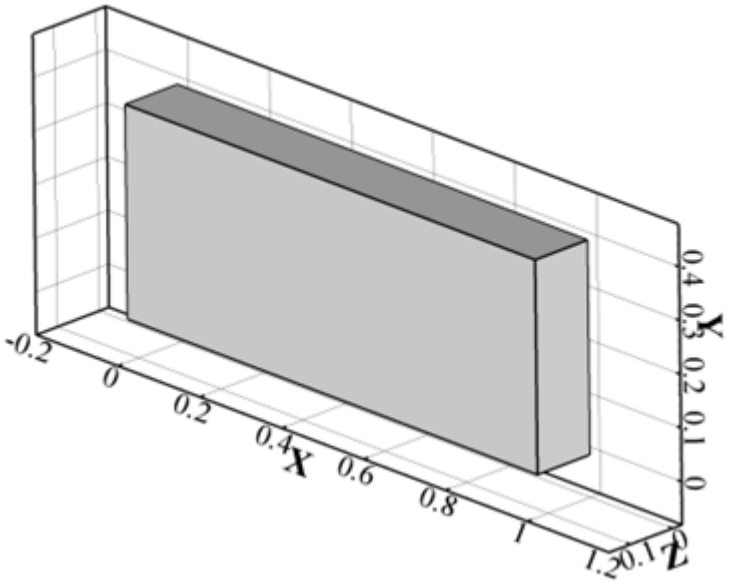
The zero pressure gradient wall boundary layer is a benchmark test case that is commonly used for turbulence model investigation due to its geometric and physical simplicity. Unlike the periodic channel test case, the wall boundary layer needs unsteady boundary conditions because

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

there is no periodicity in the streamwise direction. In the current simulations, the Vortex Method (VM) was used for these purposes (Mathey et al., 2003 15 (p. 271)).

A computational domain for this test case is shown in Figure 12.43 (p. 237). The characteristic length, which determines the geometry, is the plate length, L, of 1 [m] in the current study. Dimensions of the computational domain in x, y, and z directions are equal to L, $0.4L$, and $0.1L$ respectively.

Figure 12.43: Computational domain for a Wall Boundary Layer test case



The simulations have been performed for an incompressible fluid. A summary of physical parameters is presented in Table 12.4 (p. 237).

Table 12.4: Properties for flat plate boundary layer simulations

$\mathrm{Re}_\theta\,[-]$	**1000**	**10 000**
Inlet boundary layer thickness $\delta^\circ\,[m]$	0.032	0.032
$\Delta t\,[s]$	0.001	0.001
$\mu\,[Pa{\cdot}s]$	4.4483×10^{-6}	4.4483×10^{-7}
$\rho\,[kg{\cdot}m^{-3}]$	1.0	1.0

The geometry and the computational grid used for the test case are shown in Figure 12.44 (p. 238). The base grid is uniform in the x- and z-directions with steps 0.004 [m] and 0.002 [m] respectively. In the wall normal direction the grid was expanded by a factor of 1.15. For all computations the value of Δy^+ is less than 1, which means that the governing equations are integrated to the wall. A complete summary of all used grids is presented in Table 12.5 (p. 238).

Figure 12.44 (b) (p. 238) presents all the boundary condition types used in the simulations. The cyan color shows one of the periodic planes, the red color the no-slip wall boundary, the blue color the outlet boundary, the green color the inlet boundary, and the yellow color the symmetry boundary.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.44: Computational grid (a), (c) and applied boundary conditions (b)

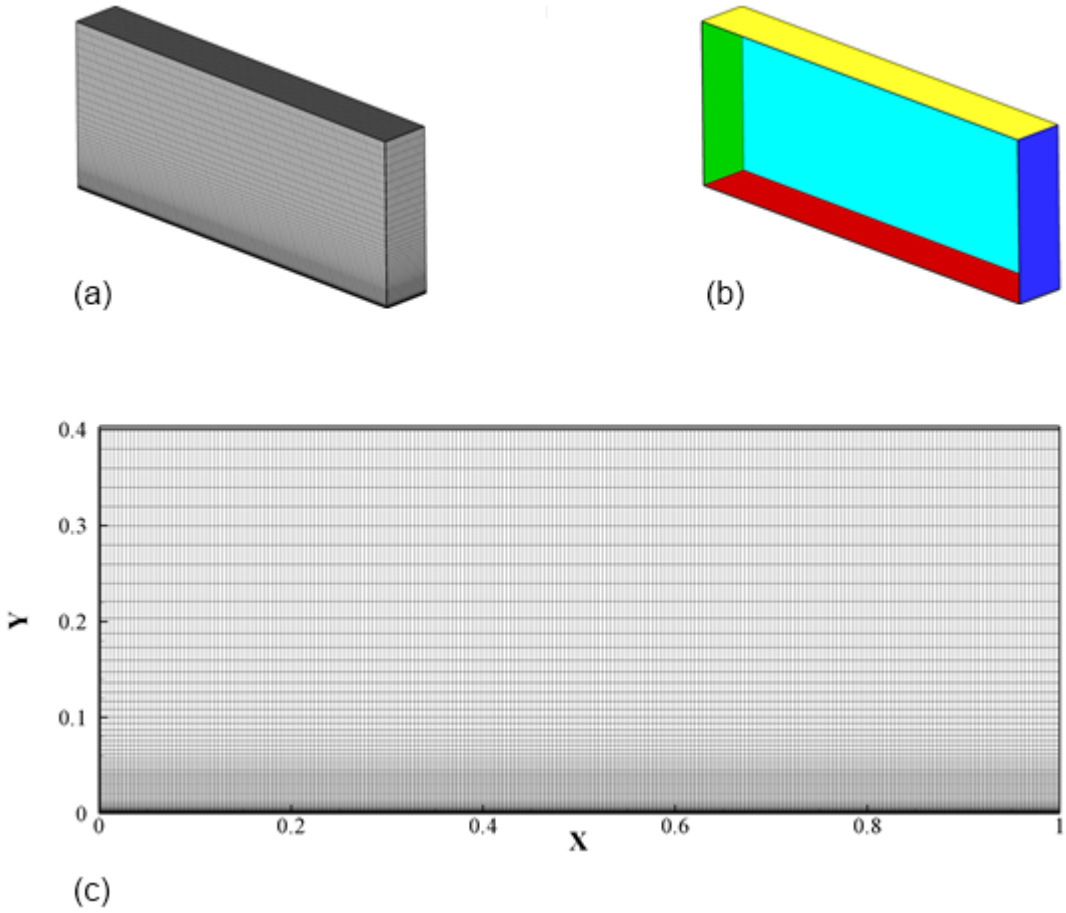


Table 12.5: Information on grids for flat plate test case

$\mathrm{Re}\Theta$	**Number of Cells**	**Number of Nodes**	Δx^+	Δy^+	Δz^+
1000	1 085 000	$251\times71\times63$	68.0	$0.30/0.80$	34.0
10 000	1 333 000	$251\times87\times63$	680.0	$0.25/0.60$	340.0

Two cases have been computed using the numerical grids with the parameters shown in Table 12.5 (p. 238). They have different inlet Reynolds numbers that are based on the boundary layer momentum thickness (Re_θ).

The Non-Iterative Time Advancement (NITA) algorithm based on the Fractional Time Step method was applied with the second order scheme for the approximation of time derivatives. The convective terms in the momentum equations have been approximated with the second order Central Difference scheme and the Green-Gauss cell-based method was used for interpolation of variables on cell faces. The Standard option was selected for the pressure interpolation scheme.

Visualizations of the flow at two values of Re_θ are shown in Figure 12.45 (p. 239). Isosurfaces of the Q-criterion that are equal to $200\,[s^{-2}]$ and colored with the velocity magnitude are depicted. It can be seen that the turbulence structures are well-developed and do not show any

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

visual decay or disruption downstream of the inlet. This indicates that the Vortex Method provides sufficiently realistic turbulent content at the inlet boundary.

Figure 12.45: Isosurfaces of Q-criterion (Q=200 [s^{-2}]) colored with velocity for a flat plate at two different Reynolds numbers

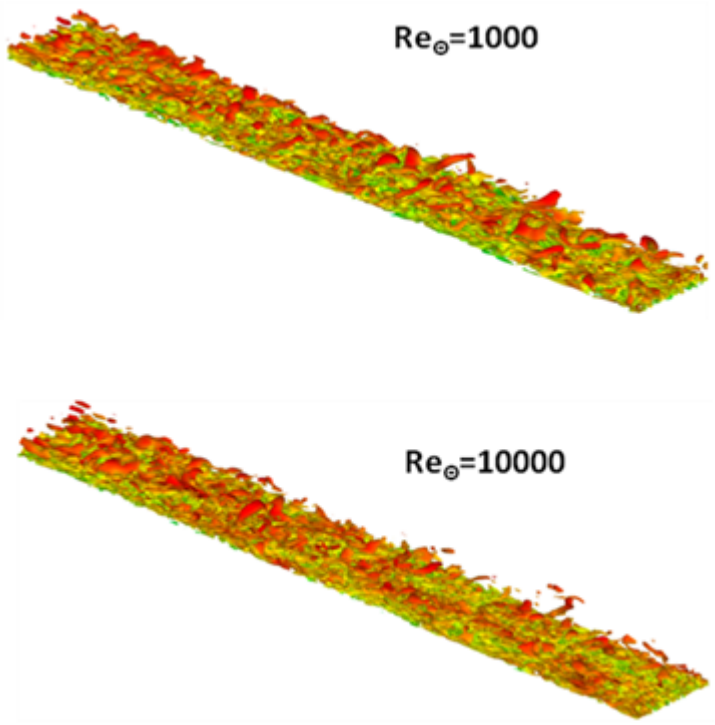


Figure 12.46 (p. 240) shows the skin-friction coefficient for the two Reynolds numbers. The results demonstrate that the inlet wall friction provided by the RANS inlet velocity profiles is maintained without any major disruption. This indicates again that the vortex method produces sensible synthetic inlet turbulence. In addition, the models react properly to the Reynolds number variation, suggesting that the WMLES can maintain a boundary layer accurately even at high Reynolds numbers, where standard LES models would fail due to a lack of resolution. Figure 12.46 (a) (p. 240) shows, in part (a), the impact of the pressure interpolation scheme, which has proven to be critical for locally stable flows. It is worth re-iterating that the PRESTO scheme requires slightly more 'running length' to recover the correct levels of turbulence and wall shear stress.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.46: Skin friction distributions along a flat plate predicted by WMLES at two Reynolds numbers (a) Re theta=1000 with different numerical settings (b) Re theta =10 000

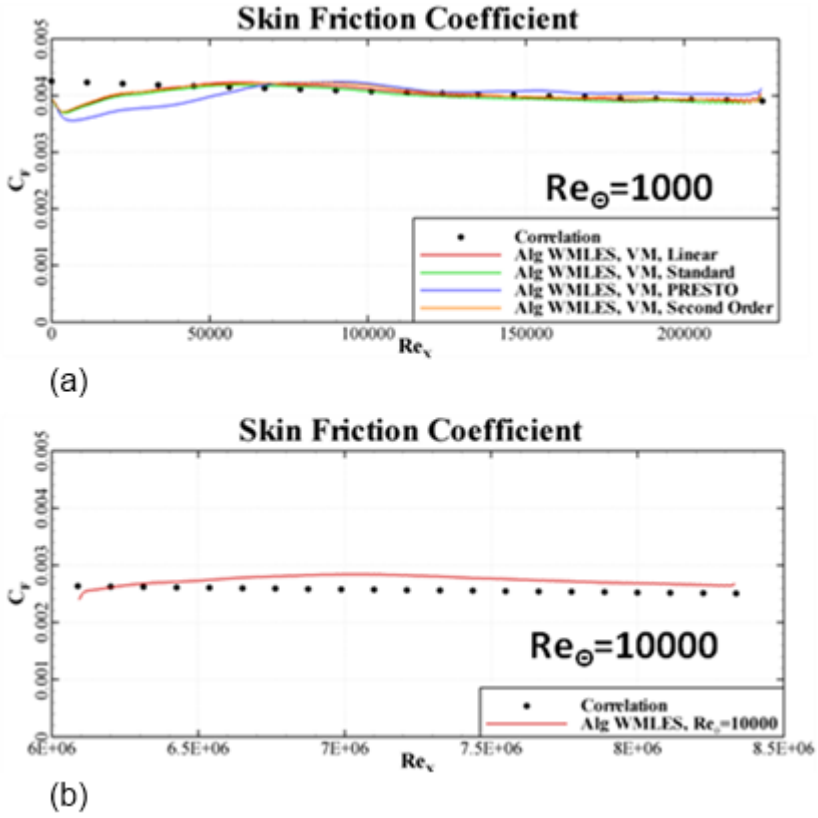
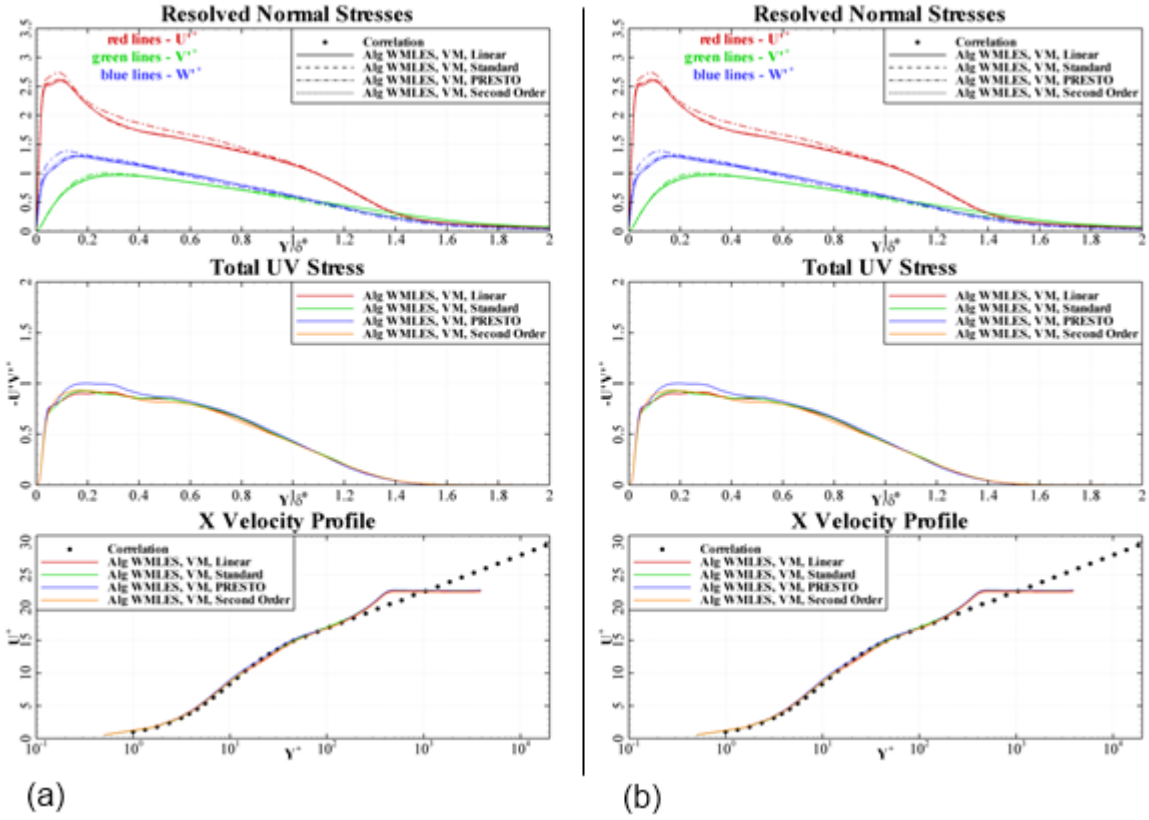


Figure 12.47 (p. 240) shows Reynolds stresses and velocity profiles from the simulations. The figure suggests that, just as for the channel flow, the quality of the simulations is fairly high in terms of both the mean flow prediction (the logarithmic profile is reproduced faithfully) and Reynolds stresses (they are well within the range expected from known DNS studies of the flat plate boundary layer).

Figure 12.47: Profiles of resolved normal and shear Reynolds stresses and mean velocity in the flat plate boundary layer predicted by WMLES at two Reynolds numbers (a) Re

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

theta=1000 with different numerical settings (b) Re theta=10 000 with the second order pressure interpolation

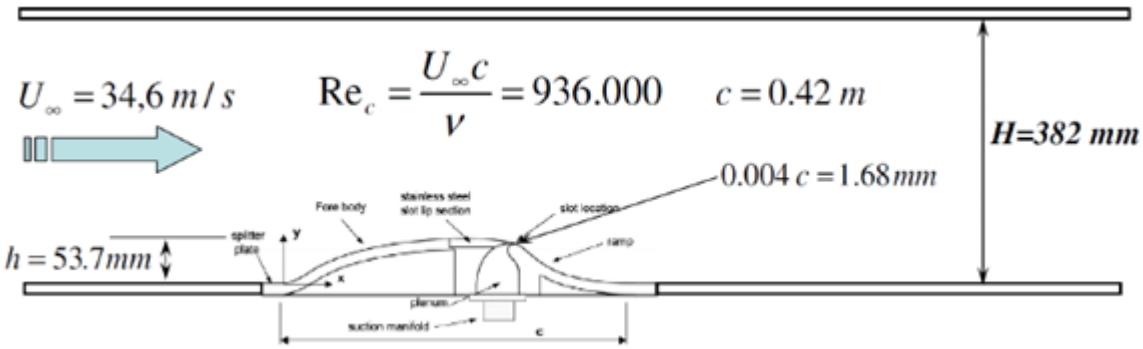


The above simulations have been carried out with Ansys Fluent. Similar results can be obtained with Ansys CFX where WMLES is the default formulation inside the LES zone of the ZFLES method.

12.1.2.3.5.3. NASA Hump Flow

A challenging test case for ELES in combination with WMLES was computed within the EU project ATAAC. The case models the flow over a hump with a relatively large separation zone on the leeward side. Figure 12.48 (p. 241) shows the experimental set-up (Greenblatt et al., 2005). Due to the limited separation zone, this flow would be categorized as a stable flow in the present context.

Figure 12.48: Experimental set-up for NASA hump flow experiment



The flow was computed with Ansys Fluent 13.0 using the SST model in the RANS zone, the vortex method at the RANS-LES interface and the algebraic WMLES option in the LES zone. The Reynolds number, based on the free-stream velocity, U_∞, and hump chord, C, is equal to 9.36×10^5. The simulation was carried out in the full domain, which extends from $-2.14C$ to $4C$ (0 corresponds to the hump beginning). In the spanwise direction, the extent of the domain is $0.2C$. The inflow boundary conditions for RANS have been set based on the preliminary flat plate boundary layer computations up to the flow section $x/C=2.14$ ($\mathrm{Re}_\theta=7200$), where the parameters of the incoming boundary layer have been measured in the experiment. At the upper wall of the channel, free-slip wall conditions have been specified.

The grid in the LES zone (see Figure 12.49 (p. 242)) consists of $200\times100\times100$ cells and was designed to provide $10\times40\times20$ cells per boundary layer volume in the streamwise, wall normal, and spanwise directions. The RANS grid is much coarser, especially in the spanwise direction. Figure 12.49 (p. 242) also presents a visualization of the turbulent structures in the LES zone that suggests a high resolution provided by the simulation (note that the momentum thickness Reynolds number at the inlet to the LES domain is relatively high ($\mathrm{Re}_\theta=7000$)). In retrospect, the set-up might not be fully optimal, as the RANS-LES interface is placed relatively close to the non-equilibrium/separation zone of the boundary layer. There are only about two boundary layer thicknesses between the interface and the bend of the geometry. A more optimal grid should cover more of the upstream boundary layer and enable the synthetic turbulence to develop over a longer running length.

Figure 12.49: (a) Grid used for the NASA hump simulation (b) Turbulent structures in the LES domain (Q-criterion colored with spanwise velocity component)

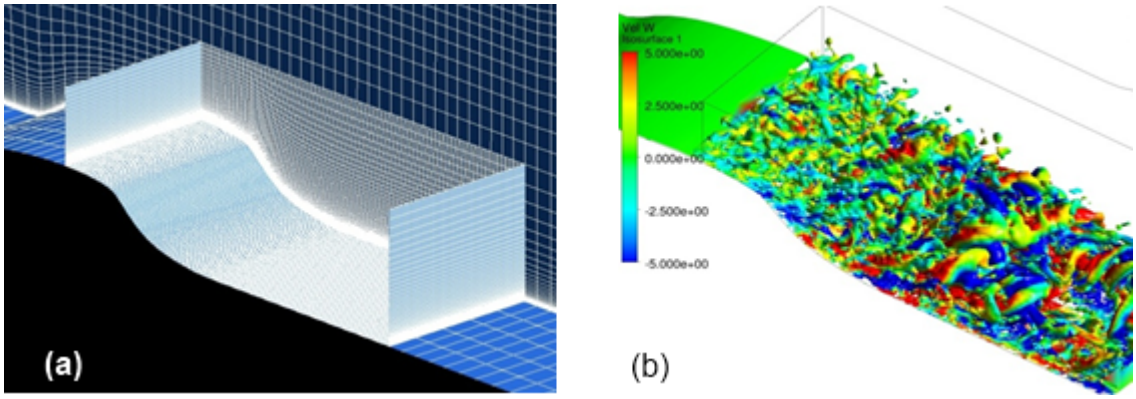
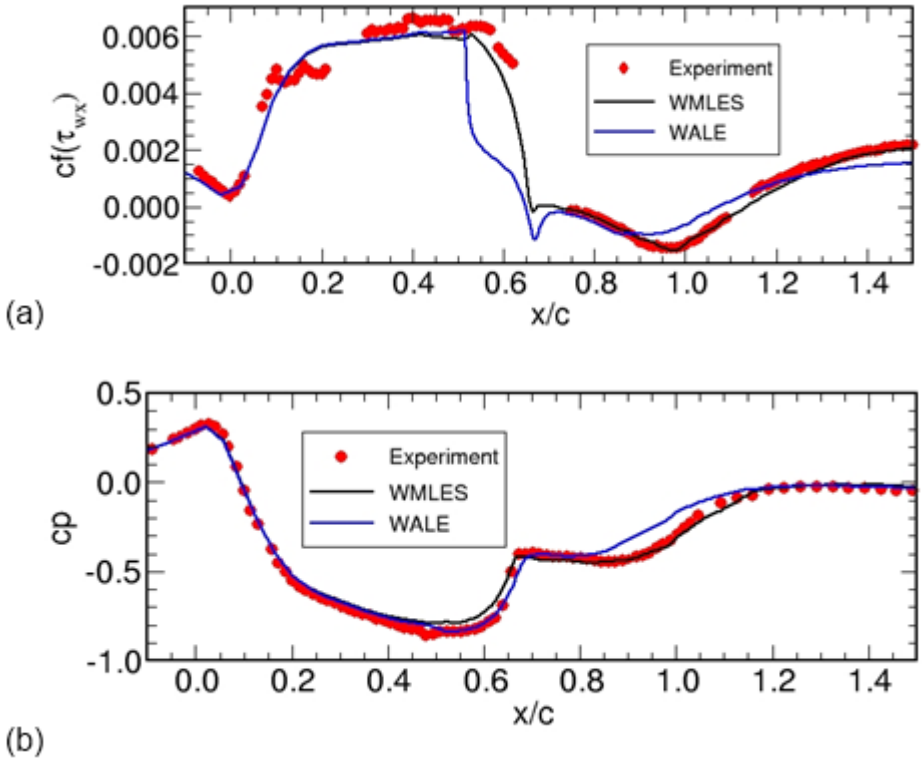


Figure 12.50 (p. 243) shows the skin-friction and wall-pressure coefficient distributions from the simulations. It can be seen that the use of ELES combined with the WMLES model in the LES zone results in very close agreement with the data, even though the skin-friction is known to be very sensitive to simulation details. A comparison of the results obtained using WMLES with those obtained using the standard WALE model in the LES zone is shown in Figure 12.50 (p. 243). The results suggest that the latter performs considerably worse than the former. In particular, in the simulations using the WALE model, the wall shear stress drops immediately after the RANS-LES interface to unrealistically small values due to the lack of resolution. The results with this model further downstream are therefore no longer reliable as the wall shear stress has a strong influence on the overall boundary layer development. Further investigations of this flow are on-going, so the results should not be considered final, but are provided to demonstrate the basic concepts.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.50: (a) Skin-friction, c_f, and (b) Wall pressure coefficients, c_p, from NASA hump flow simulations. Comparison of WMLES and WALE LES methods in the LES domain



12.1.2.3.5.4. T-Junction with Thermal Mixing

The following example is a flow through a pipe T-junction with two streams at different temperatures Westin et al., 2006 37 (p. 274). This test case was used as a benchmark of the OECD to evaluate CFD capabilities for reactor safety applications. This flow is not easily categorized in the current framework. It can be placed somewhere between a globally and a locally unstable flow. As shown below, this flow can be modeled with SAS and DDES, but special care must be taken in choosing the numerical settings. (Note that this flow has been computed with SDES and SBES models and the solutions are similar to the SAS and DDES model results.)

The set-up consists of a horizontal pipe for the cold water flow, and a vertically oriented pipe for the hot water flow. The hot water pipe is attached to the upper side of the horizontal cold water pipe. In the experiments, the length of the straight pipes upstream of the T-junction is more than 80 diameters for the cold water inlet, and approximately 20 diameters for the hot water inlet. The flow conditions are listed in Table 12.6 (p. 243).

Table 12.6: Flow conditions for T-Junction test case

	Diameter	Bulk velocity	Mass flow	Temperature	Re number
Hot Pipe	$100\ [mm]$	$1.53\ [m{\cdot}s^{-1}]$	$12\ [s^{-1}]$	$30^{\circ}C$	$\mathrm{Re}=1.9\times10^{5}$
Cold Pipe	$140\ [mm]$	$1.56\ [m{\cdot}s^{-1}]$	$24\ [s^{-1}]$	$15^{\circ}C$	$\mathrm{Re}=1.9\times10^{5}$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

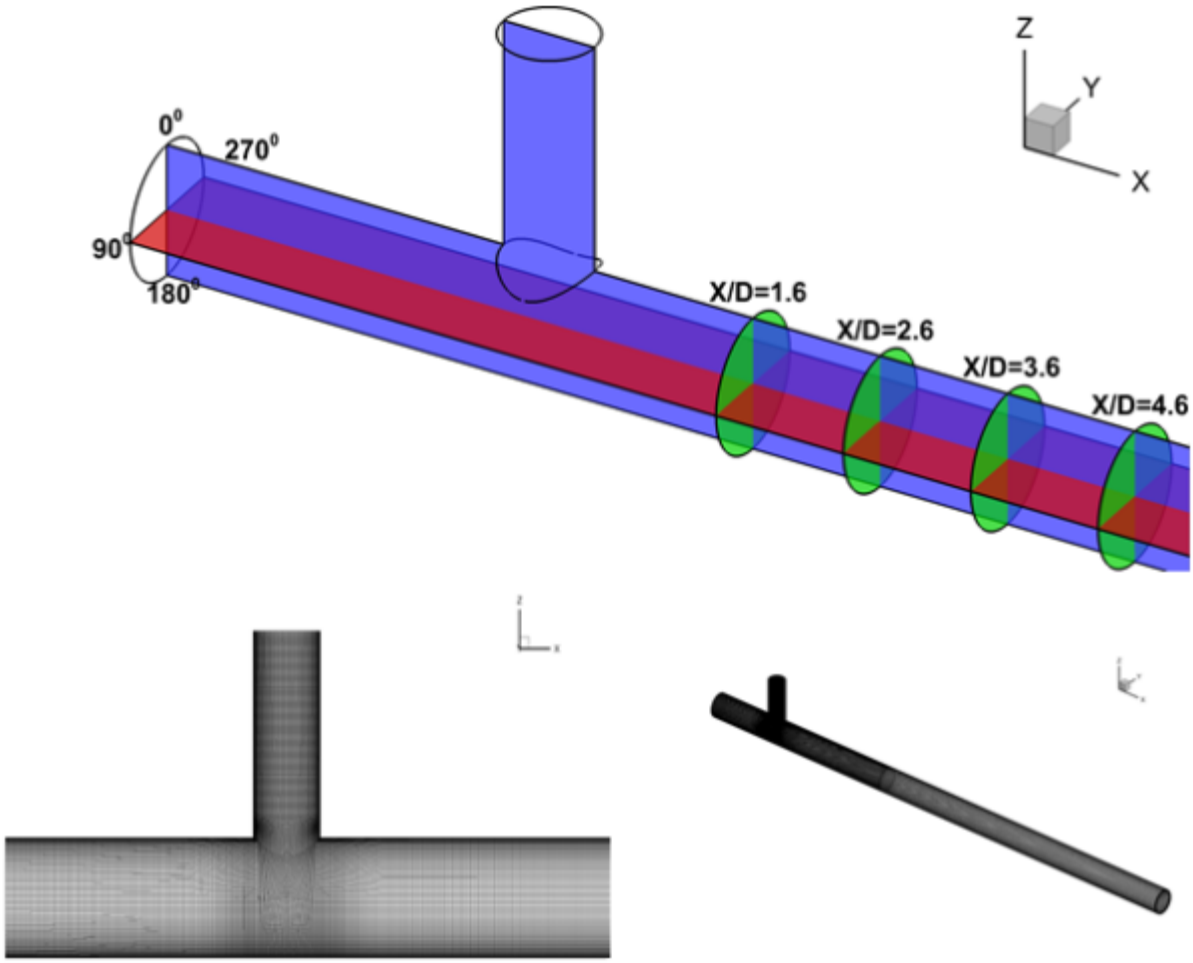
A sketch of the domain is depicted in Figure 12.51 (p. 245). The domain dimensions are as follows. The hot leg inlet is located at the z/D=22 section, the cold leg inlet is located at the x/D=−27 section, and the outlet is located at x/D=142, with D being the diameter of the cold leg of the pipe. When ELES was used, two additional interfaces have been introduced in the domain, where the synthetic fluctuations generated with the use of the Vortex Method have been specified. These sections have been placed at z/D=0.7 in the hot leg and at x/D=−1.0 in the cold leg.

The computational grid for this flow is composed of about 4.9 million hexahedral cells (see Figure 12.51 (p. 245)). The wall normal grid spacing was set to 0.0001 [m], which yields Δy^{+}=0.2−9.0 in the entire domain. The grid spacing in the axial and circumferential directions was set as follows:

- For the cold water pipe where the inlet boundary layer thickness δ_{cold} is equal to 0.07 [m], the grid spacing was chosen Δ_{axial}=0.0035$[m]$ and $\Delta_{circumferential}$=0.0021$[m]$, which yields $\delta_{hot}/\Delta_{axial}\approx 20$ and $\delta_{hot}/\Delta_{circumferrential}\approx 33$.
- For the hot water pipe the inlet boundary layer thickness δ_{hot} was set to 0.022 [m] and the grid spacing was chosen Δ_{axial}=0.0035$[m]$ and $\Delta_{circumferrential}$=0.0014$[m]$, which yields $\delta_{hot}/\Delta_{axial}\approx 6$ and $\delta_{hot}/\Delta_{circumferrential}\approx 15$.
- In wall units, the grid spacing is (Δ_{axial}^{+}, $\Delta_{circumferrential}^{+}$) ≈(195, 80) for the hot water pipe and (Δ_{axial}^{+}, $\Delta_{circumferrential}^{+}$)≈ (115, 70) for the cold water pipe, which means that the flow requires near-wall turbulence modeling. The time step was set to 0.001 [s], which leads to CFL~1 in the central mixing zone.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.51: Geometry and grid of T-Junction test case with measurement planes



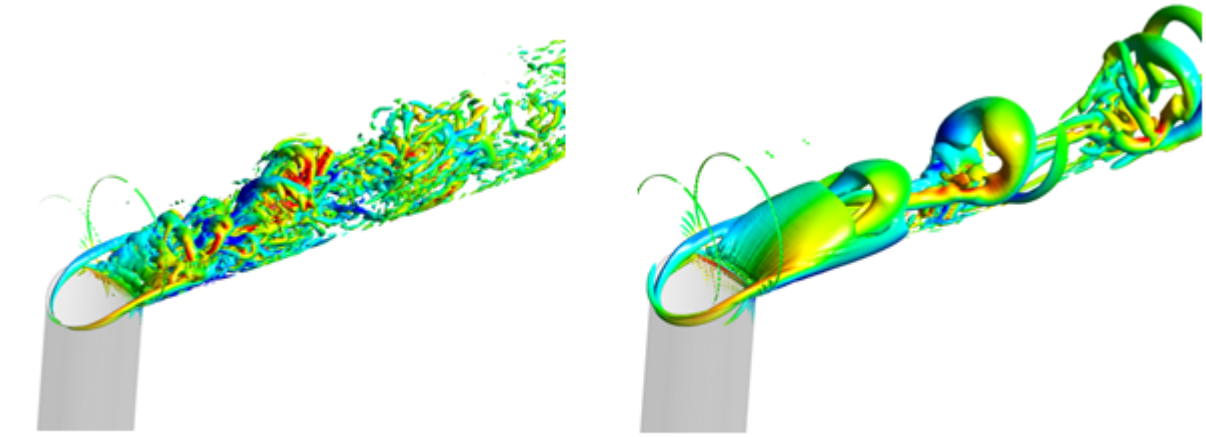
The boundary conditions for this case have been specified as follows. For the inlet boundaries, the precursor simulations of the pipe flow have been performed using the SST model. For the cold water pipe, a fully developed pipe flow was calculated using the SST model and the profiles of velocity and turbulence quantities have been specified at the inlet boundary. For the hot leg and the pipe, the profiles in the experiments were not fully developed. For this reason, a separate pipe flow simulation was conducted using constant inlet values for velocity and turbulence. The inlet profiles for the hot leg have then been extracted from this precursor simulation at the location where they matched the experimental profiles most closely.

It bears repeating that this flow is not easily categorized into one of the three groups described above, but might be described as between globally unstable and locally unstable. It was originally computed with the global SAS and DDES models. Although both simulations turn into a proper SRS mode in the interaction zone of the two streams, the results turned out to be very sensitive to numerical details and solver settings, especially for the SAS model. As an illustration, in Figure 12.52 (p. 246), the turbulence structures are shown as predicted by the SAS-SST model with the use of the CD and BCD numerical schemes. The effect of the scheme on the resolved flow is striking. This effect is an indication that the underlying flow instability is not very strong and can only be represented by the SAS model with the use of a low dissipative numerical scheme such as CD in this particular case. Under such conditions, it is not advisable to apply global methods like SAS (and to a lesser extent, DDES), as will be seen from the temperature

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

distributions later. It is important to emphasize that in more unstable flows, the difference between CD and BCD is not nearly as strong and often barely noticeable.

Figure 12.52: Turbulence structures for SAS-SST model (a) Central Difference (CD) scheme, (b) Bounded Central Difference (BCD) scheme



(a) (b)

It is therefore recommended to apply the ELES model with synthetic turbulence specified at predefined RANS-LES interfaces located in both pipes upstream of the interaction zone. Switch from the RANS to LES at these interfaces using the vortex method. In this case, the SST model was employed in the RANS zone and the WMLES approach was used in the LES part of the domain. As seen in Figure 12.53 (p. 247), with this approach resolved turbulence is generated well-upstream of the interaction zone and is then maintained through the interaction zone independent of the numerical scheme (CD or BCD).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.53: Vorticity contours for ELES/WMLES simulation (a) CD scheme, (b) BCD scheme

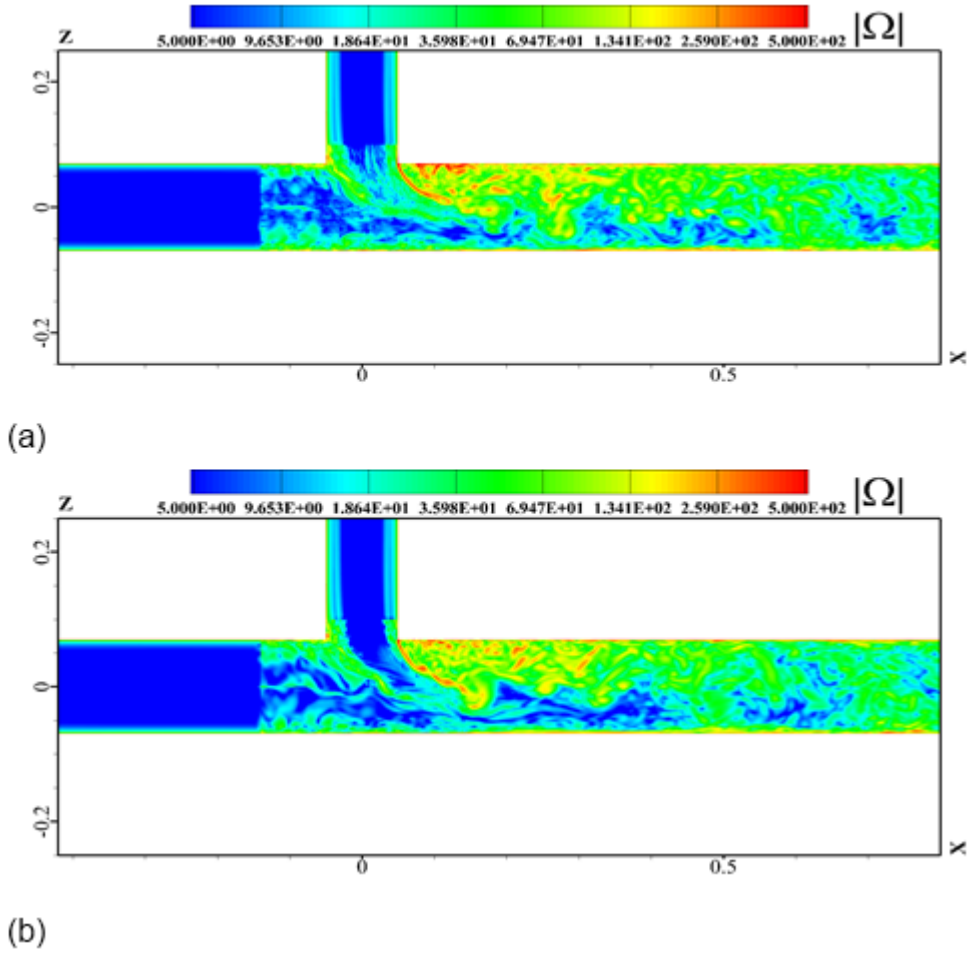
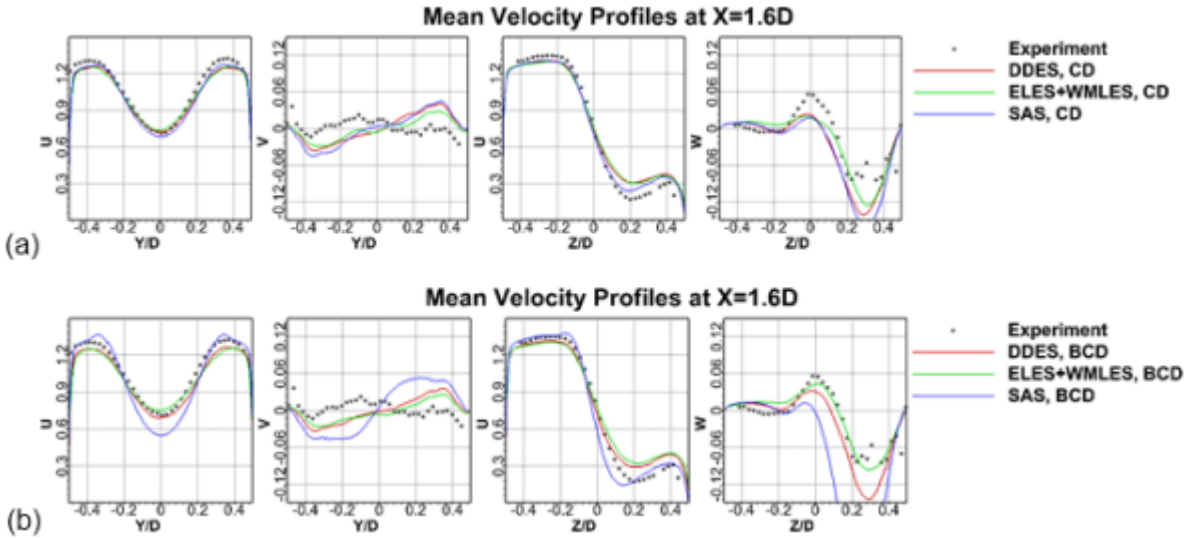


Figure 12.54 (p. 248) shows velocity profiles of different velocity components at different measurement locations (see Figure 12.51 (p. 245)). Figure 12.54 (a) (p. 248) shows results for the DDES, ELES/WMLES, and SAS simulations using the CD scheme. All simulations agree well with each other and with the experimental data. Figure 12.54 (b) (p. 248) shows the same models, but computed using the BCD scheme. As discussed, the SAS/BCD model shows marked differences compared to the experimental data, as already expected from Figure 12.52 (p. 246). It stays in URANS mode, which for this case turns out to be inadequate. The other models are less sensitive to the numerical set-up and provide almost identical results when using the BCD and the CD scheme.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.54: Comparison of the experimental and computational velocity profiles for T-Junction flow for different turbulence models (a) CD scheme (b) BCD scheme (note that scales of coordinate axes change by large factors between curves)



From an application-oriented standpoint, the most important outcome of these simulations is the thermal mixing and the resulting wall temperature distributions. Results for the different simulations are shown in Figure 12.55 (p. 249) to Figure 12.58 (p. 250). The comparison is depicted for four lines located on the wall of the main pipe downstream of the intersection at the Top (0°), Front (90°), Bottom (180°), and Rear (270°) (see Figure 12.51 (p. 245)). One can find significant differences between the global and the ELES formulations, especially on the top wall. The temperature mixing is more accurately predicted with the ELES model because the transitional process between RANS and LES is not well-defined in global models. While the solution of global hybrid models is much better than URANS (not shown here), the details can still be missed in the initial mixing zone. The ELES method is more consistent, as it provides a clear interface where modeled and resolved turbulence are exchanged (RANS-LES interface with synthetic turbulence). Because of that, well-defined resolved turbulence is already present upstream of the junction, thereby avoiding the ambiguities of the formation of resolved turbulence in the interaction zone.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.55: Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Top wall (0 degrees, see Figure 12.51 (p. 245)) of the main pipe

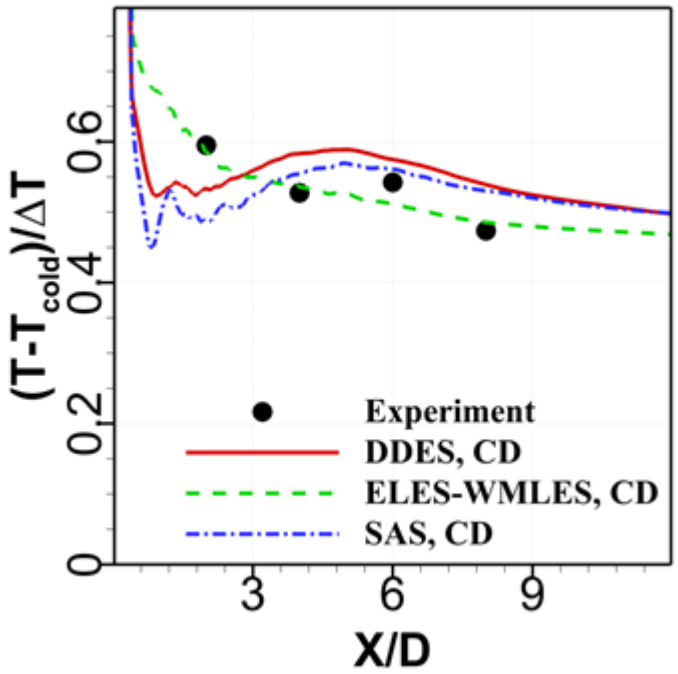
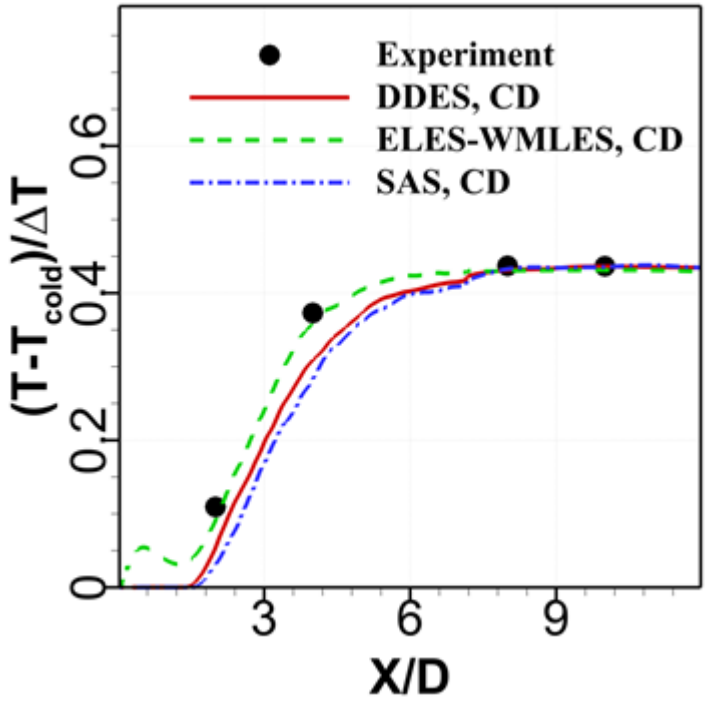


Figure 12.56: Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Front wall (90 degrees, see Figure 12.51 (p. 245)) of the main pipe



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.57: Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Bottom wall (180 degrees, see Figure 12.51 (p. 245)) of the main pipe

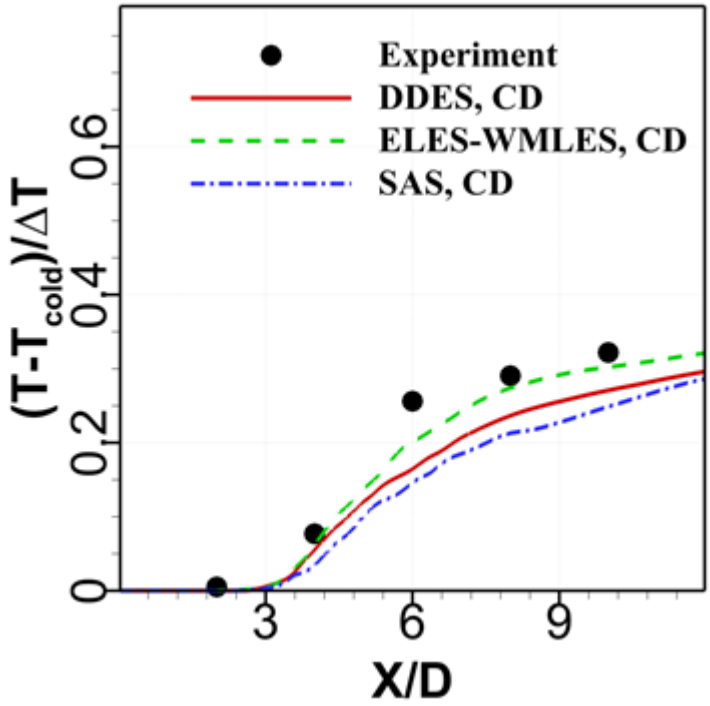
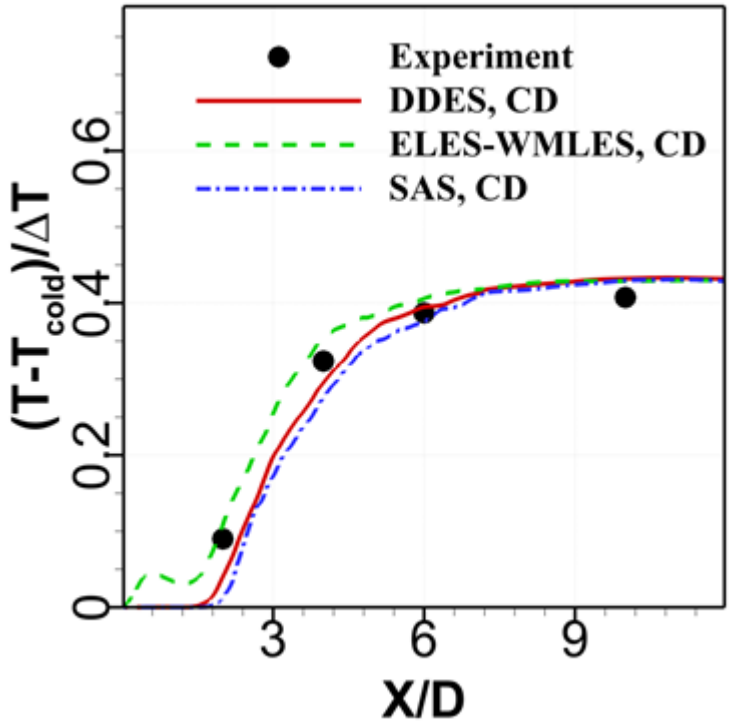


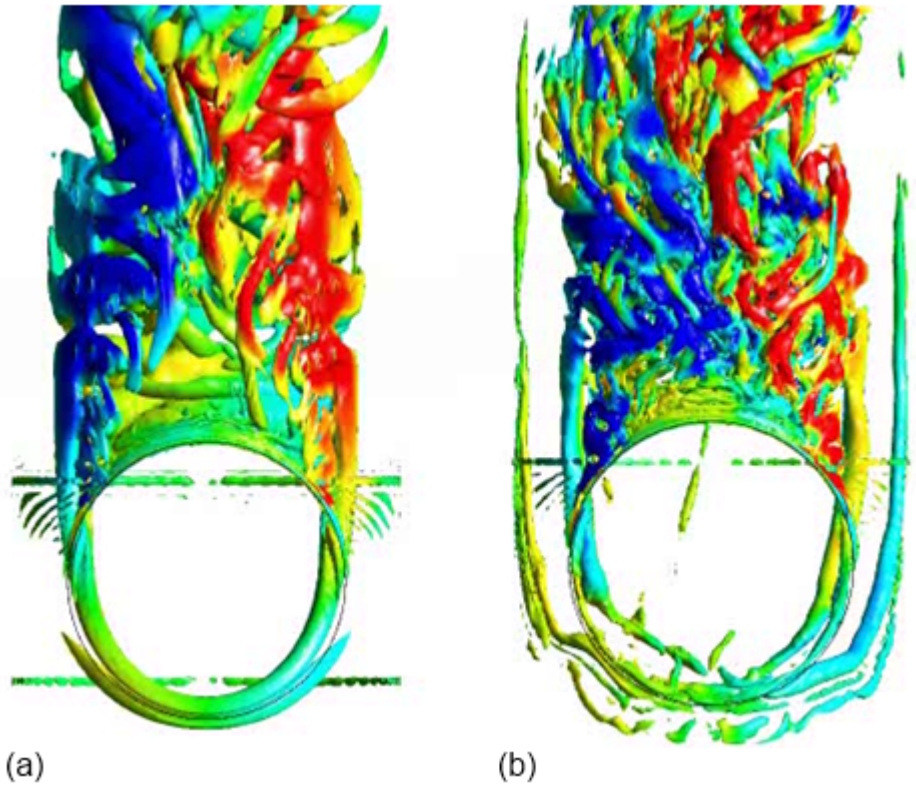
Figure 12.58: Comparison of the experimental and computational wall temperature distributions for T-Junction flow at the Rear wall (270 degrees, see Figure 12.51 (p. 245)) of the main pipe



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Details of the resolved turbulence can be seen in Figure 12.59 (p. 251), which shows the region just downstream of the pipe intersection on the Top wall (0° - Figure 12.51 (p. 245)) where the temperature predictions between ELES and DDES differ the most (Figure 12.55 (p. 249)). ELES shows significantly stronger resolved turbulence activity than DDES, confirming the arguments above. More recent simulations of this flow using the SBES model show similar solutions to DDES, but with smaller resolved turbulence structures.

Figure 12.59: Comparison of turbulence structures on the Top wall downstream of the pipe intersection (a) DDES model (b) ELES model



12.1.3. Numerical Settings for SRS

This chapter discusses:

12.1.3.1. Spatial Discretization
12.1.3.2. Time Discretization

12.1.3.1. Spatial Discretization

12.1.3.1.1. Momentum

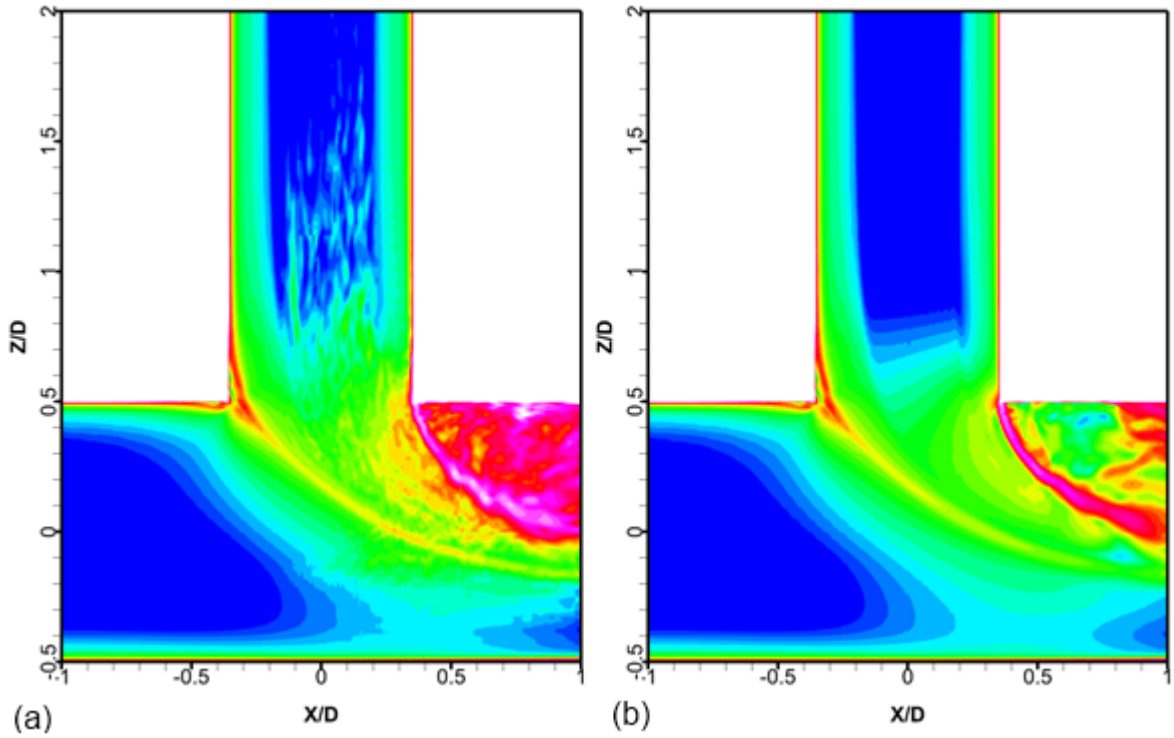
SRS models, as described in Scale-Resolving Simulation (SRS) Models – Basic Formulations (p. 183), serve the main purpose of dissipating the energy out of the turbulence spectrum at the limit of the grid resolution. The eddy viscosity is defined to provide the correct dissipation at the larger LES scales. This assumes that the numerical scheme is non-dissipative and that all dissipation results from the LES model. For this reason, one is required to select a numerical scheme in the LES region with low dissipation, relative to the dissipation provided by a subgrid LES model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Another strategy is to avoid the introduction of the LES (subgrid) eddy viscosity and provide all damping through the numerical scheme. This approach is called MILES (Monotone Integrated Large Eddy Simulation) (Boris et al., 1992 1 (p. 269)). In Ansys CFD, the standard LES methodology is followed, whereby the dissipation is introduced by a LES eddy viscosity model and the numerical dissipation is kept at a low value.

In order to achieve low numerical dissipation, you cannot use the standard numerical schemes for convection that were developed for the RANS equations (Second Order Upwind Schemes, or SOU), which are dissipative by nature. In contrast, LES is carried out using Central Difference (CD) schemes. In industrial simulations, second order schemes are typically employed, however, in complex geometries with non-ideal grids, CD methods are frequently unstable and produce unphysical wiggles (see Figure 12.60 (p. 252)), which can eventually destroy the solution. To overcome this problem, variations of CD schemes have been developed with more dissipative character, but still much less dissipative than Upwind Schemes. An example is the Bounded Central Difference (BCD) scheme of Jasak et al., 1999 14 (p. 271).

Figure 12.60: Example of scheme oscillations in T-Junction flow shown by vorticity: (a) CD, (b) BCD



The CD scheme can be used successfully for (WM)LES of simple flows on optimal grids (typically hexahedral grids with low skew) such as channel or pipe flows. For more complex geometries, ELES allows the reduction of the LES domain to a limited region with high quality grids. Under such conditions, CD can be employed inside the LES portion of the grid, while using a standard upwind biased scheme for the RANS part of the domain.

For global models, like SAS or DDES/SDES/SBES, involving RANS and LES portions without a well-defined interface between them, most cases require the use of the BCD scheme, which can also handle both the RANS and LES domains with acceptable accuracy.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

When using ELES in Ansys Fluent, one can also switch the numerical scheme between the RANS and the LES regions (see Cokljat et al., 2009 2 (p. 269)) by hand.

In Ansys CFX, the default for the SAS and SBES models is a numerical scheme that switches explicitly between a second order upwind and the CD scheme, based on the state of the flow, using a switch proposed by Strelets, 2001 33 (p. 273). This switching scheme is relatively complex. It is advisable to apply the less complex BCD scheme that is also available in the software. In Ansys CFX there is an additional parameter for the BCD scheme that allows a continuous variation of the scheme from BCD to CD. The parameter is called CDS Bound. CDS Bound=1 applies only to BCD and CDS Bound=0 applies only to CD.

12.1.3.1.2. Turbulence Equations

The spatial discretization of the convection terms of the turbulence model is second order by default. The use of second order upwind schemes can be beneficial for accurate RANS simulations. In case of numerical problems, it might be reasonable to try switching to the first order upwind scheme.

Note:

For SBES, numerical problems have been observed on complex grids with second order turbulence numerics. In such scenarios, switching to first order numerics is recommended. Note that this has little effect on the accuracy, because the RANS region is typically insensitive to this switch. In the LES region, the two-equation model is, in any case, overwritten by the selected LES model.

12.1.3.1.3. Gradients (Ansys Fluent)

The selection of a specific gradient method is not of much relevance to SRSs on high quality hexahedral meshes. For skewed or polyhedral meshes, the Least Square Method (LSM) is recommended. For the SAS model one should use the LSM or the Green-Gauss Node Based (GGNB). The latter allows a slightly higher sensitivity to initial instabilities.

12.1.3.1.4. Pressure (Ansys Fluent)

SRS can be relatively sensitive to the pressure interpolation. Validation studies have shown that the PRESTO scheme is more dissipative than the other options and should be avoided unless required for other reasons. For the validation studies, the standard pressure interpolation was typically used.

12.1.3.2. Time Discretization

12.1.3.2.1. Time Integration

Time integration should be carried out with the second order backward Euler scheme. This has proven to have sufficient accuracy for a wide range of applications. For turbulence (and other positive) variables, use the Bounded Second Order Implicit Euler scheme (this must be selected in Ansys Fluent and is the default in Ansys CFX).

The time steps should be selected to achieve a Courant number of $CFL \approx 1$ in the LES part of the domain. For complex geometries and grids with high stretching factors, the definition of the CFL

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

number is not always very reliable (for example, if the flow passes through a region of highly stretched cells). In such situations, estimates can be built upon the physical dimensions of the shear layer to be resolved. If N cubic cells are required for resolving a shear layer (say $\mathrm{N}=15\text{-}20$ across a mixing layer of thickness δ) and a certain CFL number is to be achieved, then a time step of

$$\Delta t=\frac{CFL\cdot\delta}{N\cdot U} \tag{12.33}$$

is required. Considering that δ is proportional to the RANS turbulent length scale L_t (with a constant of order 1), this estimate may be further simplified to:

$$\begin{aligned}\Delta t&=\frac{CFL\cdot L_t}{N\cdot U}\\ L_t&=\frac{k^{3/2}}{\varepsilon}=\frac{k^{1/2}}{C_\mu\omega}\end{aligned} \tag{12.34}$$

where $C_\mu=0.09$. This simplified analysis means that the time step Δt can be estimated on a precursor RANS simulation.

You can also apply a more global estimate by assessing the through flow time, which is the time required by a fluid element to pass through the LES domain of length L with velocity U: $T_{tf}=L/U$. With an estimate of how many cells, N, will be passed along this trajectory, one obtains $\Delta t=T_{tf}/N\cdot CFL$.

12.1.3.2.2. Time Advancement and Under-Relaxation (Ansys Fluent)

There are several different settings for time advancement in Ansys Fluent. The first choice is between the Iterative (ITA) and the Non-Iterative Time Advancement (NITA). NITA should be checked for any new application as it can result in significant CPU savings. As a general guideline, NITA works well on high quality grids and for flows with limited additional physical coupling between the equations. Within NITA, the fractional step scheme is recommended; however, one must be very cautious and conservative with the assessment of the time step size. An attempt to perform a simulation with CFL>1 can lead to an incorrect solution. In addition, one should reduce residual tolerance for all equations to 0.0001.

For the ITA schemes (everything except NITA), the segregated solvers are typically faster than the coupled solver. The optimal choice is in most cases the SIMPLEC scheme. The default under-relaxation parameters for this scheme are set for steady-state simulations. For SRS model simulations, they should be changed to values as close as possible to 1 to improve iterative convergence. Typically, the number of inner iteration loops required with SIMPLEC depends on the complexity of the flow problem. The most critical quantity is the mass conservation. Mass residuals should decrease by at least one order of magnitude every time step. With high under-relaxation and good grid quality, good solutions can often be achieved even with only two inner loops.

The coupled solver is slower per iteration, but it can lead to more robust convergence, and for complex cases can be advantageous. For the coupled solver, one would typically also specify under-relaxation values of (or close to) 1. The number of inner loops is typically $N_{inner}\sim$ 2-5. In Ansys CFX, the coupled solver is used in all simulations.

For flows with additional physics (multiphase, combustion, and so on), the number of inner iterations per time step can increase for all solvers.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

It is important to emphasize that the optimal under-relaxation factors and the optimal number of inner iterations is case-dependent. Some optimization might be required for achieving the most efficient results.

12.1.4. Initial and Boundary Conditions

This chapter discusses:

12.1.4.1. Initialization of SRS

12.1.4.2. Boundary Conditions for SRS

12.1.4.3. Symmetry vs. Periodicity

12.1.4.1. Initialization of SRS

In most cases it is best to initialize the SRS model using a RANS model solution. This recommendation is especially true for global hybrid RANS-LES models (SAS, DDES/SDES/SBES), which are based on an underlying RANS model.

For pure LES or WMLES, Ansys Fluent offers an option for initializing the flow by converting turbulence from RANS to LES mode (`solve/initialize/init-instantaneous-vel`) using a synthetic turbulence generation routine. This option should be used with caution as it can, at times, have a detrimental effect on the robustness of the simulation. It should be executed mainly for cases where no synthetic turbulence is generated at an inlet/interface and where the inherent flow instability is not strong enough to generate resolved turbulence on its own. A typical example would be the LES of a channel flow with periodic boundary conditions in the streamwise direction. For such flows, the solver could return a laminar solution even at super-critical (turbulent) Reynolds numbers if no initial disturbance is provided.

In Ansys CFX, synthetic turbulence is generated automatically in the first time step inside the LES region of a ZLES set-up.

12.1.4.2. Boundary Conditions for SRS

12.1.4.2.1. Inlet Conditions

Inlet conditions should be selected based on the physics of the flow and applied in a similar manner as RANS computations.

For global models (SAS, DDES/SDES/SBES), use standard (typically steady-state) RANS inlet conditions.

For LES or WMLES, provide synthetic turbulence at the inlet.

12.1.4.2.2. Outlet Conditions

If possible, outflow or average pressure is better than constant pressure outlets as vortices carry non-constant pressure distributions across the boundary. For certain acoustics calculations, like jet noise, use non-reflecting boundary conditions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.1.4.2.3. Wall Conditions

For all models except LES, use low y^+ values of around $y^+=1$. The models are formulated in a y^+-insensitive fashion, so larger values of y^+ can be tolerated as long as the overall boundary layer resolution is sufficient.

For LES, one would typically have to apply wall functions in order to avoid the large resolution requirements near the wall. The wall resolution in streamwise (x), normal (y) and spanwise (z) directions are coupled.

$$\begin{aligned} \Delta x^+<40, \quad \Delta z^+<20 \quad \text{for} \quad \Delta y^+<10 \\ \Delta x^+<4\Delta y^+, \quad \Delta z^+<2\Delta y^+ \quad \text{for} \quad \Delta y^+\geq 10 \end{aligned} \tag{12.35}$$

12.1.4.3. Symmetry vs. Periodicity

In most cases, periodicity or slip conditions cannot be employed in regions that border on zones of resolved turbulence, even if the geometry and the time-averaged flow are symmetric with respect to a given plane. The reason is that unsteady turbulence does not obey symmetry instantaneously. The application of symmetry boundaries would therefore impose an unphysical constraint onto the resolved scales. It is therefore essential to either compute the full domain, or to apply periodicity at such planes if possible (for example, if there is a matching plane at the other end of the domain).

Symmetry and slip wall conditions can be used if the resolved turbulence is confined to regions not touching these boundaries.

Periodicity conditions can lead to problems for axi-symmetric situations. As the radius approaches zero, the circumferential size of the domain goes to zero, and periodicity conditions would not enable turbulence structures of finite size to exist. An example is the flow in an axi-symmetric pipe. If you were to compute that flow in a pipe segment with periodicity conditions in the circumferential direction, you would restrict the size of the resolved eddies to zero near the axis. This effect is not correct and would substantially alter the solution. Such a simulation would therefore have to be carried out in full 360° mode. Note that the situation would be different in the case of the flow through a ring segment, where the axis is excluded from the SRS domain. Periodicity could be applied in the case of $\left(R_2-R_1\right)/R_1<C$ with R_2 being the outer radius, R_1 the inner radius of the segment and C being a constant of the order 1 or larger.

12.1.5. Postprocessing and Averaging

This chapter discusses:

12.1.5.1. Visual Inspection

12.1.5.2. Averaging

12.1.5.1. Visual Inspection

The first and most important step in any SRS model is the visual inspection of the turbulence structures. This is typically done using an isosurface of the Q-criterion. The definition of Q is:

$$\begin{aligned} Q_{Dim}&=C_Q\left(\Omega^2-S^2\right) \\ Dim&=\left[s^{-2}\right] \end{aligned} \tag{12.36}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where in different definitions the constant might be different (for historic reasons, C_Q=0.5 in Ansys Fluent and C_Q=0.25 in Ansys CFD-Post). The value of the constant C_Q is typically unimportant as we are only interested in visual impressions when using this quantity. In this definition, S is the absolute value of the Strain Rate and Ω is the absolute value of vorticity.

$$\begin{aligned} S&=\sqrt{2S_{ij}S_{ij}} \\ \Omega&=\sqrt{2\omega_{ij}\omega_{ij}} \\ S_{ij}&=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right) \\ \omega_{ij}&=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}-\frac{\partial U_j}{\partial x_i}\right) \end{aligned} \tag{12.37}$$

The rationale behind this definition is that we want to visualize vorticity, which characterizes turbulence vortices, but also to subtract the mean shear rate in order to avoid displaying steady shear layers (where $S=\Omega=1/2\left|du/dy\right|$).

There are different definitions of Q, some of them non-dimensional. Avoid using non-dimensional Q values as they can be mainly used for visualization of free vortices and their dynamics (for example, tip vortex of an airplane wing). In turbulent flows, they can elevate very weak turbulence structures to the same level as the strong ones and thereby produce an incorrect picture.

In Ansys Fluent, the variable Q is called "Q criterion" (under Turbulence) and in Ansys CFD-Post "Velocity.Invariant Q" in the variable list. Both codes also have a non-dimensional version of Q (Ansys Fluent: "Normalized q criterion ," Ansys CFD-Post: "Location / Vortex Core Region, Method = Q-Criterion"), which are not very descriptive for turbulence vortex fields.

The dimensional Q-values can be very large and can vary greatly in the domain. Frequently, values up to $Q_{mas}\sim10^8$ can be found in high Re number flows. In such cases, isosurfaces in the range of $Q_{mas}\sim10^4-10^5$ are typically sensible. You must experiment with some values for the isosurface before obtaining a suitable picture. It might be helpful to first plot Q on a fixed surface as a contour plot and select the correct scaling from that contour plot. Use positive values for the isosurface. Do not use Q=0 for visualization, as it will show very weak structures not relevant to turbulence visualizations.

It is also advisable to color the isosurface of Q with some other variable. Interesting quantities are the eddy-viscosity ratio (μ_t/μ), or a velocity component that is small or zero in RANS (such as the spanwise velocity), or the CFL number, and so on. The visual inspection should be done continuously during the entire start-up and run time of the simulations (once per day or after every 1000 time steps). It serves the following purposes (see for example Figure 12.16 (p. 209) and Figure 12.17 (p. 209)):

- Check if unsteady turbulence develops at all and at the expected locations.
- Check large scale symmetries/asymmetries of the flow.
- Check the solution for numerical wiggles (odd-even decoupling)
- Check the size of the resolved eddies and see if they are as one would expect from the grid resolution.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Check the CFL number on these eddies. It should be smaller than CFL ≈ 1. Check the eddy-viscosity ratio. It should be much smaller than RANS.
- Check for global SRS turbulence models (SAS/DDES/SDES/SBES) if the turbulence structures develop early in the separating shear layer or if a noticeable delay is observed (see Figure 12.26: Isosurfaces of the Q-criterion colored with the velocity magnitude (p. 220)).
- Check for ELES/Unsteady inlet conditions, if synthetic turbulence is reasonable and does not decay (such as in Figure 12.45 (p. 239)).
- Check the progress of the simulation towards a statistically converged solution. This means that the resolved turbulence requires some time until it has developed and has been transported through the domain. Time-averaging has to wait until that stage has been achieved.
- Include pictures of turbulence structures in any reports of the test case (slides, reports, publications, service requests).
- If possible make animations, which help to understanding of the flow physics and is also helpful for others to understand the flow.
- Add monitoring points at interesting locations and plot their development in time to demonstrate statistical convergence.

12.1.5.2. Averaging

Unsteady simulations with scale resolution require special care in postprocessing and averaging. Engineers are usually interested only in time-averaged results and not in the details of the unsteady flowfield. It is therefore important to follow a systematic approach when computing such quantities.

The typical process is to start from a RANS solution (or reasonable initial condition). When switching to any SRS model, the flow will require some time to statistically settle into a new state for the following reasons:

- The resolved turbulence requires some time to develop and be transported through the domain.
- The global flow topology might change from the initial (RANS) solution.
- Other physical effects might require longer start-up times (such as multi-phase).

The general strategy is therefore to run the simulation for some start-up time ΔT_s, before activating the averaging process (or initiating the acquisition of, for example, acoustics information). When should this process be started and how long does it take until the flow is statistically steady? This is the stage where any increase in ΔT_s would not change the averaged solutions. Unfortunately ΔT_s depends strongly on the flowfield and no general guidelines can be given. For some flows, the flow develops quickly (in a few thousand time steps). For others it takes tens of thousands of steps to reach that point. However, a first estimate can be obtained by estimating throughflow time, T_{TF}. This is the time that the mean flow requires to pass one time through the domain $T_{TF}=L_{CFD}/U_\infty$ where L_{CFD} is the length of the domain and U_∞ is the mean flow velocity. The turbulence statistics typically require several (3-5) throughflow times to establish themselves. Again, this is just a rough estimate and can depend on the particular flow.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In order to determine ΔT_s more systematically, one must monitor the simulation. It is advisable to monitor some local and some global quantities.

- Continuously inspect the solution visually with the aid of regular images and updated animations.
- Inspect solution variables at monitor points in the critical zone of simulation (pressure, velocity, temperature, and so on) as a function of time. The amplitude and frequency of local oscillations should become regular before the averaged statistics can be gathered.
- Monitor global quantities (forces on body, massflow, integrated swirl, and so on). Interesting quantities are often those that would be zero for RANS (spanwise forces, and so on) as they are sensitive to the SRS characteristics. They also help to evaluate the overall symmetry of the solution (they should fluctuate around zero) and to determine slow transients (quantities that fluctuate around zero but with low overlaid frequencies).

Only when all indicators show that the flow is no longer changing statistically (meaning only the details of the turbulence structures are a function of time) should the averaging be activated. It is important to document the number of steps that have already occurred when averaging was started and how many steps have been averaged. With respect to averaged quantities:

- Monitor time-averaged quantities and ensure that they are not "drifting." They will drift initially, but should then settle to an asymptotic value.
- Ensure that they satisfy the symmetry conditions of the flow. Any asymmetry is an indicator of non-convergence (exceptionally, there are flows that develop physical asymmetries despite a symmetric set-up. Example: some symmetric diffusers separate from one side and stay attached on the other).
- Ensure that the averaged quantities are smooth.
- In zonal/embedded simulations, check if averaged quantities are reasonably smooth across RANS-LES interfaces (they will never be perfectly smooth, but should also not change drastically).

12.1.6. Summary

An overview of hybrid Scale-Resolving Simulation (SRS) technologies was given. Due to the nature of the subject, only a rough outline of the models could be provided. The rational and the advantages/disadvantages of each model family have been discussed. Based on the description of the models, an attempt has been made to categorize flows into sub-classes, and to map the modeling strategies onto these classes. It should be emphasized again, that the proposed categories are not easily and clearly defined and have significant overlap. Still it is considered necessary to explain that no single SRS model is suitable for all applications and it is not possible to generalize about which model should be used for which type of flow.

In principle, ELES and ZFLES, in combination with WMLES, are suitable for all flows, but require a substantial amount of preprocessing work to define the corresponding zones and provide suitable grids for all of them. For complex applications, this is not always feasible/practical, and global models (SAS, DDES, SDES, SBES) are favored. Studies have shown a clear superiority of the SBES model over all the other global hybrid RANS-LES formulations. The SBES model is therefore recommended for such applications. However, as detailed, all of these models work only if a sufficient level of instability is present in the flow. If in doubt, it is better to select the safer option over the more convenient one.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Details on many aspects of SRS have been provided, ranging from numerics, to grid resolution all the way to postprocessing. Numerous examples have been shown to enable the reader to properly place the intended application into this framework. It is anticipated that the document will evolve over time, as new questions are posed by users and as the SRS models themselves will evolve.

A brief summary of the more important points is provided in the Appendices.

The following topics are discussed:

12.1.6.1. Acknowledgment

12.1.6.2. Appendix 1: Summary of Numerics Settings with Ansys Fluent

12.1.6.3. Appendix 2: Summary of Numerics Settings With Ansys CFX

12.1.6.4. Appendix 3: Models

12.1.6.5. Appendix 4: Generic Flow Types and Modeling

12.1.6.1. Acknowledgment

The material in this document was prepared with the help of members of the turbulence team at Ansys Jochen Schütze, Yuri Egorov, Richard Lechner, as well as with the help of the colleagues at NTS in St. Petersburg Mikhail Gritskevich and Andrey Garbaruk. Aleksey Gerasimov at Ansys has provided a very thorough review, which has resulted in the removal of numerous inconsistencies and has significantly improved the quality and readability of the document. Some of the test cases have been run in the framework of the EU projects DESIDER and ATAAC.

12.1.6.2. Appendix 1: Summary of Numerics Settings with Ansys Fluent

	Unsteady Simulation	**Comment**
Convection Terms	CD / BCD	CD on simple geometries (also inside LES regions). In case of wiggles in solution use BCD (most industrial cases)
Pressure Discretization	Any except PRESTO	Use PRESTO only if required for other reasons. Note that the initial formation of turbulence structures can be delayed (inhibited) with PRESTO.
Velocity Gradients	Least Squares Cell Based	No significant impact on SRS, typically Least Square Cell Based. For the SAS model one should use the Least Square Cell Based, or the Green-Gauss Node Based (GGNB). The latter allows a slightly higher sensitivity to initial instabilities.
Iterative Method	SIMPLEC	NITA/Fractional step only for simple flows Monitor convergence: at least 1 order in mass conservation. SIMPLEC with 2-5 inner loops. For cases that are difficult to converge try the coupled solver. More expensive,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Unsteady Simulation	Comment
		but potentially lower inner iterations required. Increase Under-Relaxation Factors to values ~1
Under-relaxation	URF ≈ 1	Start with all URF ≈ 1 (typically 0.8–0.95). Reduce in case of convergence problems. Lower values for additional physics (combustion, multi-phase, and so on).
Time Discretization	Second order backward Euler	Use CFL<1 in LES zones if possible. This condition can also be relaxed depending on the flow and CFL~5 was used for the T-junction test case successfully. Bounded for second order turbulence quantities (k, ω, ε) and other positive quantities (volume fraction, and so on).

12.1.6.3. Appendix 2: Summary of Numerics Settings With Ansys CFX

	Unsteady Simulation	Comment
Convection Terms	CD / BCD	CD on simple geometries (also inside LES regions). In case of wiggles in solution use BCD The default scheme for SBES and SAS is a hybrid scheme that switches automatically between High Res and CD. Recent experience indicates that BCD is generally easier to apply and often yields the same accuracy. **CDS Bound** enables shifting between the classical BCD scheme and the Central Difference scheme.
Time discretization	Second order backward Euler	Use CFL<1 in LES zones if possible. This condition can also be relaxed depending on the flow. Bounded for second order turbulence quantities is default (k, ω, ε) and other positive quantities (volume fraction, and so on).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.1.6.4. Appendix 3: Models

	Applications	Comments
Scale-Adaptive Simulation (SAS)	• Use for globally unstable flows • Use CFL~1 for best results (higher CFL possible but less resolution) • Avoid PRESTO scheme • Check Q-criterion carefully during run time to ensure SRS structures	• Safest SRS model, as it has URANS fallback position on coarse grids/time steps • Danger of falling into URANS mode if flow instability is not strong
Detached Eddy Simulation (DES)	• Use for globally unstable flows and with care also for locally unstable flows • Always use DDES over DES to reduce the impact of the DES limiter on attached boundary layers, and use the DDES shielding function • Grid in SRS region must be of LES quality, no RANS fallback position • Use CFL~1 • Avoid PRESTO scheme • Check Q-criterion carefully during run time to ensure SRS structures	• DDES is not available for CFX • More aggressive than SAS in terms of unsteadiness • Careful grid generation important, otherwise danger of gray zones or grid-induced separation
Shielded Detached	• Improved version of DDES. It	• SDES is not available for CFX

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Applications	Comments
Eddy Simulation (SDES)	provides better shielding of the RANS boundary layers and a more aggressive definition of the grid length scale – leading to more rapid transition from RANS to LES.	• Similar to DDES but improved shielding and reduction of gray area zone and grid-induced separation.
Stress-Blended Eddy Simulation (SBES)	• Optimal model that features asymptotic shielding of the RANS boundary layer as well as explicit switch to existing algebraic LES model. Can also be run in WMLES mode.	• This is the model of choice for globally and locally unstable flows. • Can also be used in WMLES mode, once LES content is introduce upstream. This can be achieved by synthetic turbulence or by a previous RANS-LES transition upstream (for example backstep).
Large Eddy Simulation (LES)	• Use for free shear flows • Use if boundary layers are laminar • Use for turbulent boundary layers only with high grid resolution at low Reynolds numbers • Use CFL~1 • Apply synthetic turbulence at inlets • Check Q-criterion carefully during run time to ensure SRS structures	• Typically too expensive for wall-bounded flows
Wall Modeled LES (WMLES)	• Use for wall boundary layers at moderate and high Reynolds numbers	• Scales much more favorably with Reynolds number than standard LES but still very expensive • Limit wall region to a small portion of flow domain (ELES)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

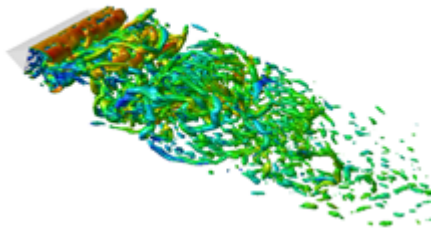
	Applications	Comments
	• Resolve boundary layer volume ($\delta \times \delta \times \delta$) by $10 \times 40 \times 20$ cells • Use CFL~1 • Apply synthetic turbulence at inlets • Check Q-criterion carefully during run time to ensure SRS structures	• Note that the SBES model (SST+WALE) can also provide WMLES capabilities.
Embedded LES (ELES) Zonal Forced LES (ZFLES)	• Use for wall boundary layers at moderate and high Reynolds numbers • Resolve boundary layer volume ($\delta \times \delta \times \delta$) by $10 \times 40 \times 20$ cells • Use CFL~1 • Apply synthetic turbulence at RANS-LES interface • Check Q-criterion carefully during run time to ensure SRS structures	• Allows flexible combination of models in different parts of the domain. • If wall boundary layers in LES domain, consider using WMLES (default in CFX)
Vortex Method (VM) – Fluent	• Use to generate synthetic turbulence at RANS-LES interface or LES (WMLES) inlet • Restrict interface zone to the minimal section where turbulence must be converted (do not extend LES	• Grid in LES region of interface must be of LES quality

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Applications	Comments
	zone far into the freestream) • If large RANS-LES interface cannot be avoided increase (and check) the number of vortices specified. Can be as high as 10^4. • Use CFL~1 • Check Q-criterion carefully during run time to ensure SRS structures	
Harmonic Turbulence Generator (HTG) - CFX	• Restrict inlet zone to the LES minimal section where turbulence must be converted (do not extend LES zone far into the freestream)	• Grid in LES region of interface must be of LES quality

12.1.6.5. Appendix 4: Generic Flow Types and Modeling

Table 12.7: Globally Unstable Flows



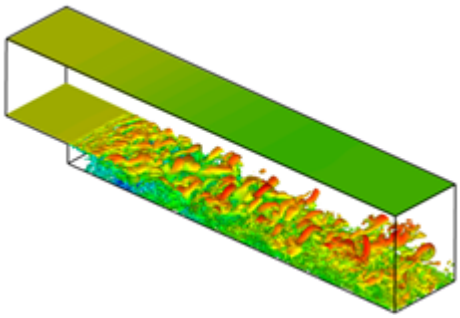
Examples	• Flows past bluff bodies – Flow past buildings – Landing gears of airplanes – Baffles in mixers

	– Side mirrors of cars – Stalled wings/sails – Trains/trucks/cars in crossflow – Tip gap of turbomachinery blades – Flows past orifices, sharp nozzles, and so on. • Flows with strong swirl instabilities – Flow in combustion chambers of gas turbines – Some tip vortex flows in adverse pressure gradients – Flows past vortex generators • Flows with strong flow interaction – Impinging/colliding jets
Modeling	• SAS model is safest option because it has RANS fall-back position. • DDES/SDES/SBES in case SAS does not show sufficient content of resolved turbulence. Provide suitable LES grid in LES region. • Often SAS and DDES give very similar solutions. • Optimal model SBES. • ELES typically not required. • The DDES model has been superseded by the SBES model family. The SBES modeling approach is recommended.
Critical	• Visually check turbulent structures

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	• Run flow until statistically converged

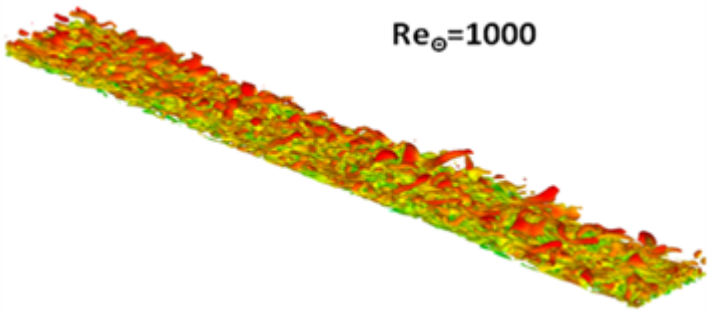
Table 12.8: Locally Unstable Flows

	
Examples	• Flows with large separation zones (< boundary layer thickness) – Backward-facing step type flows – Bump flows with large separation – Cavity flows – Mixing layer leaving plate/trailing edge • Flows with weak swirl instabilities – Flames with low or zero swirl • Flows with weak flow interaction – Jet in crossflow with low momentum ratio
Modeling	• Use ELES where geometry permits. • DDES/SDES/SBES on high quality grids and low dissipation numerics (CD/BCD) • The DDES model has been superseded by the SBES model family. The SBES modeling approach is recommended.
Critical	• Instability of Separating Shear Layer (SSL) must be resolved with DDES/SDES/SBES quickly. ELES is safer as it provides unsteady inlet to separation zone but generally much more expensive

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	• Visually check turbulent structures in SSL.

Table 12.9: Stable Flows

	
Examples	• Attached and mildly separated wall bounded flows – Boundary layers – Channel/pipe flows
Modeling	• LES in separate domain if possible – WMLES for higher Re numbers – Maybe interpolate larger domain RANS solution onto LES zone boundaries – Use unsteady (synthetic) turbulence at inlet, preferred Vortex Method • ELES in combined RANS-LES simulation – Define LES zone. Extend LES zone to leave space around critical area. – Place RANS-LES interface into region of uncritical flows (such as equilibrium boundary layers) • Global model – Use SBES model and place synthetic turbulence generator to force the model into WMLES mode
Critical	• Visually check turbulent structures • Provide sufficient grid resolution in (WM)LES zones especially for wall-bounded flows (Section 12.1.2.3.3 (p. 229)).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	• CFL number<1.

12.1.7. References for Scale-Resolving Simulations

1

Boris, J. P., Grinstein, F. F., Oran, E. S., and Kolbe, R.S.,

"New insights into Large Eddy Simulation"

Fluid Dynamic Research, Vol. 10, 199-228, 1992

2

Cokljat D., Caradi D., Link G., Lechner R., and Menter F.R.,

"Embedded LES Methodology for General-Purpose CFD Solvers"

Proc. Turbulent Shear Flow Phenomena, 6th Int. Symp. Turbulence and Shear Flow Phenomena, Seoul, Korea, 22-24 June 2009, 1191-1196

3

Comte-Bellot, G., and Corrsin, S

"Simple Eulerian time correlation of full- and narrow-band velocity signals in grid-generated, 'isotropic' turbulence"

Journal of Fluid Mechanics, Vol. 48, Part 2, pp. 273-337, 1971

4

Davidson, L

"Evaluation of the SAS-SST Model, "Channel Flow, Asymmetric Diffuser and Axi-Symmetric Hill"

Proceedings European Conference on Computational Fluid Dynamics. ECCOMAS CFD

5

Durbin, P., and Pettersson Reif, B.A

"Statistical theory of turbulent flows"

John Wiley and Sons, Ltd, UK, 2003

6

Egorov Y., Menter F.R., and Cokljat D

"Scale-Adaptive Simulation Method for Unsteady Flow Predictions. Part 2: Application to Aerodynamic Flow"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Journal Flow Turbulence and Combustion, Vol. 85, No. 1, pp. 139-165, 2010

7

Fröhlich J., Mellen C.P., Rodi W., Temmerman L., and Leschziner M.,

"Highly resolved large-eddy simulation of separated flow in a channel with streamwise periodic constrictions"

Journal of Fluid Mechanics, Vol. 526, pp. 19–66, 2005

8

Fröhlich J., and von Terzi, D

"Hybrid LES/RANS methods for simulation of turbulent flows"

Progress in Aerospace Sciences, Vol. 44, Issue 5, pp. 349-377, 2008

9

Girimaji, S. S, (2004)

"Partially-Averaged Navier-Stokes Method: A RANS to DNS Bridging Method"

10

Girimaji, S., and Abdul-Hamid, K.S

"Partially-averaged Navier-Stokes model for turbulence: implementation and validation"

AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, paper 2005-0502, 2005

11

Gritskevich, M.S. Garbaruk, A.V., Schütze, J., and Menter F.R

"Sensitization of DDES and IDDES Formulations to the $k-\omega$ Shear-Stress Transport Model"

Flow Turbulence and Combustion, 2012

12

Geurts, B. J.

"Elements of direct and large-eddy simulation"

Edwards Inc, Philadelphia, USA, 2004

13

Hanjalic, K., and Launder, B

"Modelling Turbulence in Engineering and the Environment"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Cambridge University Press, Cambridge, UK, 2011

14

Jasak H., Weller H.G., and Gosman A.D

"High Resolution Differencing Scheme for Arbitrarily Unstructured Meshes"

International Journal for Numerical Methods in Fluids, Vol. 31, pp. 431-449, 1999

15

Mathey, F., Cokljat, D., Bertoglio, J.P., and Sergent, E

"Specification of LES Inlet Boundary Condition Using Vortex Method"

4th International Symposium on Turbulence, Heat and Mass Transfer, Antalya, Turkey, 2003

16

Menter, F. R

"Two-equation eddy-viscosity turbulence models for engineering applications"

AIAA Journal, Vol. 32, No. 8, pp. 1598–1605, 1994

17

Menter F.R., Garbaruk A., and Smirnov P

"Scale adaptive simulation with artificial forcing"

Proc. 3rd Symposium on Hybrid RANS-LES Methods, 2009

18

Menter, F.R., and Egorov, Y

"Scale-Adaptive Simulation Method for Unsteady Flow Predictions. Part 1: Theory and Model Description"

Journal Flow, Turbulence and Combustion, Vol. 85, No. 1, pp. 113– 138, 2010

19

Menter, F.R., and Kuntz, M

"Adaptation of Eddy-Viscosity Turbulence Models to Unsteady Separated Flow Behind Vehicles"

Proc. Conf. The Aerodynamics of Heavy Vehicles: Trucks, Busses and Trains, Asilomar, California, Springer, 2002

20

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Michelassi V., Wissink, J. G., and Rodi W

"Direct numerical simulation, large eddy simulation and unsteady Reynolds-averaged Navier-Stokes simulations of periodic unsteady flow in a low-pressure turbine cascade: A comparison"

Journal of Power and Engineering, Vol 217, Number 4, pp. 403– 411, 2003

21

Morris S. C. and Foss J. F.

"Turbulent boundary layer to single-stream shear layer: the transition region"

Journal of Fluid Mechanics Fluid Mech., Vol 494, pp. 187–221, 2003

22

Nicoud, F., and Ducros, F

"Subgrid-scale stress modelling based on the square of the velocity gradient tensor"

Flow, Turbulence and Combustion, Vol. 62, pp. 183– 200, 1999

23

Reichardt, H

"Vollstandige Darstellung der turbulenten Geschwindigkeitsverteilung in glatten Leitungen"

Zeitschrift fur Angewandte Mathematik und Mechanik, Vol. 31, pp. 208– 219, 1951

24

Rotta, J.C

"Turbulente Strömumgen"

BG Teubner Stuttgart, 1972

25

Sagaut, P., Deck, S., and Terracol, M

"Multiscale and multiresolution approaches in turbulence"

Imperial College Press, London, 2006

26

Schildmacher, K.-U., Koch, R., Wittig, S., Krebs, W., and Hoffmann, S.,

"Experimental investigations of the temporal air-fuel mixing fluctuations and cold flow instabilities of a premixing gas turbine burner"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

ASME Paper 2000-GT-0084, 2000

27

Shur, M.L., Spalart, P.R., Strelets, M.K., and Travin, A.K

"A hybrid RANS-LES approach with delayed-DES and wall-modeled LES capabilities"

International Journal of Heat and Fluid Flow, Vol. 29, pp. 1638-1649, 2008

28

Sjunnesson, A., Henriksson, R., and Lofstrom, C

"CARS measurements and Visualization of Reacting Flows in Bluff Body Stabilized Flame"

AIAA Paper 1992–3650, 1992

29

Smagorinsky, J

"General Circulation Experiments with the Primitive Equations"

Monthly Weather Review, Vol. 91, pp. 99 – 165, 1963

30

Spalart, P.R., Jou, W., Strelets, M., and Allmaras, S

"Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach"

Advances in DNS/LES. 1st AFOSR Int. Conf. on DNS/LES, 1997

31

Spalart, P. R

"Strategies for turbulence modelling and simulations"

International Journal of Heat and Fluid Flow, Vol. 21, pp. 2, 2000

32

Spalart, P., Deck, S., Shur, M., Squires, K., Strelets, M., and Travin, A

"A New Version of Detached Eddy Simulation, Resistant to Ambiguous Grid Densities"

Journal of Theoretical and Computational Fluid Dynamics, Vol. 20, pp. 181 –195, 2006

33

Strelets, M

"Detached Eddy Simulation of massively separated flows"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AIAA Paper 2001-879, 2001

34

Travin, A., Sur M., Strelets, M., and Spalart P

"Physical and numerical upgrades in the detached eddy simulation of complex turbulent flows", In Advances in LES of complex flows"

eds. R. Friedrich, W. Rodi, Kluwer Acad, New York, pp. 239-254, 2000

35

Vogel, J.C., and Eaton, J.K

"Combined heat transfer and fluid dynamic measurements downstream of a backward-facing step"

Journal of Heat and Mass Transfer, Vol. 107, pp. 922 –929, 1985

36

Wagner, C., Hüttl, T., and Sagaut, P

"Large-eddy simulation for acoustics"

Cambridge University Press, 2007

37

Westin, J., Alavyoon F., Andersson, L. Veber, P., Henriksson, M., and Andersson, C

"Experiments and Unsteady CFD-Calculations of Thermal Mixing in a T-Junction"

OECD/NEA/IAEA Workshop on the Benchmarking of CFD Codes for Application to Nuclear Reactor Safety (CFD4NRS), Munich, Germany, pp. 1–15, 2006

38

Widenhorn, A., Noll, B., and Aigner, M

"Numerical Study of a Non-Reacting Turbulent Flow in a Turbine Model Combustor"

AIAA Paper 2009-647, Orlando Florida, 2009

39

Wilcox, D.C

"Turbulence Modeling for CFD"

DCW Industries Inc, 3rd Edition, 2006

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2. RANS Turbulence Modeling in Ansys CFD

Turbulence modeling is one of the main sources of uncertainty in CFD simulations of technical flows. This is not surprising, as turbulence is the most complex phenomenon in classical physics. Turbulent flows pose a multi-scale problem, where the dimension of the technical device is often of the order of meters (or even 10^2 meters in case of airplanes and ships) whereas the smallest turbulence vortices are of the order of 10^{-5}-10^{-6} meters for high Reynolds number flows. Direct Numerical Simulation (DNS) of turbulence is therefore restricted to very small flow domains and low Reynolds numbers. Even the reduction in scales through Large Eddy Simulation (LES), does not lead to acceptable turn-around times for most technical flow simulations, especially in cases where wall boundary layers are important, as is most often the case (see Scale-Resolving Simulations in Ansys CFD (p. 181)).

A practical solution to this dilemma is offered through the concept of Reynolds-Averaged Navier-Stokes (RANS) equations. Instead of resolving the turbulent structures in time and space and then averaging the solution to obtain the desired engineering mean flow quantities, one first averages the equations and solves directly for the time-mean (or ensemble mean) variables. While this is much more economical, it eliminates the turbulence-related physics from the equations. Turbulence models are then required to feed back that information to allow physically correct simulations. When using RANS turbulence models, one should not forget that these models are tasked to bridge many orders of magnitude in computing power relative to DNS. From this perspective, it is not surprising that RANS computations are prone to modeling errors of significant size. Unfortunately, the RANS-related uncertainty cannot be quantified reliably, as there are not only quantitative errors, but there is also the potential for qualitative failure, when predicting incorrect flow topologies. Still, properly selected RANS models work very well for many technical applications, and it is therefore essential to understand the strength and weaknesses of different models to achieve optimal solution accuracy.

The current document is not intended as a textbook on turbulence modeling. The reader is referred to the available textbooks [2 (p. 367)] – [7 (p. 367)] the ERCOFTAC best Practice Guidelines [8 (p. 368)] as well as the Ansys Fluent and the Ansys CFX Theory and User documentation for deeper studies and for more details. The goal here is to guide the user through the process of optimal RANS model selection within the Ansys CFD codes, especially Ansys Fluent and Ansys CFX. Nevertheless, some of the equations required for the discussion are provided in 'Appendix A: Theory' so that the reader does not have to continuously revert to the Ansys Theory Documentation. Another document of relevance is Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401) .

The following topics are discussed:

12.2.1. General Considerations
12.2.2. Best Practice RANS
12.2.3. Model Evaluation
12.2.4. Numerical Settings
12.2.5. Summary
12.2.6. Acknowledgment
12.2.7. References
12.2.8. Appendix A: Theory

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.1. General Considerations

For CFD users, it is important to understand that turbulence modeling is an advancing technology and that Best Practice advice given some years ago might not be the Best Practice advice of today. When reading the current document, one will find that there is a consolidation in one of the most important subjects in RANS modeling – namely the choice of a scale-equation. While in Ansys CFX, there has historically been a strong emphasis on the ω-equation, this is not the case in the Ansys Fluent community, which still has a significant use of $k-\varepsilon$ models. To simplify turbulence model selection on the user side, as well as turbulence model development on the Ansys side, it is desirable to unite most users behind a single model family. The model family of choice in Ansys-CFD is based on the ω-equation. This does not mean that ε-equation based models will be discontinued from the codes, but it does mean that the majority of Ansys turbulence modeling efforts will focus on the ω-equation family. Making such a choice requires good arguments, and one of the intentions of this document is to provide that rationale and to convince the Ansys CFD user base of its merits.

One of the historic reasons for having a multitude of turbulence models in a CFD code is the ability to select the most accurate model for a given application. When focusing on a single model family, it is necessary to address this issue and to provide the same flexibility within that given formulation. This is achieved by a concept called Generalized $k-\omega$ (GEKO) model [Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)] and [10 (p. 368)]. GEKO is an ω-equation-based turbulence model framework, which introduces free parameters into the equations, which can be selected and tuned by the user within given ranges, without negative impact on the basic model calibration. Instead of switching between different turbulence models to optimize accuracy, it is anticipated that the user stays within the GEKO model and optimizes its free parameters. The parameter range of the GEKO model is designed to cover a wider solution space than the one historically available through different model families. Furthermore, specific parameter choices for GEKO constitute an exact transformation of the standard $k-\varepsilon$ model (albeit with a superior $k-\omega$ wall treatment), whereas another parameter setting mimics e.g. the SST model. The need to use historic models is thereby drastically reduced. There is also a detailed Best Practice Report on how to apply the GEKO model [Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)].

Consistent with the above reasoning, the current document will not go through all historic model choices available in both CFD codes, but will provide a guide as to the most optimal selection and usage of the preferred models. This will make for a much easier read, which is already a significant benefit of the proposed strategy. However, the rationale of selecting the ω-equation will be provided, and the pros-and-cons of different modeling concepts will be discussed.

One-equation models are also part of the current Best Practice discussion but will be treated only briefly. The reason is that these models are not suitable for a building-block approach, where different elements of physics need to be combined seamlessly to form a complete modeling framework. This does not mean that the use of such models cannot be suitable in the specific application area for which they have been developed, but they do not form the main track of turbulence models in Ansys CFD codes.

12.2.2. Best Practice RANS

The following topics are discussed:

12.2.2.1. Managing Uncertainty

12.2.2.2. Steady vs Unsteady vs Convergence

12.2.2.3. Turbulence Model Selection

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.2.4. Additional Physics

12.2.2.1. Managing Uncertainty

If confronted with a new application, it is prudent to ensure that the turbulence model intended to be used is validated for this type of flows – or at least for the underlying flow features observed. Ideally, one would start with a validation study of a similar case for which experimental data are available and optimize the CFD set-up:

- Geometry
  - Represent geometry as closely as possible – small simplifications in geometry can sometimes have large effects on the solution.
  - Avoid overly tight domains – try to place inlets in regions of well-defined flow and avoid outlets in regions of strong non-equilibrium flow dynamics, especially separation/backflow zones.
- Conditions
  - Select correct physical properties for density, viscosity, …
  - Ensure correct representation of flow physics (rotating systems, porous media, …).
- Optimal grid (see Mesh Resolution Requirements (p. 345))
  - Select an optimal grid topology for given flows.
  - Ensure fine resolution of wall boundary layers (see Fully Turbulent Boundary Layers (p. 347)).
  - Perform simulations on successively refined grids until the solution no longer changes.
- Boundary conditions
  - Match experimental/application boundary conditions closely.
  - Avoid inflow at outlets. Often the outlet flow can be accelerated through contraction of the domain in this region to avoid backflow.
- Numerics
  - Decide on steady versus unsteady settings (see Steady vs Unsteady vs Convergence (p. 278)).
  - Use 2nd order numerics if possible, also for the turbulence equations.
  - Do not use small under-relaxation factors (URF). Small values can slow down convergence. The selection of optimal URF is as a balance between convergence speed and robustness. Default values tend to be on the conservative side towards robustness.
- Turbulence model
  - Compare different models or modify GEKO coefficients and establish if the flow is sensitive to model changes.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - Select models/coefficients which are best calibrated for the given application – or which match best the validation case at hand.
  - Optimize GEKO coefficients if indicated by experimental data.
  - Decide if additional terms need to be activated
    → Curvature Correction.
    → Corner Correction.
    → Rough walls.
    → Buoyancy.
    → …

- Run case in automated/scripted way to avoid user errors during set-up and post-processing.
- Test solution sensitivity to any arbitrary assumptions made above by variation of such parameters.

The process outlined cannot be followed in all industrial CFD projects. However, it is helpful to recall what an optimal process would look like and be aware of any shortcuts taken. Experience shows that many CFD errors result from non-optimal set-ups and meshes and not from deficiencies of the turbulence model. The most common source of error being under-resolved meshes or non-converged solutions.

12.2.2.2. Steady vs Unsteady vs Convergence

The first question to consider before starting a RANS simulation is if the flow is steady or unsteady. As all turbulent flows are inherently unsteady in a physical sense, the question more precisely is if the flow is steady in the framework of RANS modeling. This might be one of the most difficult questions to answer a priori. In addition, the answer to this question can depend on the RANS model selected. RANS models that predict large zones of flow separation will be more likely to develop unsteady solutions than models that predict small or no separation zones. Often the models that predict larger separation zones are closer to the experimental data.

For cases where unsteadiness is imposed from outside, like flows with moving geometries, or unsteady boundary conditions, the situation is clear. However, there are also many flow scenarios where the set-up is 'steady', but where the flow still exhibits unsteady characteristics:

- Vortex shedding behind bluff bodies.
- Meandering/precessing Vortex (e.g. combustion chamber).
- Rotating stall in axial turbomachines.
- …

If a steady state solution is expected, the usual strategy is to set numerics to 'steady conditions' and run the case. If residuals converge 'deeply' a steady solution is achieved. Unfortunately, the question 'How deep is enough?' cannot easily be answered. While in some cases, three orders of magnitude are enough, whereas other cases might require 4-6 orders. It is therefore recommended

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

to display monitor quantities at monitor points which are sensitive to solution convergence. One can also display distributions (e.g. xy-plots, Contour plots) of quantities during the simulation. A case in point would be to display the wall shear-stress in a cut of the geometry to judge if the solution has settled down. This is particularly important in cases where a transition model is employed as laminar-turbulent fronts tend to settle down slowly.

Unfortunately, in many complex applications, 'deep' convergence cannot be achieved, as there will invariably be small regions, where the solution might not fully settle down, partly for physical and partly for numerical or mesh quality reasons.

To judge the steadiness of the simulation in such scenarios, it is again recommended to set monitor points and plot them during the solution. This can be global parameters, like lift or drag forces, pressure drop between inlet and outlet, massflow etc. and/or local quantities like primary solution variables (u,p,T,…) at specific locations. These locations should be placed in regions of high flow complexity and the highest likelihood of convergence problems. In cases where either the residuals, or the monitor points do not settle down to an acceptable level, it is recommended to switch to unsteady mode and continue the simulation. There are cases, where switching to unsteady mode helps to converge to steady state. If this is not the case, users must decide if the unsteadiness is of relevance, or just a small local disturbance without impact on the simulation outcome. If the unsteadiness is important, then time-mean values can only be obtained by running the simulation unsteady and averaging during run-time. It can also help to create animations to understand the cause and nature of unsteady behavior.

In a worst-case scenario, a steady state set-up can force the solution into an incorrect flow topology from which it cannot 'escape'. This is accompanied by non-convergence of the residuals which is not easy to distinguish from non-convergence due to small local oscillations. Such situations are more likely if small under-relaxation coefficients are used, as they tend to 'freeze' non-physical solutions. In such cases, switching to unsteady settings can overcome the problem and will typically lead to a strong change in the flow characteristics (as indicated by local monitor points and global forces).

12.2.2.3. Turbulence Model Selection

For most industrial applications, eddy-viscosity models provide the optimal balance between accuracy and robustness. Reynolds Stress Models (RSM) are not recommended for general use, as they often result in robustness problems without a reliable increase in accuracy. The additional physical effects accounted for in RSM can in most cases also be added to eddy-viscosity models, through Curvature Correction, Corner Correction and Buoyancy extensions and finally the use of Explicit Reynolds Stress Models (EARSM).

The following topics are discussed:

12.2.2.3.1. Spalart-Allmaras (SA) One Equation Model

12.2.2.3.2. Two-Equation Models

12.2.2.3.3. Wallin-Johansson Explicit Algebraic Reynolds Stress Models (WJ-EARSM)

12.2.2.3.4. Reynolds Stress Models (RSM)

12.2.2.3.5. Limiters

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.2.3.1. Spalart-Allmaras (SA) One Equation Model

The one-equation model of Spalart-Allmaras [11 (p. 368)] is widely used for external aerodynamical applications in the aeronautical industry and is well suited for such applications. It provides an improved performance relative to $k-\varepsilon$ models for flows with adverse pressure gradients and separation. Overall, the accuracy to predict separation is lower than for optimal two-equation models like SST and GEKO. On the other hand, the model requires the solution of only one transport equation instead of two. The SA model is not recommended for general use, as it is not well calibrated for free shear flows. It does predict accurate spreading rates for mixing layers but fails for plane and round jet flows, which are strongly dissipated (overly large spreading rate) by the model. In addition, the model does not predict decay of freestream turbulence which is of importance for some types of laminar-turbulent transition predictions.

The SA model in Ansys Fluent is also not extended to include:

- Laminar-turbulent transition
- Buoyancy
- Stress-Blended Eddy Simulation (SBES)

12.2.2.3.2. Two-Equation Models

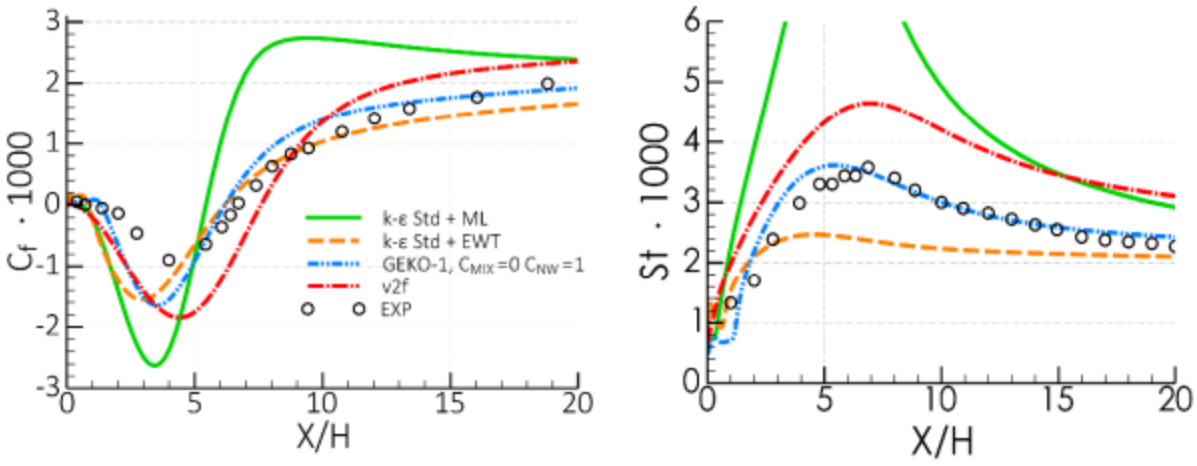
Two-equation models are the main model family for industrial flow simulations. They form the foundation of a building block system which can include all elements of RANS modeling.

Within the two-equation model family, the $k-\omega$ based models are recommended. They offer a superior wall treatment compared to $k-\varepsilon$ based models and are therefore much more flexible and accurate especially for non-equilibrium flows. This can be seen in Figure 12.61: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow (12). (p. 281) for the Vogel and Eaton backward facing step [12 (p. 368)]. This flow provides experimental data for the wall shear stress coefficient, C_f, and the heat transfer coefficient, St, on the wall downstream of the step. The mesh for this study had a fine near wall resolution of $y^+<1$.

Figure 12.61: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow (12). (p. 281) shows a model comparison for this flow. All model variants shown are based on the same standard $k-\varepsilon$ model. The $k-\varepsilon$ ML model is a representative of a $k-\varepsilon$ low-Re model and inhibits the known $k-\varepsilon$ model deficiency of substantial over-prediction of C_f and St near the reattachment point [5 (p. 367)]. The same $k-\varepsilon$ model combined with the 2-Layer based Enhanced Wall Treatment (EWT) shows an entirely different behavior, with a very flat distribution of the heat transfer coefficient (but a better match of C_f). The $k-\varepsilon$ model in combination with the V2F approach [13 (p. 368)] gives an over-prediction of the separation bubble size and an overly high heat transfer coefficient distribution. The V2F model is only included for comparison and is not offered in any of the Ansys CFD codes. The $k-\omega$ based GEKO model with C_{SEP}=1.0 (short GEKO-1.0) model shows the best agreement of both C_f and St. The GEKO-1.0 model is an exact transformation of the $k-\varepsilon$ to a $k-\omega$ formulation, except for the sublayer model. Note that other $k-\omega$ models like BSL/SST models produce very similar results to GEKO. This example shows the superior behavior of the ω-equation based turbulence model for predicting wall shear-stress and heat transfer distributions compared to other approaches.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.61: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow (12 (p. 368)).



Turbulence models from the $k-\omega$ family offer additional benefits when predicting adverse pressure gradient flows and separation onset as will be shown in Adverse Pressure Gradients and Flow Separation (p. 294). Finally, $k-\omega$ models are compatible with models for laminar-turbulent transition and rough wall treatments. All $k-\omega$ models in Ansys CFD are implemented with a y^+-insensitive wall treatment, avoiding the discussion concerning the optimal selection of wall formulations in $k-\varepsilon$ models (see e.g. Near-Wall Treatment (p. 390)). It is important to note that the grid resolution requirements between $k-\varepsilon$ and $k-\omega$ models are the same. In case of coarse grids (large y^+) the y^+-insensitive wall treatment will switch automatically to a wall function – so that there is no advantage in explicitly selecting a wall function (a choice required in some $k-\varepsilon$ models).

It should be noted that there can always be cases where one turbulence model happens to be more robust than another. In case of robustness problems with models from the $k-\omega$ model family, it can be a prudent strategy to switch to a $k-\varepsilon$ model.

Standard k-ε Model [14 (p. 369)]

- Use for cases where backwards compatibility is required to compare to previous simulations using this model.
- Be aware that for this model no limiter (see Limiters (p. 284)) is activated so that excessive turbulence production can affect the simulations. Note that the lack of a limiter can lead to improved convergence (for the wrong reasons).
- It is recommended to activate the 'Production Limiter'.
- Note that the GEKO model with C_{SEP}=1.0 and C_{NW}=1.0 (an example is shown in NASA CS0 Diffuser (p. 294) for the model comparison) is an exact transformation of the standard. $k-\varepsilon$ model – albeit with a superior wall treatment and an automatically activated realizability limiter.
- On coarse meshes, use scalable wall functions, for finer meshes use the EWT based on the 2-layer formulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Realizable k-ε Model (RKE) [15 (p. 369)]

- Use cases where backwards compatibility is required to compare to previous simulations using this model.
- Be aware that the Realizability limiter of this model is only partially effective – it allows for large turbulence production in non-shear regions (see Buoyancy Flows (p. 340)). The model can be combined with the 'Production Limiter'.
- On coarse meshes, use scalable wall functions, for finer meshes use the EWT based on the 2-layer formulation (Near-Wall Treatment (p. 390)).

RNG $k-\varepsilon$ Model (RNGKE) [16 (p. 369)]

- Use for cases where backwards compatibility is required to compare to previous simulations using this model.

Standard $k-\omega$ Model [7 (p. 367)]

- Do not use the standard $k-\omega$ model – it has a strong dependency of the solution on freestream values of ω outside of shear layers (see Figure 12.161: Effect of freestream omega-levels on velocity profile for free mixing layer solution. Left: std. k-omega model without CD term. Right: GEKO model including CD term. (p. 381)). The model is available mostly for historic reasons.

BSL/SST Model [17 (p. 369)], [18 (p. 369)]

- The SST model is recommended for most industrial applications. It has a high accuracy for flows with adverse pressure gradients and separation. Outside the boundary layer it reverts back to a $k-\varepsilon$ model setting.
- The SST model's ability to accurately predict separation is based on the SST-limiter (Equation 12.68 (p. 383)) which reduces the eddy-viscosity in such flows. Larger separation zones can lead to unsteady behavior, which might be physically correct, but undesirable from a project standpoint. In such cases the a_1 coefficient (default a_1 = 0.31) can be increased to values in the range of a_1 = 0.31-1.0 to reduce separation. Note that the a_1 coefficient cannot be reduced below its default value of a_1 = 0.31 without violating the basic calibration of the model for boundary layers.
- Alternatively, one can switch to the BSL model, which de-activates the SST limiter entirely.
- BSL and SST models automatically activate the 'Production Limiter'.
- BSL and SST models automatically activate y^+-insensitive wall-treatment.
- Do not use the 'low-Reynolds number correction' with any of the $k-\omega$ models. It is a historic feature. It is not required to integrate the equations to the wall but can lead to pseudo-transition – meaning a non-calibrated laminar-turbulent transition effect.

GEKO Model [10 (p. 368)]

- The model is intended to replace eventually all other two-equation models.
- The model can and should be used for all industrial applications.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The model has the flexibility to allow users to tune the model against experimental data.
- The model has a realizability limiter automatically activated.
- The model activates the y^+-insensitive wall-treatment.
- The model can be tuned to mimic existing model like std. $k-\varepsilon$ or SST.
  - The SST is mimicked by default settings. Note that this does not mean an exact transformation of SST to GEKO.
  - The $k-\varepsilon$ model is recovered for C_{SEP}=1.0, C_{NW}=1.0. (An example is shown in NASA CS0 Diffuser (p. 294) for the model comparison)
- The GEKO model has an adjoint formulation in Ansys Fluent which can be used as a basis for machine learning.
- Note that the GEKO model is not fully published – this could lead to issues in case users want to publish their results in a scientific journal.
- There is an extensive Best Practice Guide for this model available: [Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)].

12.2.2.3.3. Wallin-Johansson Explicit Algebraic Reynolds Stress Models (WJ-EARSM)

- The WJ-EARSM [19 (p. 369)] can be used in case secondary flows are of importance (an example of the corner flows is shown is Corner Flows (p. 308)). Note however, that similar effects can be achieved by activating the simpler Corner Flow Correction (CFC) available with GEKO (and in Ansys Fluent with all other $k-\omega$ models) [20 (p. 369)].
- Use in combination with BSL (WJ-BSL-EARSM) or better with GEKO model (a β feature in Fluent .
- EARSM models will not provide the same sensitivity to streamline curvature and system rotation as full RSM and additional curvature correction terms might need to be added.

12.2.2.3.4. Reynolds Stress Models (RSM)

- These models are prone to numerical problems on complex applications and non-optimal grids. They are therefore only recommended for applications, where they have shown superior performance over eddy-viscosity models.
- Examples of such applications are flows with strong curvature and or system rotation. Note however, that similar effects can be achieved when activating the Curvature Correction model in eddy-viscosity models.
- In case RSM are used, it is recommended to combine them with the ω-equation (BSL or GEKO). The GEKO-RSM model formulation is based on the stress-omega model [7 (p. 367)]. The model solves the Reynolds stress equations in combination with the ω-equation from the GEKO model instead of the original Wilcox model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.2.3.5. Limiters

The eddy-viscosity assumption alters the production term, P_k, in the k-equation from a term linear in the velocity gradients to a quadratic term ($P_k=\mu_t S^2$). This can cause problems in regions with non-shear layer related strain rate, S, like in inviscid stagnation or in acceleration zones. When using two-equation eddy-viscosity models, limiters must therefore be employed. Limiters are not required when running the WJ-EARSM as it restricts the level of production over dissipation automatically. Examples for the use of limiters are given in Limiters (p. 389).

For $k-\varepsilon$ models, the user needs to activate a limiter. For the std. $k-\varepsilon$ model no limiter is active by default (representing the published version of the model) and the production limiter should be activated. For the RKE model a realizability limiter is built-in, but experience shows that it does not work effectively, due to the specifics of its formulation. Also, for this model the production limiter is recommended. Note that when writing a report/publication the activation of limiters should be clearly indicated to allow a proper interpretation and reproduction of the results.

All ω-equation based models in Ansys Fluent and Ansys CFX have the production limiter automatically activated. The GEKO model features in addition a proper realizability limiter.

When models for laminar-turbulent transition are activated, it was found that in some cases the production limiter was not sufficient to prevent a minor build-up of turbulence in the stagnation zone of airfoils. This is usually unnoticeable in fully turbulent mode but can slightly affect the transition location. For this reason, the Kato-Launder limiter [21 (p. 370)] is additionally activated for such flows. This limiter can affect other parts of the flow, especially flows with swirl and curvature. In case this is not acceptable, this limiter can be deactivated.

- For $k-\varepsilon$ models, activate the production limiter manually.
- For $k-\omega$ models no action is required.
- For transition models – no action is required unless the Kato-Launder limiter is not wanted.

12.2.2.4. Additional Physics

The following sections describe model additions which can be used in combination with two-equation models. Particularly, Corner Correction [20 (p. 369)] and Curvature Corrections [22 (p. 370)], [23 (p. 370)] are meant to eliminate some deficiencies of eddy-viscosity models relative to fully Reynolds-Stress Models. Other additions, like rough walls and laminar-turbulent transition are required for all RANS models if the effects are important for the intended application.

The following topics are discussed:

12.2.2.4.1. Laminar-turbulent transition
12.2.2.4.2. Curvature Correction
12.2.2.4.3. Corner Correction
12.2.2.4.4. Buoyancy Correction
12.2.2.4.5. Wall Roughness Correction

12.2.2.4.1. Laminar-turbulent transition

There are four models for laminar-turbulent transition in Ansys Fluent and Ansys CFX:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The algebraic intermittency model (Ansys Fluent)
- The One-equation intermittency model (Ansys Fluent, Ansys CFX)
- The two equation $\gamma-\mathrm{Re}_\theta$ model (Ansys Fluent, Ansys CFX)
- $\mathrm{k}-\mathrm{kl}-\omega$ Model (Ansys Fluent)

The first three models are based on the concept of 'Local-Correlation based Transition Modeling – LCTM'. They form a series of models as developed over time starting with the $\gamma-\mathrm{Re}_\theta$ model [24 (p. 370)]–[26 (p. 370)] which was then simplified to a one-equation model and finally to an algebraic model. The models are calibrated in a similar fashion, but offer differences in their predictions. Users are therefore advised to test which model best suits their test cases. Obviously, the CPU cost increases with the number of transport equations being solved. The algebraic model might therefore serve as a good starting point. The $\mathrm{k}-\mathrm{kl}-\omega$ formulation models the kinetic energy in the pre-transitional phase of the boundary layer and triggers transition based on this quantity. It is not widely used and not recommended, as it is only compatible with the standard $k-\omega$ model.

Including the effect of transition into the CFD simulation requires finer mesh resolution (see Transitional Boundary Layers (p. 355)) and should therefore be employed with caution.

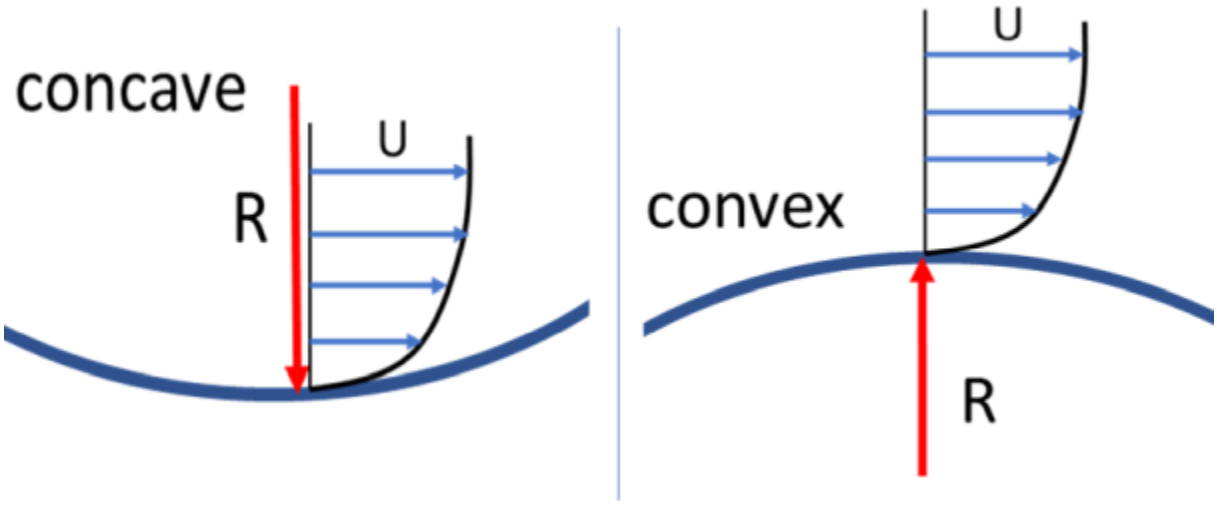
- Transition prediction is important for aerodynamic and turbomachinery flows, as transition can strongly affect overall performance. It is also important to include in some heat transfer cases like impinging jets, as laminar flows have a much lower heat transfer coefficient than turbulent flow.
- In case transition is important, it has strong implications on the set-up from a modeling, numerical and mesh resolution standpoint.
  - Select a suitable transition model
  - The mesh resolution requirements for predicting transition are much higher than for fully turbulent flow simulations. This is discussed in Transitional Boundary Layers (p. 355).
  - Numerical settings might need to be adjusted.
  - Monitor points/charts on transition location motion should be included to check convergence. Plotting charts of wall shear stress of C_f during iterations is recommended to visually see if the transition locations are settling down.

12.2.2.4.2. Curvature Correction

Streamline curvature and swirl can have a significant effect on turbulence. This is illustrated in Figure 12.62: Schematic of the boundary layer on curved surfaces. (p. 286) which shows on the left a boundary layer on a concave wall and on the right a boundary layer on a convex wall. In the concave case, turbulence is enhanced and for the convex case it is damped. Standard eddy-viscosity models do not account for this effect and need augmentation to sensitize them. The effect of curvature is even stronger for swirling flows, where turbulence is typically strongly suppressed (see Swirl Flows (p. 323)).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.62: Schematic of the boundary layer on curved surfaces.



The Curvature Correction [22 (p. 370)] used in Ansys-CFD is based on the so-called Spalart-Shur correction [23 (p. 370)]. The details of the formulation can be found in the theory documentation and are not reported here. This correction detects changes in streamline direction due to changes in the orientation of the strain rate tensor along a streamline. This then results in a correction factor which multiplies the production terms of both the k- and the ω-equations. For stabilizing curvature, the factor is smaller and for de-stabilizing flows larger than one.

The CC is most relevant not for boundary layers but for free vortex flows, where the effect of curvature results in strong damping of turbulence inside the vortex. Models without CC provisions produce much too high eddy-viscosities in such scenarios and strongly damp the vortex.

- Turbulence can be dampened or enhanced based on streamline curvature (see Swirl Flows (p. 323)). The Correction should be activated if curvature effects are strong and important. This is typically the case if the radius of curvature is of the same order or smaller than the boundary/shear-layer thickness. Note that for most boundary layer flows this is not the case.

- This correction is most beneficial for swirling flows (tip vortex of a wing, vortex inside a hydro-cyclone). Eddy-viscosity models without this correction create excessive eddy-viscosity levels in such flows and reduce vortex strength too strongly. In addition, they enforce a solid body-like rotation, whereas a real vortex often has significant regions following inviscid laws.

- In principle, the correction does not hurt flows without or with only weak curvature but increases computational costs.

12.2.2.4.3. Corner Correction

Another physical effect which cannot be represented by eddy-viscosity models is the secondary flow caused by turbulence through anisotropy of the normal stresses (meaning that the normal stresses in the wall parallel and wall normal directions are different, driving a secondary flow). This effect appears typically in corner flows (like rectangular channels). One strategy to enable eddy-viscosity models to account for this effect is to extend them through a non-linear stress-strain relationship (see EARSM models in Corner Flows (p. 308)). The simplest formulation is however the Quadratic Closure Relation (QCR) proposed by Spalart [20 (p. 369)]. It consists of extending the linear part of the eddy-viscosity formulation by a quadratic term:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\tau_{ij} = -\rho\overline{u_i u_j} = \mu_t 2S_{ij} - \frac{2}{3}\rho k\delta_{ij} - C_{CORNER}\frac{1.2\ \mu_t}{max\left(0.3\omega, \sqrt{(S^2+W^2)/2}\right)}\left(S_{ik}W_{kj} - W_{ik}S_{kj}\right) \tag{12.38}$$

with

$$S_{i,j} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right); \qquad W_{i,j} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right)$$

The effect is important for:

- Flows parallel to corners, like those observed in wing-body junctions or rectangular channels, which develop secondary flow directed into the corner. This effect (Prandtl's secondary flows of 2nd kind, 27 (p. 371)) cannot be captured by eddy-viscosity models, as the effect is driven by the anisotropy of the normal stresses of the Reynolds Stress tensor.
- Without this correction, the flow in such corners can show premature separation if encountering an adverse pressure gradient. This can have a significant effect on the overall flow and can potentially lead to incorrect flow topologies in the CFD simulation. The resulting errors can be very large, as a change in flow topology (e.g. separation from a corner instead from the clean part of the wing/blade) can affect most other flow parameters.
- Simulations will only benefit from this correction if there is a sufficiently fine grid resolution in the corner region. Hexahedral meshes are optimal for such flows, as they allow for an easy adaptation of the surface grid lines into the corner.
- This correction should not produce negative side-effects in terms of accuracy if activated in cases where it is not required. However, it can potentially affect robustness and CPU cost.

12.2.2.4.4. Buoyancy Correction

Flows with gradients in density in a gravity field can exhibit similar effects as flows with curvature. Flows where the force vector due to gravity and the density gradient point in the same direction are stabilized (decrease in turbulence) whereas the opposite is true for flows where the two vectors point in opposite directions. Buoyancy terms have been developed originally for the $k-\varepsilon$ model. The source term in the k-equation reads (g_i – gravity vector):

$$G_b = -g_i\frac{\mu_t}{Pr_t}\frac{1}{\rho}\frac{\partial\rho}{\partial x_i} \tag{12.39}$$

To judge if such effects are important, this term has to be set in relation to the production of turbulence kinetic energy. For shear layers one gets (with ΔU, $\Delta\rho$ – velocity, density differences across shear layer, δ thickness of layer, $Pr_t \sim 1$):

$$R = \left(-g_i\frac{\mu_t}{Pr_t}\frac{1}{\rho}\frac{\partial\rho}{\partial x_i}\right)\Big/\left(\mu_t S^2\right) \approx g\frac{\Delta\rho}{\rho}\frac{\delta}{\Delta U^2} \tag{12.40}$$

In case this ratio is larger than a threshold (of the order of $R > 0.1$) the term G_b should be included. As outlined in the theory documentation, the term can additionally be included into the scale-equation (ε/ω-equation). Since the benefit of doing so is not consistent, the default is to include the term only into the k-equation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The effect of buoyancy is similar to the effect of curvature – it can stabilize or destabilize turbulence.
- Turbulence is enhanced/damped if the density gradient and gravity vector point in opposite/same directions.

12.2.2.4.5. Wall Roughness Correction

Rough walls can have a significant effect on the performance of technical devices. Roughness has two essential effects. The first is that it increases wall shear stress levels (and thereby also heat transfer) and the second is that it forces laminar-turbulent transition to more upstream locations.

There are several roughness options in Ansys CFD:

- The standard method is a simple shift of the logarithmic layer based on wall roughness. In addition, the cell center of the wall cell is virtually shifted as: $y^+ \rightarrow y^+ + h_{s^+}/2$ where h_s is the roughness height. This option is used in Ansys Fluent.
- The second method is a blend of the Wilcox wall roughness model [7 (p. 367)] and the log-layer shift, based on y^+. This option is the default for all ω-equation based models in Ansys CFX.
- The last option is proposed by Aupoix [28 (p. 371)] and modifies the near wall boundary values of the turbulence quantities to achieve the desired increase in wall-shear. It can be applied to the Spalart-Allmaras model and all ω-based two-equation models.

The first two options are compatible with all y^+ values if y^+ lies in the inner region (~0.1 of the boundary layer thickness).

The Aupoix formulation requires a fine near wall mesh (y^+~1) but can cover roughness heights which are a significant fraction of the boundary layer height.

Roughness effects on transition location can also be included into the SST $\gamma - \mathrm{Re}_{\theta t}$ transition model (4-equation model). The effect of roughness is to push the transition location upstream. While the effect of fully turbulent flow is described by the sand-grain roughness, the effect on transition is determined by the height of the roughness elements (geometric roughness), as even a single roughness element can trigger transition.

12.2.3. Model Evaluation

The key consideration for selecting a turbulence model is the accuracy that the model can provide for a given flow, or a class of flows. As all turbulence models are more-or-less calibrated for the same base-flows (e.g. flat plate boundary layer, basic free shear flows and decaying turbulence), the model accuracy outside this narrow 'calibration box' can only be determined by further validation studies. Such studies are based on 'building block' test cases, which typically add one more element of complexity to the calibration cases, like adverse pressure gradients, separation, swirl, etc. As such cases do not account for the complex interactions seen in industrial flows, they can still not provide the full picture, but they can give a much better idea on the suitability of a model for a certain type of flows. Such tests are also the limit to which models are tested during model development and calibration, as more complex industrial flows typically lack detailed experimental data to draw meaningful conclusions on specific model deficiencies. In the following sections, model performance will be discussed for such building-block test cases, with each section focusing on a different physical effect. It

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

is important for CFD users to understand the differences in model performance to make an educated decision during model selection.

Since this is a Best Practice Document, only the most necessary information on the test cases and their numerical set-up is provided. All solutions have been ensured to be mesh independent and, in all cases, the boundary conditions match the experiment as closely as possible.

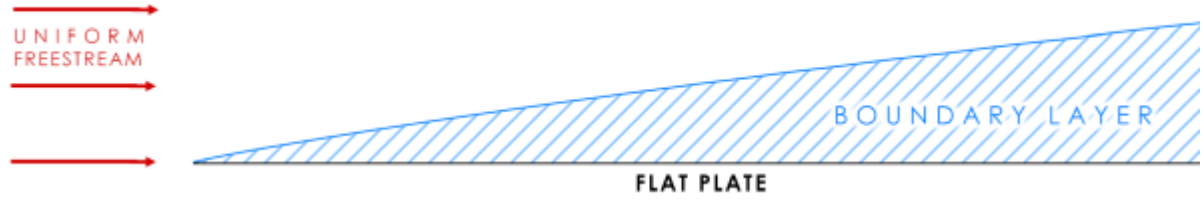
The following topics are discussed:

12.2.3.1. Flat Plate Flow
12.2.3.2. Adverse Pressure Gradients and Flow Separation
12.2.3.3. Corner Flows
12.2.3.4. Swirl Flows
12.2.3.5. Reattachment Flows
12.2.3.6. Impinging Flows
12.2.3.7. Buoyancy Flows
12.2.3.8. Effect of Limiters
12.2.3.9. Mesh Resolution Requirements

12.2.3.1. Flat Plate Flow

The setup of the zero pressure gradient flat plate boundary layer flow (see Figure 12.63: Schematic of the flow over a flat plate. (p. 289)) is based on the experiment of Wieghardt & Tillmann, [29 (p. 371)]. The Reynolds number based on inlet velocity and the length of the flat plate is $\mathrm{Re} = 10^7$, and the wall is maintained at a constant temperature. No heat transfer measurements have been carried out. Therefore, in order to assess the capability of turbulence models to predict heat transfer to the wall, an empirical correlation is used to compute the Stanton number at the wall from the measured distribution of skin friction coefficient.

Figure 12.63: Schematic of the flow over a flat plate.



The computational domain with boundary conditions is shown in Figure 12.64: Computational domain with boundary conditions and mesh for the flow over a flat plate (p. 290). Constant velocity and temperature values are assigned at the inlet plane. The difference between inlet and wall temperatures is 10° [K]. Turbulence characteristics at the inlet are also assumed constant with values corresponding to a free-stream turbulence intensity of Tu=1%, and an eddy-viscosity ratio equal to TVR=0.2. Computational meshes (see Table 1) cover three near-wall scenarios: viscous sublayer, logarithmic region and a "buffer layer" resolution.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.64: Computational domain with boundary conditions and mesh for the flow over a flat plate

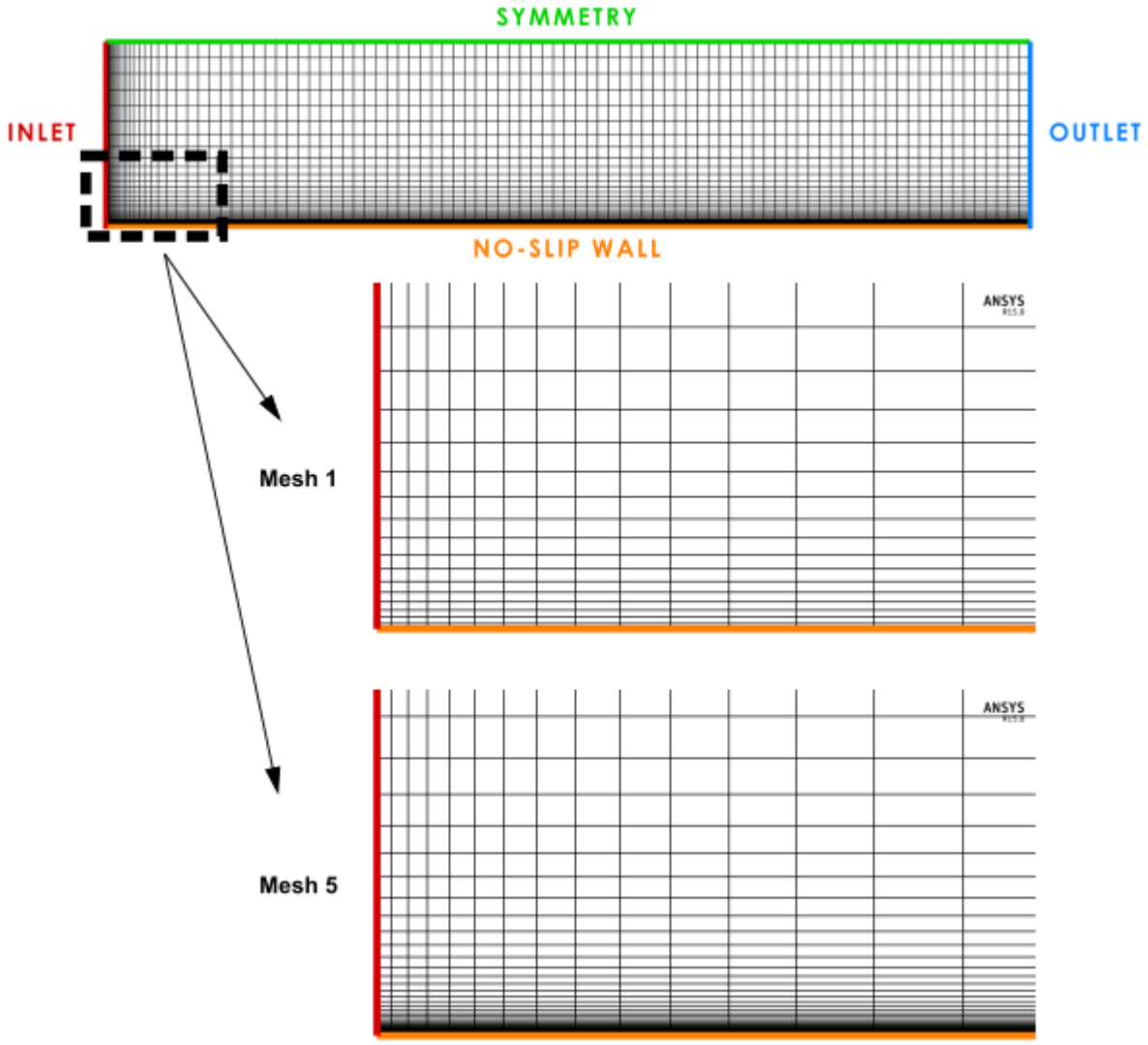


Table 12.10: Parameters of the computational meshes

	Mesh 1	Mesh 2	Mesh 3	Mesh 4	Mesh 5
y+max	56.9	22.6	14.4	3.1	0.3
y+mean	33.6	10.9	6.1	1.1	0.1
Number of cells	3800	4600	5000	6000	8000

Model Comparison

Figure 12.65: Comparison of wall shear stress coefficient, C_f (Left) and wall heat transfer coefficient, St. (Right) for flat plate boundary layer (28) and y^+~1. (p. 291) shows the comparison of the results of different turbulence models (SST, GEKO, SA and RKE) on the fine mesh: Mesh-5. As expected, all the considered models perform well for the flat plate simulations, both in terms of wall shear stress, Cf, and wall heat transfer coefficient, St, as shown in Figure 5. Also, the velocity profiles are in good

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

agreement with the logarithmic profile, as can be seen in Figure 12.66: Comparison of velocity profiles in wall-law for flat plate boundary layer at $Re_x=8.7\cdot10^6$. (p. 291).

Figure 12.65: Comparison of wall shear stress coefficient, C_f (Left) and wall heat transfer coefficient, St. (Right) for flat plate boundary layer (28 (p. 371)) and y^+~1.

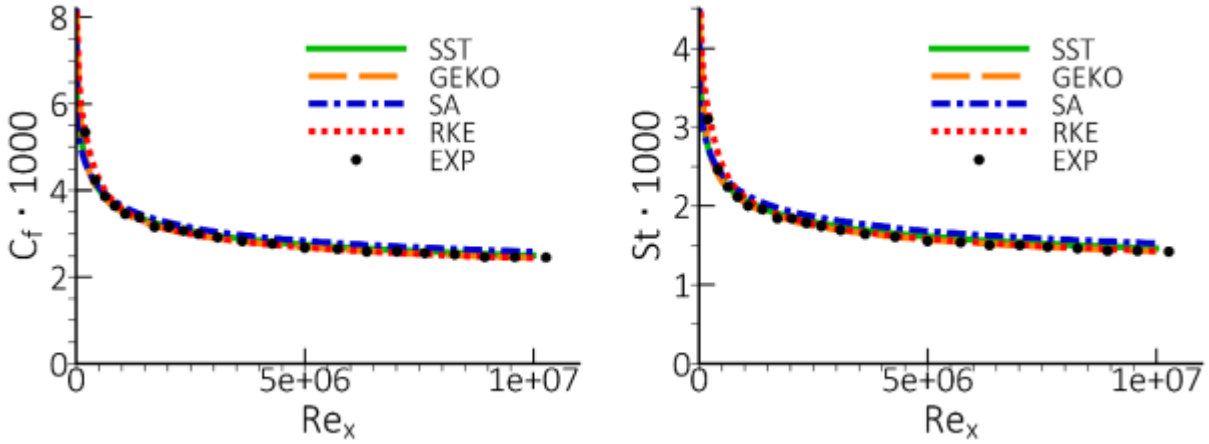
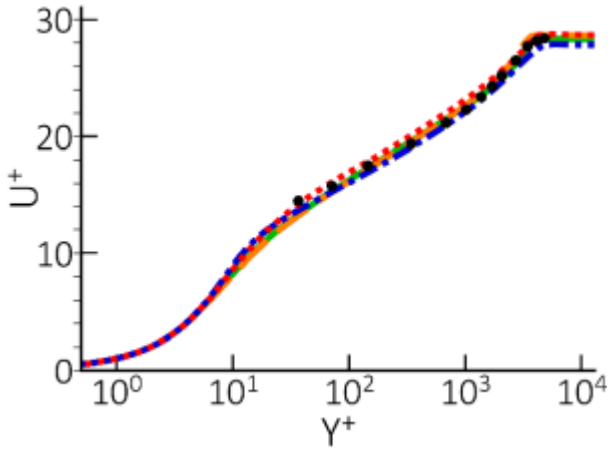


Figure 12.66: Comparison of velocity profiles in wall-law for flat plate boundary layer at $Re_x=8.7\cdot10^6$.

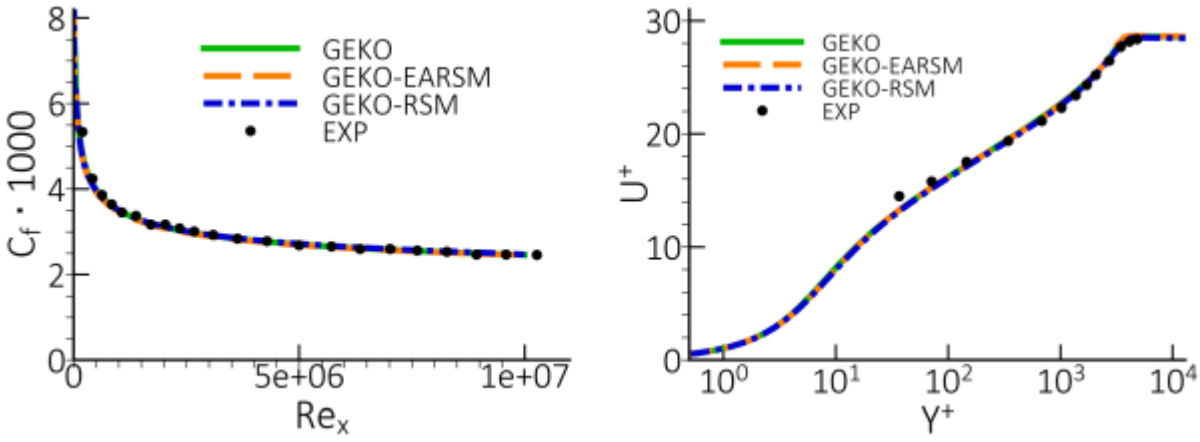


Non-linear effects

For the simple wall-bounded flows the non-linear effects of the EARSM or even RSM models do not affect the main flow. Figure 12.67: Comparison of wall shear stress coefficient (Left) and velocity profile at $Re_x=8.7\cdot10^6$ (Right) predicted with the GEKO model without and with EARSM/RSM options for the flat plate boundary layer (28). (p. 292) shows the prediction of the skin friction coefficient and wall-law velocity profile for the GEKO model without and with EARSM/RSM options.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.67: Comparison of wall shear stress coefficient (Left) and velocity profile at $Re_x=8.7\cdot10^6$ (Right) predicted with the GEKO model without and with EARSM/RSM options for the flat plate boundary layer (28 (p. 371)).



Mesh Sensitivity

For the coarser meshes, the computational accuracy depends on the wall treatment. The use of the automatic y^+-insensitive wall treatment for the $k-\omega$ turbulence models provides virtual independence of the solution to the near wall resolution. An example of the SST model solution is shown in Figure 12.68: Comparison of wall shear stress coefficient, C_f (upper left), wall heat transfer coefficient, St. (upper right) and log-law velocity profile at $Re_x=8.7\cdot10^6$ (lower right) for the flat plate boundary layer using the SST model [28]. (p. 293). On the other hand, the accuracy of prediction of wall shear stress for high-Re $k-\varepsilon$ turbulence models depends on the choice of the wall treatment. The standard wall function (SWF) designed for the high-Re $k-\varepsilon$ computational grids is not adequate for the low-Re fine $y^+\sim1$ grids, as shown in Figure 12.69: Distribution of the skin friction coefficient over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28]. (p. 293) (Left). The use of the scalable wall function (ScWF) improves the situation considerably, as shown in Figure 12.69: Distribution of the skin friction coefficient over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28]. (p. 293) (Right). For the rest of the test cases, the ScWF wall function is used for the $k-\varepsilon$ models. For the mathematical formulation of the different wall treatments see Near-Wall Treatment (p. 390).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.68: Comparison of wall shear stress coefficient, C_f (upper left), wall heat transfer coefficient, St. (upper right) and log-law velocity profile at $Re_x=8.7\cdot10^6$ (lower right) for the flat plate boundary layer using the SST model [28 (p. 371)].

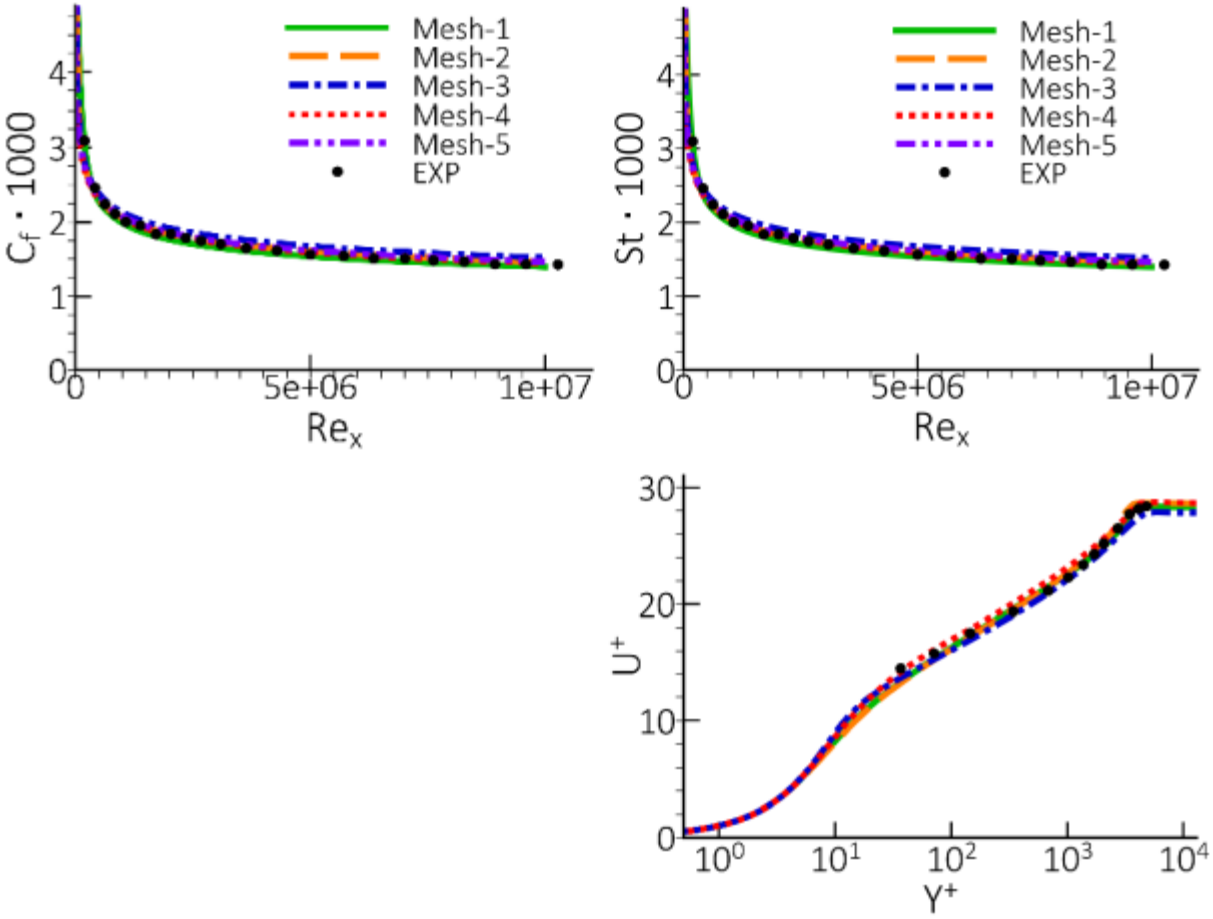
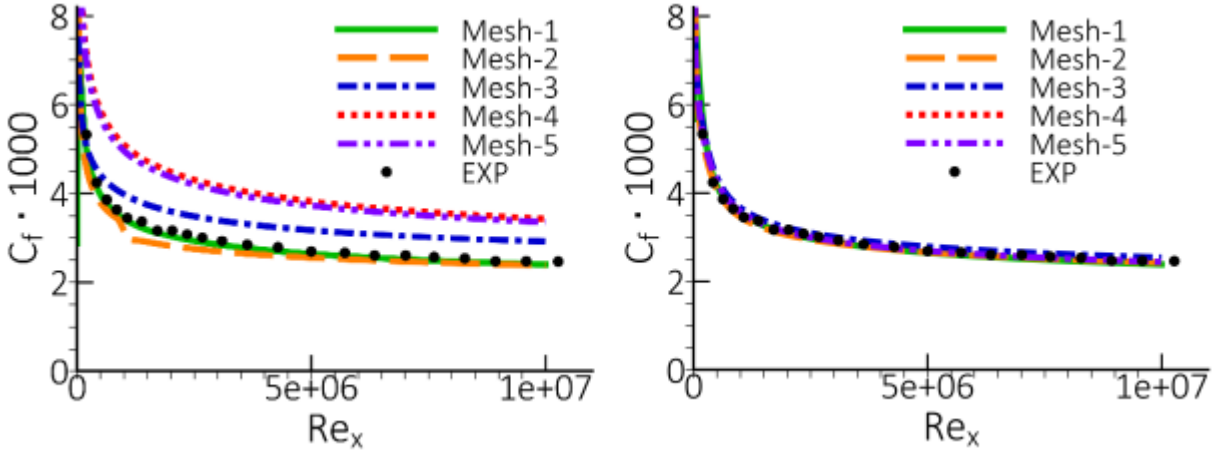
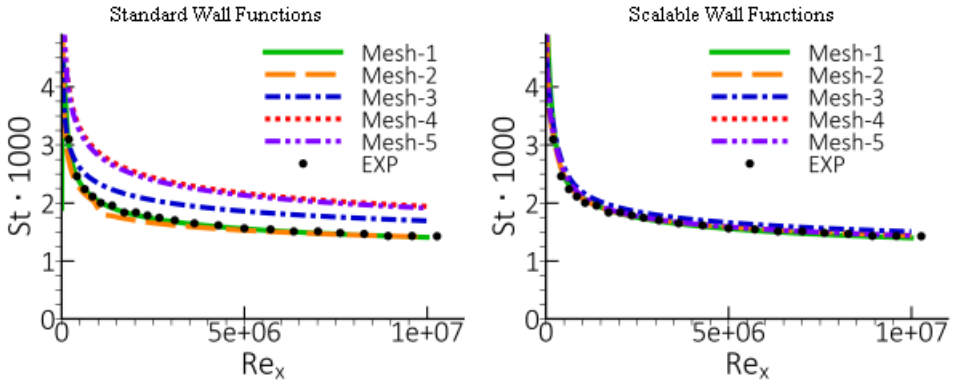


Figure 12.69: Distribution of the skin friction coefficient over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28 (p. 371)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.70: Distribution of the Stanton number over the flat plate computed with RKE model with SWF (left) and ScWF (right) for the flat plate boundary layer [28 (p. 371)].



12.2.3.2. Adverse Pressure Gradients and Flow Separation

Arguably the most important additional physical effect when developing turbulence models beyond equilibrium boundary layer flows, concerns their ability to accurately predict flows with pressure gradients and separation from smooth surfaces. This is of major importance in any external aerodynamic flow, especially airfoil and wing flows, where flow separation and stall are the defining factors of the performance envelope. However, separation from smooth surfaces is also relevant for internal flows, like diffusers and blades/vanes etc. For this reason, and because many turbulence models originate from the aeronautical community, separation prediction has always been a major focus in turbulence modeling. In more 'industrial' type flows, separation is often not caused by a pressure gradient, but by a shape change in the geometry, like a step or a corner. In such cases, the separation point/line is fixed and there is much reduced turbulence model sensitivity. This needs to be considered during the discussion of model differences for smooth wall separation. In the following 'separation' refers to separation from a smooth wall.

It is accepted knowledge in the turbulence modeling community that $k-\omega$ models are more suitable for predicting separation than$k-\varepsilon$ models [7 (p. 367)]. This will be demonstrated for several flows in the following discussions. Since the Spalart-Allmaras 1-equation model is also a popular turbulence model in aeronautics, it is included in some of the comparisons.

The following topics are discussed:

12.2.3.2.1. NASA CS0 Diffuser

12.2.3.2.2. Airfoil Flows

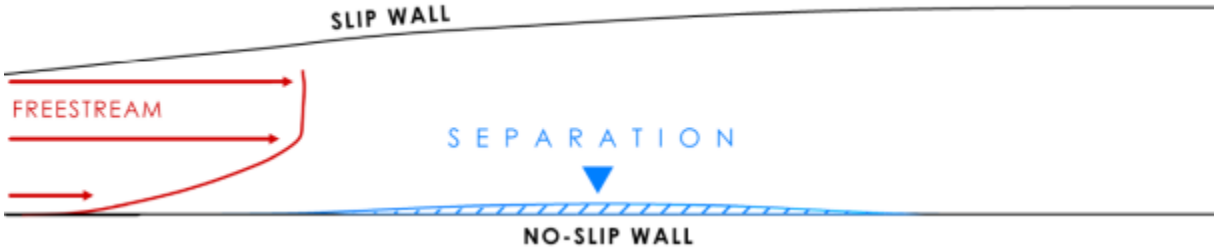
12.2.3.2.3. Transonic Bump Flow

12.2.3.2.1. NASA CS0 Diffuser

One of the most widely used test cases for evaluation of flow separation is the NASA CS0 diffuser of Driver [30 (p. 371)], where a relatively shallow separation region has been created in the experiment. The test case geometry (partly shown in Figure 12.71: Schematic of the flow in CS0 diffuser. (p. 295) near the separation zone) consists of an axisymmetric diffuser with an internally mounted cylinder along the centerline. The boundary layer develops along the axis of the cylinder. The expansion of the diffuser wall causes an adverse pressure gradient. The Reynolds number based on the freestream velocity, U_∞ and the cylinder diameter D, is equal to $\mathrm{Re}=10^6$.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.71: Schematic of the flow in CS0 diffuser.



The computational domain with boundary conditions is shown in Figure 12.72: Figure 12: Computational domain with boundary conditions and mesh for CS0 diffuser. (p. 295). At the inlet section, uniform streamwise velocity and turbulent quantities were specified. The outer radial boundary of the domain represents a streamline of the inviscid flow defined in the experiment, and so a free slip condition is imposed at this boundary. A no-slip condition is used on the internal cylinder wall everywhere except for the initial part where a symmetry condition is imposed (the length of the non-slip wall is adjusted to provide the correct boundary layer thickness at the first experimental section). Finally, a constant pressure condition is used at the outlet boundary.

Figure 12.72: Figure 12: Computational domain with boundary conditions and mesh for CS0 diffuser.

Model Comparison

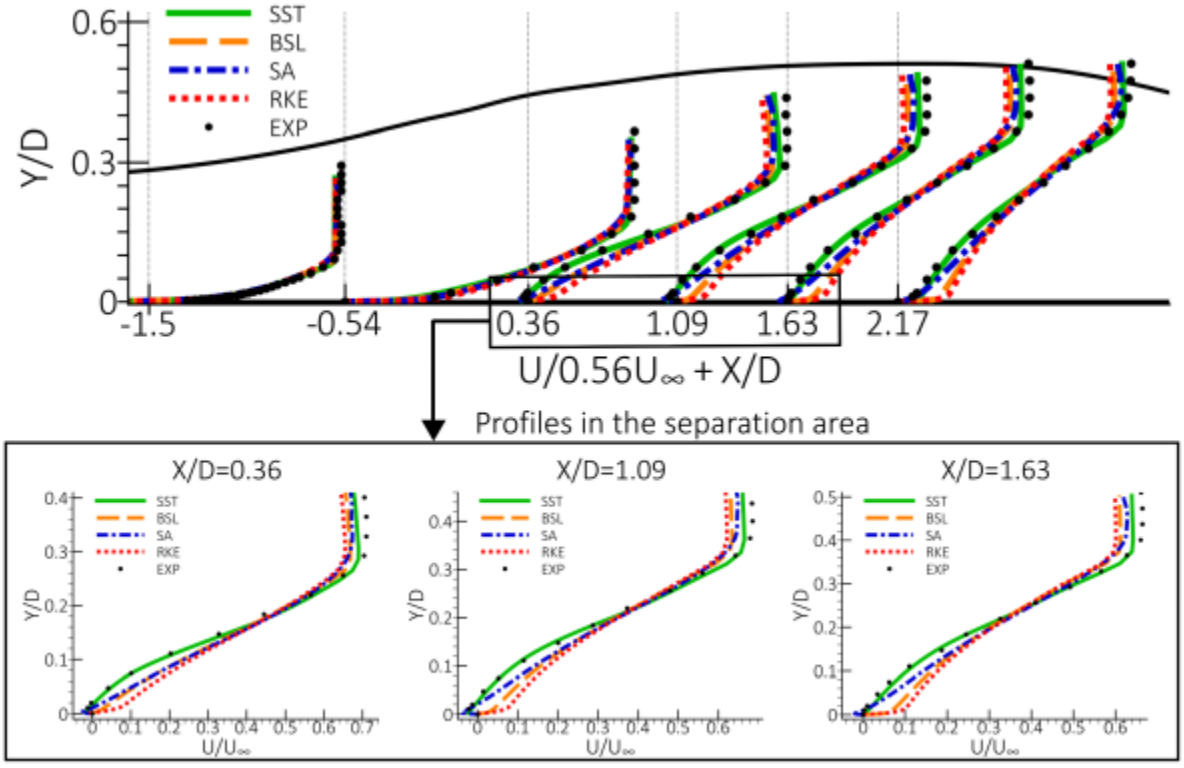
Figure 12.73: Comparison of velocity profiles predicted with different turbulence models for CS0 diffuser [30] (p. 296) Figure 13 shows a comparison of velocity profiles for different turbulence models at different locations in the diffuser compared to experimental data. The curves clearly indicate that the $k-\varepsilon$ model family shows a strong tendency to under-predict the effect of the adverse pressure gradient and the onset/amount of separation $0<X/D<2$. The SA is capable to reproduce negative wall shear stress, however the model underpredicts the height of the separation bubble.

Note that the tuned GEKO model $(C_{SEP}=1.0, C_{NW}=1.0)$ is an exact transformation of the standard $k-\varepsilon$ models and therefore reproduces the $k-\varepsilon$ model behavior which is shown in Figure 14. It is also interesting to note that even when activating RSM (Reynolds Stress Model) model in combination with the ε-equation, the separation is still not predicted properly. Figure Figure 15 shows the wall shear-stress and the wall-pressure coefficients C_f and C_p for the same models. The results confirm the conclusions reached from the velocity profiles.

It is interesting to note that $k-\varepsilon$ family models predict firmly attached velocity profiles, even though the wall shear stress approaches zero at around $X/D=0$. This is the result of the two-layer formulation (detailed described in section 9.3.4) which allows for a very thin backflow region near the wall, even though the overall profile is attached. This is also consistent with the C_p distribution, which shows a lack of flow displacement for the $k-\varepsilon$ models. Figure 16 shows the difference of the flow structure for the $k-\omega$ based and $k-\varepsilon$ based models.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.73: Comparison of velocity profiles predicted with different turbulence models for CS0 diffuser [30 (p. 371)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.74: Comparison of velocity profiles for k-ω model family and GEKO models for CS0 diffuser [30 (p. 371)].

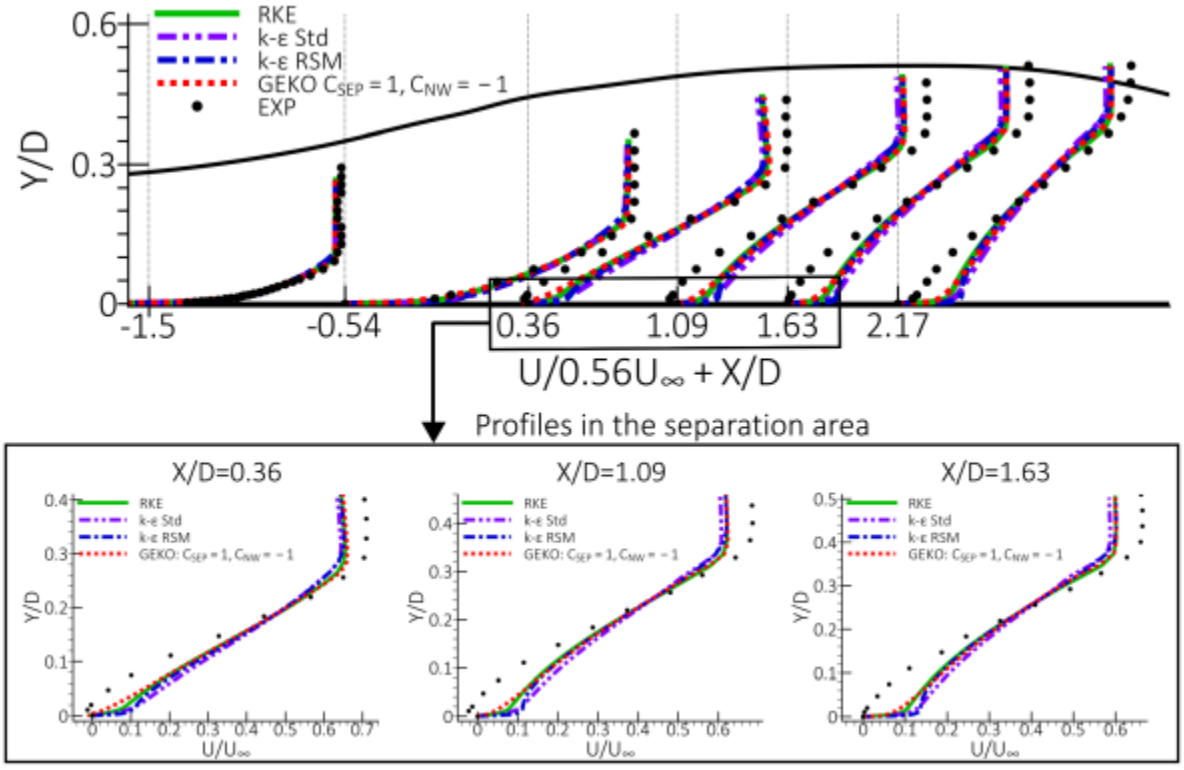


Figure 12.75: Prediction of wall characteristics for k-ω family and GEKO models for CS0 diffuser [30 (p. 371)]. Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p.

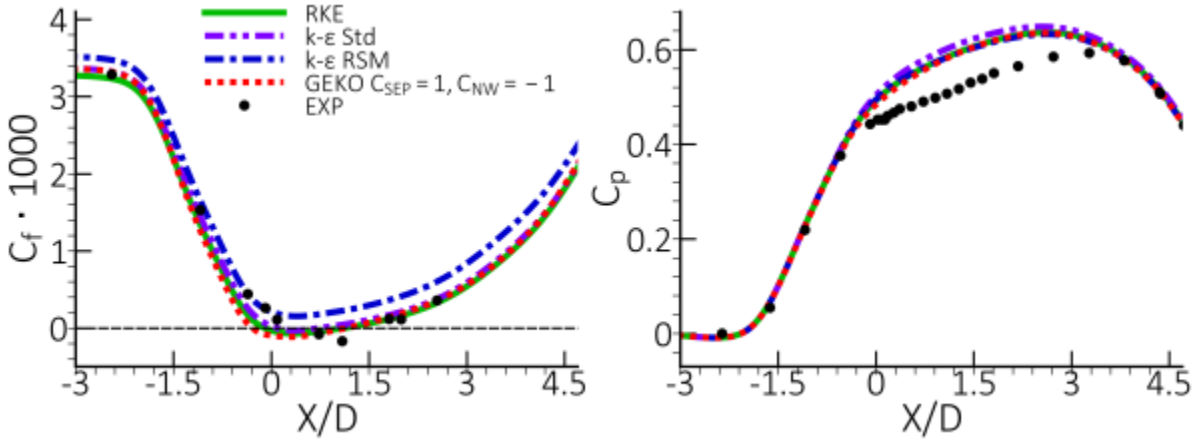
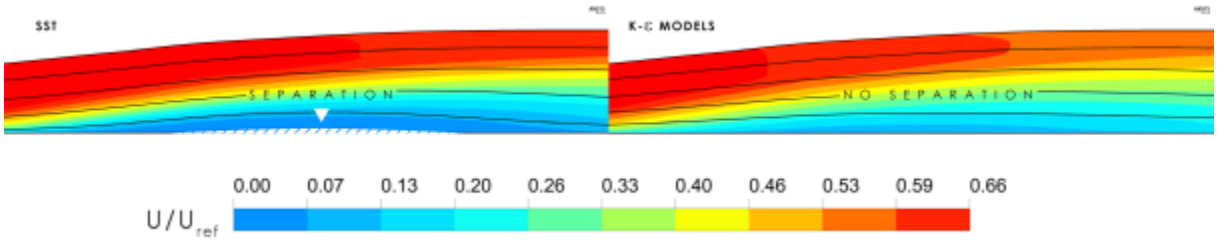


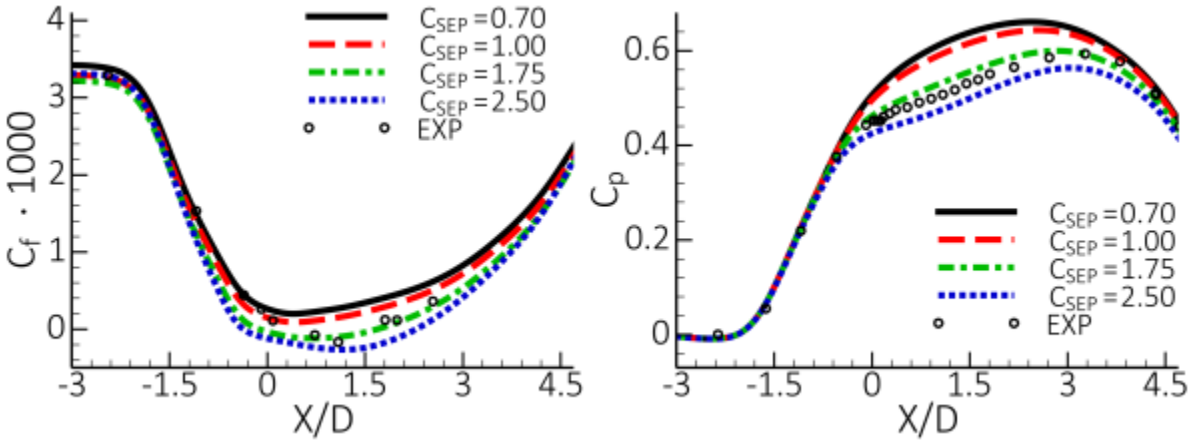
Figure 12.76: Flow structured visualized with the streamwise velocity field and streamlines (black lines) for CS0 diffuser [30 (p. 371)].



GEKO Model Versatility

Variations in the CSEP coefficient for the GEKO turbulence model are shown in Figure 12.77: Figure 17: Impact of variation in C_{SEP} coefficient of the GEKO model for CS0 diffuser flow [30]. Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p. (p. 298) - Figure 12.80: Figure 20: Impact of variation in C_{SEP} coefficient of the GEKO model on turbulence shear stress profiles for CS0 diffuser flow [30]. (p. 299). As expected, with increasing of C_{SEP}, the model becomes more sensitive to the adverse pressure gradient in the diffuser and improves its separation predict up to a value of C_{SEP} = 2.0. Higher values of C_{SEP} lead to over-separation. It is the desired behavior of the GEKO model to allow for a wide range of calibration engulfing the experimental solution.

Figure 12.77: Figure 17: Impact of variation in C_{SEP} coefficient of the GEKO model for CS0 diffuser flow [30 (p. 371)]. Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.78: Figure 18: Impact of variation in C_{SEP} coefficient of the GEKO model on velocity profiles for CS0 diffuser flow [30 (p. 371)].

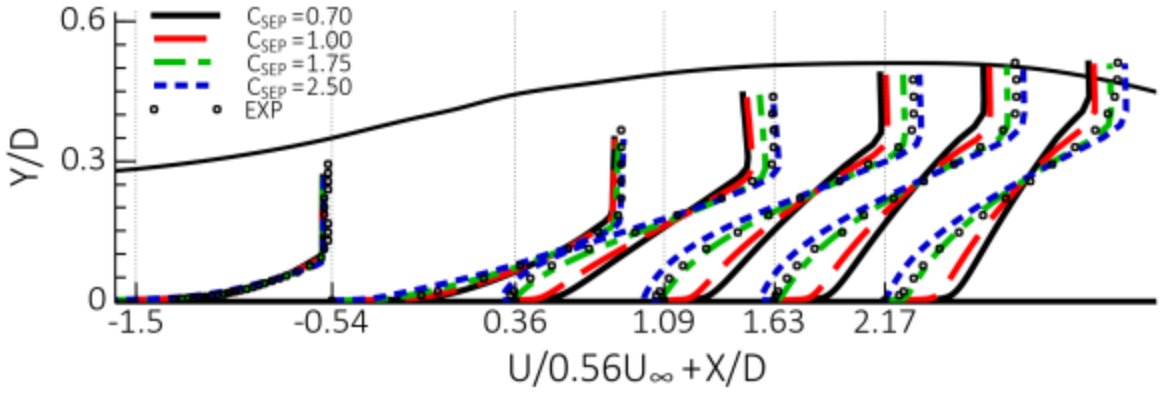


Figure 12.79: Figure 19: Impact of variation in C_{SEP} coefficient of the GEKO model on turbulence kinetic energy profiles for CS0 diffuser flow [30 (p. 371)].

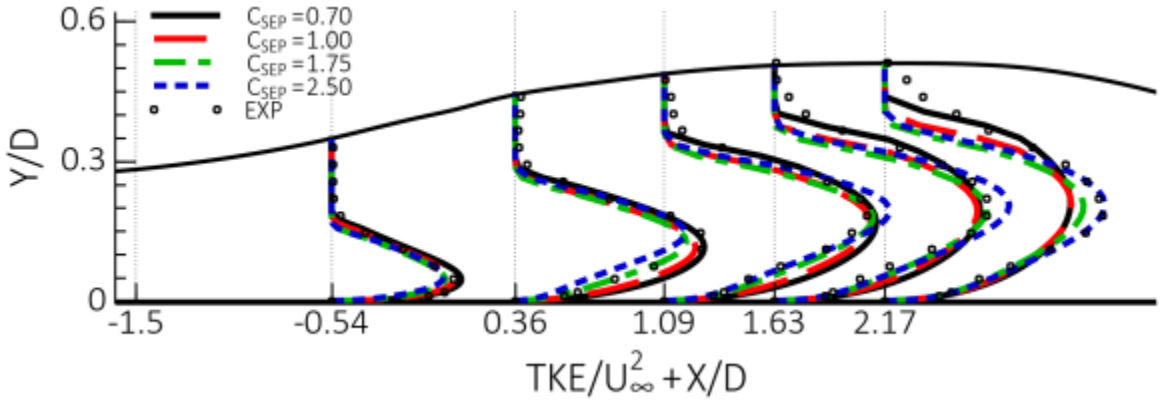
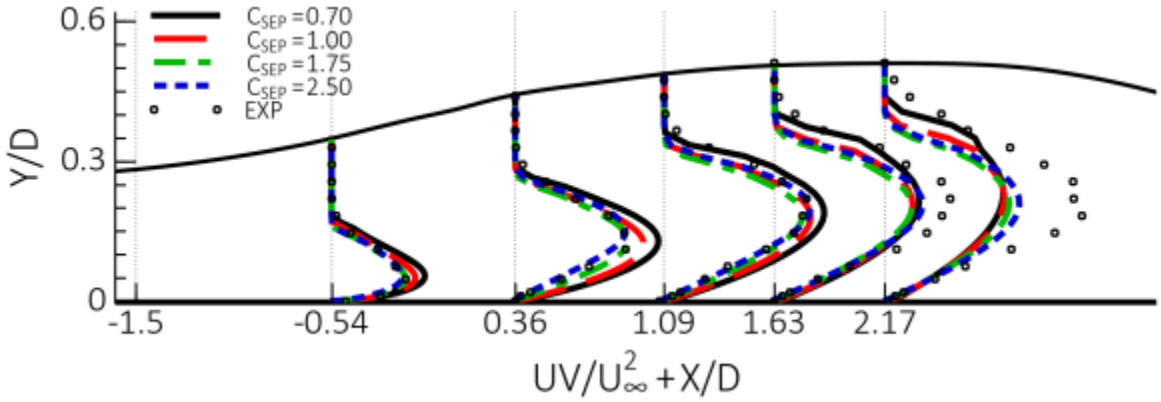


Figure 12.80: Figure 20: Impact of variation in C_{SEP} coefficient of the GEKO model on turbulence shear stress profiles for CS0 diffuser flow [30 (p. 371)].



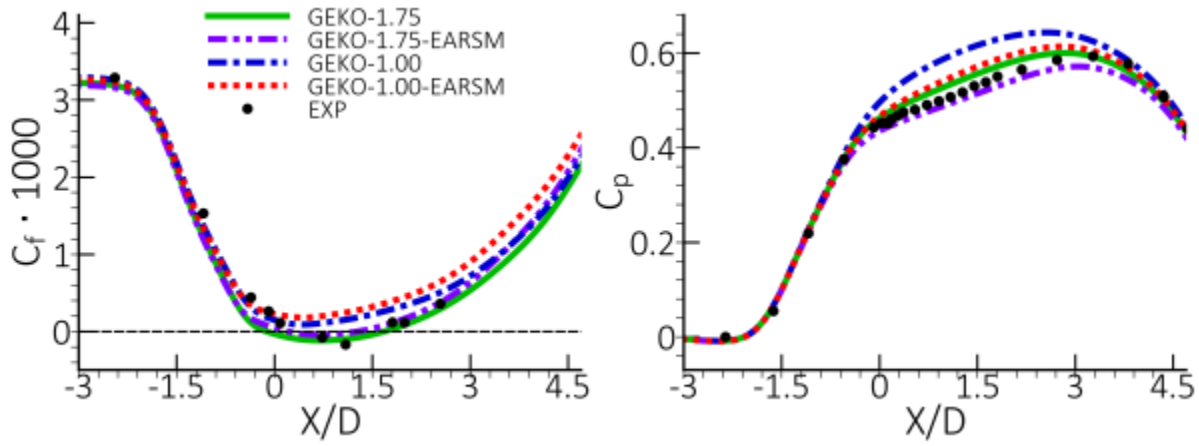
Non-linear effects

Figure 12.81: Figure 21: Impact of non-linear terms for the GEKO model for CS0 diffuser flow (30). Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p. (p. 300) Figure 21 shows

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the results of the GEKO model with (EARSM) and without non-linear terms. Here GEKO-1.00/1.75 corresponds to different values of C_{SEP} = 1.00/1.75. Considering turbulence anisotropy of the Reynolds stress tensor has negligible effect on the prediction of the skin friction coefficient, however the prediction of the pressure distribution is very sensitive. The flattening of the CP-distribution indicates that the flow is more separated when including the EARSM terms. This is expected as the EARSM model has a similar effect as the SST limiter, by accounting for the transport of the principal shear stress and thereby reducing the increase in the shear stress level as separation is approached. The effect of including the EARSM formulation is therefore like an increase in C_{SEP} for the standard GEKO model.

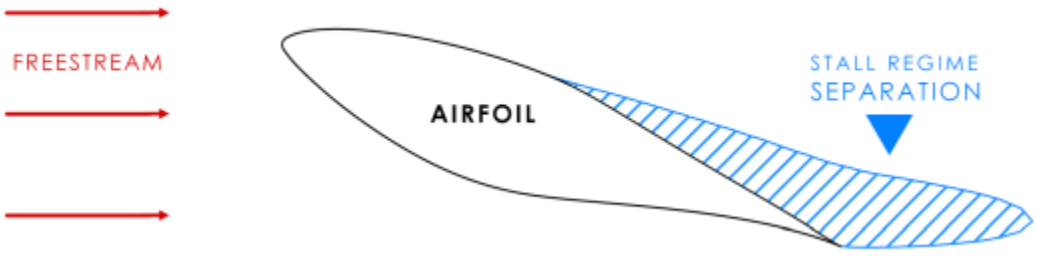
Figure 12.81: Figure 21: Impact of non-linear terms for the GEKO model for CS0 diffuser flow (30 (p. 371)). Left: wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p.



12.2.3.2.2. Airfoil Flows

The accurate prediction of airfoil characteristics especially in regimes near stall where the flow is separated and maximal lift is achieved, is an important task for aviation and wind power, as well as for turbomachinery flows. Figure 12.82: Figure 22: Schematic of the flow around an airfoil for stall regime. (p. 300) shows the schematic for such a situation, where the separation zone on the suction side is significant and plays a key role in the prediction of the airfoil characteristics.

Figure 12.82: Figure 22: Schematic of the flow around an airfoil for stall regime.



Six aerodynamic airfoils with different shapes and thicknesses (from 13% to 30%) are considered [31 (p. 371)]–[35 (p. 372)]. Experimental investigations were carried out in low turbulence rectangular wind tunnels (Tu < 1%) at relatively high Reynolds numbers (Re > 10^6) based on airfoil chord C and freestream velocity. The airfoil shapes and more detailed information about the experiments

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

are shown in the Table 12.11: Table 2: Considered airfoils and parameters of wind tunnel and flow in the experiments (p. 301). For all the airfoils, except DU-97-W-300, the boundary layer was tripped with the use of a short rough tape placed at the leading edge. It is therefore assumed that the flow around these airfoils is fully turbulent and that only for the "clean" DU-97-300 airfoil a model for laminar turbulent transition should be considered. For this airfoil, the additional intermittency equation [36 (p. 372)] is therefore solved together with the k and ω-equations. Since the experimental Mach number did not exceed 0.15, incompressible flow is assumed in all cases.

The computations are carried out with inclusion of the wind tunnel walls. Inlet and outlet boundaries are located 10C upstream of the leading edge and downstream of the trailing edge airfoil, respectively. A constant velocity is specified at the inlet section of the computational domain. No-slip conditions are used on the airfoil boundary and constant pressure is specified at the outlet. Symmetry boundary condition are specified on the wind tunnel upper and lower walls for imitation of slip-walls. The inlet turbulence kinetic energy matches the experimental turbulence intensity and the specific dissipation rate is specified as $\omega = 10 \cdot U_\infty / C$.

Table 12.11: Table 2: Considered airfoils and parameters of wind tunnel and flow in the experiments

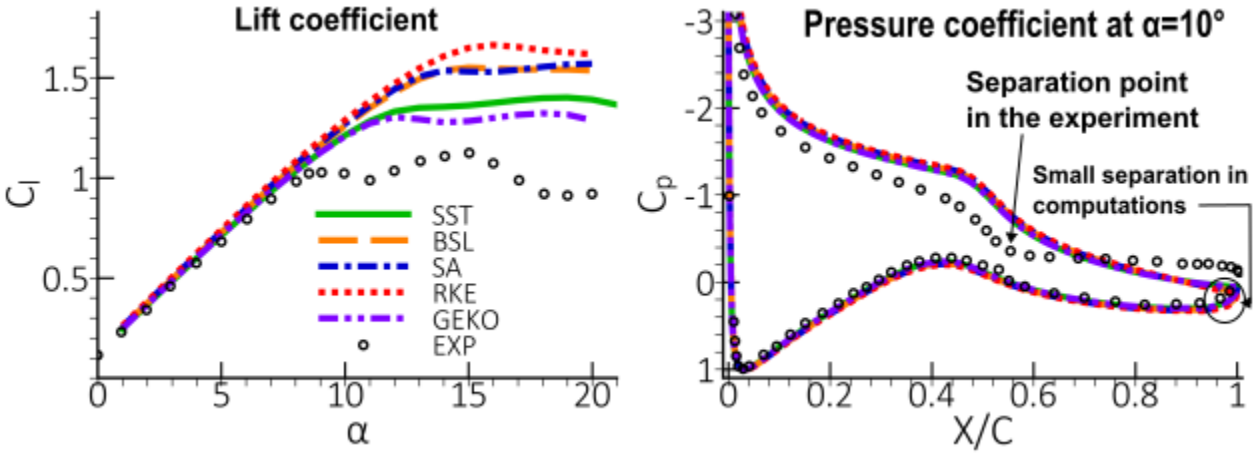
Airfoil	Thickness	Surface	Wind Tunnel Height	$Re/10^6$
S805	13.5%	Tripped	3.60C	1.00
S825	17.10%	Tripped	5.00C	2.00
S809	21.00%	Tripped	3.00C	2.00
S814	24.00%	Tripped	2.76C	1.50
DU-97-W-180	18.00%	Tripped	3.00C	3.00
DU-97-W-300	30.00%	Clean	3.00C	3.00
NACA-4412	12.00%	Tripped	2.73C	1.64

Model Comparison

For flows around airfoils, the maximum lift coefficient and corresponding angle of attack are systematically overpredicted by the Reynolds Averaged Navier-Stokes (RANS) approach in combination with standard RANS turbulence models. For example, the predicted lift coefficient for the different turbulence models for the S809 airfoil is shown in Figure 12.83: Prediction of lift coefficient in wide range of angle of attacks for different turbulence models for S809 airfoil at $Re = 2 \cdot 10^6$ [33] (p. 302). The disagreement between computations and experimental data is caused by delay of turbulent boundary layer separation under adverse pressure gradient on the suction side of the airfoils which is shown in Figure 12.83: Prediction of lift coefficient in wide range of angle of attacks for different turbulence models for S809 airfoil at $Re = 2 \cdot 10^6$ [33] (p. 302). Even the most aggressive SST and GEKO (default conditions) models predict separation too late (at $X/C \approx 0.95$) while in the experiment the separation on the suction side occurs near the mid-chord (at $X/C \approx 0.5$). Since the separation position is controlled by the turbulence model, one of the ways to improve the accuracy of the airfoil characteristics prediction is a special tuning of the models for this class of flows.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.83: Prediction of lift coefficient in wide range of angle of attacks for different turbulence models for S809 airfoil at Re=$2 \cdot 10^6$ [33 (p. 371)]



GEKO Model Tuning

The simplest way of model turning without violating the basic flows calibration is to increase the C_{SEP} coefficient of the GEKO model. This coefficient (default value is C_{SEP} =1.75) affects the size of separation bubble: for higher C_{SEP} values the separation is larger. Figure 12.84: Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 [35] (Left) and DU-97-W-300 [35] (Right) airfoil at Re=$3 \cdot 10^6$ (p. 303) – Figure 12.88: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for α=18° (Right) with GEKO model for S814 airfoil at Re=$1.5 \cdot 10^6$ [34] (p. 305) compare lift curves for 1.75 ≤C_{SEP}≤ 2.5 for all the considered airfoils. In these comparisons, it should be kept in mind that 2D simulations are not correct in the post-stall region, due to the formation of 3D structures in the experiments [37 (p. 372)]. Despite this, it is possible to adjust the GEKO model for the prediction of such flows for angles of attack up to stall using 2D simulations. The GEKO-2.50 (GEKO with C_{SEP} = 2.5) model predicts earlier separation on the suction side of the airfoils than other model variants, which improves agreement of the predicted both integral (lift coefficient) and local (pressure coefficient) airfoil characteristics with the experimental data near stall. However, the GEKO-2.50 model underpredicts the lift coefficient for low angles of attack for the thickest DU-97-W-300 and S814 airfoils (30% thickness) which is undesirable. The optimal C_{SEP} value is therefore between C_{SEP}=2.00-2.50.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.84: Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 [35 (p. 372)] (Left) and DU-97-W-300 [35 (p. 372)] (Right) airfoil at Re=$3 \cdot 10^6$

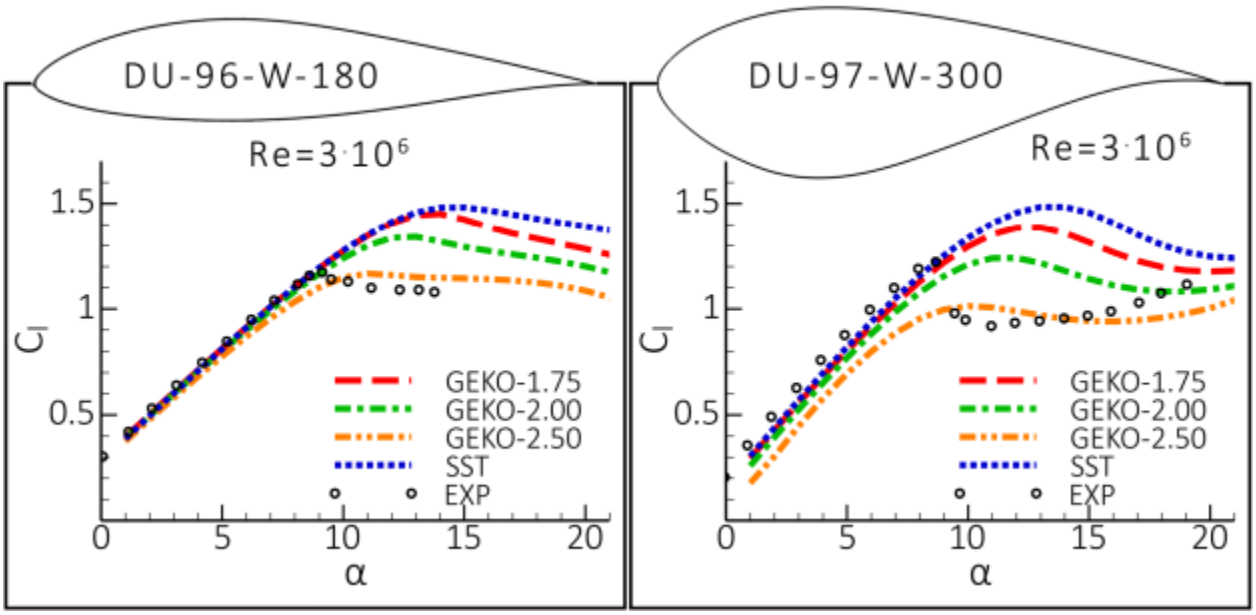
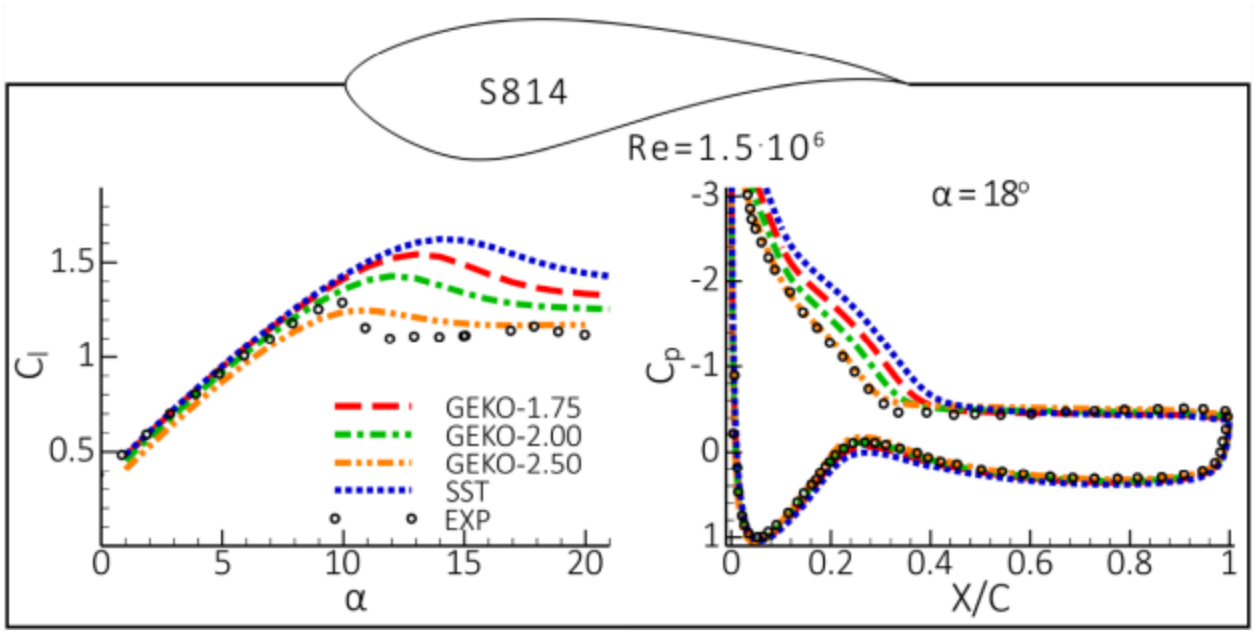


Figure 12.85: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha = 12°$ (Right) with GEKO model for S805 airfoil at Re=$1 \cdot 10^6$ [31 (p. 371)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.86: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=17°$ (Right) with GEKO model for S825 airfoil at Re=$2\cdot10^6$ [32 (p. 371)]

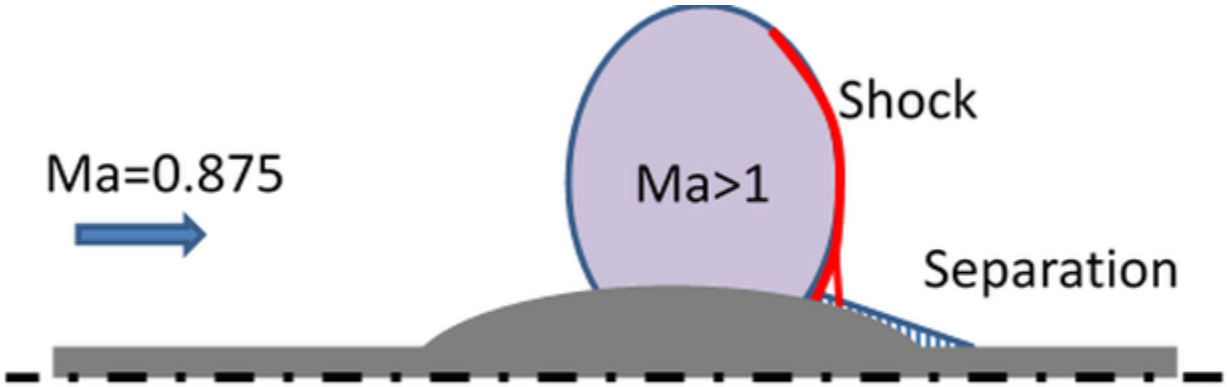
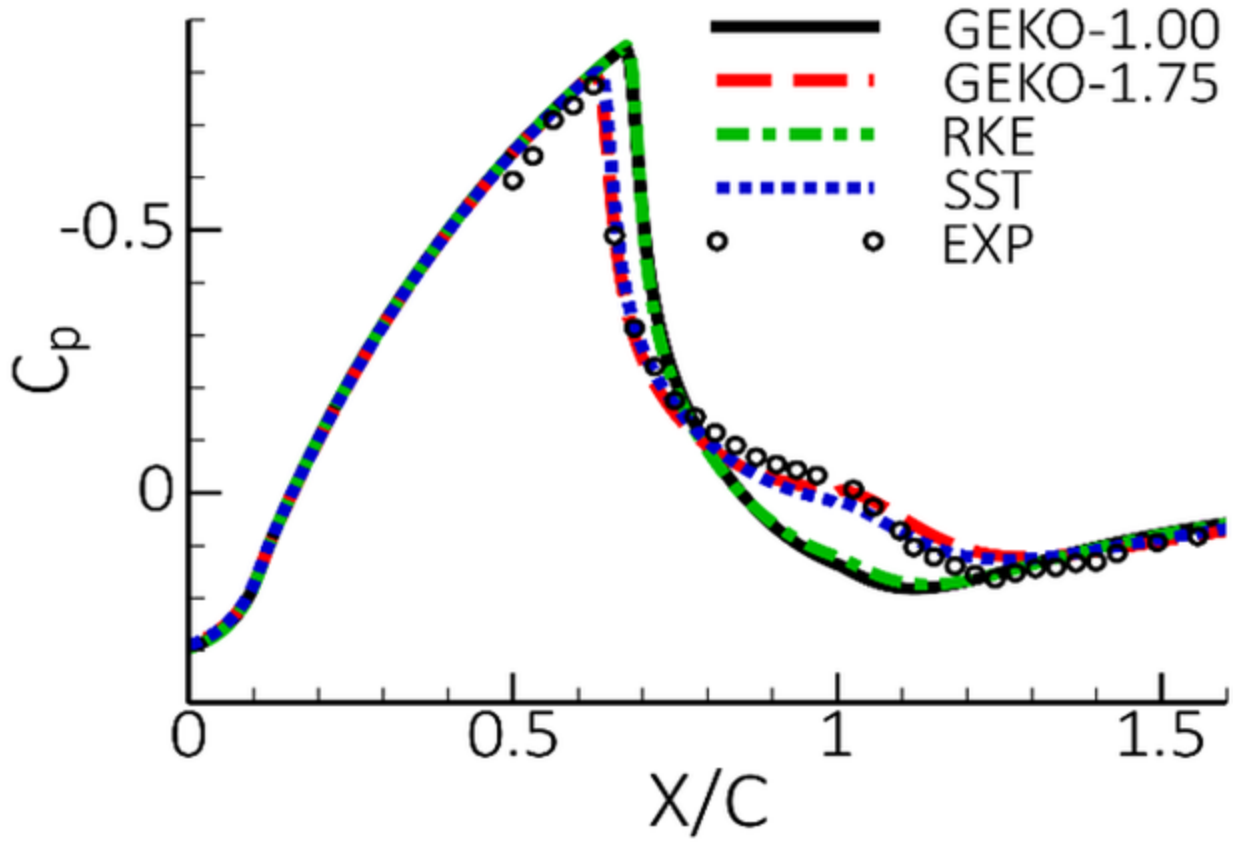
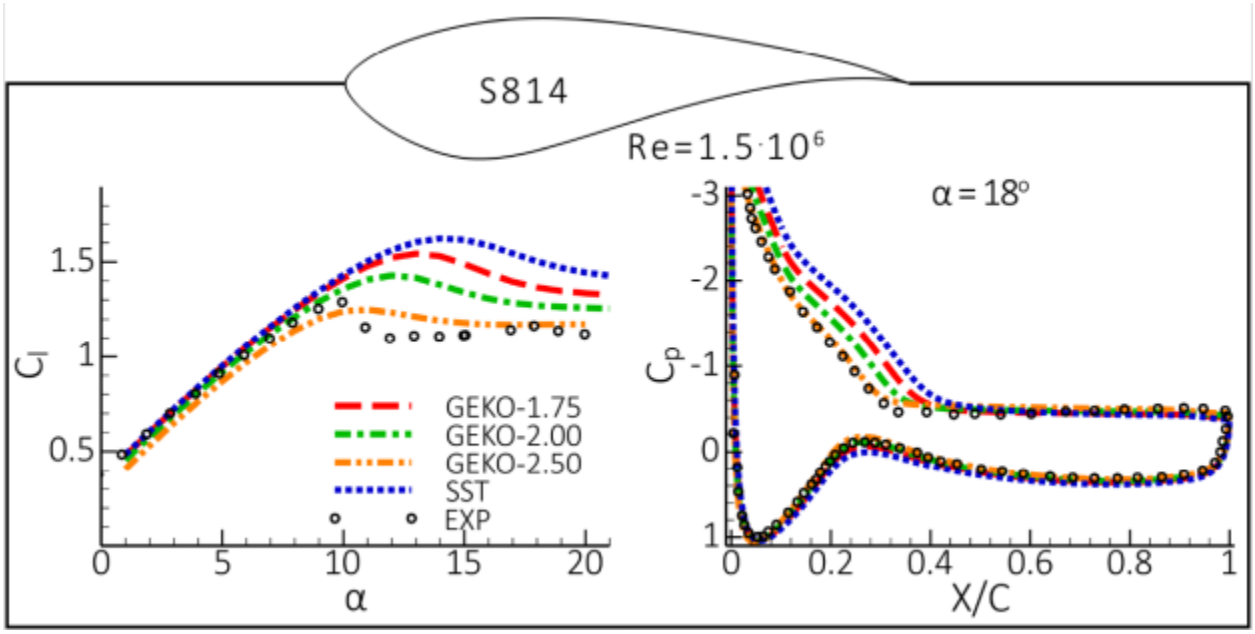


Figure 12.87: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=10°$ (Right) with GEKO model for S809 airfoil at Re=$2\cdot10^6$ [33 (p. 371)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

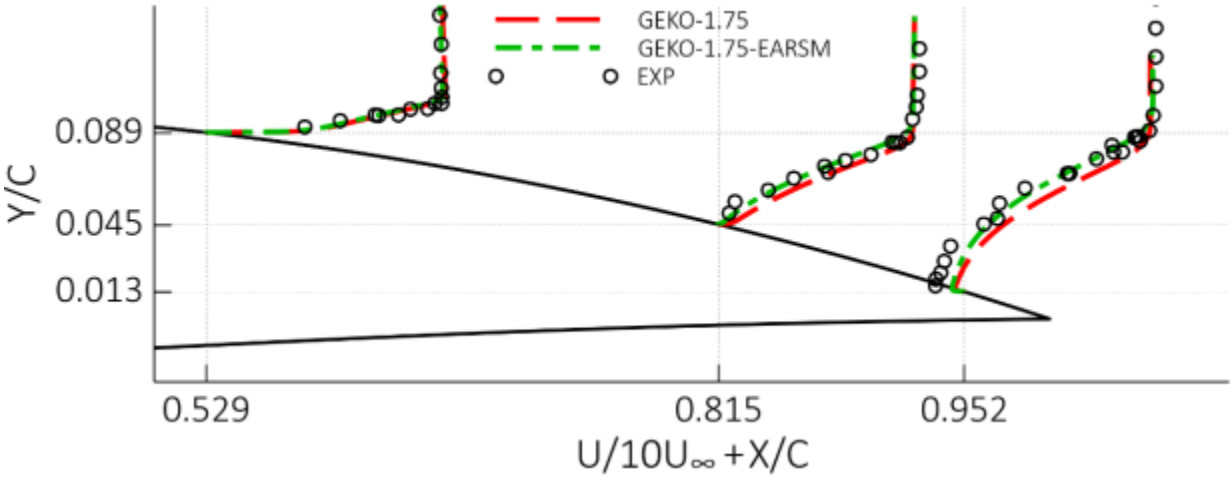
Figure 12.88: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for α=18° (Right) with GEKO model for S814 airfoil at Re=$1.5 \cdot 10^6$ [34 (p. 372)]



Non-linear effects

An example of considering non-linear effects for the GEKO model using an EARSM model is presented next. Figure 12.89: NACA-4412 airfoil near the trailing edge with streamwise velocity profiles at measurement locations predicted with linear and non-linear GEKO-1.75 model compared with the experimental data [38] (p. 305) shows the effect of the non-linear WJ-EARSM terms for the NACA-4412 airfoil for the GEKO-1.75 (C_{SEP} = 1.75) model. The use of the non-linear terms slightly increases separation on the suction side of the airfoils. However, the effect is much weaker than the model tuning with the C_{SEP} coefficient shown above.

Figure 12.89: NACA-4412 airfoil near the trailing edge with streamwise velocity profiles at measurement locations predicted with linear and non-linear GEKO-1.75 model compared with the experimental data [38 (p. 372)]



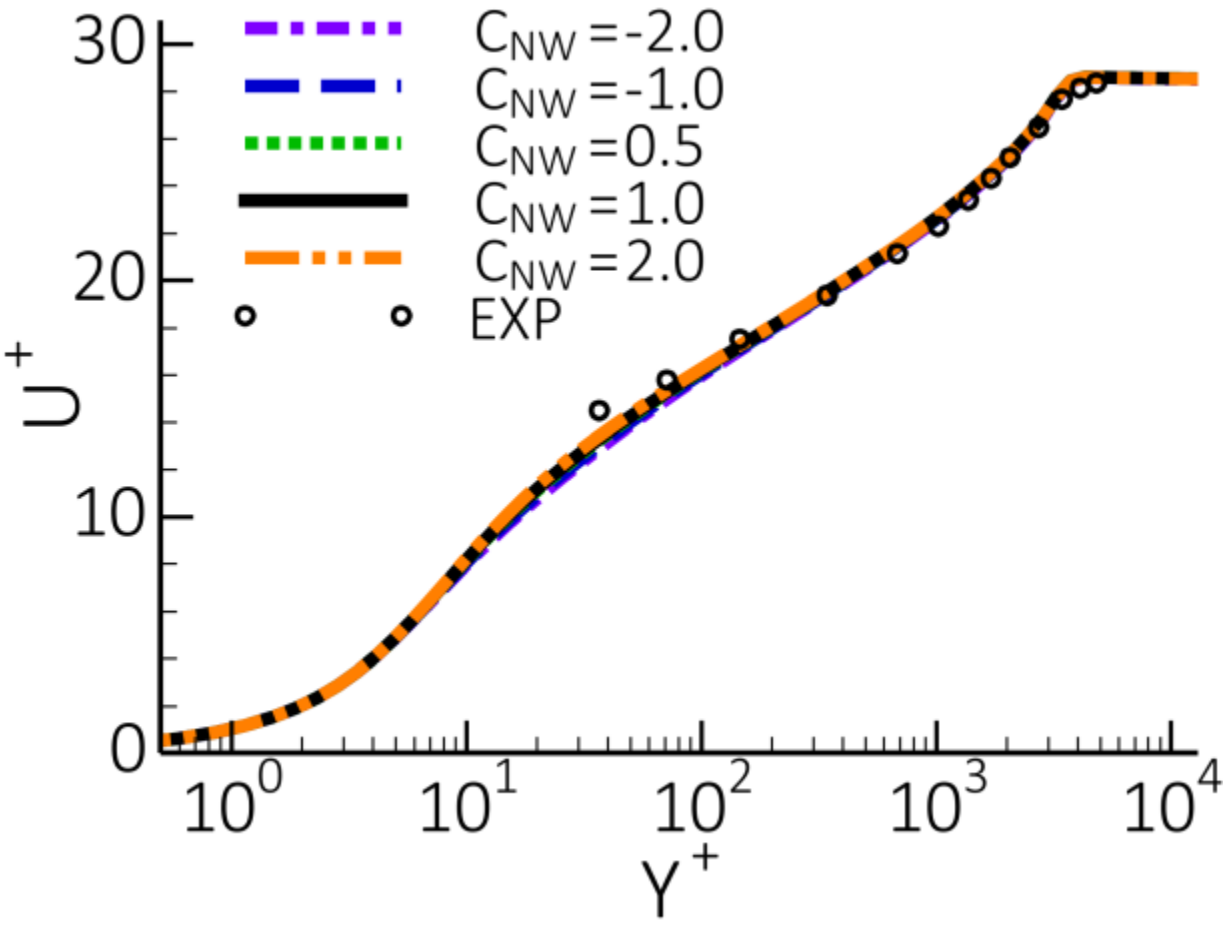
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.3.2.3. Transonic Bump Flow

Transonic compressible flow past an axisymmetric bump was simulated at conditions corresponding to the experiments of Bachalo & Johnson [39 (p. 372)]. The bump was placed over a cylindrical pipe, and the entire geometry is axisymmetric. The incoming Reynolds number based on the bump chord length was Re = 2.763·106. The dynamic viscosity was assumed to adhere to Sutherland's law and the Prandtl number was set to Pr = 0.71.

The NASA Bump flow features a subsonic inflow with Ma=0.875. The flow is then accelerated to supersonic speed over the bump and then reverts to subsonic speed through a shock wave. The shock causes the boundary layer behind the shock to separate, which in turn interacts with the shock by pushing it forward. The ability to predict the shock location is therefore directly linked to a model's ability to predict boundary layer separation.

Figure 12.90: Scheme of the transonic bump flow

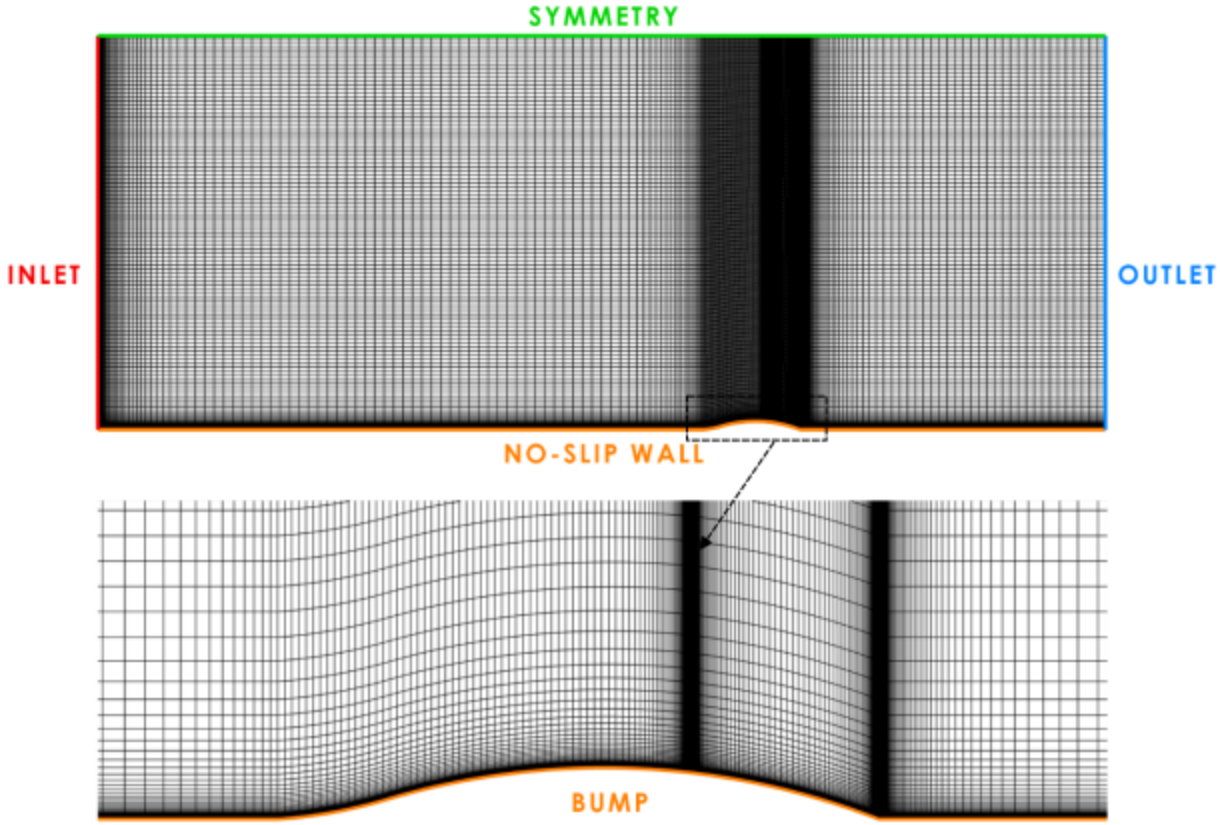


The size of the computational domain as illustrated in Figure 12.91: Computational domain with boundary conditions and mesh for the NASA Transonic Bump [39] (p. 307) was about 12C by 4C (C is the bump length). The inlet boundary is located 6.89C upstream of the bump. Constant total pressure of P_{tot} = 106595 [Pa] and total temperature of T_{tot} = 322 [K] are assigned at the inlet section of the computational domain. Turbulence characteristics over the inlet are also assumed constant with values corresponding to a free-stream turbulence intensity of Tu=0.1%, and an eddy-viscosity ratio of TVR=1. A no-slip condition is used on the wall and a constant pressure is specified at the outlet boundary. The computational mesh shown in Figure 12.91: Computational

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

domain with boundary conditions and mesh for the NASA Transonic Bump [39] (p. 307) is refined in streamwise direction over the bump for proper resolution of the shock wave.

Figure 12.91: Computational domain with boundary conditions and mesh for the NASA Transonic Bump [39 (p. 372)]



As expected, again, the models form two groups with GEKO-1.75/SST and GEKO-1.00/RKE as seen from Figure 12.92: Comparison of wall pressure coefficient, C_p for the transonic bump flow [39]. The bump starts at X/C=0 location. Left: comparison of the different turbulence models, right: versatility of the GEKO model with the C_{SEP} variation (GEKO-1.75: C_{SEP}=1.75) (p. 308). Only the GEKO-1.75/SST model can predict the shock location and the post-shock separation zone properly. The GEKO-1.00/RKE models fail due to their lack of separation sensitivity. Like for the other described cases, the non-linear terms show less influence on the size of separation zone than the choice of the turbulence model, Figure 12.93: Comparison of different C_{SEP} settings and ERSM inclusion for the transonic bump flow [39]. Left: wall shear stress coefficient, C_f, right: wall pressure stress coefficient C_p (p. 308).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.92: Comparison of wall pressure coefficient, C_p for the transonic bump flow [39 (p. 372)]. The bump starts at X/C=0 location. Left: comparison of the different turbulence models, right: versatility of the GEKO model with the C_{SEP} variation (GEKO-1.75: C_{SEP}=1.75)

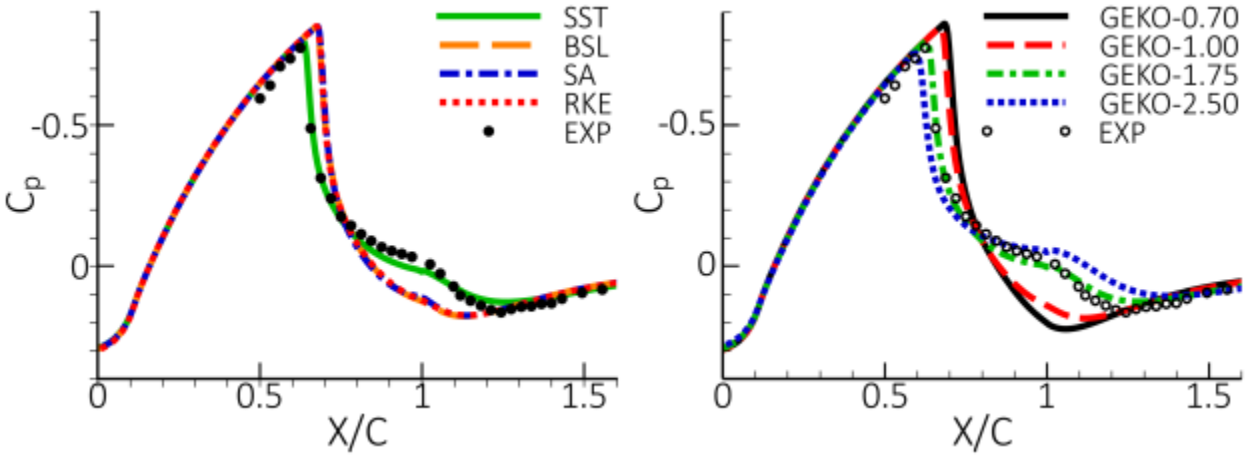
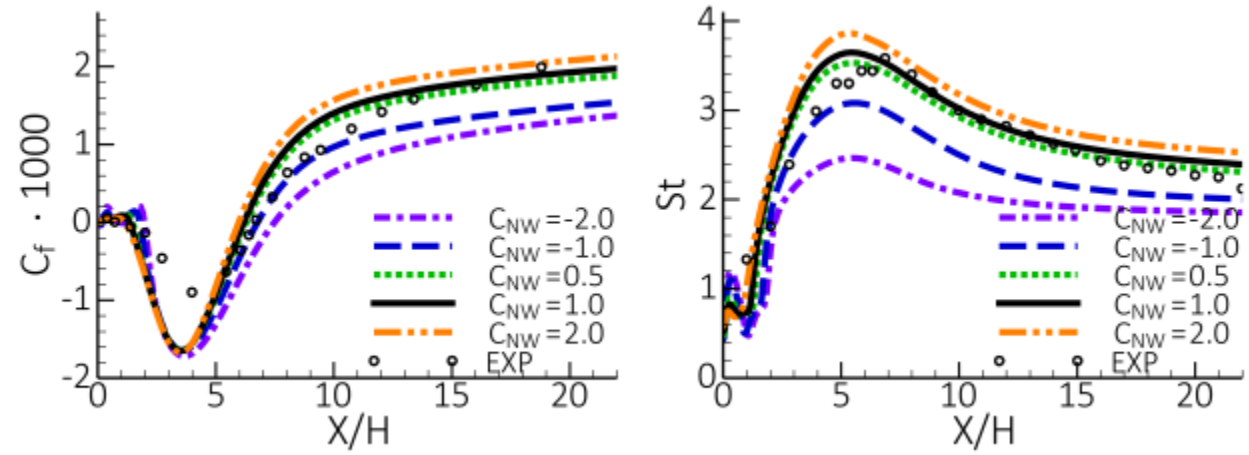


Figure 12.93: Comparison of different C_{SEP} settings and ERSM inclusion for the transonic bump flow [39 (p. 372)]. Left: wall shear stress coefficient, C_f, right: wall pressure stress coefficient C_p



12.2.3.3. Corner Flows

As described in section 3.4.3, eddy-viscosity models need special corrections for the prediction of the secondary flows in corners. In this section, the effect of the use of CFC (Corner flow Correction) [22 (p. 370)], [23 (p. 370)] and EARSM terms from the Wallin-Johansson [19 (p. 369)] model (see section 3.3.3) in combination with different k-ω models is shown for flows in rectangular channels and diffusers. For these flows, only the inclusion of the non-linear terms allows to predict the secondary flows in the corners.

Thus, the main goal of this section is to demonstrate the improvement in the prediction of the secondary motion in corners with the use of the non-linear turbulence models. The non-linear effects are shown for the GEKO model in combination the Corner Flow Correction (CFC) and EARSM. The effect of these terms on other models like the SST model would be similar.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

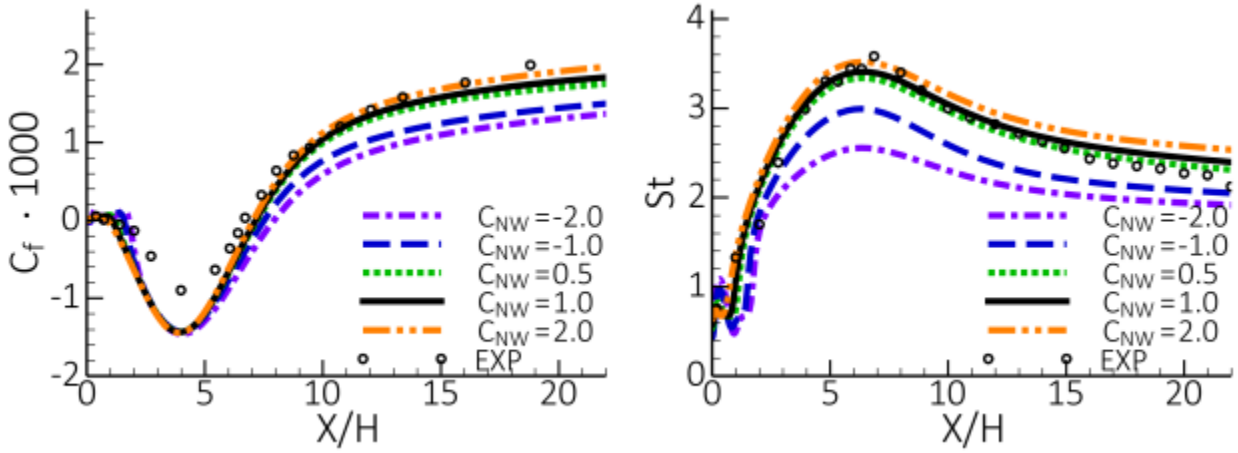
The following topics are discussed:

12.2.3.3.1. Developed Flow in Square Duct
12.2.3.3.2. Flow in Rectangular Diffusers
12.2.3.3.3. Flow around DLR F6 aircraft
12.2.3.3.4. Conclusions

12.2.3.3.1. Developed Flow in Square Duct

The simplest case demonstrating the impact of the secondary motion on the main flow is the developed flow in square duct. For this flow, the linear eddy-viscosity models are not capable to predict secondary motion into the corners. This effect is shown in Figure 12.94: Flow topology in the square duct case predicted using the GEKO model with CFC (left) and without CFC (right). Blue lines: iso-surfaces of the streamwise velocity, red lines in the corners: streamlines of the secondary motion (p. 309).

Figure 12.94: Flow topology in the square duct case predicted using the GEKO model with CFC (left) and without CFC (right). Blue lines: iso-surfaces of the streamwise velocity, red lines in the corners: streamlines of the secondary motion

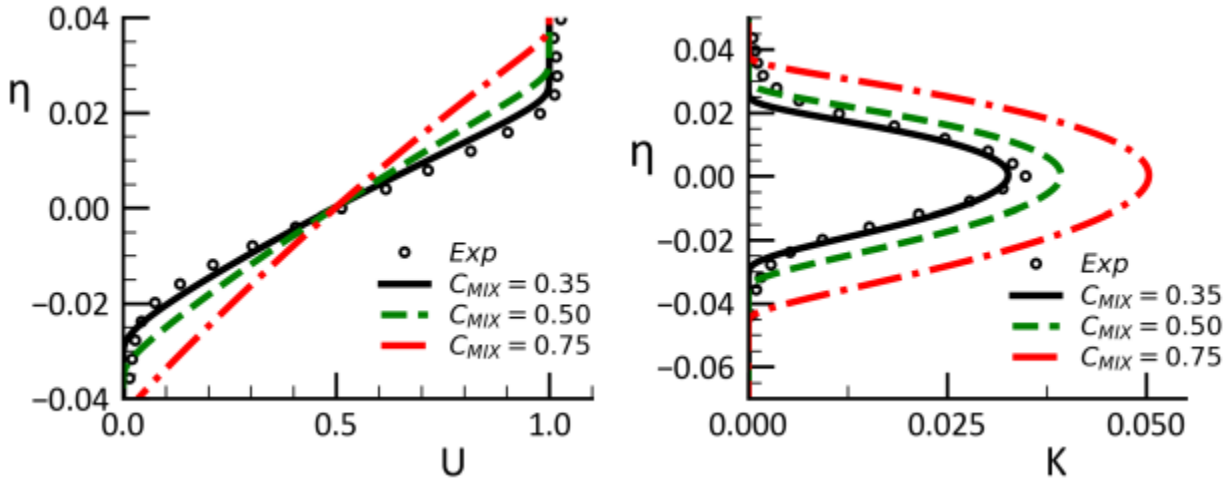


The computations are performed with the linear and non-linear GEKO-CFC model under the conditions of the reference DNS data [40 (p. 372)], namely at a Reynolds number based on the averaged friction velocity u_τ and the channel width, H, equal to Re_τ =1200. The computations were carried out in a "2.5D mode", which include momentum equations for all the 3 velocity components but assumed that their streamwise derivatives are zero. A constant streamwise pressure gradient is specified to achieve the DNS Reynolds number. At solid walls, no-slip boundary conditions are imposed.

The streamlines in the cross-section show that the non-linear GEKO model can predict secondary flow while the linear GEKO model cannot (see Figure 12.94: Flow topology in the square duct case predicted using the GEKO model with CFC (left) and without CFC (right). Blue lines: iso-surfaces of the streamwise velocity, red lines in the corners: streamlines of the secondary motion (p. 309)). The flow structure of GEKO-1.00, GEKO-1.75 and WJ-BSL-EARSM model is virtually the same (not shown). Streamwise and lateral velocity profiles along a diagonal line (X=Y) predicted by the GEKO-EARSM models are in reasonable agreement with the DNS data (see Figure 12.95: Streamwise (Left) and lateral (Right) velocity profiles on the diagonal of the square duct case [40] (p. 310)).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.95: Streamwise (Left) and lateral (Right) velocity profiles on the diagonal of the square duct case [40 (p. 372)]



12.2.3.3.2. Flow in Rectangular Diffusers

Flow in rectangular diffusers is a much more challenging test case than the fully developed channel flow, since it involves the effect of turbulence anisotropy combined with an adverse pressure gradient causing separation of the flow. Two diffusers are considered: the symmetric DLR diffuser [41 (p. 373)] and the asymmetric Stanford diffuser [42 (p. 373)], [43 (p. 373)] where the flow is highly sensitive to the secondary flow prediction.

DLR-Diffuser

Flow is modeled based on the experimental study in the framework of DLR project $VicToria$ to investigate 3D separated flow, which occur at end wall junctions. The diffuser configuration with the expansion ratio of ER = 2.0 was considered in the CFD study is shown in Figure Figure 36. Computational domain together with boundary conditions is shown in Figure 12.97: Computational domain with the boundary conditions for the DLR diffuser [41] (p. 311). The inlet condition is unform flow with U=10m/s Tu=1% and TVR=1. A no-slip condition is used at the wall and a constant pressure is specified at the outlet boundary. The origin the of coordinate system is located on the centerline of the flat wall coincident with the start of the ramp. The X-axis is aligned with the streamwise direction, the Y-axis defines the spanwise direction while the Z-axis coincides with the expansion direction of the channel. Reynolds number based on the inlet bulk velocity and cross-section of internal width h is Re = $5.1 \cdot 10^4$

For this case, the linear models predict two recirculation zones starting from the inclined-side wall corner as shown in Figure 12.98: The surface streamlines near the inclined and side walls plotted for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41]. (p. 312) – Figure 12.99: Flow topology for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41] (p. 312). In contrast, the secondary flow predicted with non-linear model prevents corner separation and pushes the recirculation zone to the inclined wall, while flow on the side wall is virtually attached. The dramatic difference in flow topology, visualized by the wall-streamlines on the inclined wall with the linear and non-linear GEKO model is shown in Figure 12.98: The surface streamlines near the inclined and side walls plotted for the linear (left) and non-linear GEKO model with corner flow correction (right)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

for the DLR diffuser [41]. (p. 312)– Figure 12.99: Flow topology for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41] (p. 312) .

Figure 12.96: DLR diffuser configuration [41 (p. 373)]

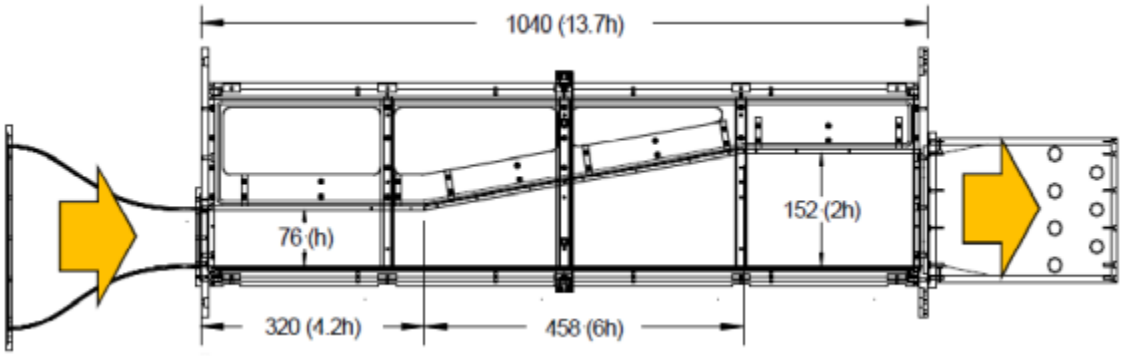
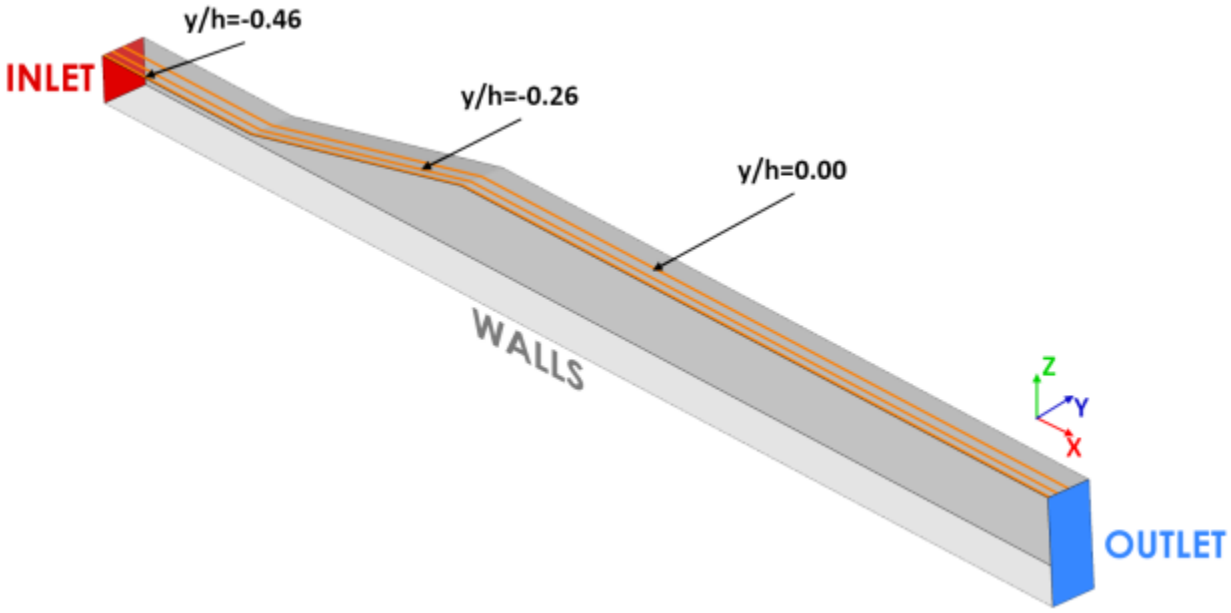


Figure 12.97: Computational domain with the boundary conditions for the DLR diffuser [41 (p. 373)]



A comparison of the predicted skin friction coefficient with the experimental data (Figure 12.100: Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for turbulence models with and without non-linear terms for the DLR diffuser [41] (p. 312)) shows that the use of the non-linear models improves the prediction of the flow near the corner (y/h=0.46) where the linear models predict large corner separations. In contrast, the flow near the midsection (y/h=0) of the diffuser in not so sensitive to the non-linear effects. However, all the models predict an attached boundary layer at the midsection, while in the experiment the flow separates. Despite the different flow topologies for linear and non-linear models, the pressure distribution at the midsection (Figure 12.101: Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the DLR diffuser [41] (p. 313)) agrees with the experimental data for all the models.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.98: The surface streamlines near the inclined and side walls plotted for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41 (p. 373)].

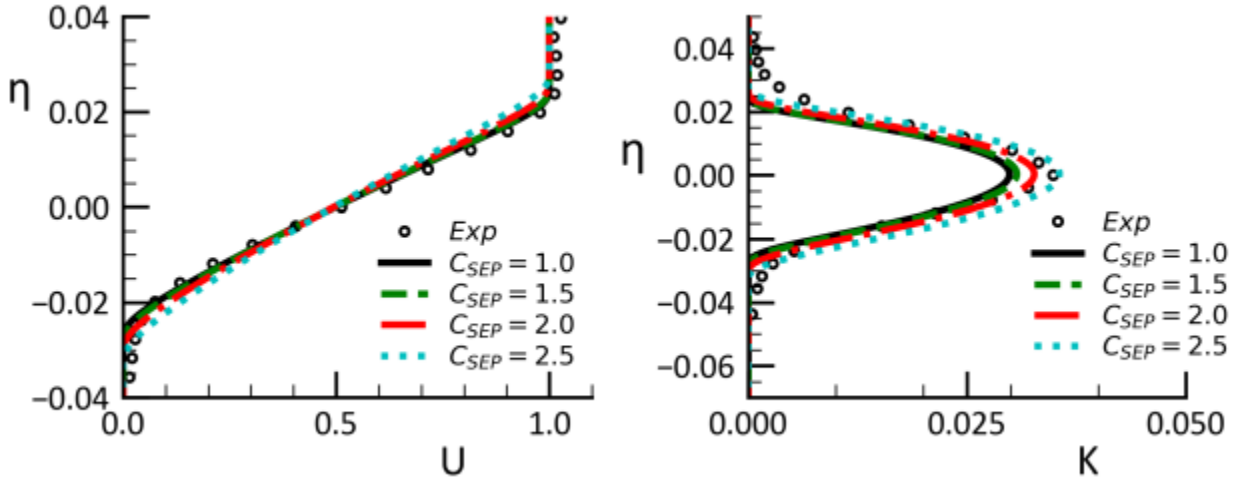
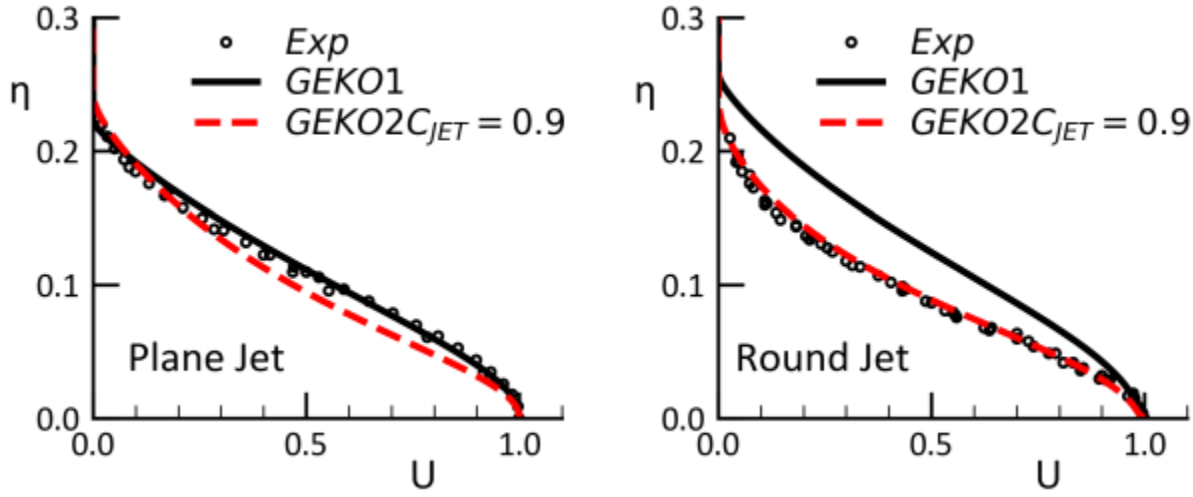


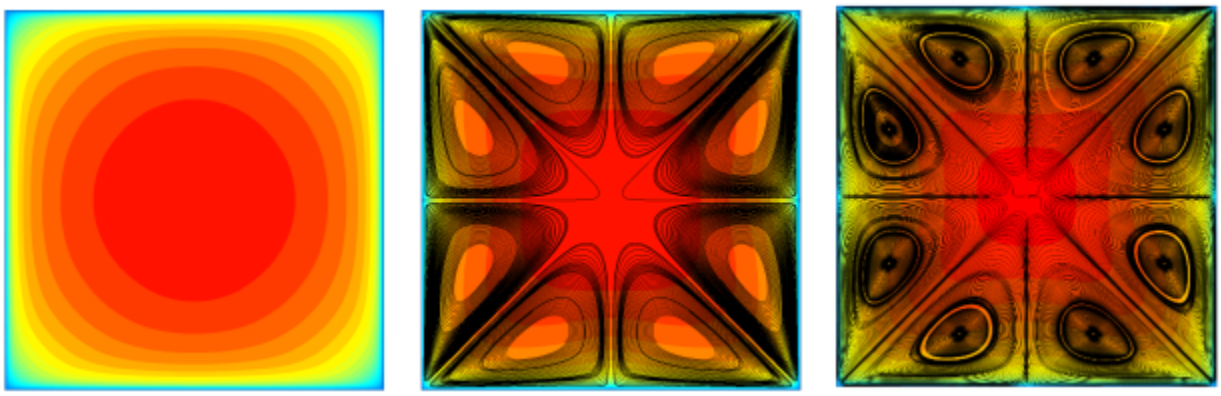
Figure 12.99: Flow topology for the linear (left) and non-linear GEKO model with corner flow correction (right) for the DLR diffuser [41 (p. 373)]

Figure 12.100: Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for turbulence models with and without non-linear terms for the DLR diffuser [41 (p. 373)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.101: Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the DLR diffuser [41 (p. 373)]



Stanford Diffuser

This flow studied experimentally by Cherry et al. [42 (p. 373)], [43 (p. 373)] is even more challenging than the flows considered above. For this case, separation has proven very sensitive to details of turbulence modeling. It seems clear that the anisotropy of the normal stresses must be accounted for to avoid the formation of an incorrect flow topology.

The geometry of the diffuser is shown in Figure 12.102: Geometry of Stanford diffuser [42], [43]. (p. 314). The top wall of the diffuser is inclined by an angle of 11.3 degrees. To achieve asymmetry, one the side walls is inclined by 2.56 degrees. The coordinate system used in the computations is shown in Figure 12.103: Computational domain with boundary conditions for the Stanford diffuser [42], [43]. (p. 314). The origin of the coordinate system in the X-direction is located at the cross-section of the diffuser corresponding to the intersection of its straight and inclined walls in the Y- and Z-directions; it is in the vertex of the dihedral angle formed by the straight walls of the diffuser.

In accordance with the experimental setup, the inlet flow is considered as fully developed flow in a rectangular duct with the bulk velocity providing a Reynolds number of Re = 10^4. The velocity in the inlet plane is specified using precursor computations of the developed rectangular channel flow with the same turbulence model used later for the diffuser simulation. The computations were carried out with the GEKO model with and without corner flow correction (CFC), GEKO-EARSM model and differential Reynold-Stress GEKO model (GEKO-RSM).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.102: Geometry of Stanford diffuser [42 (p. 373)], [43 (p. 373)].

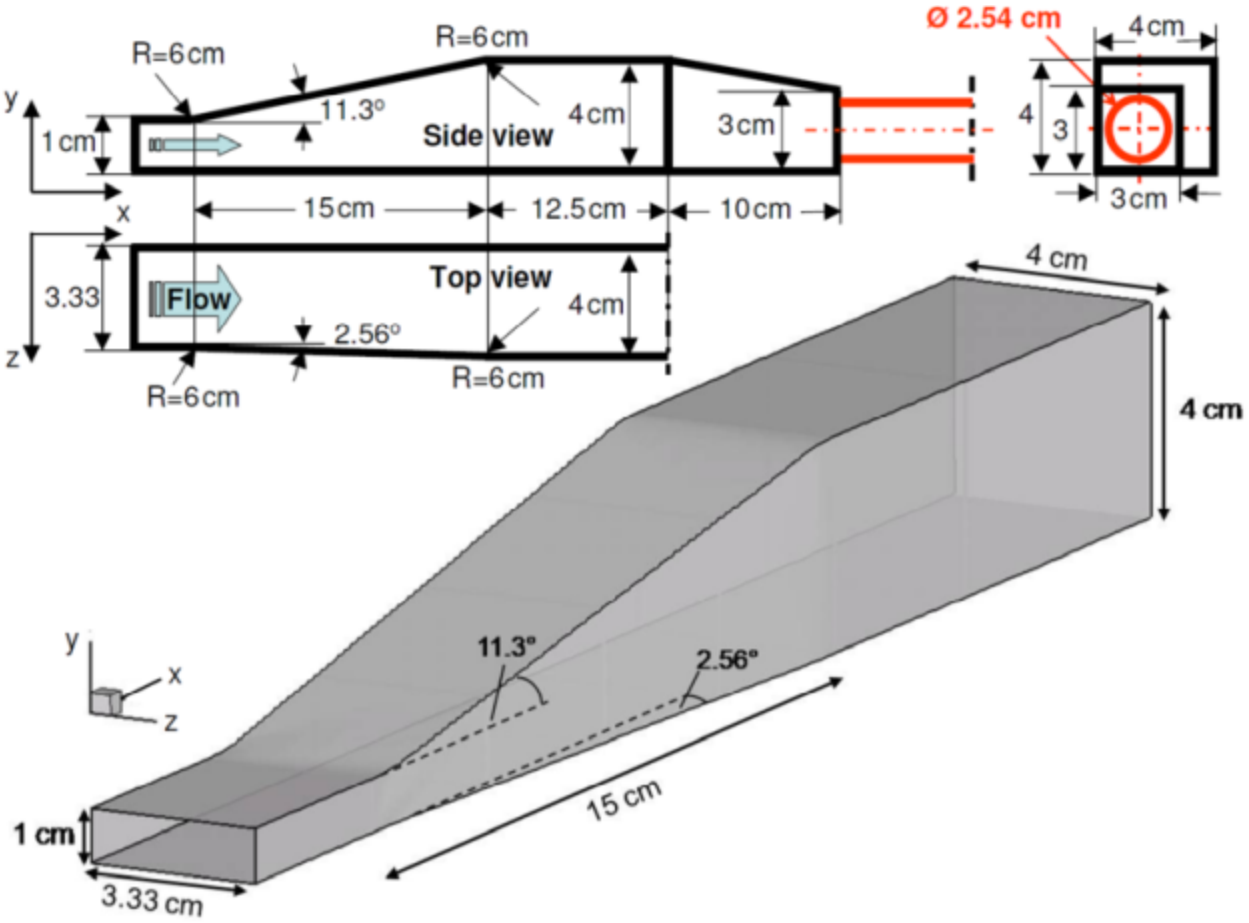


Figure 12.103: Computational domain with boundary conditions for the Stanford diffuser [42 (p. 373)], [43 (p. 373)].

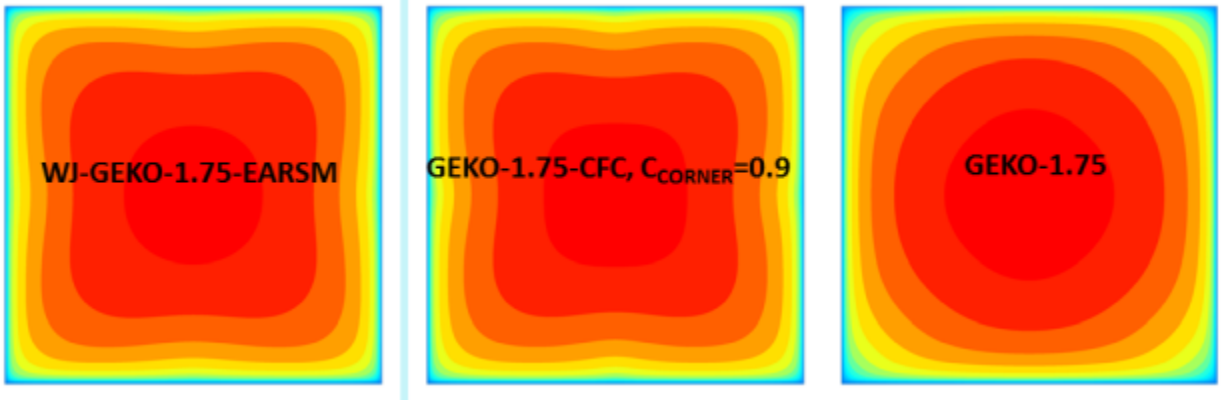


Figure 12.104: Flow topology for the linear (left) and non-linear CFC-GEKO model (middle) for the Stanford diffuser. Experimental data (right) [42], [43]. (p. 315) shows the 3D flow topology for the linear, the non-linear GEKO model and the experiment (note that these pictures are turned by 180° for visualization purposes). It is well seen that in the experiment the separation occurs mainly on the strongly inclined wall. The linear GEKO model predicts massive corner separation in the expanding part of the diffuser and separation occurs on the non-inclined side wall. The non-linear model decreases the size of the corner separation zone which dramatically changes the flow to-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pology. For the non-linear model, the separation occurs mainly on the strongly inclined wall which is in better agreement with the experimental flow topology. The effects of the nonlinearity on the flow topology are also shown on Figure 12.105: Streamwise velocity contours predicted by the linear and non-linear GEKO (C_{SEP}=1.00) model at X5 section. Blue colors indicate recirculation zone. (p. 316) where the streamwise velocity contours at section X5 are plotted (blue colors indicates negative values). The non-linear terms change the separation zone location (from the side wall to the inclined wall). The results of all the non-linear models are close to each other.

Figure 12.104: Flow topology for the linear (left) and non-linear CFC-GEKO model (middle) for the Stanford diffuser. Experimental data (right) [42 (p. 373)], [43 (p. 373)].

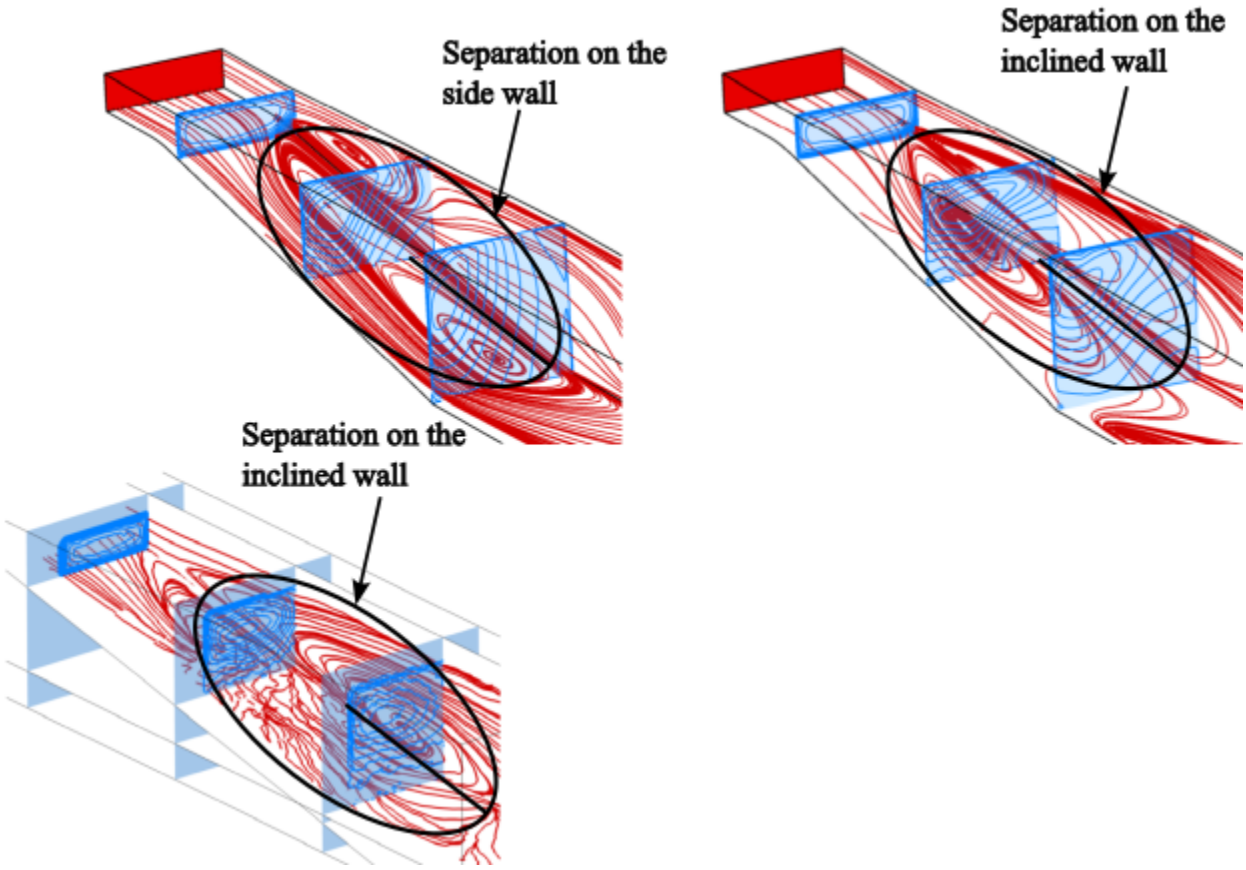
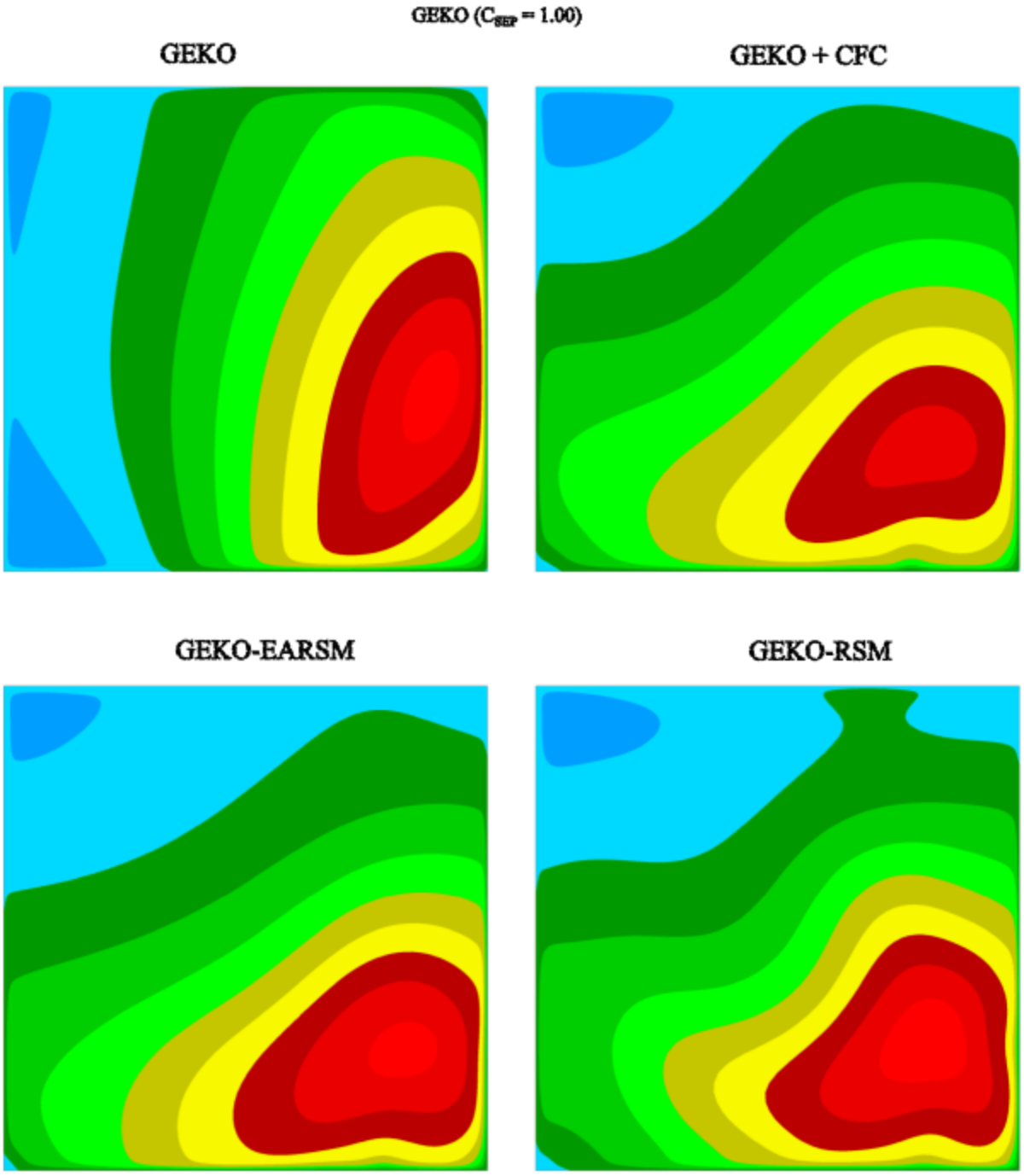


Figure 12.105: Streamwise velocity contours predicted by the linear and non-linear GEKO (C_{SEP}=1.00) model at X5 section. Blue colors indicate recirculation zone.



Interestingly, the flow topology for the non-linear models also depends on the baseline linear model. Figure 12.106: Streamwise velocity contours predicted be the linear and non-linear GEKO (C_{SEP}=1.75) model at X5 section. Blue colors indicate recirculation zone. (p. 318) shows that the increase of C_{SEP} in the GEKO model from C_{SEP} = 1.00 to CSEP = 1.75 leads a separation on both, the side and inclined wall. The effect of the C_{SEP} value on the prediction of the pressure coefficient is shown in Figure 12.107: Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the Stanford diffuser [42], [43]. (p. 319).

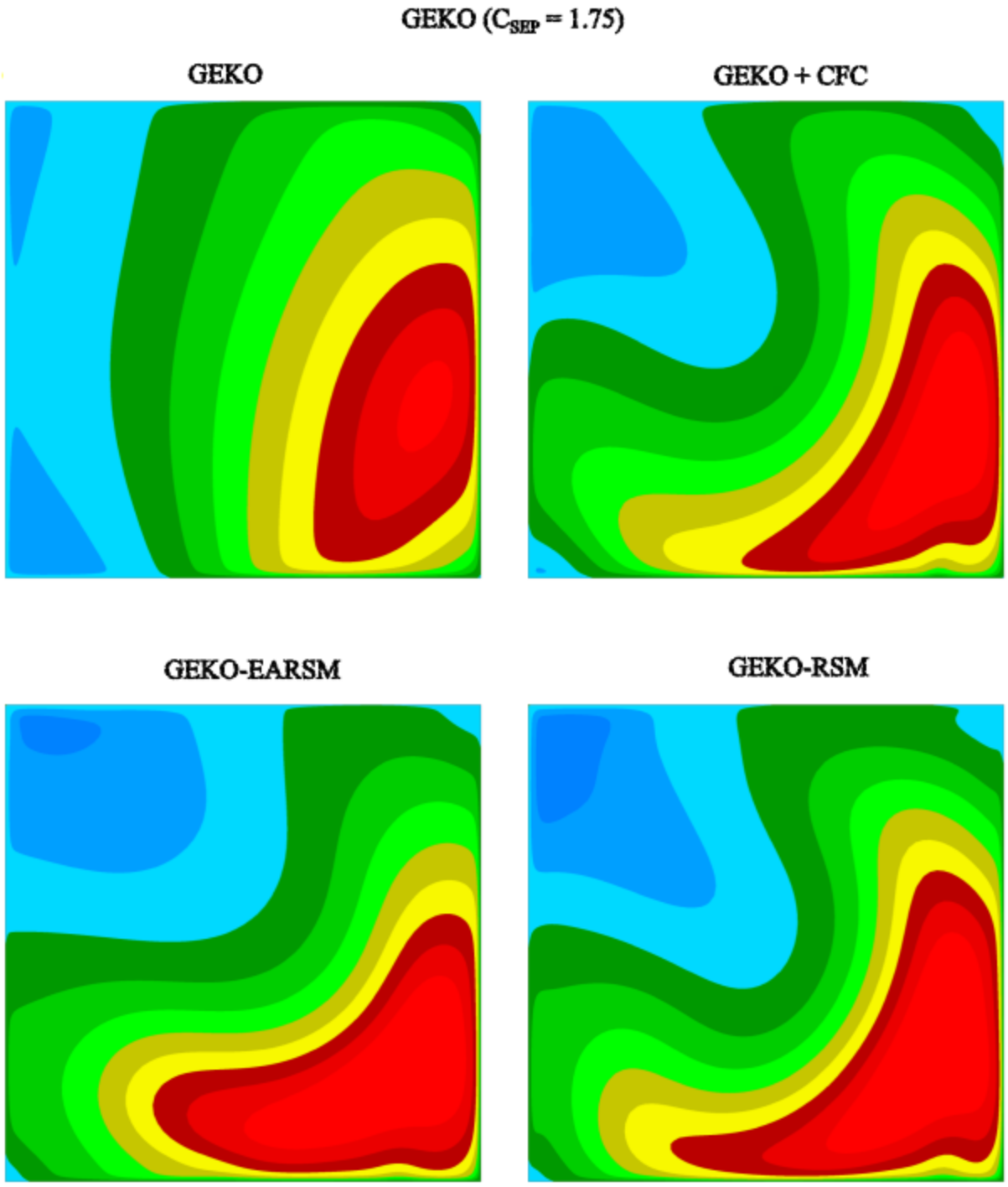
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

As seen, the more aggressive C_{SEP} settings (higher C_{SEP}) fail to capture the overall pressure distribution.

As the tests indicate, this flow is extremely sensitive to modeling details, as it allows for topology changes in the solution based on model settings. Poor results are obtained for C_{SEP}=1.75, as typically chosen for aerodynamic flows. This might be a result of the large separation zone predicted by the model. Such zones tend to be poorly predicted by RANS as the reattachment dynamics can be dominated by small-scale shedding from the separation line. As such effects are not included in RANS, it can lead to overly large separation zones. The error can be reduced by more conservative settings e.g. for C_{SEP}=1, which delays separation and thereby reduces the separation zone. While giving a better overall agreement with data, it could be the result of a cancelation of errors. This example shows that seemingly simple experimental configurations can still pose severe challenges to RANS modeling.

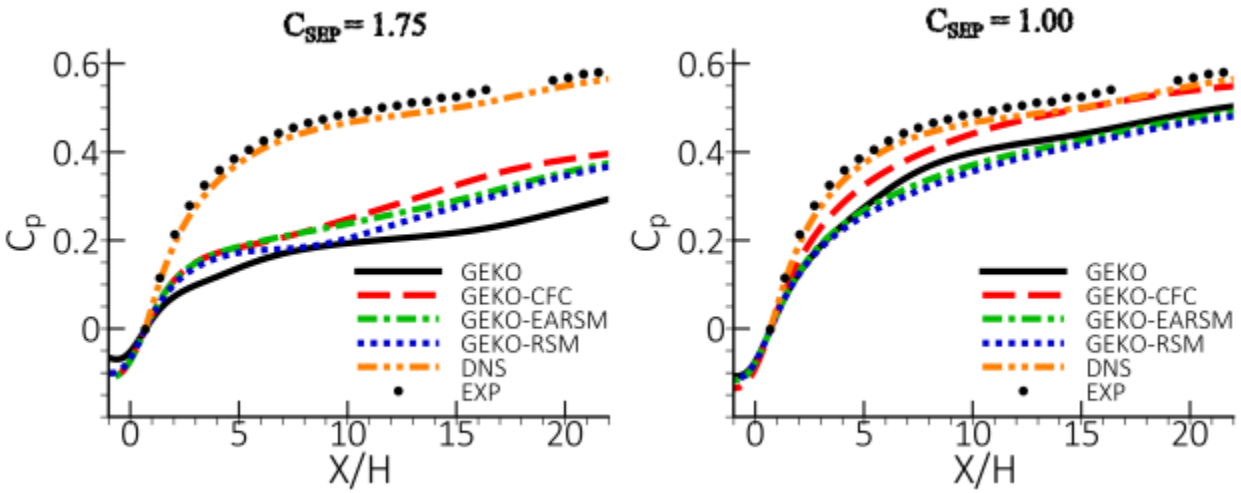
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.106: Streamwise velocity contours predicted be the linear and non-linear GEKO (C_{SEP}=1.75) model at X5 section. Blue colors indicate recirculation zone.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.107: Distribution of the pressure coefficient along the diffuser at midsection for turbulence models with and without non-linear terms for the Stanford diffuser [42 (p. 373)], [43 (p. 373)].

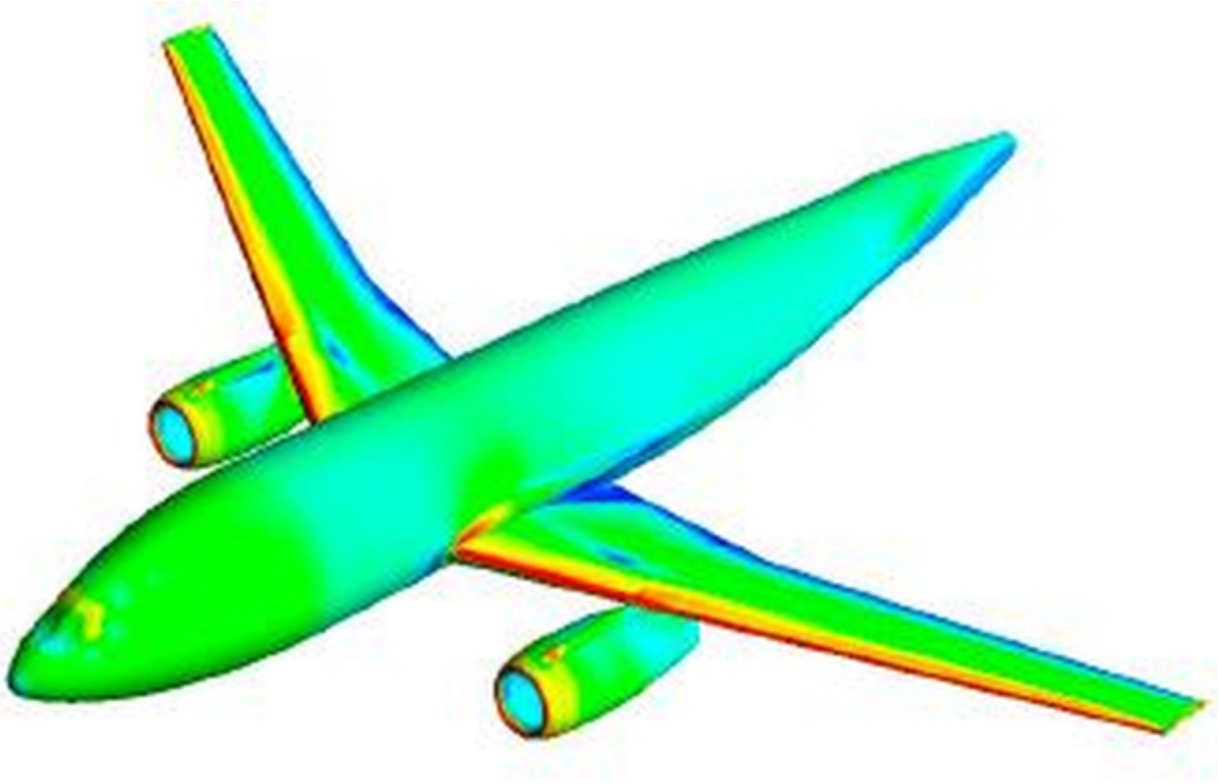


12.2.3.3.3. Flow around DLR F6 aircraft

Transonic flow past the DLR F6 airplane configuration with mounted engine shown in Figure Figure 48 at Mach number of Ma = 0.75 and Re = 3·106 was considered. The variant for the geometry with mounted engine was selected for comparison, with an angle of attack of 1 degree, providing a lift coefficient of about CL=0.5. Computations were performed in the Ansys CFX on a block-structured grid of 8.4 million elements. References to the measurement carried out at ONERA are available in 2nd AIAA CFD drag prediction workshop (https://aiaa-dpw.larc.nasa.gov/Workshop2/workshop2.html).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

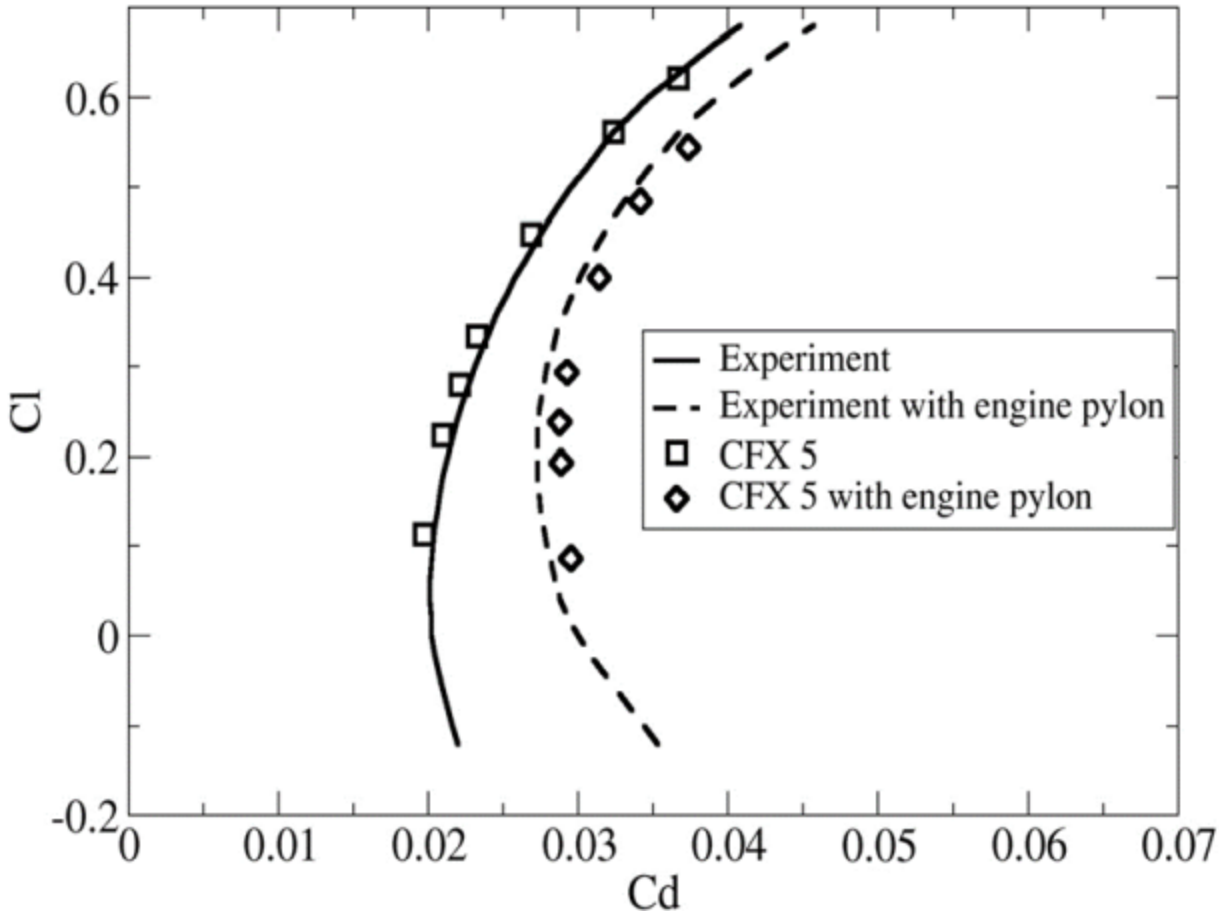
Figure 12.108: Geometry DLR F6 Wing-Body-Pylon-Nacelle testcase



The main challenge in the workshop was to predict the drag differences between a wing-body without and with the engine-nacelle installed. Figure 12.109: Lift-Drag Polar for Wing body without and with engine pylon + nacelle (p. 321) shows the lift-drag polar (different points are for different angles of attack) using the SST model. As can be seen, the data were very well represented in the simulations.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

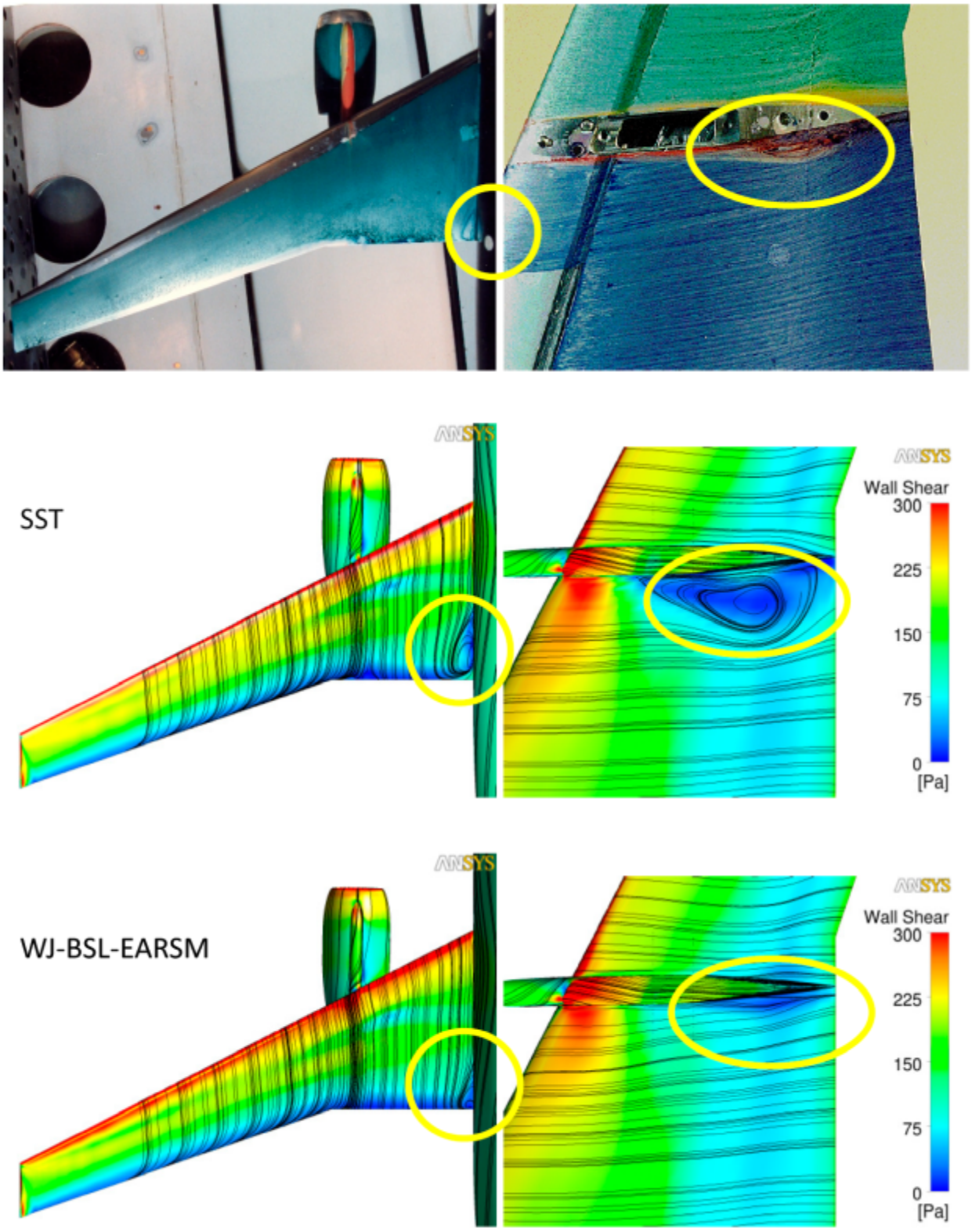
Figure 12.109: Lift-Drag Polar for Wing body without and with engine pylon + nacelle



However, in the current section, the interest is more in the details of the corners flows and the effect of non-linear models in such areas. The SST model tends to overpredict the size of the corner separation zones, due to the lack of secondary flow prediction in the corner, as shown in Figure 12.110: Separation zone at the upper wing-fuselage junction (left) and lower wing surface behind the engine (right). Top - oil film visualization in the experiment. (p. 322). The WJ-BSL-EARSM model improves the result for the wing-fuselage corner separation on the upper wing surface (left set of pictures in Figure 12.110: Separation zone at the upper wing-fuselage junction (left) and lower wing surface behind the engine (right). Top - oil film visualization in the experiment. (p. 322)). The other corner separation zone sits on the lower wing surface where the engine is mounted to the wing. The separation there is also over-predicted by the SST model. With the WJ-BSL-EARSM model, the separation size is substantially diminished in agreement with the experiment.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.110: Separation zone at the upper wing-fuselage junction (left) and lower wing surface behind the engine (right). Top - oil film visualization in the experiment.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.3.3.4. Conclusions

Non-linear effects play a significant role for the flows with secondary motions in corners. Linear eddy-viscosity models fail to predict the correct flow topology in the corners. For flows with pressure gradient and separation, this can lead to incorrect predictions of the flow topology. Non-linear models like CFC or EARSM formulations, significantly improve predictions of such flows. The use of the differential Reynolds Stress models does not lead to a significantly further improvement in comparison with the EARSM model.

12.2.3.4. Swirl Flows

This section shows two examples of the use of the curvature correction (described above in section 3.4.2): flow in a hydro-cyclone and a NACA-0012 wing tip vortex flow. For both cases, the curvature correction is used in combination with the SST model (SST-CC).

The following topics are discussed:

12.2.3.4.1. NACA-0012 Wing Tip Vortex

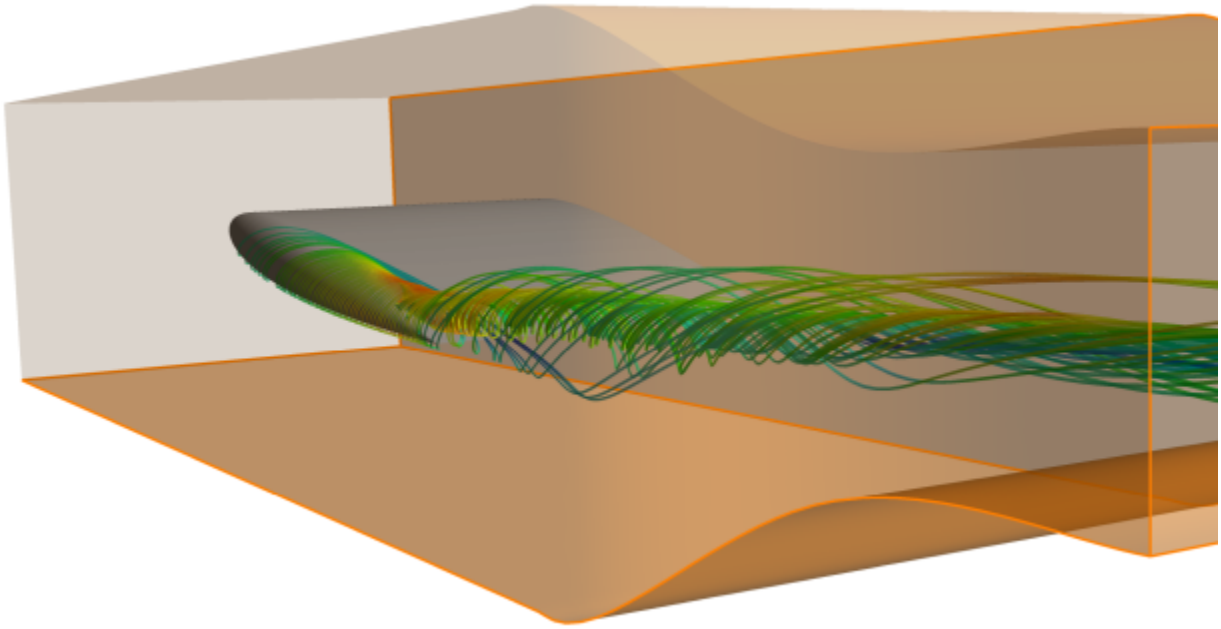
12.2.3.4.2. Flow in Hydro-Cyclone

12.2.3.4.1. NACA-0012 Wing Tip Vortex

The NACA 0012 wing tip vortex test case is based on the experiment of Chow et al. [44 (p. 373)]. There is a number of detailed experimental data available at various downstream locations including velocity fields, pressure and Reynolds stresses. The 3D wing shown on Figure 12.111: NACA-0012 wing setup in a wind tunnel showing wing tip vortices as visualized by streamlines. (p. 324) has a rounded tip and is placed inside a wind tunnel. Due to the lift, the flow from the pressure side travels around the rounded wing tip and rolls up into a strong vortex downstream. The chord-based Reynolds number is Re_C=4.6 million, the Mach number is approximately 0.1 and the angle of attack is α=$10°$. In the experiment, the flow is tripped at the leading edge so that the flow can be considered as fully turbulent.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.111: NACA-0012 wing setup in a wind tunnel showing wing tip vortices as visualized by streamlines.

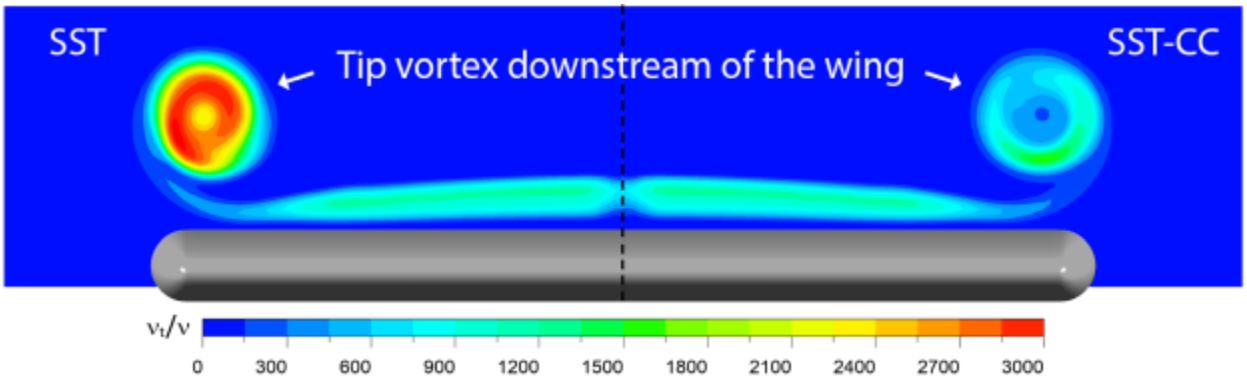


The following boundary conditions are adopted for the present simulations. At the inlet section, a total pressure value of P_{tot} = 1760 [Pa] above the atmospheric pressure is specified. Turbulent characteristics at the inlet section are computed from the turbulent intensity value of Tu = 0.15% and an eddy-to-molecular viscosity ratio equal to TVR = 5. A mass flow rate of $\dot{M}$ = 67.25 [kg/s] is imposed at the outlet boundary. This results in an area averaged inlet velocity of U_{inlet} = 51.81 [m/s] matching the experimental value. The boundary layer on the wind tunnel walls is not considered. Thus, symmetry boundary conditions are specified on the wind tunnel walls for the slip wall imitation and no-slip condition is specified on the NACA-0012 wing.

The mechanism of the curvature correction can be seen through the eddy-viscosity distribution shown in Figure 12.112: Front view of eddy-viscosity ratio contours computed with the use of SST and SST-CC turbulence models downstream of the NACA-0012 wing [44]. (p. 325) for station X/C = 0.67 downstream of the trailing edge. The lower turbulent viscosity in the vortex core region provided by the SST-CC model is a result of the reduction of production of turbulent kinetic energy by the CC modification. This prevents a premature decay of the core axial velocity to a value well below that found in the free-stream.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.112: Front view of eddy-viscosity ratio contours computed with the use of SST and SST-CC turbulence models downstream of the NACA-0012 wing [44 (p. 373)].

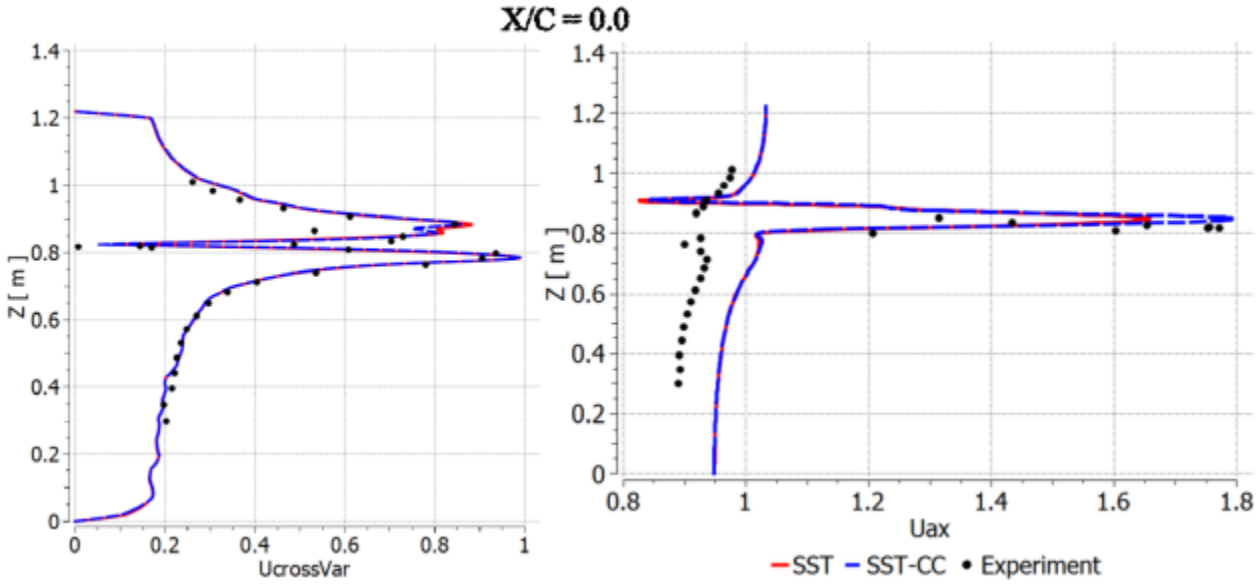


The predicted non-dimensional cross flow velocity and axial velocity at three planes located downstream of the trailing edge are shown in Figure 12.113: Non-dimensional cross flow velocity (UcrossVar) and axial velocity (Uax) at three planes located downstream of the trailing edge of the NACA 0012 wing tip [44]. (p. 326) in comparison with the experimental data. The coordinate X/C is based on the distance from the trailing edge. One can see that the vortex strength is better captured by the SST-CC model as measured by the maximum of the streamwise velocity. The original SST model provides too fast vortex decay, which is clear also through the axial velocity plots. The SST-CC model shows a significant improvement at the station $X/C = 0.24$ when compared to SST. However, at the far downstream location, $X/C = 0.67$, even the SST-CC model is no longer able to reproduce the experimental velocity profiles, although though the eddy-viscosity is significantly reduced by the correction. There is however also some mismatch in the axial freestream values which points to a potential discrepancy between the experiments and the CFD set-up.

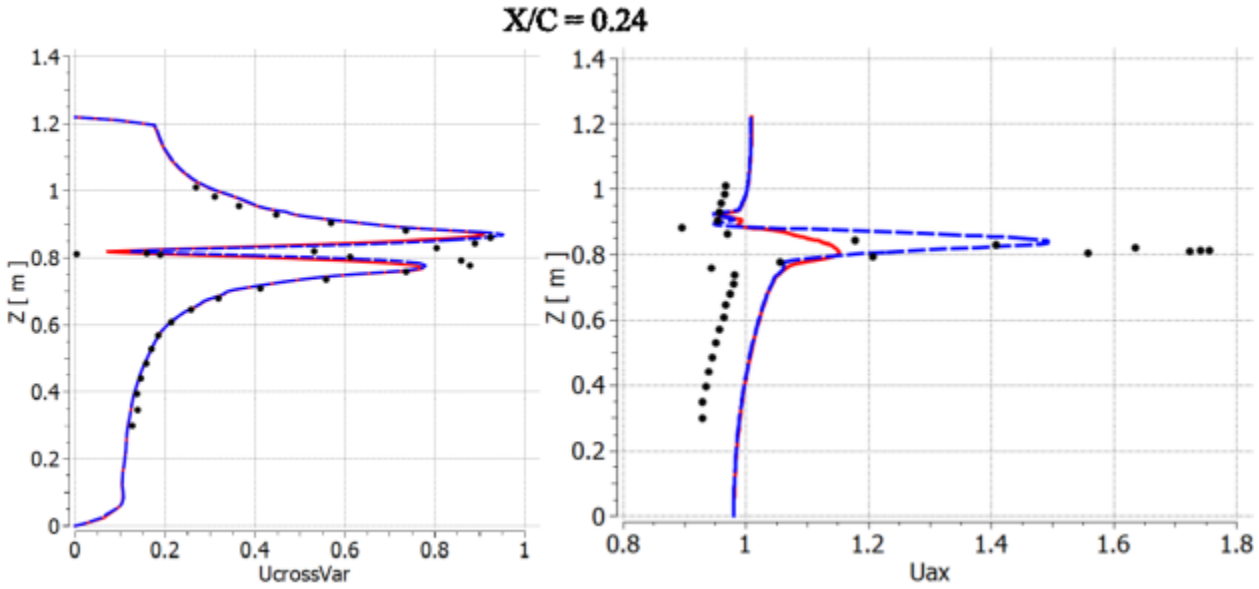
Figure 12.115: Computed distributions of the axial velocity at the plane passing through the vortex core for the NACA 0012 wing tip vortex [44]. Top view. (p. 328) shows the effect of the CC in a contour plot of the axial velocity for a plane going through the vortex core. The increased region of flow acceleration when using the CC extension is clearly visible. It breaks down however just upstream of the $X/C = 0.67$ measurement station.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.113: Non-dimensional cross flow velocity (UcrossVar) and axial velocity (Uax) at three planes located downstream of the trailing edge of the NACA 0012 wing tip [44 (p. 373)].



X/C=0.0



X/C=0.24

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.114:

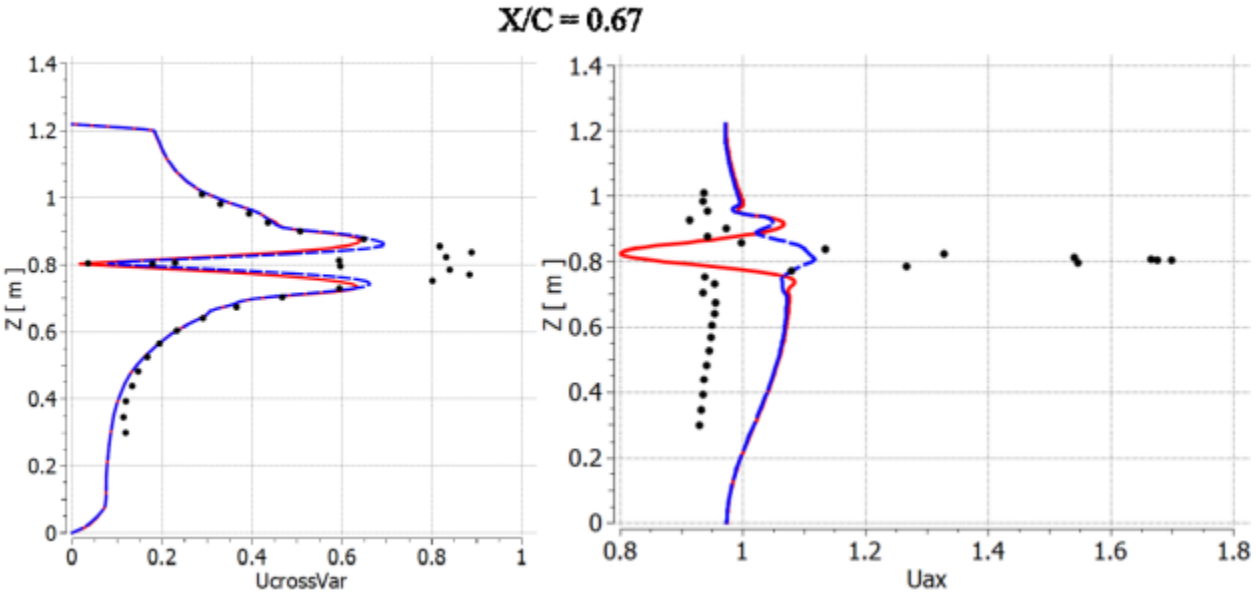
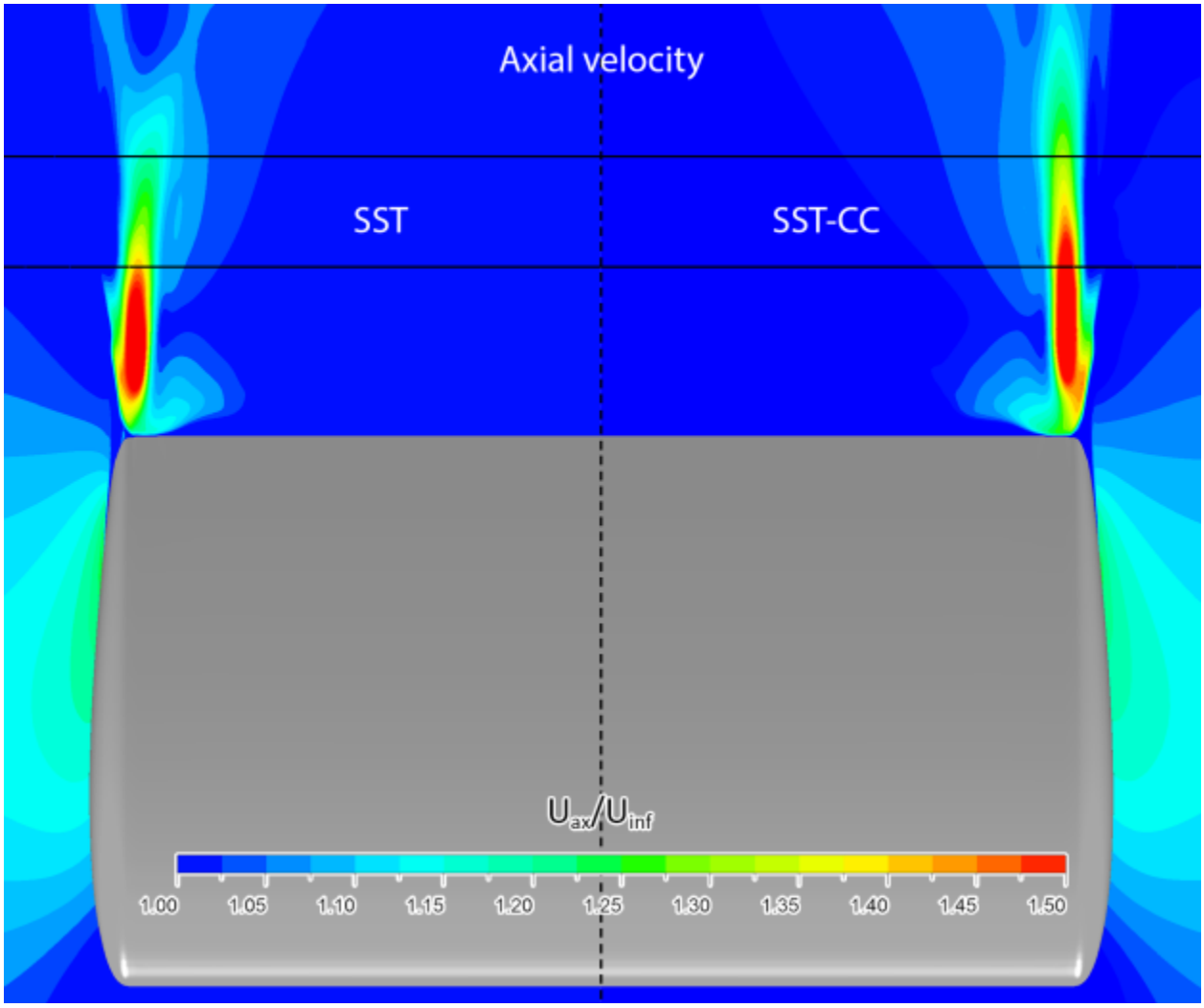


Figure 12.115: Computed distributions of the axial velocity at the plane passing through the vortex core for the NACA 0012 wing tip vortex [44 (p. 373)]. Top view.

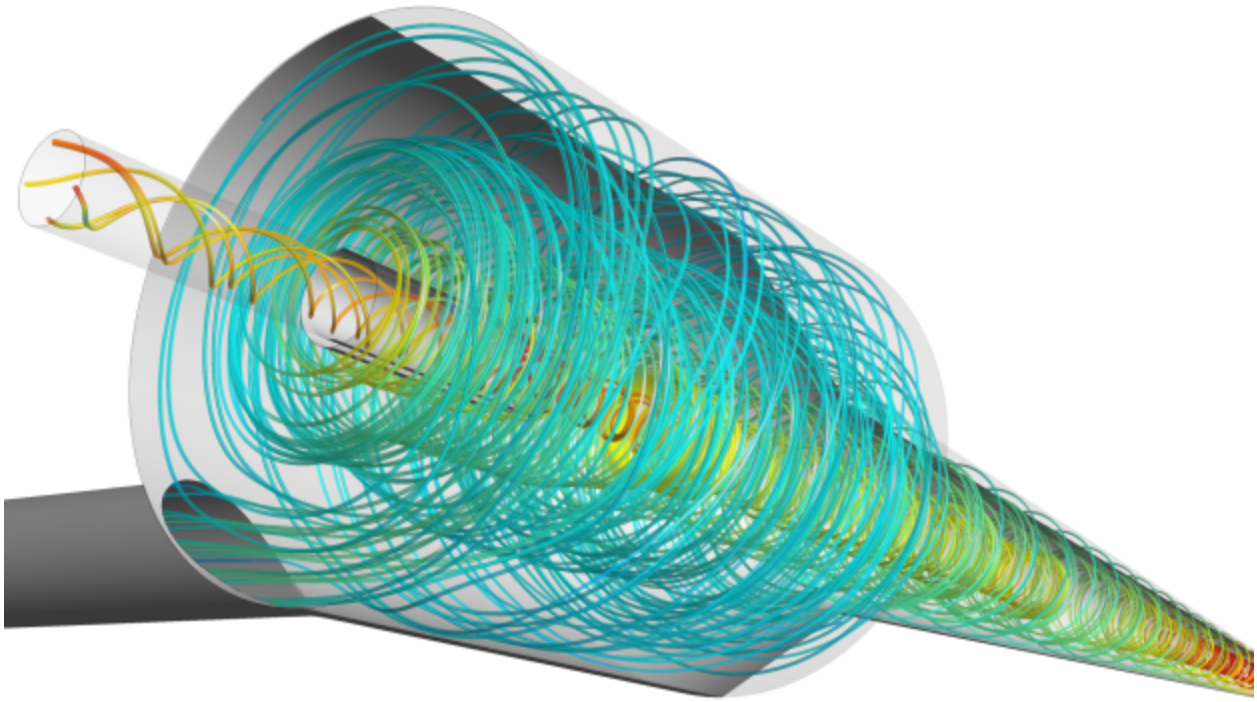


12.2.3.4.2. Flow in Hydro-Cyclone

The hydro-cyclone's typical flow structure and geometry is shown in Figure 12.116: Flow structure in the hydro-cyclone visualized by the streamlines. (p. 329). The principle behind the cyclone separation is to utilize the strong radial force acting on particles in a strongly swirling flows. The fluid/particle mixture is injected tangentially into the cyclone and spirals downwards in the cyclone barrel (cylindrical section) and then in the cyclone conical section. Under the influence of centrifugal forces, the heavy particles are pushed towards the wall and exit the cyclone due to gravity at the lower exit. The light phase moves towards the cyclone axis, where it joins the upwards flow in a central vortex, leaving the cyclone through the upper outlet. Flow inside a hydro cyclone is characterized by the formation of a strong vortex core in the central region. To predict such flows accurately, the correct representation of the turbulence is a challenging task for turbulence models.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.116: Flow structure in the hydro-cyclone visualized by the streamlines.

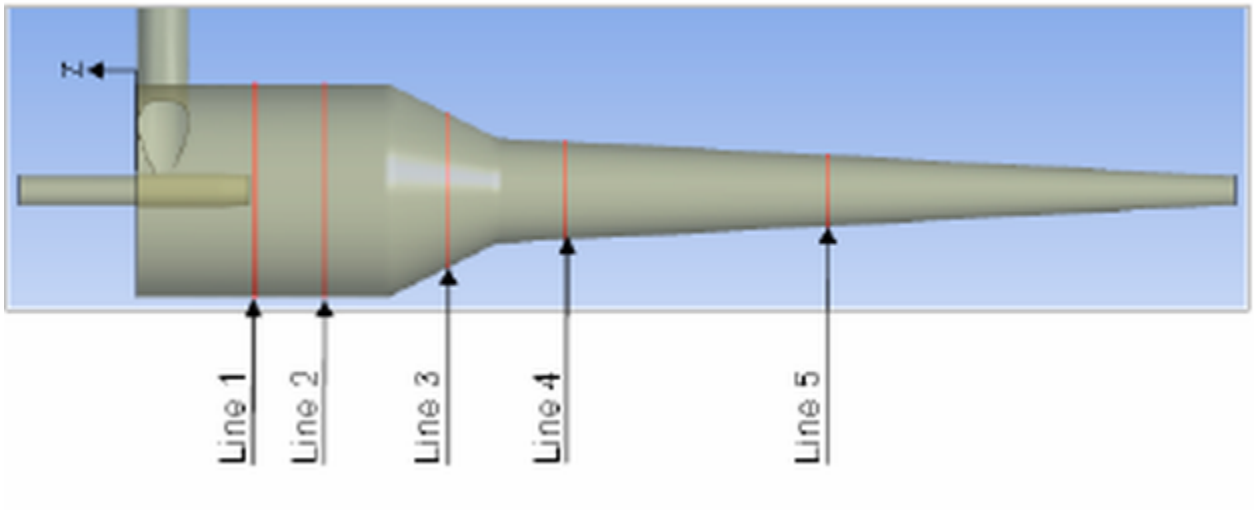


The current simulations are carried out for the hydro-cyclone investigated experimentally by Hartley [45 (p. 373)]. Experimental data of axial and tangential velocities are available at various vertical locations in the cyclone. It is important to note that with a steady-state approach, no physically correct results are obtained, so the case is run in unsteady mode. This is physically correct as it is known that the vortex core is not stable and meanders around the axis of the cyclone.

Figure 12.118: Time-averaged profiles of the tangential velocity in the hydro cyclone [45]. (p. 331) shows the sensitivity of the simulated tangential velocity to the choice of turbulence model for various z-locations in the hydro-cyclone as indicated in Figure 12.117: Measurement planes in the hydro cyclone [45]. (p. 330). The location z = −20 mm is just below the vortex finder. Here, the typical Rankine-vortex is clearly visible from the experimental data, showing an 'inviscid' vortex at the outer radii and a solid-body rotation close to the axis. The original SST model does not predict the 'inviscid' vortex at large radii, with a tendency towards a solid body rotation everywhere. A substantial improvement can be obtained by applying the curvature correction method for the SST model. The main mechanism of the curvature correction lies in capturing the stabilizing effect of "solid body rotation" near the axis which results in a strong reduction of the turbulence (and therefore eddy-viscosity) in that region. In the comparison there is also the combination of the SST model with the Kato-Launder production limiter (see Limiters (p. 389)) included. This modification does influence vortex flows as vorticity and strain rate are no longer equal. However, the effect is small compared to the CC modification.

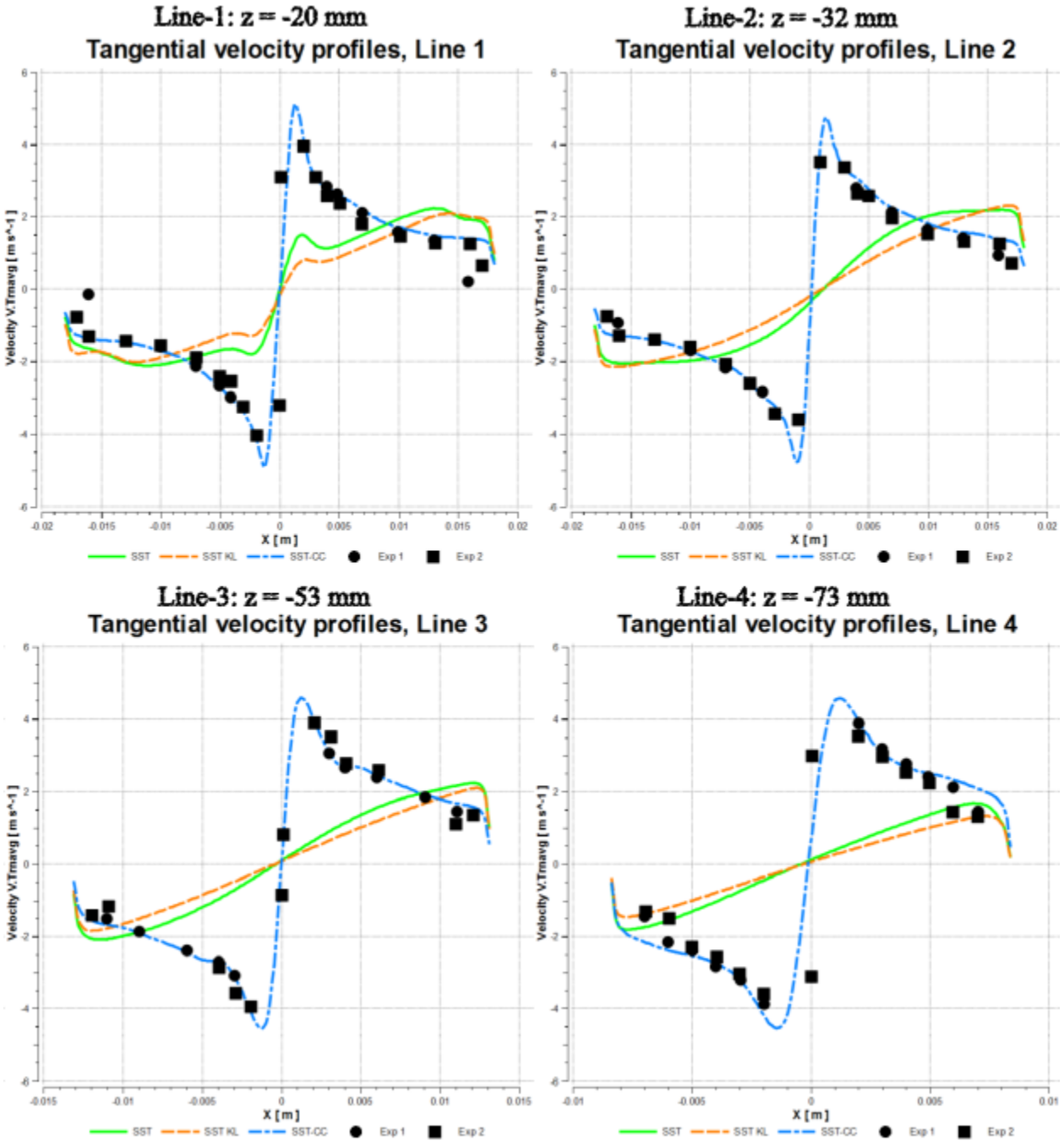
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

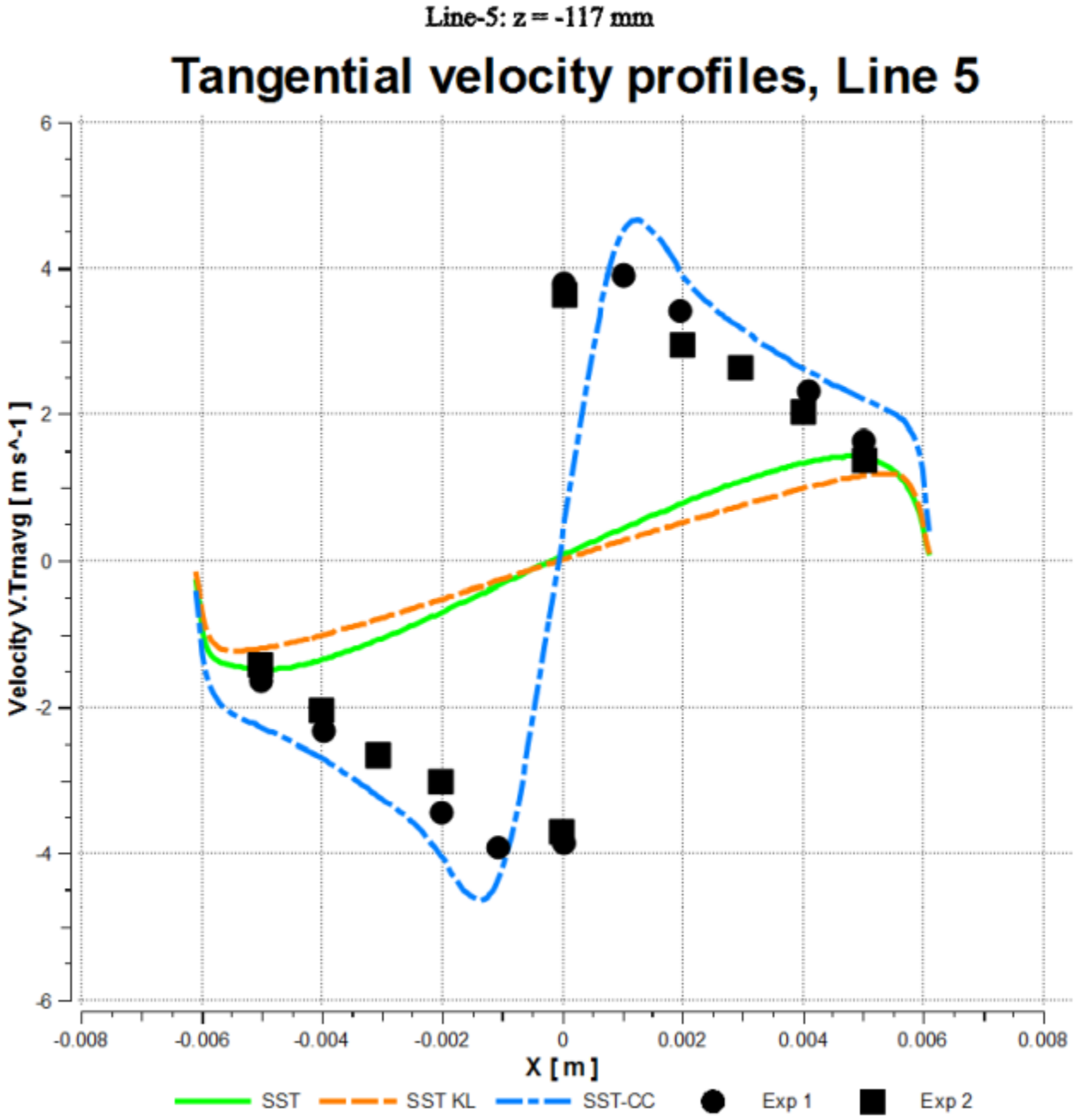
Figure 12.117: Measurement planes in the hydro cyclone [45 (p. 373)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.118: Time-averaged profiles of the tangential velocity in the hydro cyclone [45 (p. 373)].



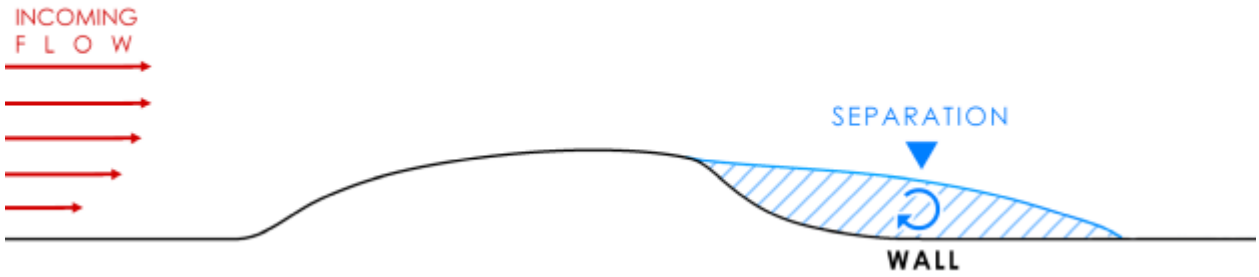


12.2.3.5. Reattachment Flows

The primary focus of the NASA Hump Flow [46 (p. 373)]–[48 (p. 374)] (Figure 12.119: Schematic of the NASA hump flow. (p. 333)) is to assess the ability of turbulence models to predict 2-D separation from a smooth body (caused by an adverse pressure gradient) as well as the subsequent reattachment and boundary layer recovery. Since its introduction, this case has proved to be a challenge for all known RANS models.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.119: Schematic of the NASA hump flow.



Models tend to underpredict the turbulent shear-stress and the turbulent kinetic energy in the separated shear layer, and therefore tend to predict too long a separation bubble. This is an issue with all RANS models and is one the main remaining deficiencies of this model family. It is important to note that the effect is not observed for cases where the separation is fixed by the geometry, like in a backward facing step (see Figure 12.61: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow (12). (p. 281)). It seems that the difference between these cases lies in a potential unsteadiness of the separation line for the hump flow, which could result in small-scale vortex shedding. Such an effect is in principle outside the realm of RANS models, as it constitutes unsteady flow and not turbulence. Naturally, there is a desire to model such effects within the existing RANS models, as the alternative would be much more expensive Scale-Resolving Simulations. For this reason, different, highly ad-hoc RANS model enhancements have been proposed. They are not used widely and should not generally be activated. However, in some applications, they have shown some improvements and one of the methods is therefore discussed here. The model presented is implemented in Ansys CFX. The formulation has also been adopted in a NASA TM by Rumsey [49 (p. 374)] using different coefficients.

As the SST model typically predicts the onset of separation with good accuracy, there is a need to introduce additional production of turbulence under such conditions. The Reattachment Modification (RM) model introduces the production term P_{RM} into the SST model to enhance turbulence levels in the separating shear layers emanating from walls.

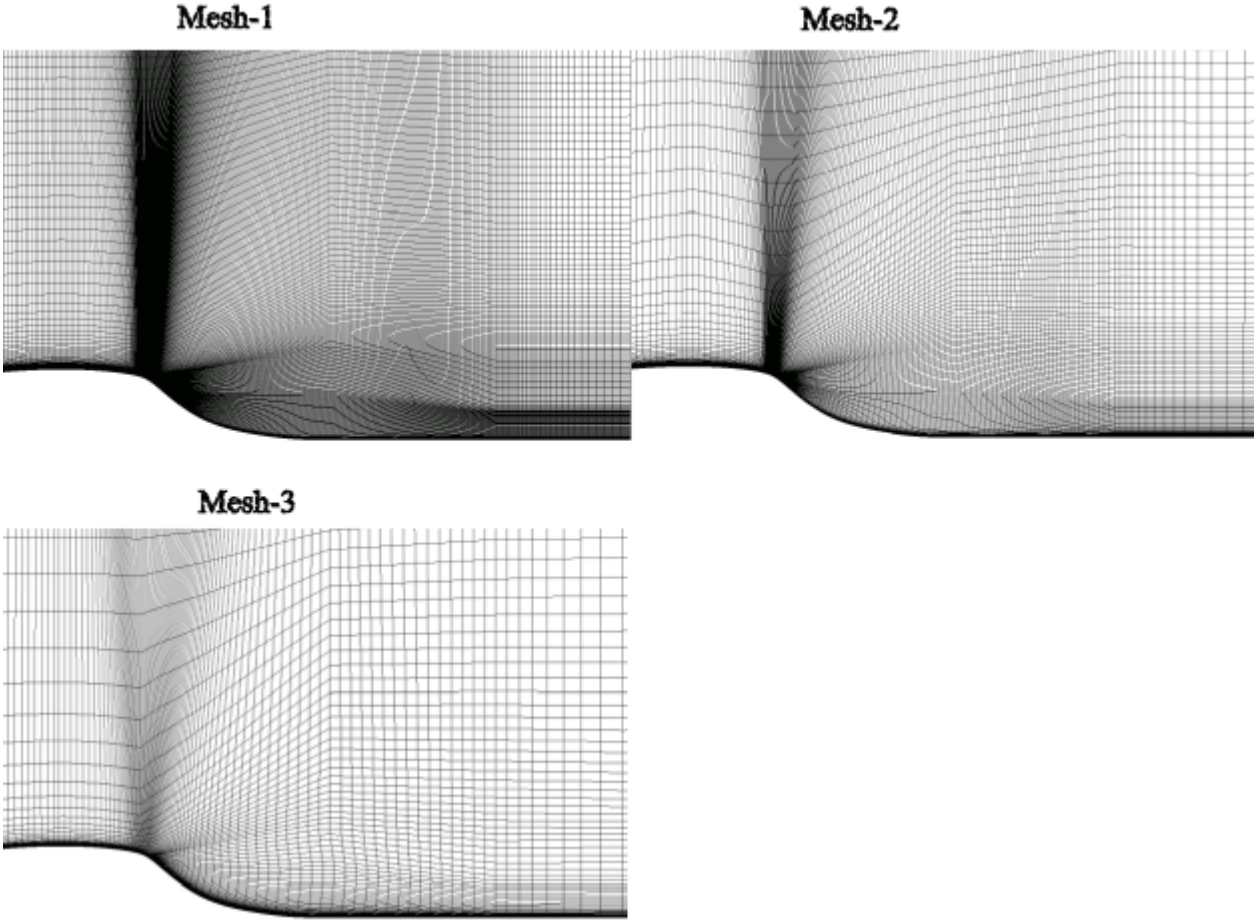
$$P_{RM}=P_K\min\left(4\max\left(0,\frac{\min(S^2,\Omega^2)}{0.09\omega^2}-1.6\right),1.5\right)F_{LR}$$
$$F_{LR}=\tanh\left(\left(\frac{k}{10\omega\nu}\right)^2\right) \tag{12.41}$$

The motivation for this formulation is that in the separating shear layer, the ratio $\frac{\min(S^2,\Omega^2)}{0.09\omega^2}$ increases significantly beyond one, which can then be used as an indicator to trigger the additional production term. Unfortunately, the reattachment modification is very sensitive to the combination of the computational mesh resolution and numerics for the turbulence equations. Three meshes as shown in Figure 12.120: Computational meshes for the Hump flow. (p. 334) have been tested, with Mesh-1 being the finest and Mesh-3 the coarsest. Figure 12.121: Comparison of wall shear stress coefficient for NASA Hump flow. (p. 335) shows the effect of the RM on the solution, as a function of mesh and discretization. The RM improves the prediction of the reattachment point only on the coarse meshes (Mesh-3) for the first order upwind scheme. The results of the original SST model and the SST-RM model using 2nd order numerics are virtually the same for all the meshes. The efficiency of the model can be improved by reducing the limiter of 1.6 in the equation, but this can have the effect that the modification will affect generic flows, especially turbulent mixing layers. The spreading rate of mixing layers would then be over-predicted. The NASA version of the model effectively re-

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

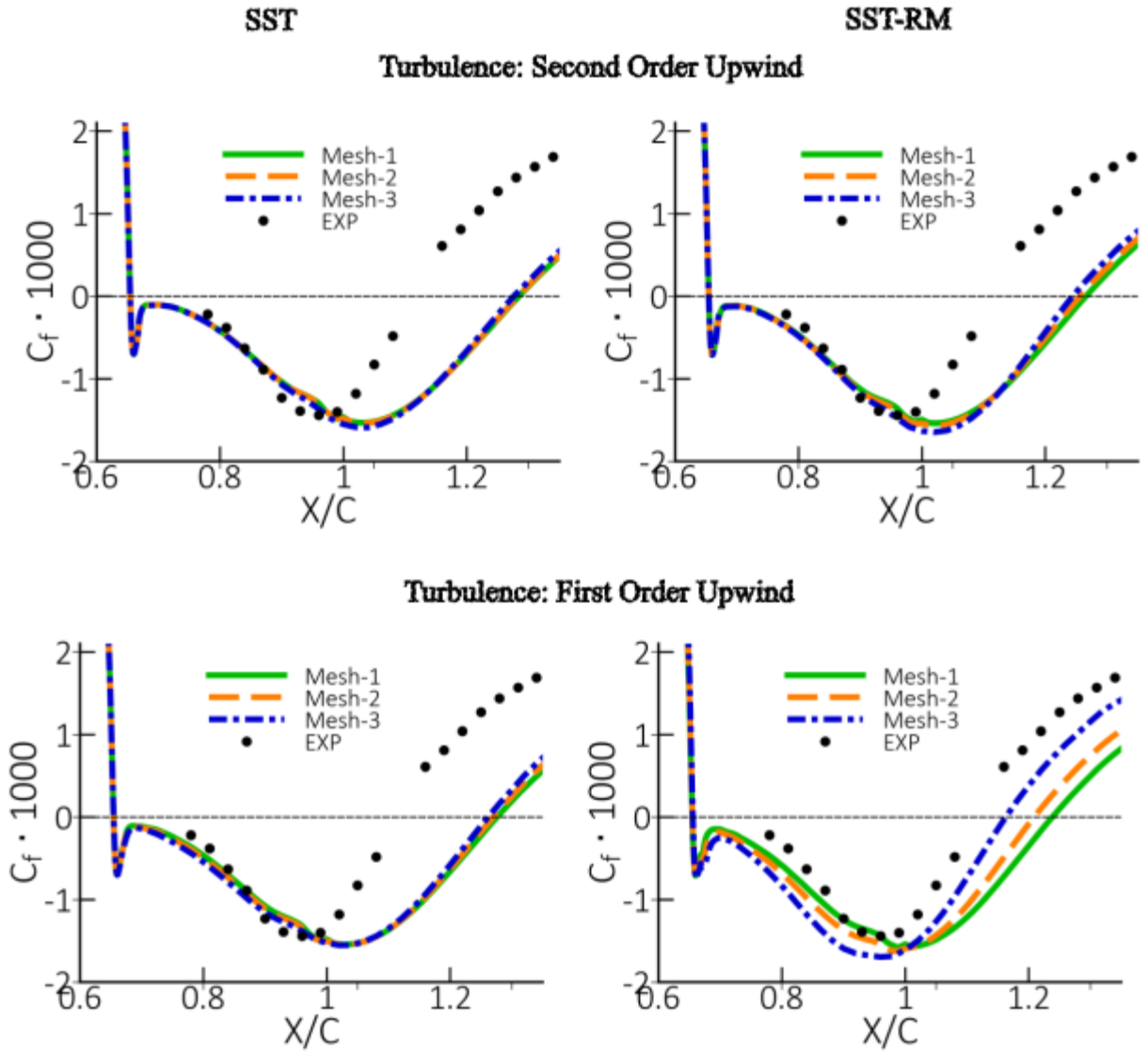
duces the limiter from 1.6→1.25 resulting in a more active model, but at the price of affecting generic flows.

Figure 12.120: Computational meshes for the Hump flow.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.121: Comparison of wall shear stress coefficient for NASA Hump flow.



An improvement of the reattachment prediction can also be achieved with the turning of the GEKO model. Figure 12.122: Comparison of wall shear stress coefficient predicted with GEKO model for NASA Hump flow [48]. (p. 336) shows the effect of the decrease of the C_{SEP} coefficient on the prediction of the reattachment point. The prediction of the velocity and Reynolds stresses are also improved for the C_{SEP} = 1.00 value (Figure 12.123: Effect of the GEKO CSEP on the streamwise velocity for NASA Hump flow [46], [47]. (p. 336)) albeit at the expense of delayed separation onset prediction. However, the Reynolds normal stresses $\overline{u'u'}$ are still underpredicted in the separation region and virtually independent of the C_{SEP} value which is shown in Figure 12.124: Effect of the GEKO CSEP on the Reynolds shear stresses for NASA Hump flow [46], [47]. (p. 337).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.122: Comparison of wall shear stress coefficient predicted with GEKO model for NASA Hump flow [48 (p. 374)].

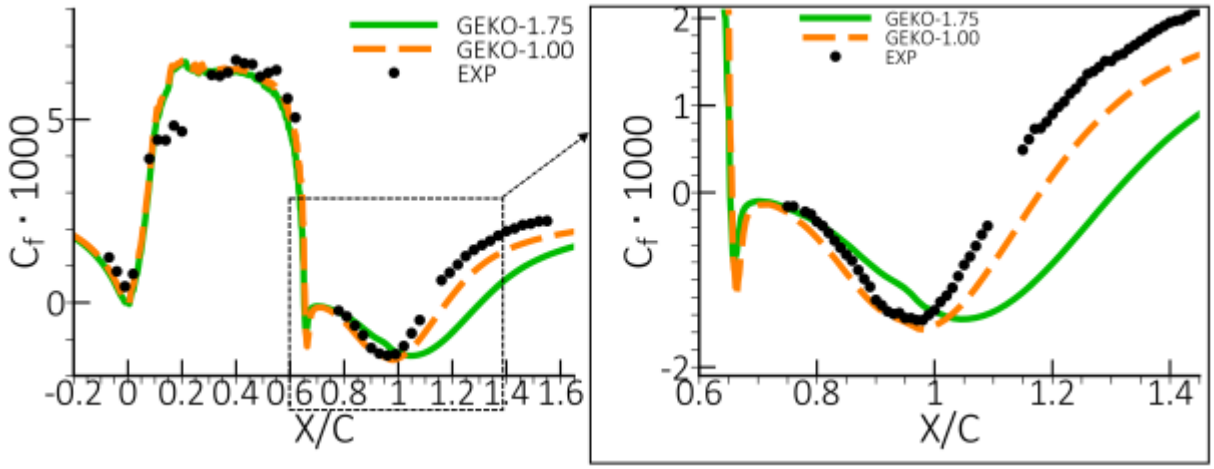
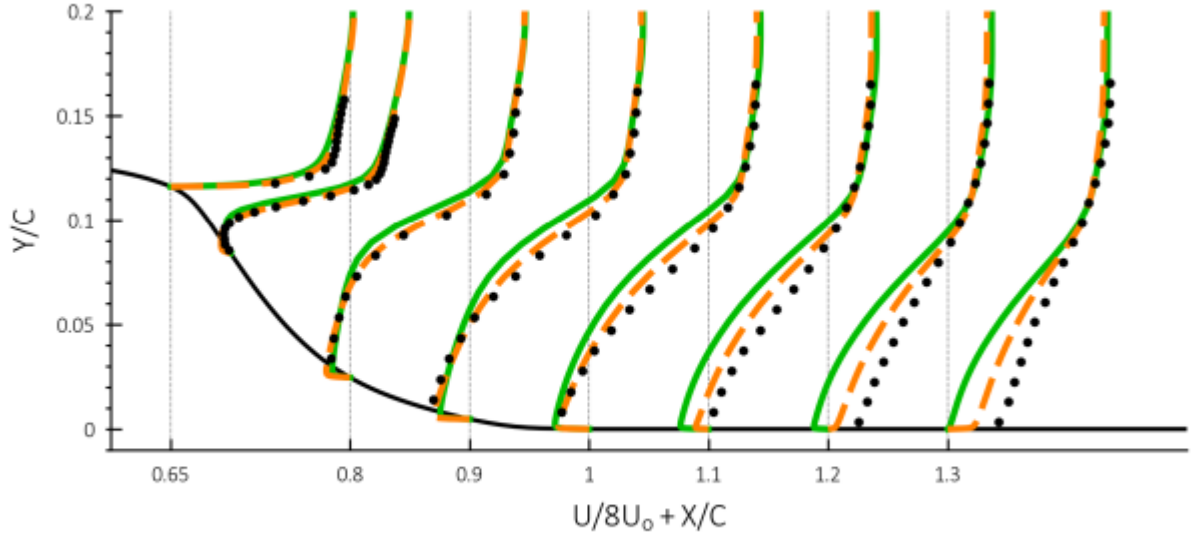
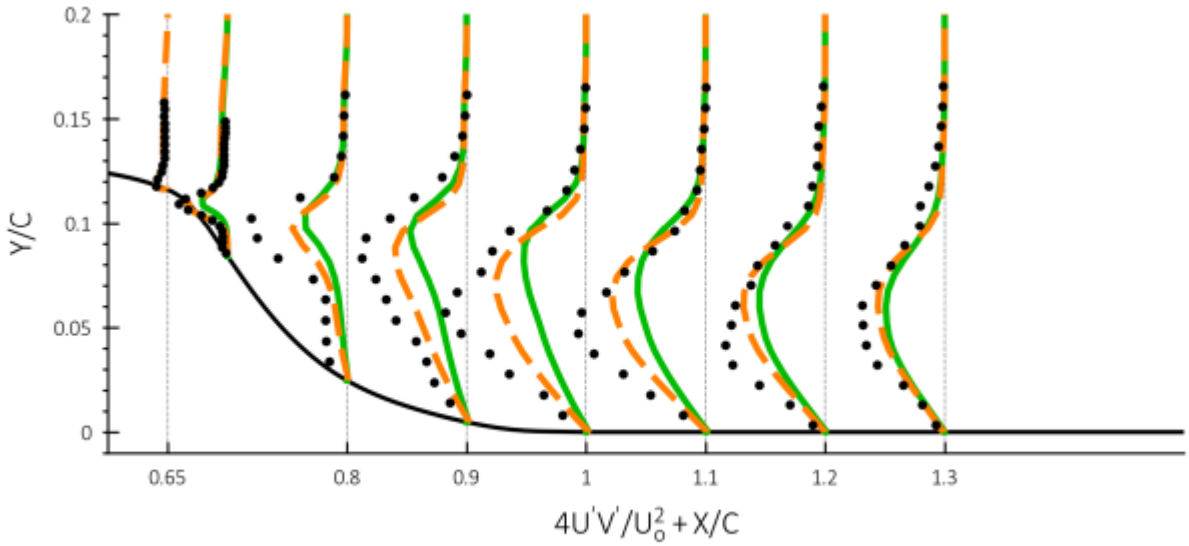


Figure 12.123: Effect of the GEKO CSEP on the streamwise velocity for NASA Hump flow [46 (p. 373)], [47 (p. 374)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.124: Effect of the GEKO CSEP on the Reynolds shear stresses for NASA Hump flow [46 (p. 373)], [47 (p. 374)].



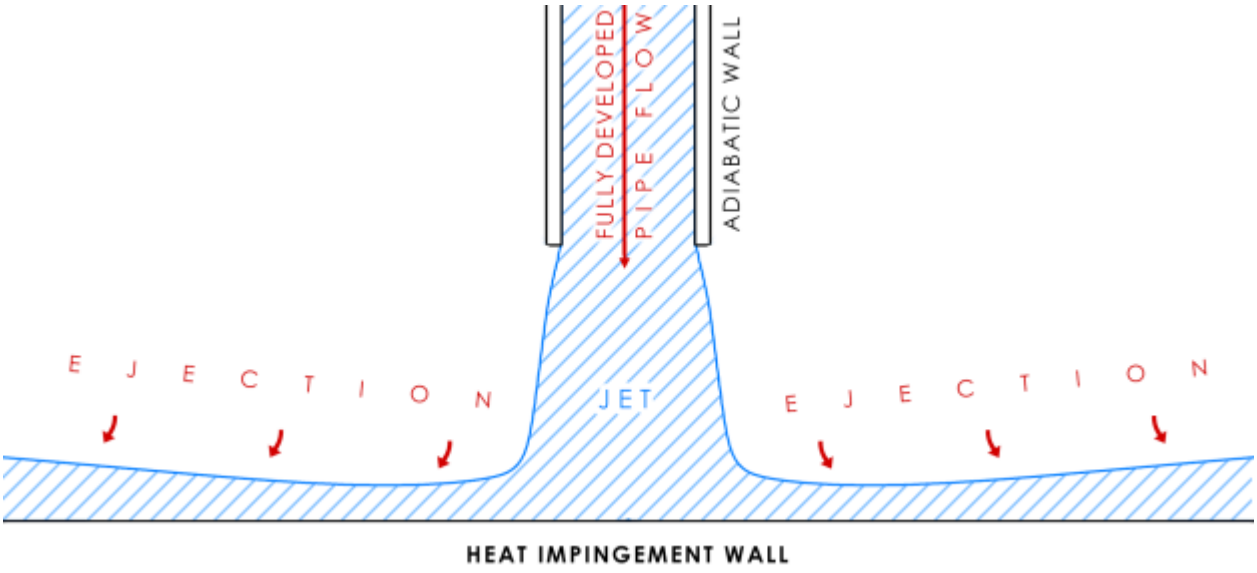
Again, it needs to be stressed that the change to C_{SEP}=1.00 is not considered a model for the true physics of these flows, but can help in practical applications to obtain better agreement with the data.

12.2.3.6. Impinging Flows

Impinging jets are a very effective way to enhance surface heat transfer and are frequently employed in industrial devices (impingement cooling). The considered test case is based on the experiments carried out by Baughn et al. [50 (p. 374)]. The flow being modeled is the incompressible flow of a turbulent jet impinging onto a flat plate. Figure 12.125: Scheme of the impinging jet flow (p. 338) shows the geometry of the flow domain. The size of the computational domain in the plane is 13D by 13D, where D = 0.0265 [m] is the diameter of the pipe.

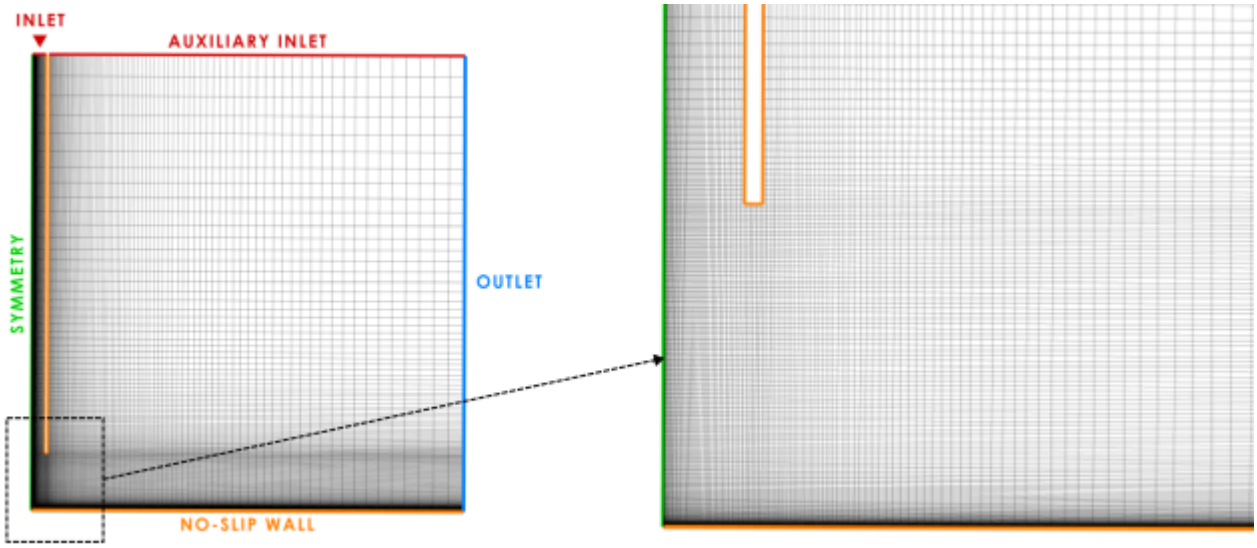
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.125: Scheme of the impinging jet flow



The computational domain with boundary conditions is shown in Figure 12.126: Computational domain with boundary conditions and mesh for the impinging jet. (p. 338). The inflow conditions for velocity and turbulence are specified using profiles for fully developed turbulent flow through a pipe, which are calculated in separate simulations. The Reynolds number based on the bulk inlet velocity, U_{bulk}= 15.45 [m/s], and pipe diameter is Re = $2.3 \cdot 10^4$. The heated surface is modeled as a wall with a specified constant heat flux, q= 300 [W/m]. All other walls are treated as adiabatic walls. An auxiliary inlet boundary condition is employed to allow for entrainment of fluid from outside. Zero gauge total pressure along with the inflow direction is assumed on this segment. The right boundary is modeled as a modified outlet boundary which changes to an inlet with 10 degrees flow direction to the boundary, in case fluid enters the domain. This treatment avoids massive flow entrainment at that boundary.

Figure 12.126: Computational domain with boundary conditions and mesh for the impinging jet.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.127: Local Nusselt number computed using different turbulence models (left) and SST model with and without taking into accout the laminar turbulent transition (right). (p. 339)(left) compares the predicted Nusselt number distributions for different turbulence models. The $k-\varepsilon$ and SA models underestimate Nu over almost the entire length of the heated wall, while the SST model gives a reasonable agreement downstream from $r/D = 1.5$. However, all the models fail in predicting the dip in the Nu number at $r/D < 1.5$. To investigate this further, a simulation is performed, where the SST model is combined with the γ-transition model (36 (p. 372)). Figure 12.127: Local Nusselt number computed using different turbulence models (left) and SST model with and without taking into accout the laminar turbulent transition (right). (p. 339) (right) demonstrates that the inclusion of the laminar-turbulent transition phenomena is essential in predicting the local minima. It coincides with the location of laminar-turbulent transition of the boundary layer developing on the impingement wall, which is triggered by the shear layer from the outer boundary of the impinging jet. Note that in case of the transition computations, the computational mesh (see Figure 12.128: Computational mesh for the fully turbulent regime (left) and regime with taking into account laminar-turbulent transition (Right). (p. 339)) was refined in radial direction ($r/D < 1.5$) for the proper resolution of the laminar-turbulent transition zone (this refinement had no effect on the fully turbulent simulations).

Figure 12.127: Local Nusselt number computed using different turbulence models (left) and SST model with and without taking into accout the laminar turbulent transition (right).

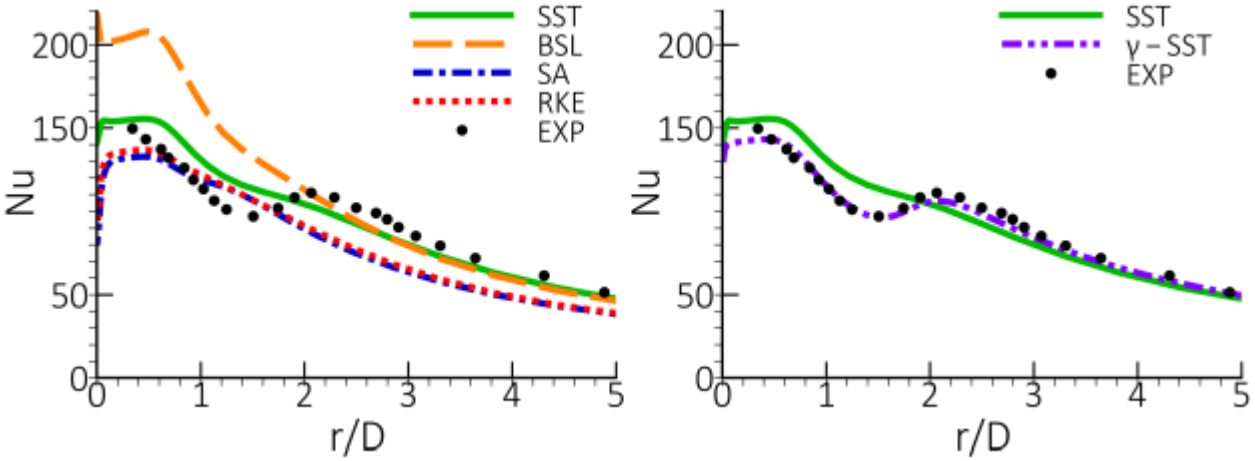
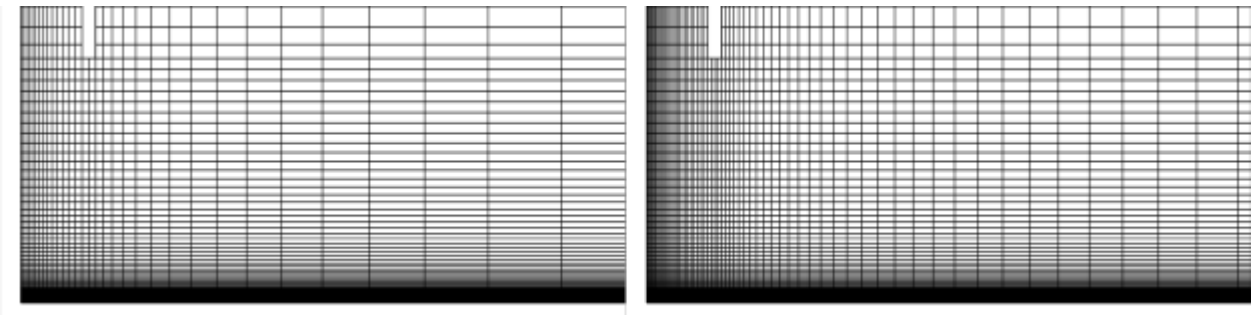


Figure 12.128: Computational mesh for the fully turbulent regime (left) and regime with taking into account laminar-turbulent transition (Right).



12.2.3.7. Buoyancy Flows

The following topics are discussed:

12.2.3.7.1. Stratified Mixing Layer

12.2.3.7.1. Stratified Mixing Layer

A stably-stratified mixing layer flow between fresh water and saline stream was experimentally investigated by Uittenbogaard [51 (p. 374)] at the Delft Hydraulics laboratory. Their installation is sketched in Figure 12.129: Scheme of the stratified mixing layer (p. 340). Two flows with different densities enter the domain horizontally separated by a splitter plate. The upper flow is a fresh water stream with a density of 1015 [kg/m3] and an average velocity of 0.52 [m/s]. The lower current is a salt water solution with density of 1030 [kg/m3] and an average velocity of 0.32 [m/s]. The flow can be considered two dimensional in the CFD setup as shown in Figure 12.129: Scheme of the stratified mixing layer (p. 340). The flow properties are presented in Table 3.

Figure 12.129: Scheme of the stratified mixing layer

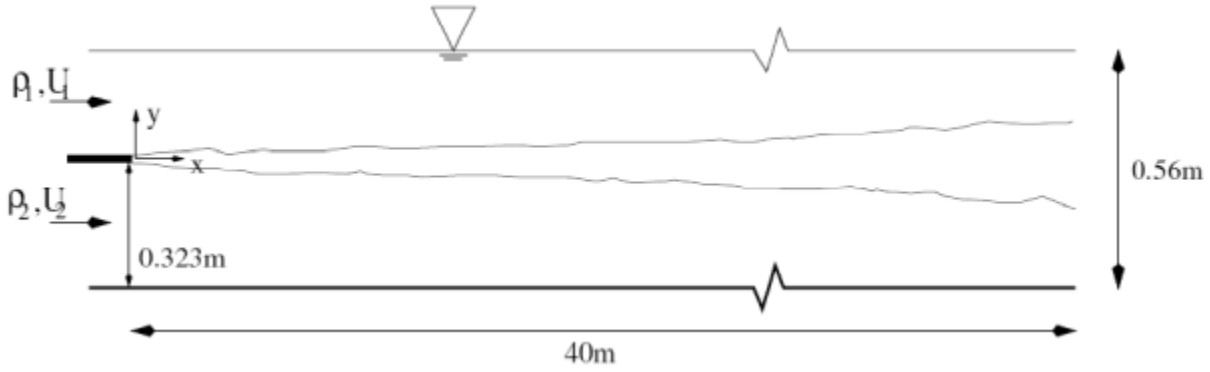


Figure 68: Scheme of the stratified mixing layer.

Table 12.12: Flow parameters of the stratified mixing layer

ρ_{fresh}, **kg/**m^3	ρ_{salt}, **kg/**m^3	μ **Pa s**	**Thermal Conductivity, W/(m K)**	**Mass Diffusivity,** m^2**/s**	**Specific Heat, J/(Kg K)**	**Molecular Weight, kg/kmol**
1015	1030	8.899×10^{-4}	0.6069	10^{-9}	4182	18.0152

The computational domain with boundary conditions is shown in Figure 12.130: Computational domain with boundary conditions and mesh for the stratified mixing layer. (p. 341). The length and height of domain are 40 [m] and 0.56 [m] corresponding to the experimental test section. At the inlet boundary, the thickness of the fresh/salt water layer is 0.237 and 0.323 [m] respectively.

Computations are carried out with Standard $k-\varepsilon$ model and SST and GEKO models without and with buoyancy corrections. For all the cases, the boundary conditions are as follows. At the inlet boundary, the experimental velocity profiles, turbulence characteristics and component concentration are specified. Constant pressure is specified on the outlet boundary. The lower boundary is modeled as the non-slip wall and upper boundary of the flow is a free surface.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.130: Computational domain with boundary conditions and mesh for the stratified mixing layer.

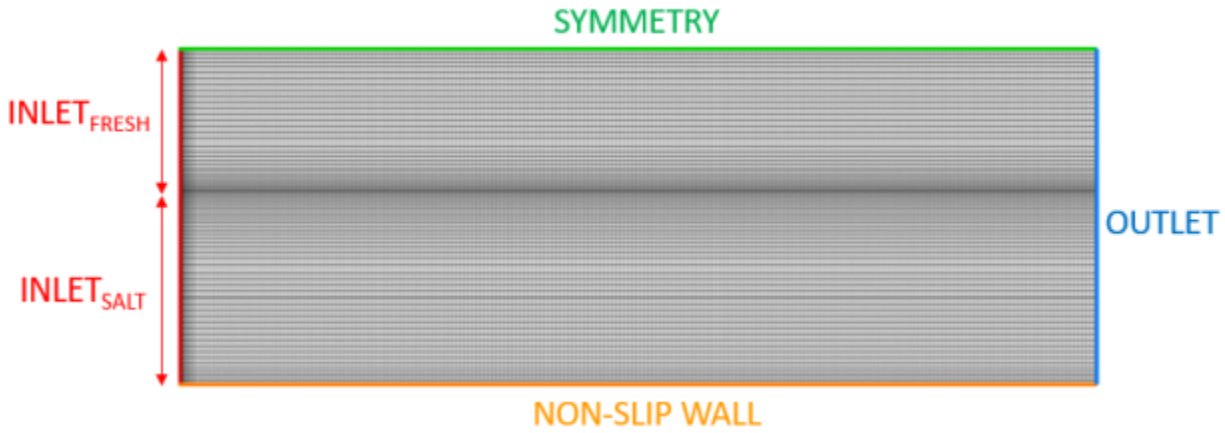


Figure 70 shows comparison of the density and velocity profiles at three different sections for the SST model without and with buoyancy correction (SST-BC). X = 5 [m] is the section near the inlet boundary and X = 40 [m] corresponds to the outlet location. The non-dimensional density variable computes as $\frac{\rho}{\rho_{ref}} = \frac{\rho - \rho_{fresh}}{\rho_{salt} - \rho_{fresh}}$. It is evident from Figure 12.133: Density field for the SST (Left) and SST-BC (Right) model. (p. 342) that without considering buoyancy effects the stabilizing effect of the density stratification is not felt and the rate of mixing of the two fields is excessive. The buoyancy production terms reduce the turbulence kinetic energy generated in the shear layer as shown in Figure 12.134: Turbulence kinetic energy field for the SST (Left) and SST-BC (Right) model. (p. 342). It leads to a reduction of the eddy viscosity and the Reynolds shear stresses. As a result, the mixing layer develops in better agreement with the experiment. The station furthest downstream at X=40[m] indicates that the buoyancy correction imposes too strong a damping effect on turbulence as seen by the lack of mixing in both the density and the velocity profiles. Default settings were used and no effort was undertaken to fine-tune the model.

Figure 12.131: Density and streamwise velocity profiles at different sections for the stratified mixing layer.

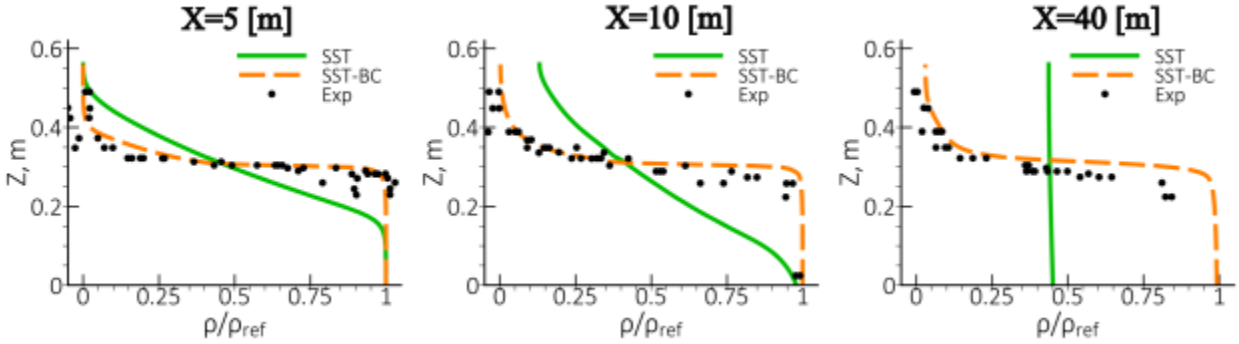


Figure 12.132:

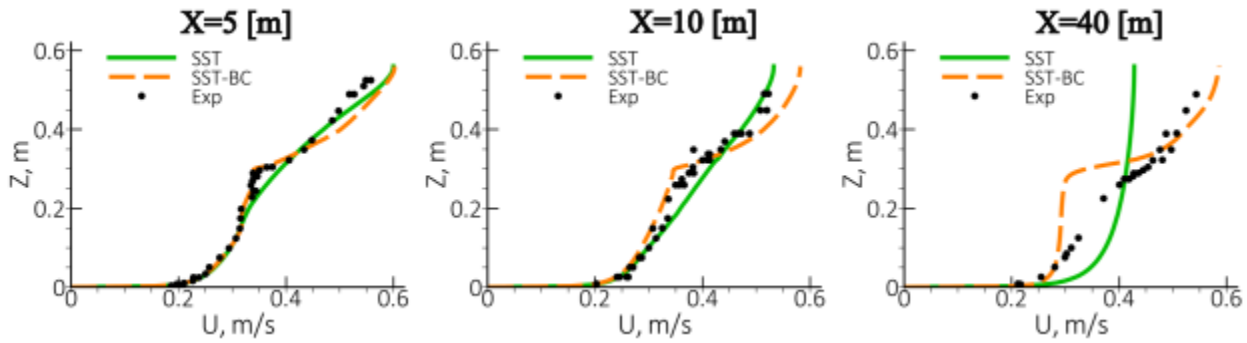


Figure 12.133: Density field for the SST (Left) and SST-BC (Right) model.

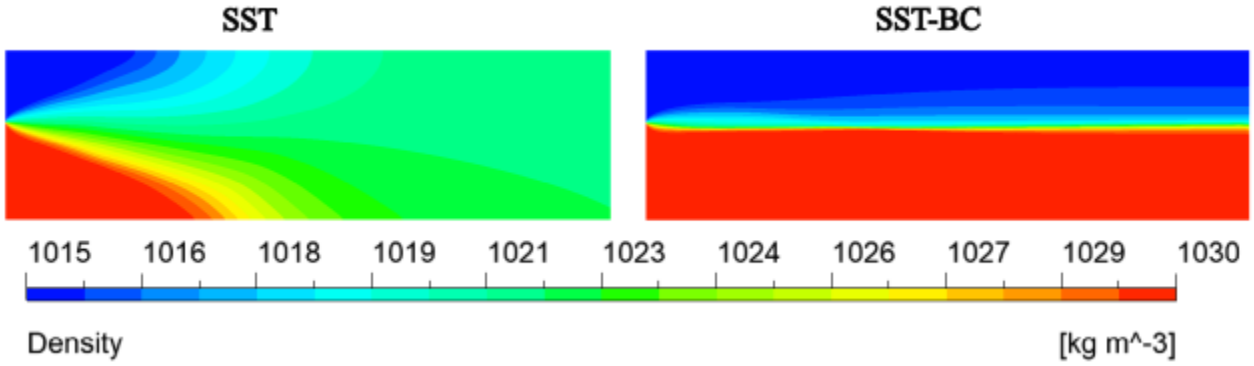
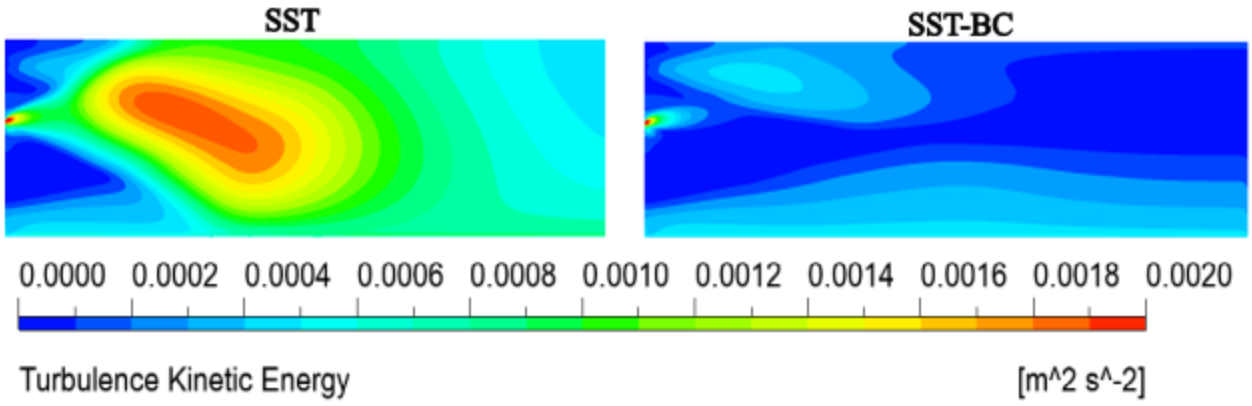


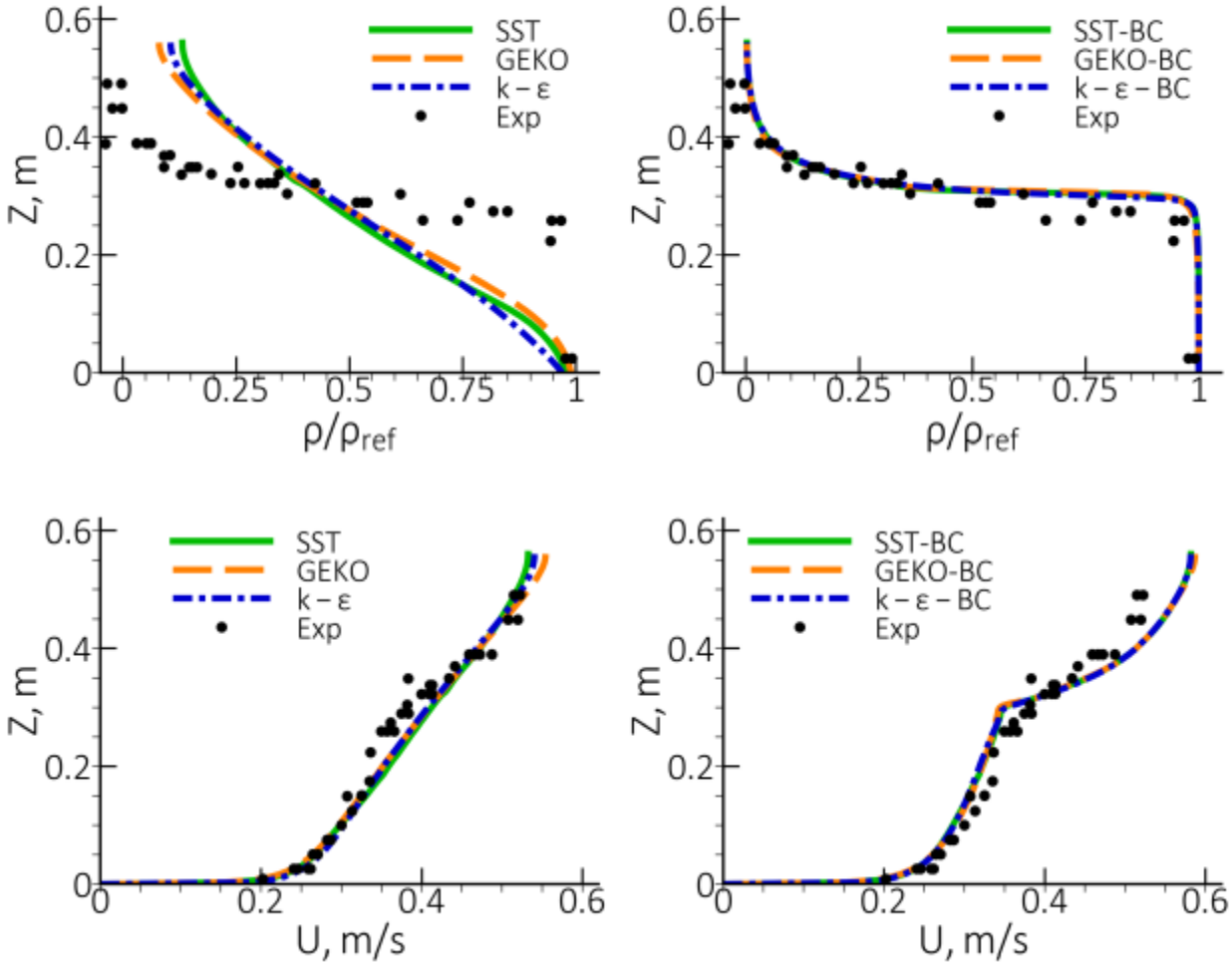
Figure 12.134: Turbulence kinetic energy field for the SST (Left) and SST-BC (Right) model.



The comparison of different turbulence models shown in Figure 12.135: Density (upper row) and streamwise velocity (lower row) profiles for diferent models for the stratified mixing layer at X = 10 [m] section. (p. 343) demonstrates that the sensitivity to the turbulence model is negligible and much smaller than the buoyancy effects. Due to similar results for different sections only X = 10 [m] is shown.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.135: Density (upper row) and streamwise velocity (lower row) profiles for diferent models for the stratified mixing layer at X = 10 [m] section.



12.2.3.8. Effect of Limiters

As mentioned in section 3.3.5 the use of limiters in the formulation of the eddy-viscosity or the production term prevents the formation of unphysically high turbulent kinetic energy levels and, therefore, high eddy-viscosity levels near stagnation regions. It should be noted that the unphysical build-up of turbulence does not only occur in stagnation regions, but in any other region with non-zero shear strain rate, S, outside shear layers (e.g. also in accelerating inviscid flow).

Detailed descriptions of the limiters are given in Limiters (p. 389). This section demonstrates the effect of realizability limiters, the production limiter and the Kato-Launder limiter for different turbulence models for flow around a NACA-4412 airfoil at α=14°. Note that RKE model features an internal realizability limiter which cannot be turned off. This limiter is not the same as the realizability limiter used in the GEKO model. As shown below, it is also not as effective as the GEKO limiter.

For all the Figures the following notation is used:

- $\mathrm{Real}_{\mathrm{lim}}$ – Realizibility limiter (Eq (9.34))
- $\mathrm{P}_{\mathrm{k,lim}}$ – Production term limiter (Eq (9.53))

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- $P_{k,KL}$ – Kauto-Launder production term formulation (Eq (9.52))

As seen in Figure 12.136: Effect of limiters for the GEKO model near the stagnation point for flow around airfoils (p. 344) Figure 74, Figure 12.137: Effect of production limiter for the $k-\varepsilon$ Standard model near the stagnation point for flow around airfoils. (p. 344) Figure 75 and Figure 12.138: Eddy-viscosity field predicted with the RKE turbulence model near the stagnation region for flow around airfoils. (p. 345), all the considered limiters avoid the eddy-viscosity and turbulent kinetic energy build-up in the stagnation regions of a NACA 4412 airfoil. For the GEKO model, the default $\mathrm{Real}_{\mathrm{lim}}$ has the same effect as other limiters. The effect of the production limiter is strongest for the $k-\varepsilon$ Standard model. On the other hand, the in-built realizability limiter of the $k-\varepsilon$ RKE model improves the solution in the stagnation region, however, the model still predicts high eddy-viscosities outside the boundary layer. Again, the use of the $P_{k,\mathrm{lim}}$ fixes this issue.

In case of laminar-turbulent transition modeling, the effectiveness of limiters is very important, even small turbulence build-up can affect the transition onset location. Typically, a strong combination of limiters is used (Pk limiter and Kato-Launder for the SST model).

Figure 12.136: Effect of limiters for the GEKO model near the stagnation point for flow around airfoils

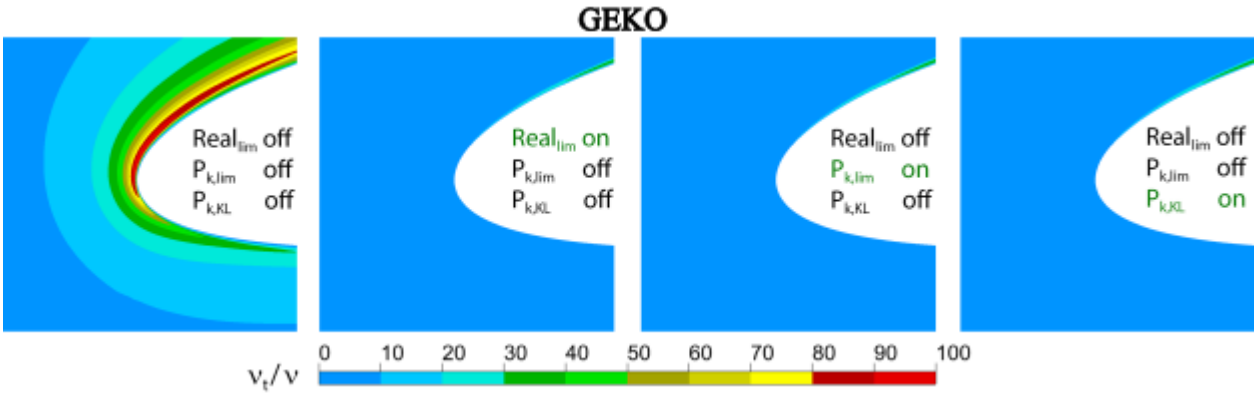
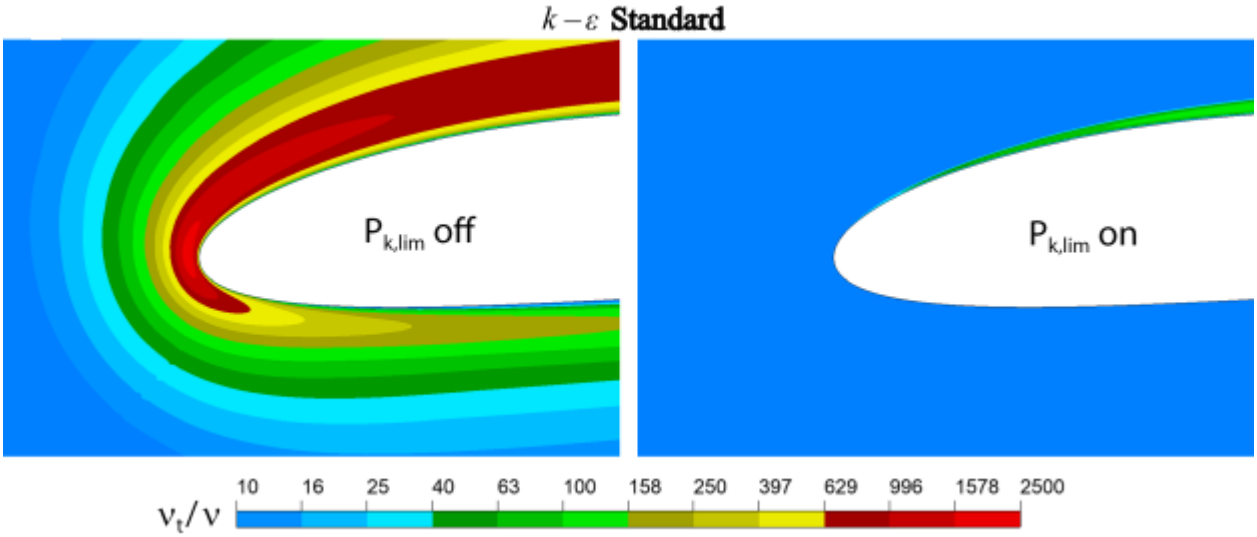
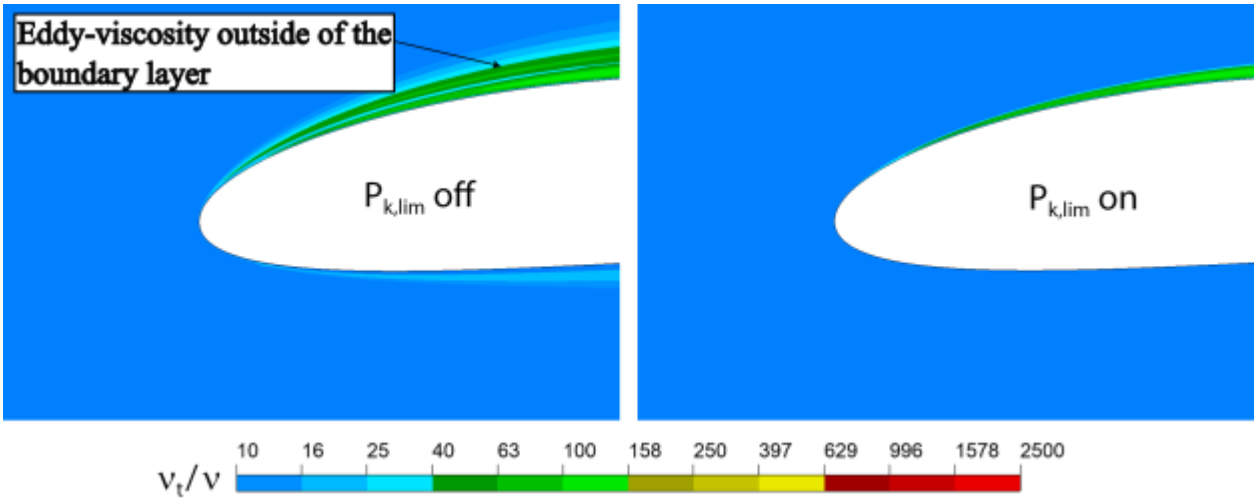


Figure 12.137: Effect of production limiter for the $k-\varepsilon$ Standard model near the stagnation point for flow around airfoils.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.138: Eddy-viscosity field predicted with the RKE turbulence model near the stagnation region for flow around airfoils.



12.2.3.9. Mesh Resolution Requirements

Mesh resolution and mesh quality are key elements for a successful CFD simulation. There are several nested requirements for mesh resolution:

The following topics are discussed:

12.2.3.9.1. Inviscid Flow
12.2.3.9.2. Free Shear Flows
12.2.3.9.3. Fully Turbulent Boundary Layers
12.2.3.9.4. Transitional Boundary Layers
12.2.3.9.5. Corner Flows

12.2.3.9.1. Inviscid Flow

The most basic requirement is that the grid must be able to resolve the inviscid parts of the flow. This means that regions of flow acceleration/deceleration as well as streamline curvature etc. must be properly resolved. This also includes the resolution of strong gradients, especially shocks. Inviscid flow resolution is typically of importance in regions of geometrical changes (corners, edges, trailing edges, leading edge suction peaks, inviscid vortex flows, ….). Insufficient mesh resolution in this context is typically noticeable through changes in pressure distribution under mesh refinement.

12.2.3.9.2. Free Shear Flows

Free shear flows, like mixing layers, jets or wakes typically require at least ~10 cells normal to the layer. The resolution in stream/spanwise direction is usually of the order of the shear layer thickness.

There are two main challenges with free shear layers. The first is that they can be very thin at their origin. It is therefore hard to capture the initial formation of such layers – in most cases they are therefore under-resolved. The second challenge is that their location is often not known before

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the simulation. One should therefore investigate the solutions and refine the meshes if important shear layers have been missed or under-resolved by the mesh. Alternatively, the layers can be targeted by mesh adaptation using adaptation criteria indicative of the layer (gradients of velocity or turbulent kinetic energy etc.).

Figure 12.139: Computational meshes for the mixing layer. (p. 346) shows examples of coarse, medium and fine meshes which correspondingly have ~ 5/10/20 cells across the mixing layer. For all the meshes the streamwise resolution is identical. The coarse mesh does not provide the grid-independent solution while results on the medium and fine meshes are in perfect agreement. This is shown on Figure 12.140: Comparison of the velocity (left), turbulence kinetic energy (middle) and Reynolds shear-stress (right) profiles on different meshes for the mixing layer. (p. 346) where self-similar profiles of the streamwise velocity component, turbulence kinetic energy and Reynolds shear stress are plotted for the SST model.

Figure 12.139: Computational meshes for the mixing layer.

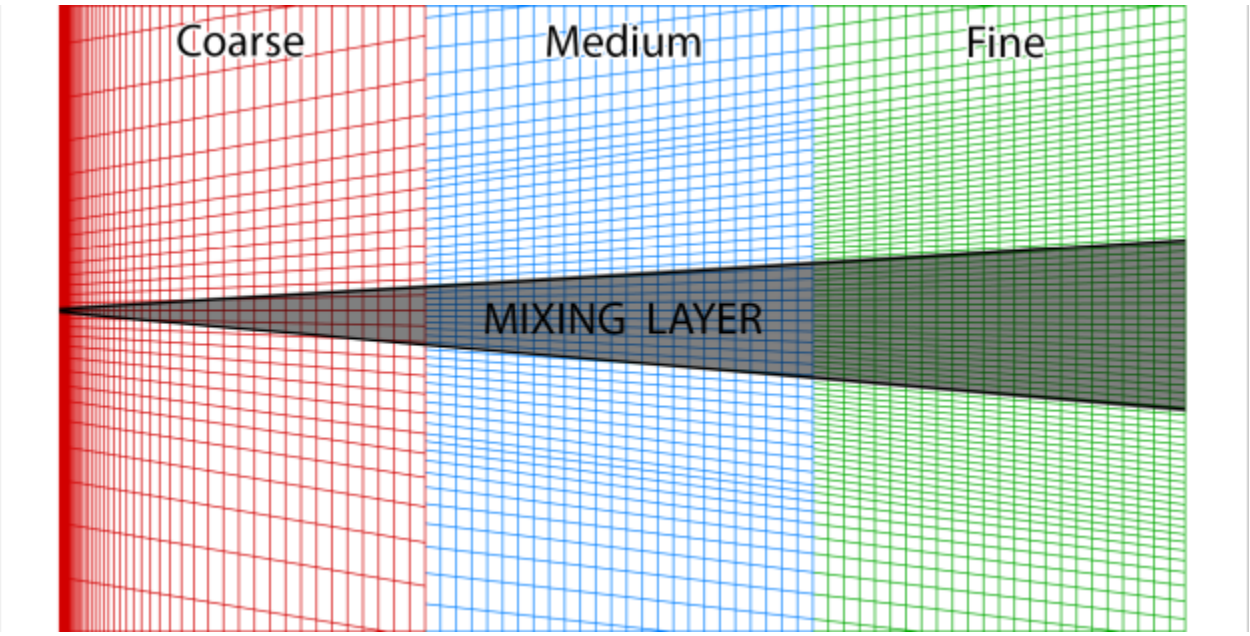
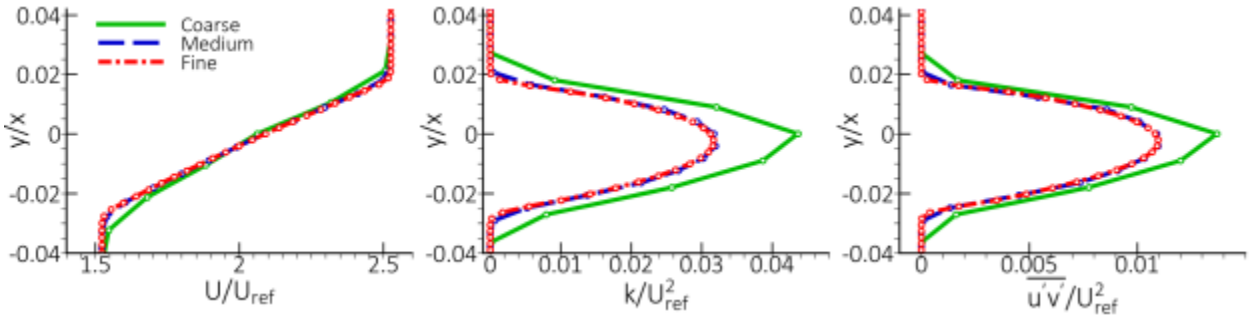


Figure 12.140: Comparison of the velocity (left), turbulence kinetic energy (middle) and Reynolds shear-stress (right) profiles on different meshes for the mixing layer.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.3.9.3. Fully Turbulent Boundary Layers

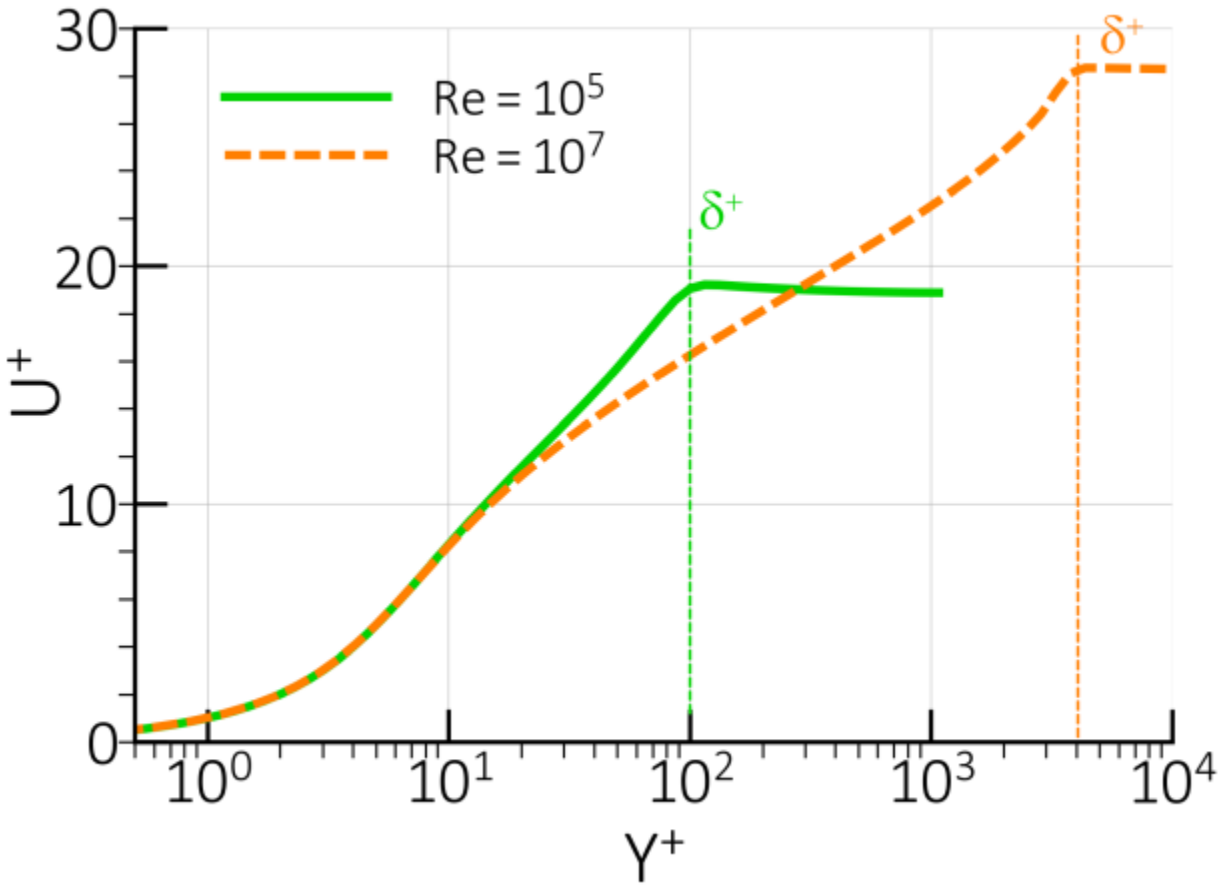
Wall boundary layers are the most demanding in terms of mesh resolution, as the solution gradients inside boundary layers can be very high. This is true for both, primary solution variables, like velocity or temperature, but even more so for turbulence quantities like k, ω ? or ε.

Most CFD users are fixated on the y^+ values of the first cell center of the wall cells and assume that achieving a given threshold y^+ value ensures enough mesh resolution inside the boundary layer. This is not the case as a given y^+ value does not by itself ensure that there are enough cells inside the boundary layer. It is important to understand that for the same y^+-value and the same grid expansion ratio (growth rate between cells

$$ER=\Delta y_{j+1}/\Delta y_j$$

there would be many more cells inside a high Reynolds number boundary layer than in a low Reynolds number one. This is clear from Figure 12.141: Boundary layer thickness in wall units for low and high Reynolds number. (p. 347) which shows that the boundary layer thickness in terms of δ^+ is much larger for the high than for the low Reynolds number case.

Figure 12.141: Boundary layer thickness in wall units for low and high Reynolds number.

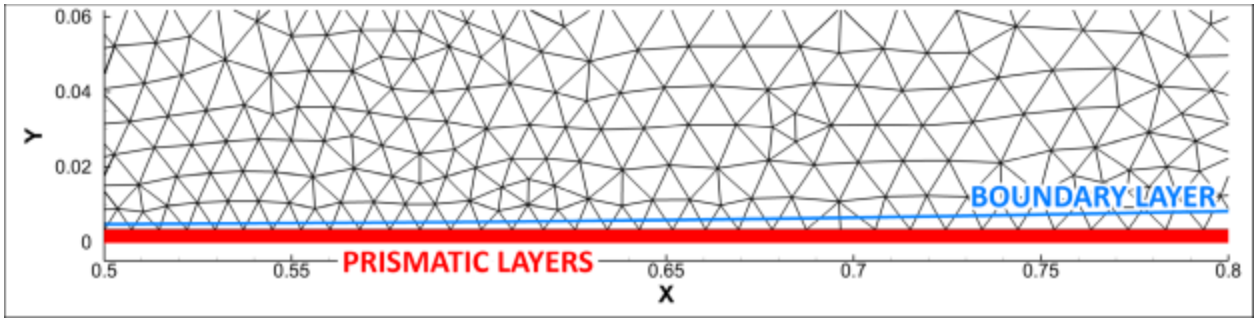


The main quality criterion for boundary layer meshes is therefore not y^+, but the number of cells (prism layers) inside the boundary layer (estimates of boundary layer thickness are given in Section

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

9.4). It is important to stress the word 'inside', as prism layers generated with a given number of cells do not ensure that the layers are resolving the boundary layer – it is easy to create prism layers where the boundary layer is much thinner or much thicker than the prism layer height. If prismatic layers cover only a part of the boundary layer (see Figure 12.142: Computational unstructured mesh with prismatic layers (red) thinner than the boundary layer (blue line) thickness. (p. 348)) it is under-resolved due to a large edge length change from the prism layers to the tetra/poly cells. In this case, the outer part of the boundary layer cannot be resolved.

Figure 12.142: Computational unstructured mesh with prismatic layers (red) thinner than the boundary layer (blue line) thickness.



There is no single number of how many cells across a boundary layer are optimal, as this depends on the type of flow and the required accuracy. The highest accuracy demands are for external aeronautical applications with safety concerns – especially aircraft simulations. For such flows, boundary layers are often resolved with 30-40 cells. For most industrial flows, 10 layers inside the boundary layer are reasonable. However, there are also many technical flows, where only smaller cell counts can be afforded, and reasonable solutions can often be achieved with only 3-5 layers.

The need to accurately resolve the boundary layer increases with the need:

- To resolve flow separation from smooth surfaces like airfoils (contrary to separation from geometric discontinuities like edge or corners).
- Flows with heat transfer, where the resolution of conditions in the very near wall region is important. This is even more important for flows with high Prandtl numbers where the thermal sublayer is very thin.
- Flows with laminar-turbulent transition where the laminar boundary layers are typically very thin and needs to be resolved for the transition model to operate properly (see below).

Streamwise resolution is typically several times the boundary layer thickness (say1-5) whereas 'spanwise' grid spacing can be even higher depending on the gradients in that direction (for 2D flows it can be arbitrarily coarse). However, in most technical flows, there is no clear distinction between streamwise and spanwise and the grid surface spacing is then typically selected as 'isotropic'.

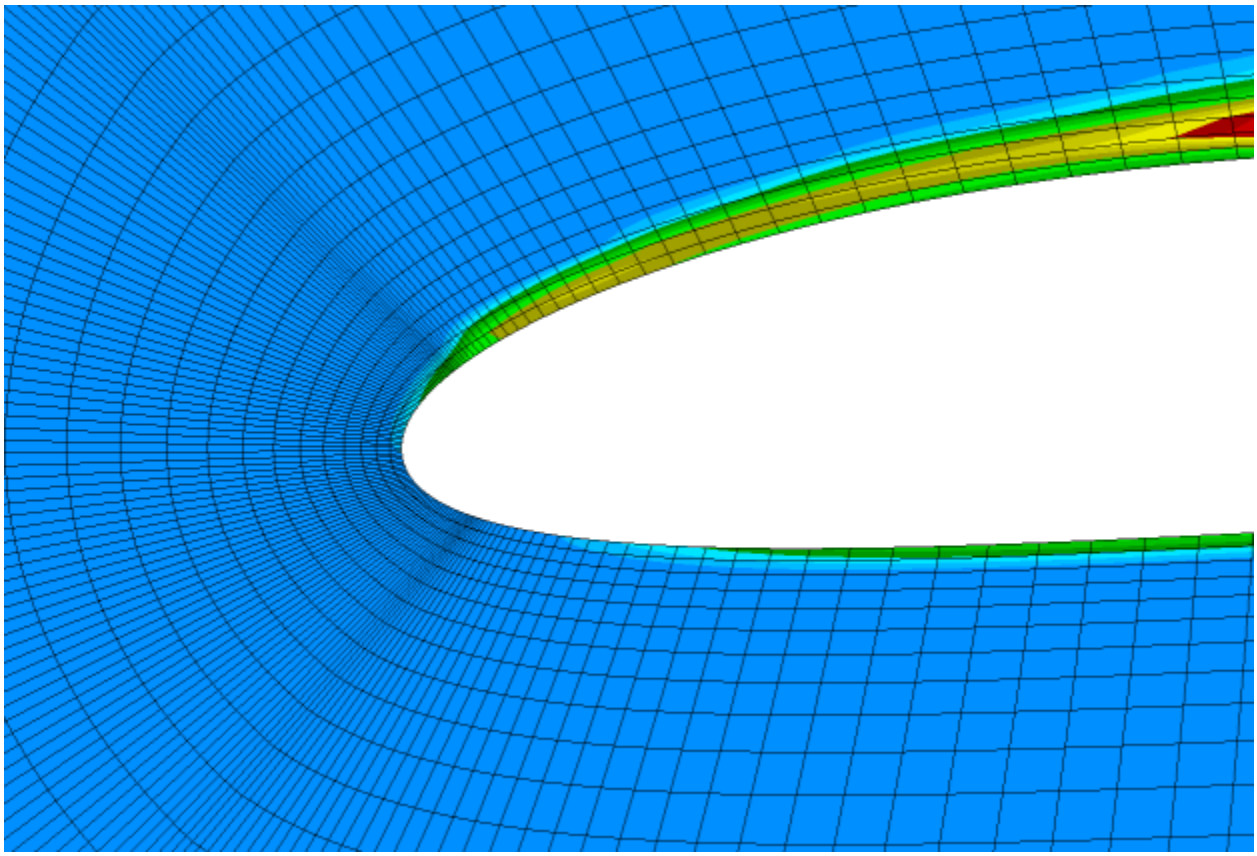
Grid resolution of boundary layers should be checked during and/or after the simulation. A good quantity to visualize is the ratio of turbulent to molecular viscosity ($TVR=\mu_t/\mu$). It has a maximum near the center of the boundary layer and decays to low freestream values outside the layer. Visualizing TVR in a contour plot with overset grid lines allows a good estimate of grid resolution by counting the wall normal cells which are inside the elevated TVR levels near the wall. This is shown in Figure 12.143: The boundary layers thickness visualized by eddy-viscosity ratio field for

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

different meshes around NACA-4412 airfoil. Blue color shows the area of the inviscid flow. (p. 349) for different grids for a NACA 4412 airfoil. Here the fine grid provides a grid-independent solution while the boundary layer is under resolved near the leading edge for the coarse and medium grids. There are ~ 5 cells across the boundary layer near the leading edge for the medium mesh and the coarse mesh does not resolve the boundary layer at all. As a result, the boundary layer is thicker for these meshes which leads to earlier separation (see Figure 12.145: Comparison of velocity profiles on different meshes for flow around the NACA-4412 airfoil. (p. 351)) on the suction side of the airfoil. The local mesh refinement in wall normal direction in the leading edge region shown in Figure 12.146: Local mesh refinement on the leading edge for the medium mesh. (p. 352) improves the prediction of the boundary layer. Figure 12.147: Comparison of the velocity profiles on the medium and refined medium (medium+LE) meshes. (p. 352) shows that the results on the refined medium mesh and on the fine mesh are in good agreement.

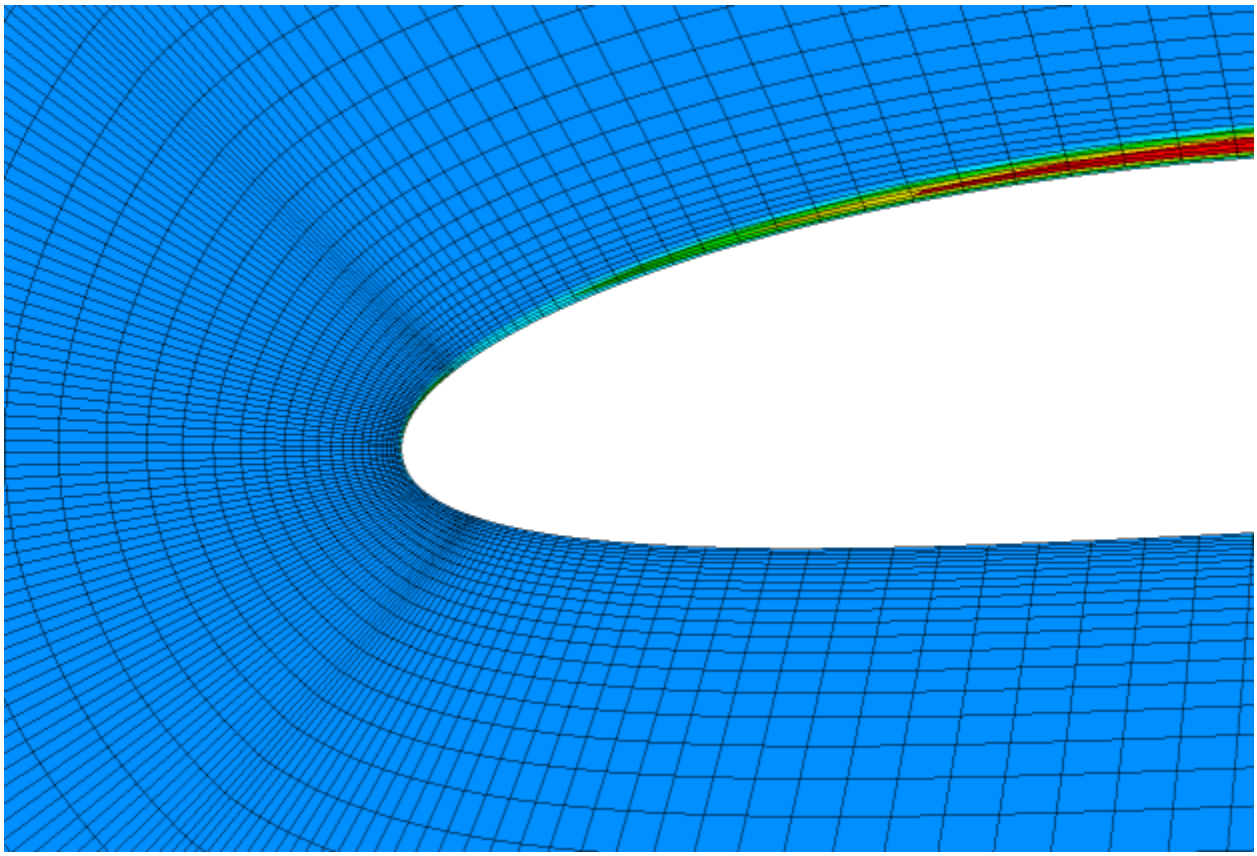
Coarse

Figure 12.143: The boundary layers thickness visualized by eddy-viscosity ratio field for different meshes around NACA-4412 airfoil. Blue color shows the area of the inviscid flow.



Medium

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Fine

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.144:

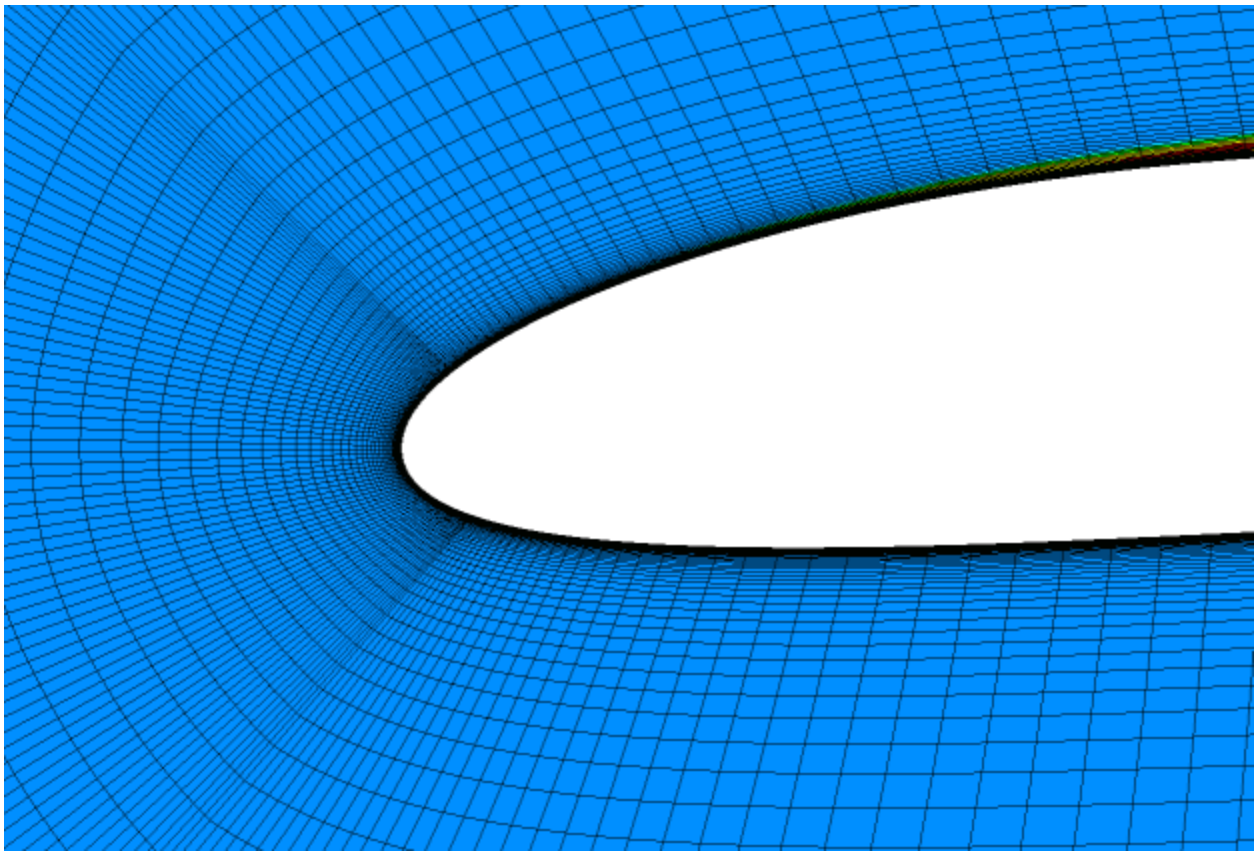
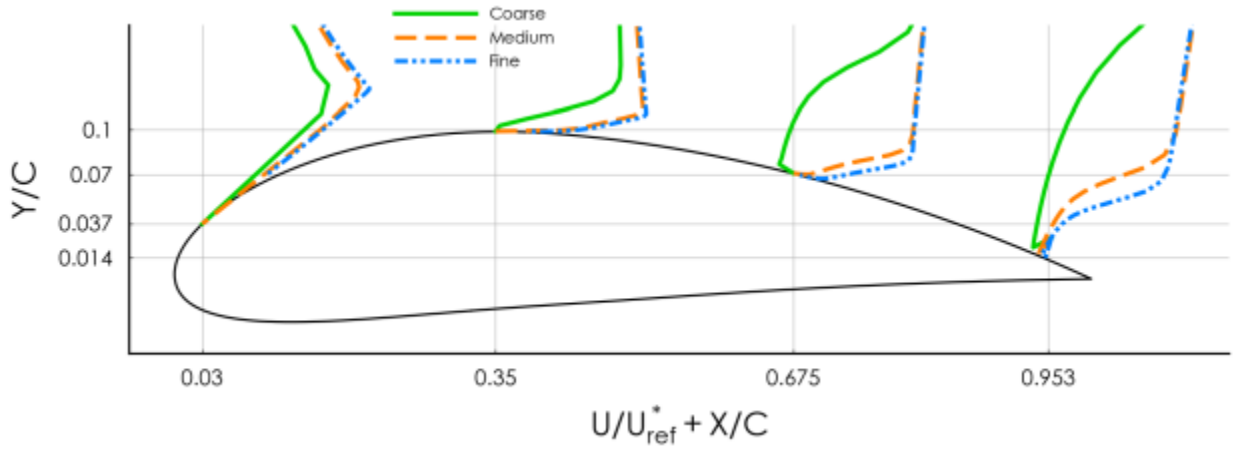


Figure 12.145: Comparison of velocity profiles on different meshes for flow around the NACA-4412 airfoil.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.146: Local mesh refinement on the leading edge for the medium mesh.

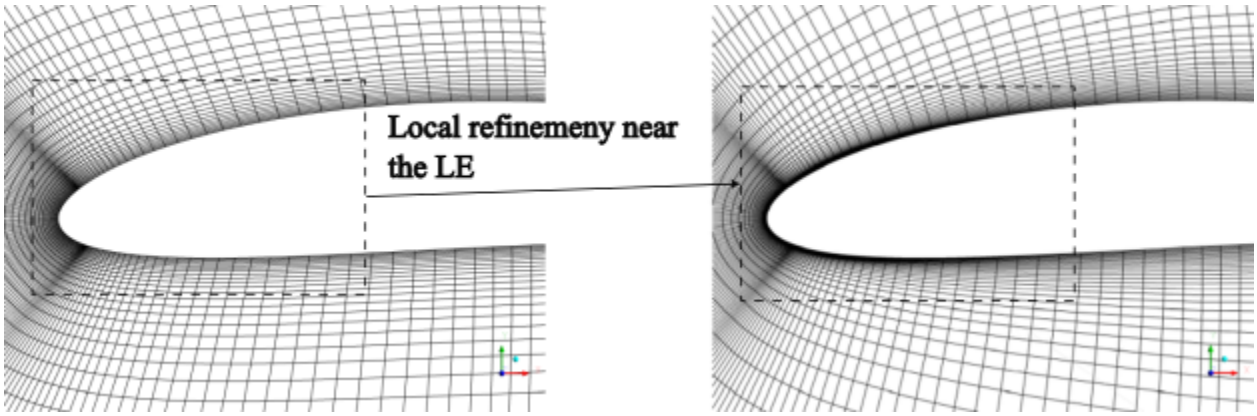
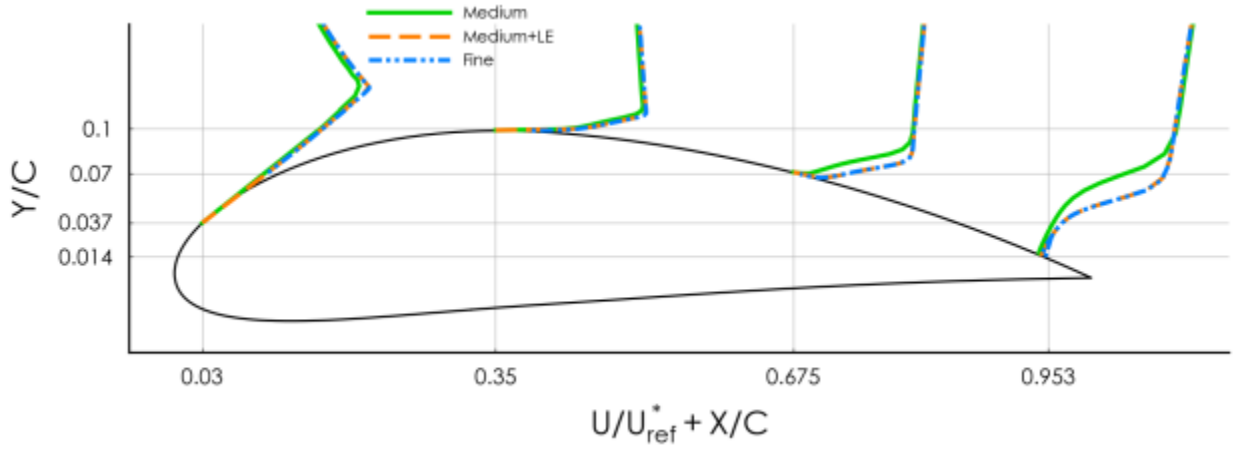


Figure 12.147: Comparison of the velocity profiles on the medium and refined medium (medium+LE) meshes.



Another grid-quality measure is the smoothness by which prism layers merge into the outer unstructured mesh. Grids with large size jumps in this region should be avoided as they can inhibit the proper growth of the boundary layer (which then cannot grow outside the prism layer mesh due to low resolution). The optimal mesh structure:

- Prismatic layers cover the entire boundary layer
- The number of prismatic layers should be sufficient for the boundary layer resolution.
  - The first near wall grid step ensures $y^+<1$ without wall treatment.
  - However, higher y^+ values can also be applied if the overall number of cells across the layer is sufficient.
- Smooth connection of the prismatic layers and tetra/poly cells.
- Streamwise grid step at the wall should be sufficient for the proper resolution of the flow features.
  - stagnation region

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- separation
- shock wave
- transition
- corner flow

Figure 12.148: Optimal unstructured mesh topology for flow around airfoil. (p. 354) illustrates the optimal mesh topology and resolution for an unstructured mesh for the flow around an airfoil.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.148: Optimal unstructured mesh topology for flow around airfoil.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.3.9.4. Transitional Boundary Layers

Boundary layers undergoing transition are more sensitive to the mesh resolution than fully turbulent boundary layer. Meshes for fully turbulent simulations are not sufficient in the transition region especially in the streamwise direction. Unfortunately, the transition location is unknown without precursor simulations in most cases. Therefore, a two-stage approach is recommended for the computations using transition models:

- 1st stage – computations on the baseline fine fully turbulent mesh and estimation of the transition region.
- 2nd stage – mesh refinement in streamwise, wall-normal and spanwise (in case of the fully three dimensional flow) directions in the transition region. Repeat until grid-converged solution is achieved.

It should be noted that the mesh requirements also depend on the transition model choice. Here the three transition models described in section 3.4.1 are considered. Figure 12.150: Effect of the resolution of transition region in streamwise direction for different transition models for the T3A flat plate boundary layer. (p. 357) shows the effect of streamwise mesh resolution on the transition prediction on the flat plate for different transition models. The baseline fully turbulent mesh with streamwise grid step $\Delta x=10\delta$ (δ is boundary layer thickness) is only sufficient for the $\gamma-\mathrm{Re}_{\theta}-\mathrm{SST}$ model but not for the other two models and should be refined in the transition region. The baseline and refined meshes are shown on Figure 12.149: Computational meshes for the fully turbulent (upper) and transitional (lower) boundary layer. (p. 356). For the refined mesh, the streamwise grid step is $\Delta x=1\delta$ in the transition region and $\Delta x=10\delta$ in the laminar and fully turbulent parts. In this context it is important to note that for many technical devices, transition is triggered through laminar-turbulent bubble separation. In such cases, the streamwise mesh resolution is even more important in order to detect and resolve the laminar bubble. It is usually required to resolve the bubble with 5-10 streamwise cells. For this reason, it is not recommended in general to use coarse Δx spacings in transition simulations, as the mechanism of transition is often not known a priori.

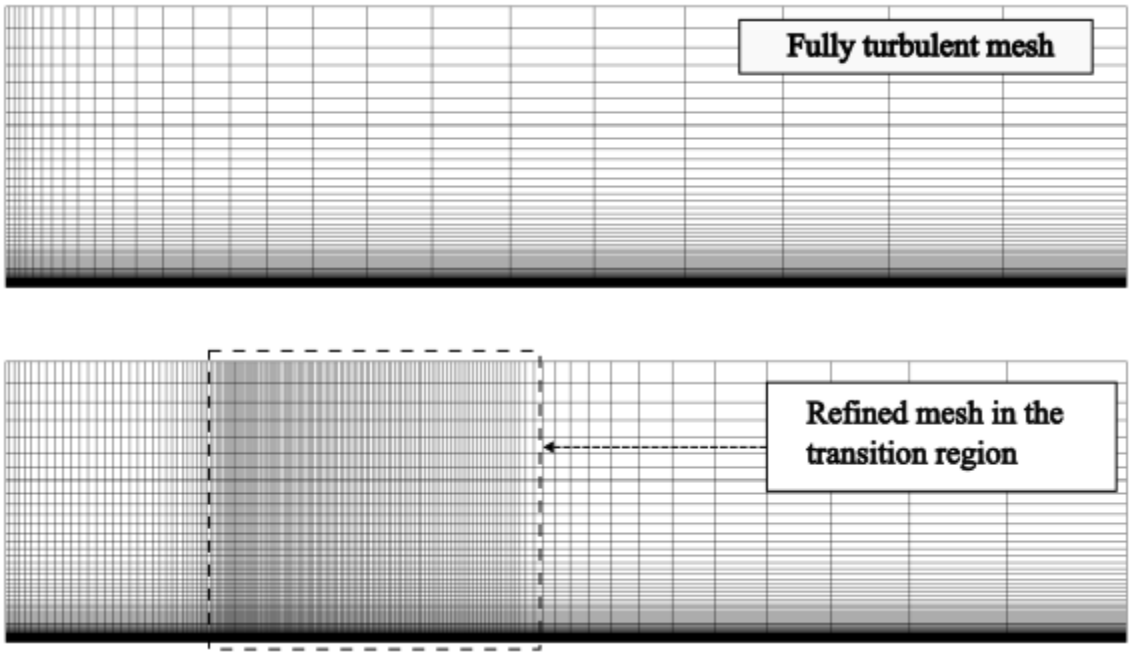
In case of first near wall height variation the transition location is insensitive to $y_1{}^+<1$ in the transition region (see Figure 12.151: Effect of the resolution of transition region in wall-normal wall direction for different transition models for T3A the flat plate boundary layer. (p. 358)). For higher values of $y_1{}^+$ in this area, the transition location moves upstream. For both streamwise and wall-normal mesh studies the two-equation model ($\gamma-\mathrm{Re}_{\theta t}-\mathrm{SST}$) is much less sensitive to the mesh resolution than one-equation intermittency transition model ($\gamma-\mathrm{SST}$), or the algebraic intermittency model ($\gamma-\mathrm{alg}-\mathrm{SST}$), which is most sensitive to the mesh resolution.

Finally the basic recommendation for all the transition models for the mesh generation.

- $\Delta x=1\delta$ in the transition region
- $\Delta y_1{}^+<1$ in the transition region
- $ER<1.1$ in the transition region ($ER=\Delta y_{j+1}/\Delta y_j$)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.149: Computational meshes for the fully turbulent (upper) and transitional (lower) boundary layer.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.150: Effect of the resolution of transition region in streamwise direction for different transition models for the T3A flat plate boundary layer.

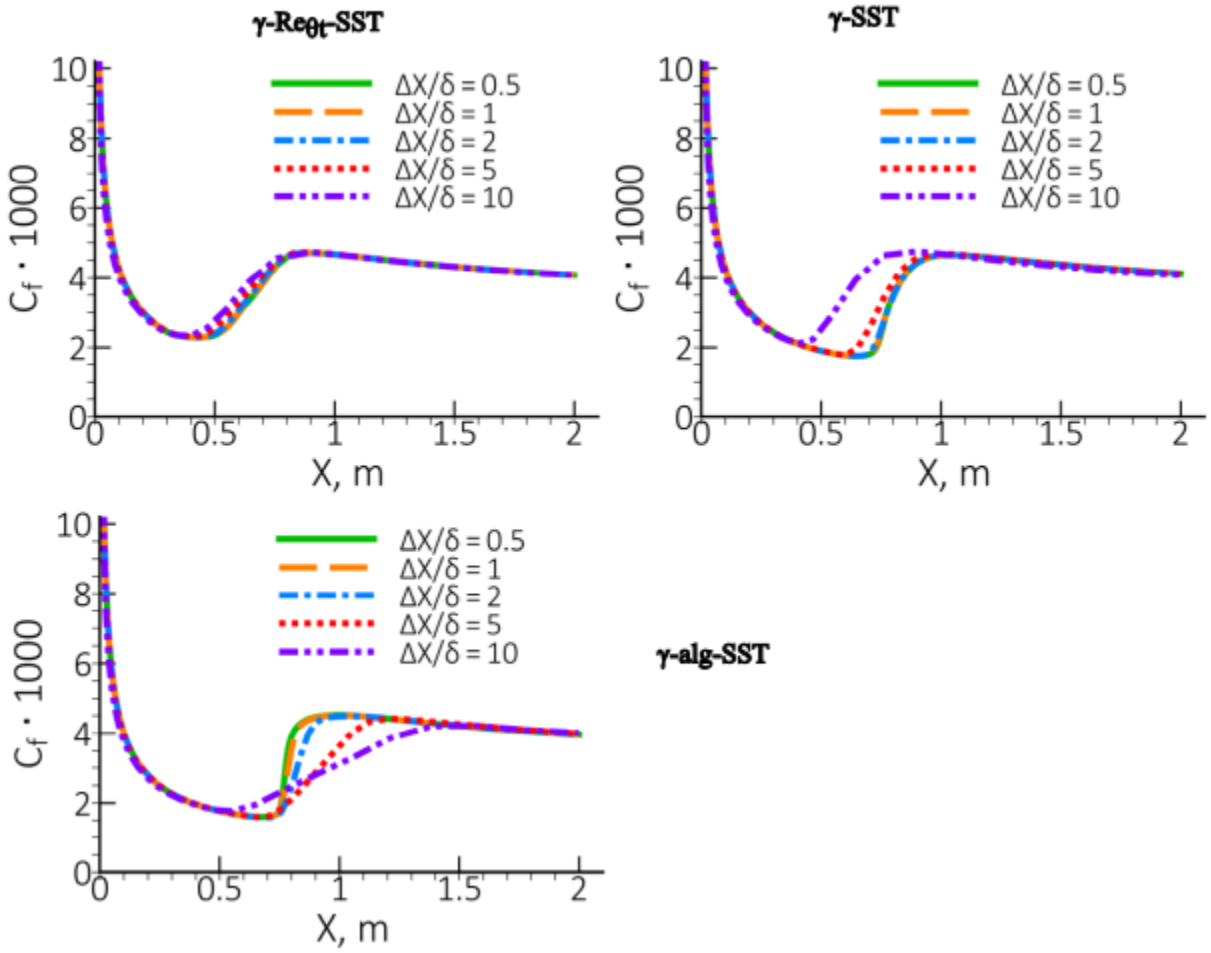
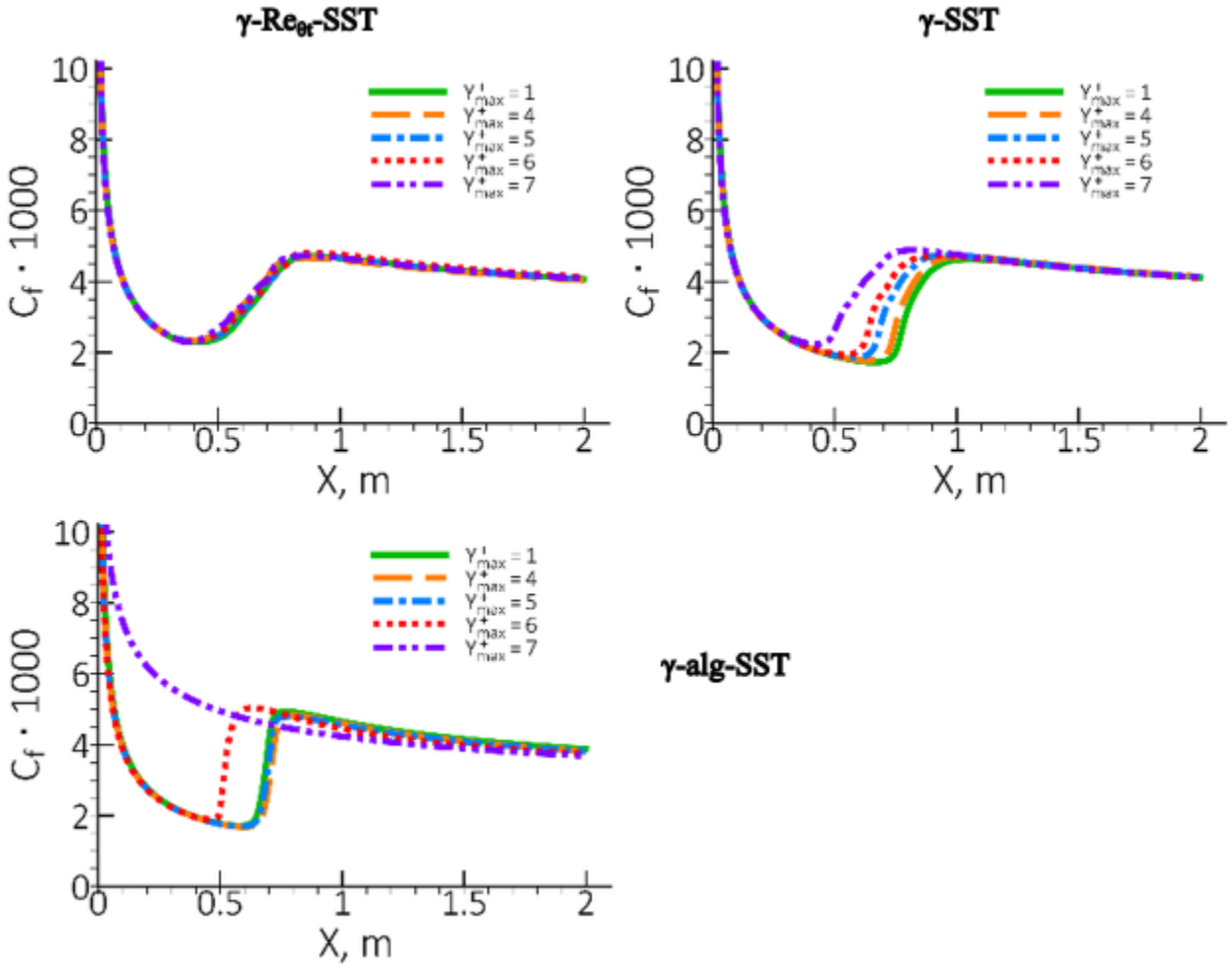


Figure 12.151: Effect of the resolution of transition region in wall-normal wall direction for different transition models for T3A the flat plate boundary layer.



12.2.3.9.5. Corner Flows

When activating the Corner Flow Correction or an EARSM to compute the secondary flow into corners, the mesh resolution into the corners especially for unstructured meshes (tetrahedral, polyhedral or cartesian) is critical.

Ideal meshes for corner flows are structured meshes as shown in Figure 12.152: Baseline structured mesh with fine resolution in the corners for the DLR-Diffuser [41]. (p. 360) where the wall normal refinement for one wall ensures automatically proper refinement into the corner for the adjacent walls (meaning small edge length on the wall going into the corner). Such corner mesh refinement avoids additional mesh refinement in streamwise direction which should be done for the unstructured meshes.

The meshing requirements to resolve the corner flow are especially challenging for the DLR diffuser (41 (p. 373)) described in Section 4.3.2. For this test case, a uniform flow (U=10m/s, Tu=1%, TVR=1) was specified at the inlet. This results in a very thin boundary layer at the onset of the diffuser. While this boundary layer can easily be resolved in wall normal direction, it is much harder to resolve the flow dynamics into the corner (the secondary flow is taking place inside the boundary

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

layer and has to be resolved in the plane normal to the mean flow direction). While the structured mesh (Str mesh) in Figure 12.152: Baseline structured mesh with fine resolution in the corners for the DLR-Diffuser [41]. (p. 360) has sufficient resolution in that area due to its mesh topology, the resolution of this zone by unstructured meshes is much harder. Two unstructured meshes, one without refinement (PH-1) and one with refinement (PH-2) into the corner are shown in Figure 12.153: Baseline mesh PH-1 (left) and refinement mesh PH-2 (right) near the corners for the unstructured meshes for the DLR-diffuser [41]. (p. 360) and Figure 12.154: Cross section of typical unstructured mesh PH-1 in rectangle channel (left) and collapsing of the prismatic cells near the corners on the refined mesh PH-2 (right) for the DLR-Diffuser [41]. (p. 361). Figure 12.155: Eddy viscosity of the developing boundary layer at the straight part of diffuser and mesh cross section mesh resolution for different meshes. (p. 362) shows contours of the eddy-viscosity ratio with the mesh plotted on top. It is intuitively clear that mesh PH-1 does not have sufficient resolution into the corner. While the mesh can be refined into the corners (PH-2, it results in a disproportionably large mesh count due to local refinement in all directions. For the current example, the mesh size is increased by a factor ten between the two grids shown. Another issue is that due to the refinement into the corner, the thickness of the prismatic layer shrinks in the corner region (Figure 12.154: Cross section of typical unstructured mesh PH-1 in rectangle channel (left) and collapsing of the prismatic cells near the corners on the refined mesh PH-2 (right) for the DLR-Diffuser [41]. (p. 361)).

The effect of mesh refinement on the structure and strength of the corner vortex can be seen in Figure 12.156: Streamlines of crossflow pattern at the start of the diffuser for three different meshes. (p. 362). Clearly, mesh PH-1 is not sufficient to resolve the vortex. Figure 12.157: Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41]. (p. 363) shows the skin friction prediction on the inclined wall for the DLR diffuser for the two unstructured meshes in comparison with a structured Hex-mesh (Str) using the SST-CFC model. It is well seen that the mesh resolution near the corners is not only important near the corners but also affects the flow in the center. In the current test case, the effect on the center part of the flow is moderate and the predicted pressure distribution (Figure 12.158: Distribution of the pressure coefficient along the diffuser at midsection for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41]. (p. 363)) is in good agreement with the experimental data for both unstructured meshes. Note however that the incorrect prediction of corner flow separation can in the worst-case result in incorrect flow topologies which in turn can have a strong impact on global flow parameters.

It is important to stress, that the challenge for the DLR diffuser lies in the thin boundary layer in the corner region. For flows with a fully developed inlet profile, like the Stanford diffuser [42 (p. 373)], the resolution requirements are much less demanding. In any case, meshes with intersecting mesh lines from the two side walls (like in the structured mesh) are ideal for such simulations.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.152: Baseline structured mesh with fine resolution in the corners for the DLR-Diffuser [41 (p. 373)].

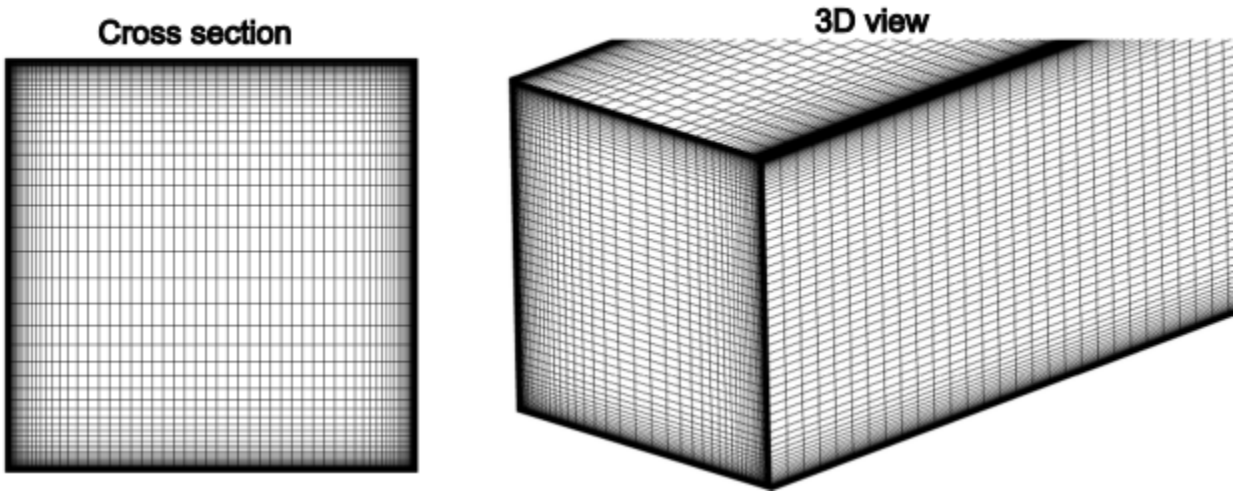
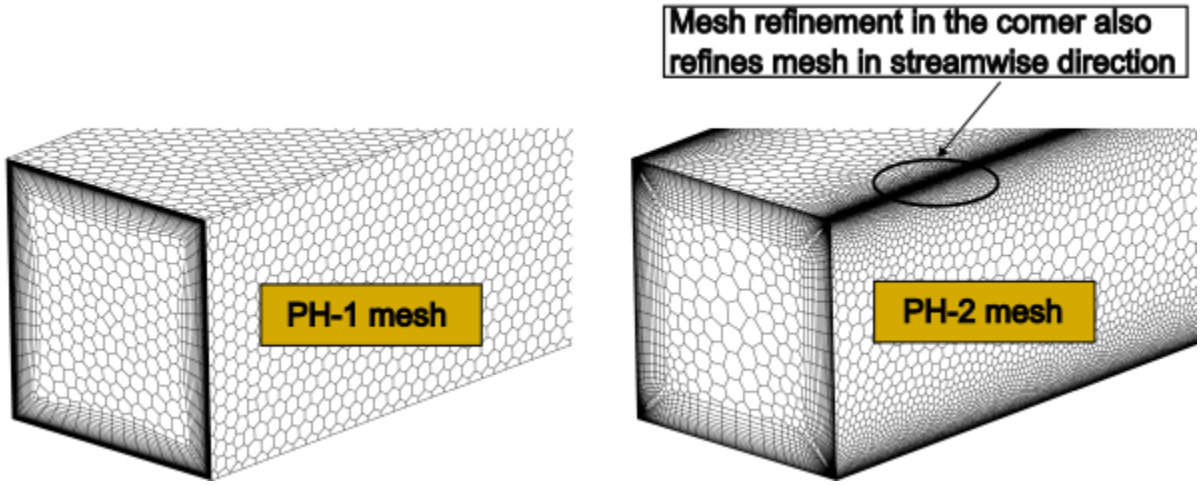
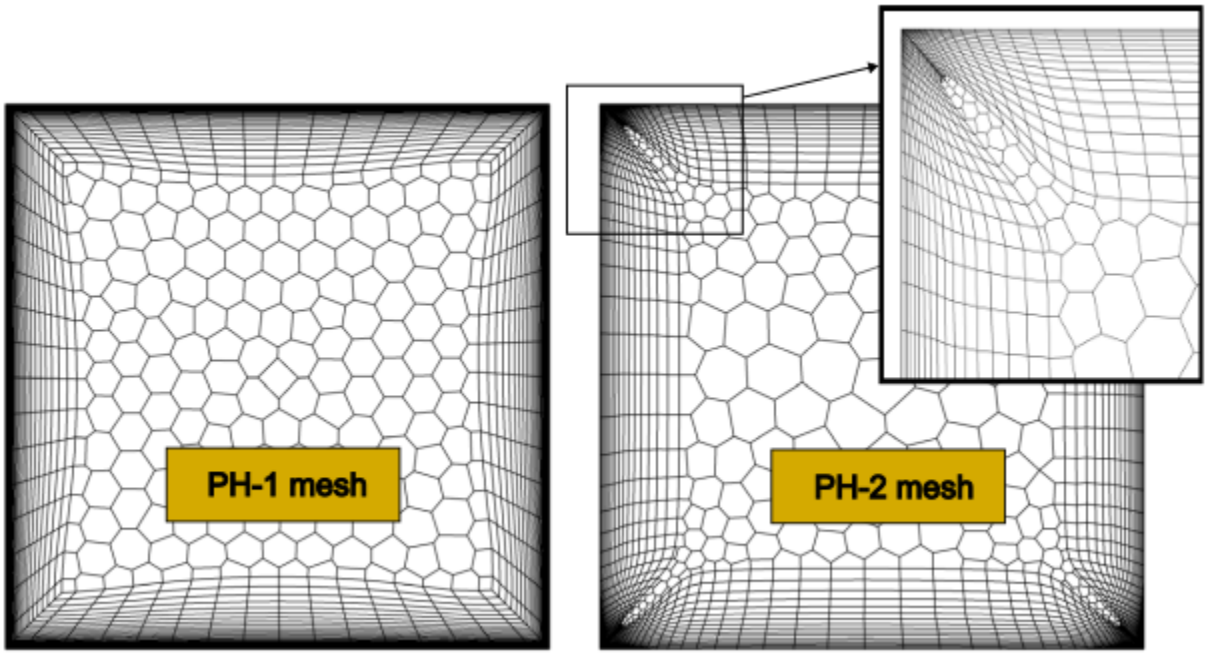


Figure 12.153: Baseline mesh PH-1 (left) and refinement mesh PH-2 (right) near the corners for the unstructured meshes for the DLR-diffuser [41 (p. 373)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.154: Cross section of typical unstructured mesh PH-1 in rectangle channel (left) and collapsing of the prismatic cells near the corners on the refined mesh PH-2 (right) for the DLR-Diffuser [41 (p. 373)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.155: Eddy viscosity of the developing boundary layer at the straight part of diffuser and mesh cross section mesh resolution for different meshes.

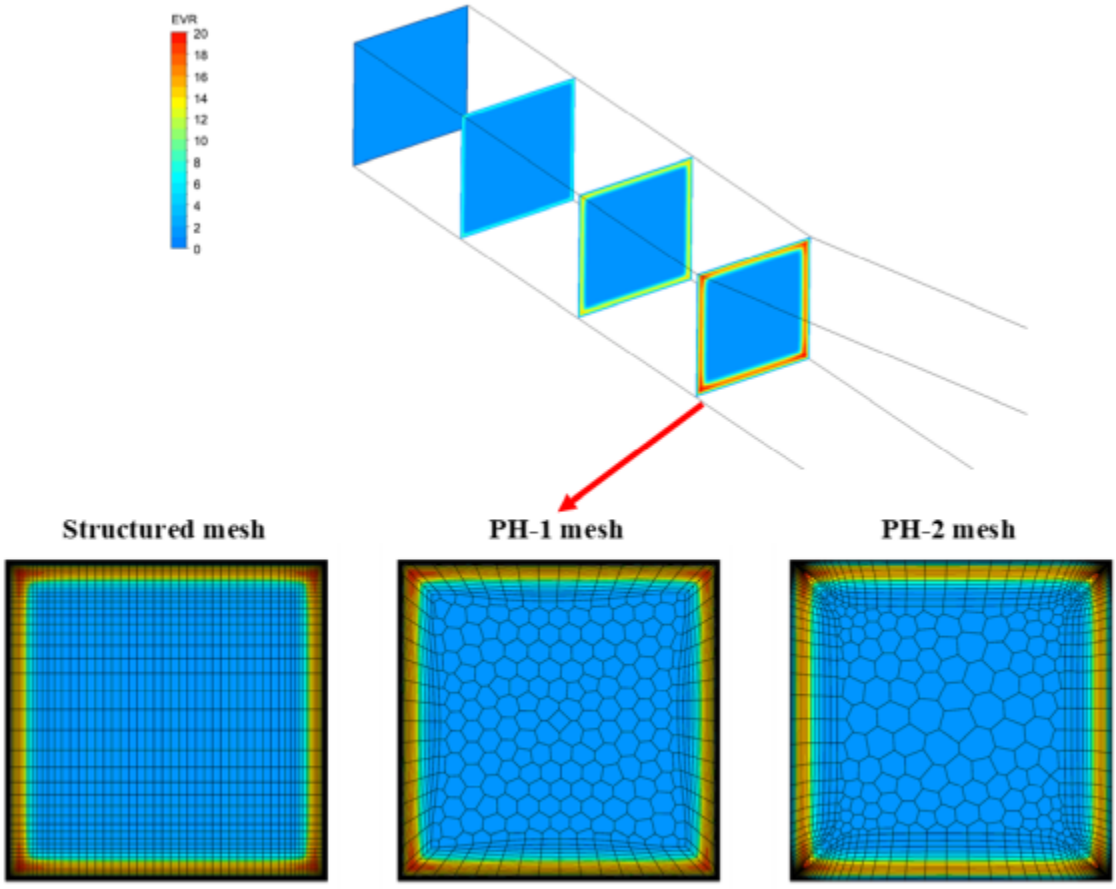
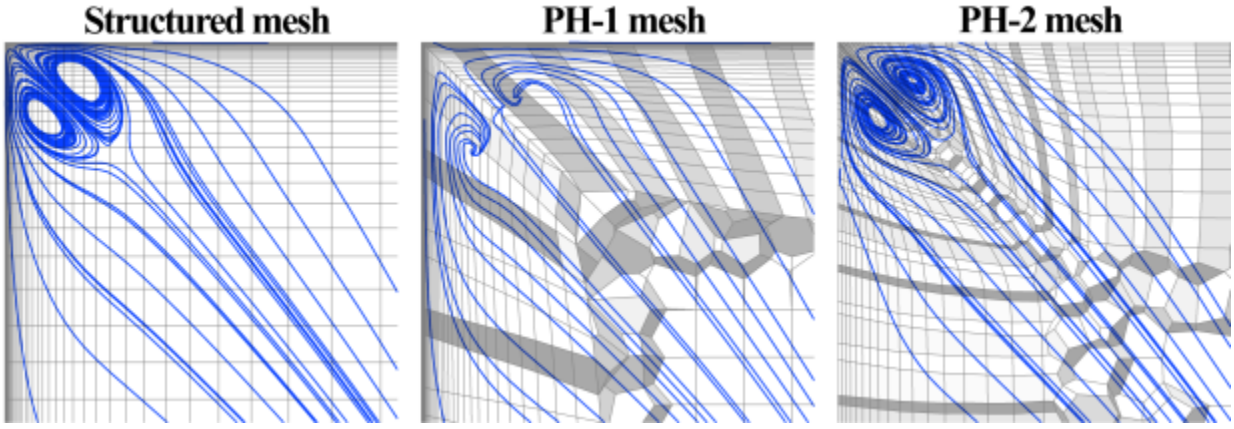


Figure 12.156: Streamlines of crossflow pattern at the start of the diffuser for three different meshes.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.157: Distribution of the skin friction along the diffuser at midsection (left) and near the corner (right) for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41 (p. 373)].

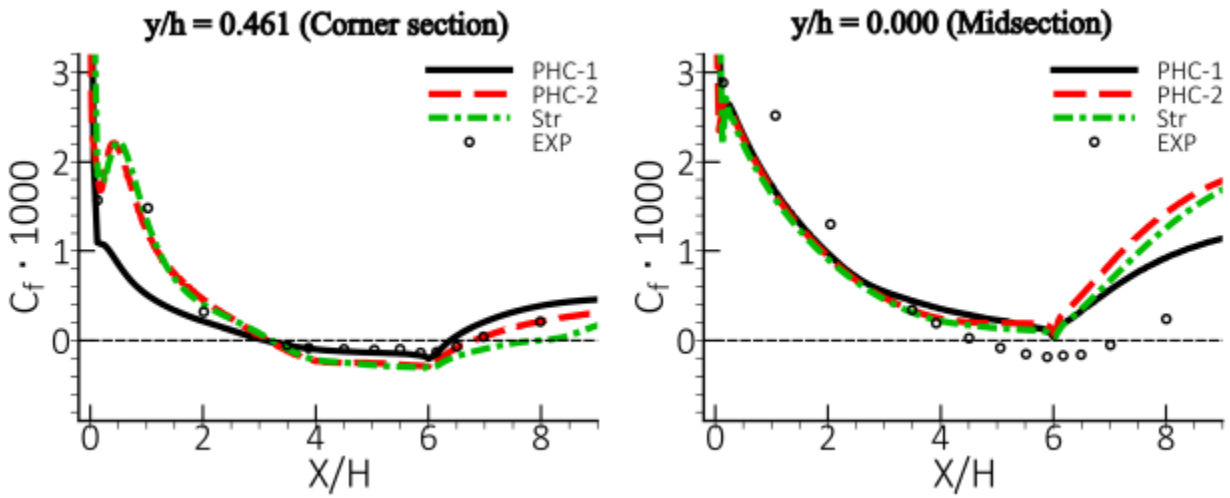
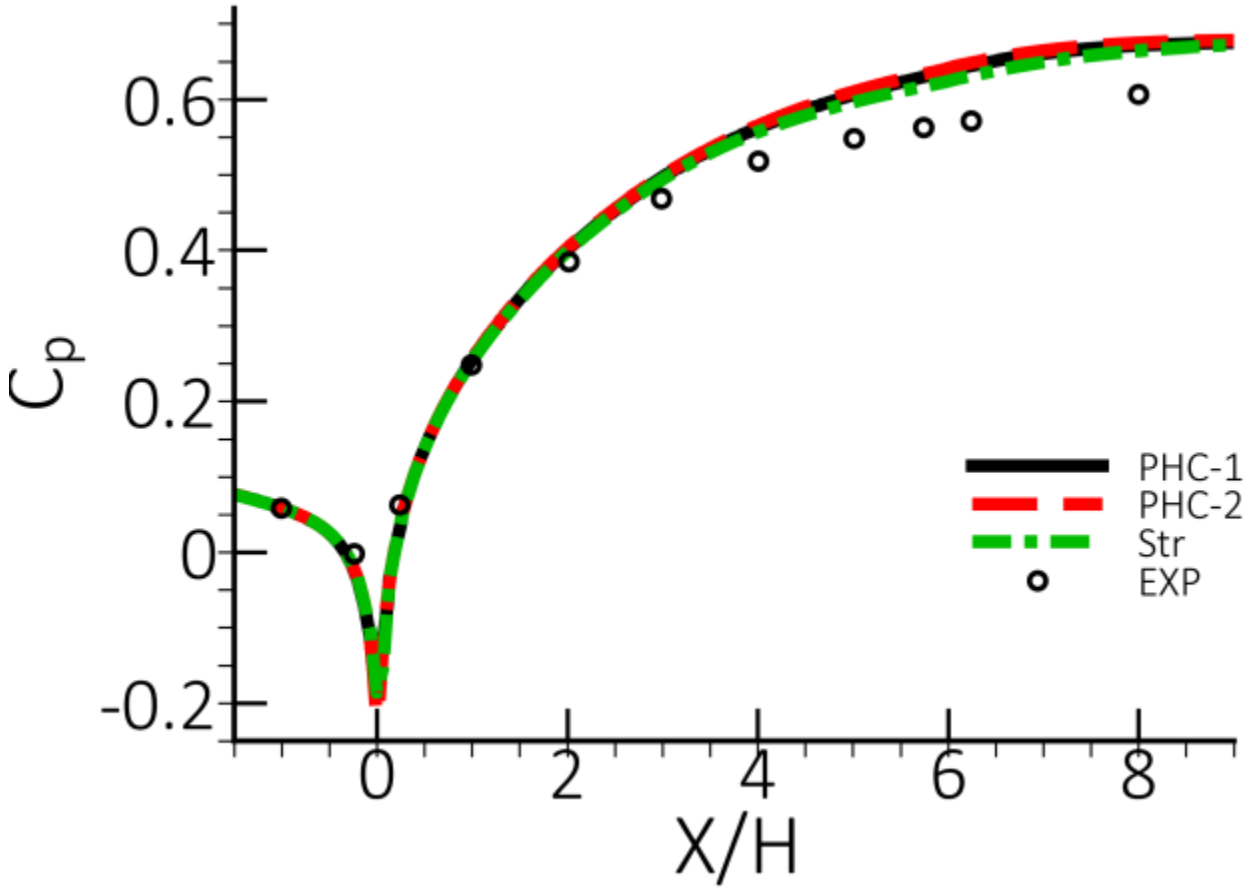


Figure 12.158: Distribution of the pressure coefficient along the diffuser at midsection for SST-CFC for the DLR diffuser for two unstructured and one structured mesh [41 (p. 373)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.4. Numerical Settings

CFD simulations based on RANS models are typically carried out in 'steady-state' model (see Section 3.2). In most cases the following settings are appropriate:

- Coupled solver
- Pseudo-transient
  - Use default proposed time step
  - In case of convergence problems vary time step
- Second order momentum
- Second order turbulence
  - This is very helpful for unstructured grids and or grids not well aligned with the flow. For hexahedral grids with good flow alignment the difference between 1st and 2nd order space discretization is typically moderate.
  - Experience has shown that 2nd order space discretization for turbulence does not have an overall negative influence on convergence – it is therefore recommended.

The following topics are discussed:

12.2.4.1. Example: High-Lift Aircraft

12.2.4.1. Example: High-Lift Aircraft

A drastic example for the danger of non-converged solutions is the following flow around a generic airplane geometry with high-lift devices (slats upstream and flaps downstream of the wing employed) deployed. The simulations for the JAXA geometry were carried out in the framework of the 3rd AIAA CFD High Lift prediction Workshop (https://hiliftpw.larc.nasa.gov/Workshop3/geometries.html). A first simulation with the SST model using default model and steady state settings resulted in a substantial under-prediction of the maximum lift coefficient (C_{Lmax}) due to a massive separation on the main wing. This can be seen in Figure 12.159: Figure 96: Wall shear stress for high-lift airplane simulation for α =18 . Left: Simulation in 'steady-state mode'. Right: Simulation in 'unsteady mode'. (p. 365) (left) showing the skin friction on the surface for the 'steady-state' set-up for an angle of attack of α=$18°$. At first, this was attributed to the SST model's separation prediction capabilities, which were apparently interpreted as being overly 'aggressive' for this case. For this reason, the a_1 coefficient in the SST model was increased to a_1=1 to reduce separation sensitivity. However, it was found that convergence for the case with the default SST model setting was poor and the case was therefore re-run with unsteady solver settings. The difference due to this change in numerical settings was dramatic as shown in Figure Figure 96 (right) which shows the skin friction distribution for the unsteady run. The unsteady set-up allowed the simulation to 'escape' from the incorrect flow topology and move to a more realistic and much less separated condition.

The effect on the lift coefficient can be seen in Figure 12.160: Lift coefficient, Cl, for high-lift aircraft simulation versus angle of attack (AOA). RANS-SST with default and 'steady state mode', SST with a1=1.0 and 'steady state mode' and URANS-SST with default SST and 'unsteady mode' set-up. (p. 365) which contains three curves. One for SST-default and 'steady state mode' one for SST with a_1=1.0 (instead of default a_1=0.31) and again 'steady state mode' and finally with SST default (a_1=0.31)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

and 'unsteady mode'. The change in the a1 coefficient results in less separation and therefore a better agreement with the exp. data. However, such a change in a_1 is not justified from a physics standpoint, when considering at 2D airfoil performance (see 4.2.2). Alternatively, the solution was also improved significantly by just switching from the steady-state set-up to the unsteady set-up. The lift for the URANS simulation is still somewhat lower than the experiment, but by a much lower factor compared to the steady run. There is also indication that including laminar-turbulent transition in the simulation will further increase the lift curve, so that the remaining difference can partly be attributed to this effect. Besides such details, the important message here is that the difference between the steady and unsteady solutions with identical turbulence model settings can be larger than the differences between strongly different turbulence model settings.

Figure 12.159: Figure 96: Wall shear stress for high-lift airplane simulation for α =18 . Left: Simulation in 'steady-state mode'. Right: Simulation in 'unsteady mode'.

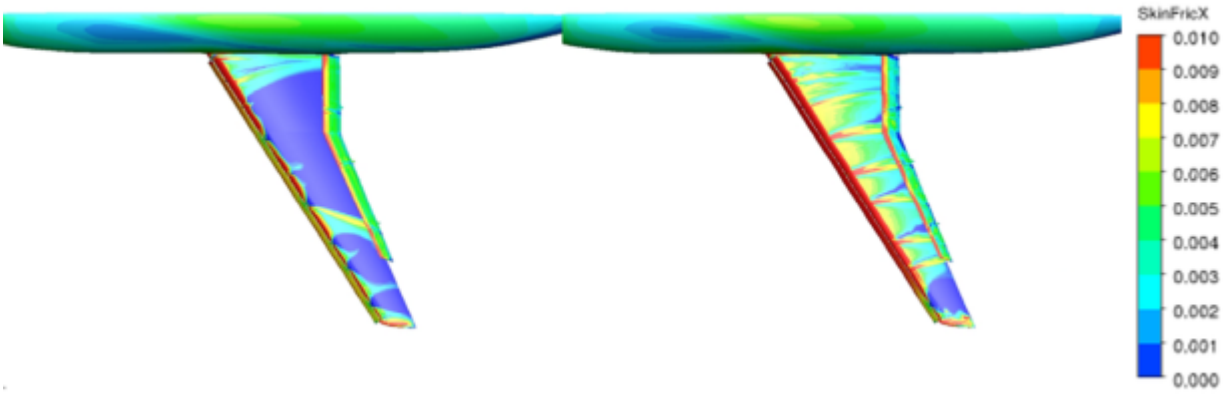
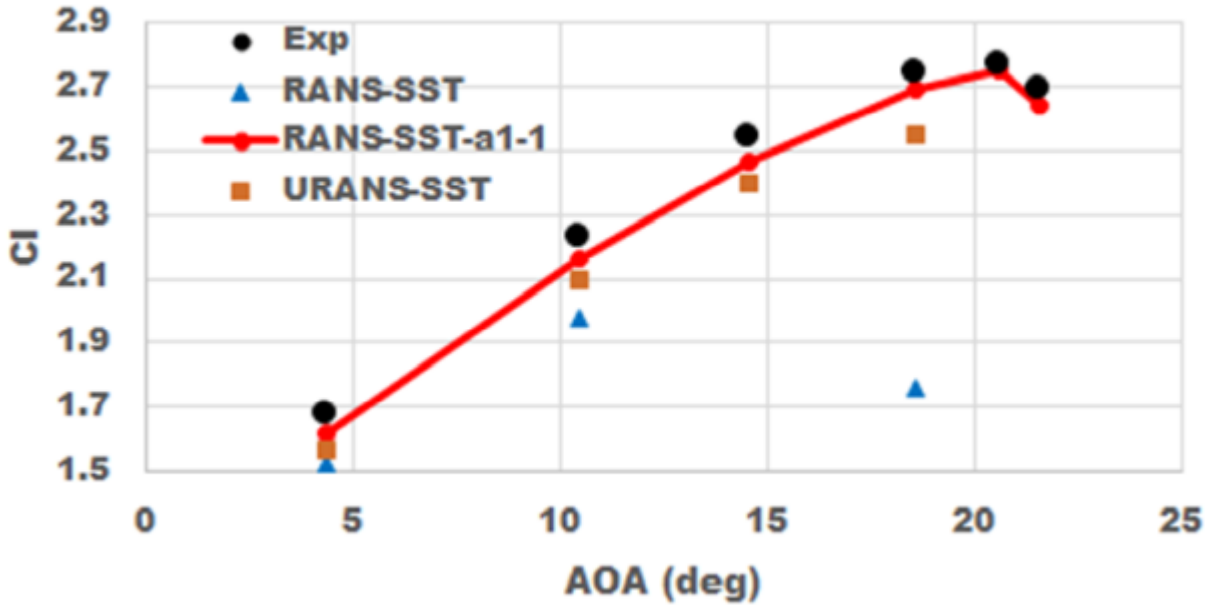


Figure 12.160: Lift coefficient, Cl, for high-lift aircraft simulation versus angle of attack (AOA). RANS-SST with default and 'steady state mode', SST with a1=1.0 and 'steady state mode' and URANS-SST with default SST and 'unsteady mode' set-up.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

It is important to stress that situations like the one shown above appear very seldomly, but they can happen in CFD. The exact cause for such scenarios is not easy to determine. However, there are some contributing factors:

- Large variation in size of geometric features (in this case the thin pylons holding the slat vs the size of the airfoil).
- Large variation in cell size between different regions (often because of the first bullet).
- Large aspect ratio grids in the domain.
- Potential for topology changes in the flowfield – meaning that the flow can change from 'attached' to stalled' on certain sections of the wing.
- Local regions of unsteady flow with relevance to the overall flow topology.

12.2.5. Summary

The current document has been compiled to provide information about the optimal selection and usage of turbulence models. It was pointed out that not all differences between CFD simulations and experimental data are the responsibility of the turbulence model, but that many different set-up details play a crucial role.

The turbulence modeling strategy at Ansys, designed around $k-\omega$ models was discussed and different sub-models of that family have been presented in comparison with different types of $k-\varepsilon$ models. It was shown that the $k-\omega$ based models offer significant advantages in terms of integration to the wall, as well as increased accuracy and robustness for complex flows, especially flows with separation from smooth walls.

The motivation behind the tunable GEKO model has been presented. The influence of the different parameters has been shown for some building-block flows. A more detailed Best Practice document for the GEKO model is available ([Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)]), in addition to a Best Practice Document for Scale-Resolving Simulations [1 (p. 367)].

Numerous sub-models and model extensions have been discussed and their effect on the simulation has been presented.

An area of active research in turbulence modeling is Machine Learning (ML). The topic has not reached the maturity required to be included in a Best Practice document. However, Ansys is currently developing a comprehensive infrastructure for optimization of GEKO parameters using ML. This infrastructure is based on an adjoint solver for the turbulence model in Ansys Fluent including non-linear EARSM coefficients. The infrastructure will enable users to employ ML concepts to their own CFD applications in an integrated and convenient fashion.

12.2.6. Acknowledgment

The authors would like to thank Jeff Dawson and David Whitaker from Hewlett-Packard Enterprise (HPE) for providing support and generously recognize HPE for providing computational resources for the simulation of the JAXA high-lift aircraft configuration.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.7. References

1

F.R. Menter,

"Best Practice: Scale-Resolving Simulations in ANSYS CFD"

ANSYS Report

2

P.A. Durbin, P.A. and Petterson Reif, B.A.

"Statistical Theory and Modeling for Turbulent Flows."

Wiley, 2010

3

Leschziner, M.,

"Statistical Turbulece Modelling for Fluid Dynamics - Demystified: An Introductory Text For Graduate Engineering Students."

Imperial College Press, London, 2016

4

Pope, S.B.,

"Turbulent Flows."

Cambridge University Press, 2000

5

Hanjalic, K. and Launder, B,

"Modelling Turbulence in Engineering and the Environment."

Cambridge University Press, 2011

6

Rotta, J.C.

"Turbulente Strömungen - Eine Einführung in die Theorie und ihre Anwendung."

Springer Verlag, 1972

7

D.C. Wilcox D.C.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Turbulence Modeling for CFD."

DCW Industries, 2006

8

ERCOFTAC

"Industrial Computational Fluid dynamics of Single Phase Flows."

2000, [Online], Available: https://www.ercoftac.org/publications/ercoftac_best_practice_guidelines/single-phase_flows_spf

9

F.R. Menter, R. Lechner, and A. Matyushenko,

"Best Practice: Generalized k-ω Two-Equation Turbulence Model in ANSYS CFD (GEKO),"

ANSYS Germany GmbH, ANSYS Report, 2019

10

F.R. Menter, A. Matyushenko, and R. Lechner

"Development of a Generalized K-ω Two-Equation Turbulence Model," in New Results in Numerical and Experimental Fluid Mechanics XII,"

2020, pp. 101–109, doi: 10.1007/978-3-030-25253-3_10.

11

P. Spalart and S. Allmaras,

"A One-Equation Turbulence Model for Aerodynamic Flows,"

AIAA, vol.439, Jan 1992, doi: 10.2514/6.1992-439.

12

J. C. Vogel and J. K. Eaton,

"Combined Heat Transfer and Fluid Dynamic Measurements Downstream of a Backward-Facing Step,"

Journal of Heat Transfer, vol. 107, no.4, pp.922–929, Nov.1985, doi: 10.1115/1.3247522.

13

D. Cokljat, S.-E. Kim, G. Iaccarino, and P. Durbin,

"A Comparative Assessment of the V2F Model for Recirculating Flows,"

presented at the 41st Aerospace Sciences Meeting and Exhibit, Reno, Nevada, Jan.2003, doi: 10.2514/6.2003-765.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

14

W. P. Jones and B. E. Launder,

"The prediction of laminarization with a two-equation model of turbulence,"

International Journal of Heat and Mass Transfer, vol. 15, no. 2, pp. 301–314, Feb. 1972, doi: 10.1016/0017-9310(72)90076-2.

15

T.-H. Shih, W. W. Liou, A. Shabbir, Z. Yang, and J. Zhu,

"A new $k-\varepsilon$ eddy viscosity model for high Reynolds number turbulent flows,"

Computers&Fluids, vol. 24, no. 3, pp.227–238, Mar.1995, doi: 10.1016/0045-7930(94)00032-T.

16

O. S. A et al

"Renormalization Group Modeling and Turbulence Simulations."

Near-Wall Turbulent Flows, pp. 1031–1046, 1993

17

F. R. Menter,

"Two-equation eddy-viscosity turbulence models for engineering applications,"

AIAA Journal, pp. 1598–1605, Aug. 1994, doi: 10.2514/3.12149.

18

F. R. Menter, M. Kuntz, and R. Langtry,

"Ten Years of Industrial Experience with the SST Turbulence Model"

Proceedings of the 4th International Symposium on Turbulence, Heat and Mass Transfer, Begell House Inc., West Redding, pp.625-632

19

S. Wallin and A. V. Johansson,

"An explicit algebraic Reynolds stress model for incompressible and compressible turbulent flows"

J. Fluid Mech., vol. 403, pp. 89–132, doi: 10.1017/S0022112099007004.

20

P. R. Spalart,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Strategies for turbulence modelling and simulations,"

p. 12, 2000

21

Kato M and Launder B.E.,

"The Modeling of Turbulent Flow Around Stationary and Vibrating Cylinders"

presented at the Ninth Symposium on Turbulent Shear Flows, Kyoto, Japan, 1993

22

P. E. Smirnov and F. R. Menter,

"Sensitization of the SST Turbulence Model to Rotation and Curvature by Applying the Spalart–Shur Correction Term,"

Journal of Turbomachinery, vol. 131, no. 4, 2009, doi: 10.1115/1.3070573.

23

P. R. Spalart and M. Shur,

"On the sensitization of turbulence models to rotation and curvature"

Aerospace Science and Technology, vol. 1, no. 5, pp. 297–302, Jul. 1997, doi: 10.1016/S1270-9638(97)90051-1.

24

F. R. Menter, R. B. Langtry, S. R. Likki, Y. B. Suzen, P. G. Huang, and S. Völker,

"A Correlation-Based Transition Model Using Local Variables—Part I: Model Formulation"

Journal of Turbomachinery, vol. 128, no. 3, pp. 413–422, Mar. 2004, doi: 10.1115/1.2184352.

25

R. B. Langtry, F. R. Menter, S. R. Likki, Y. B. Suzen, P. G. Huang, and S. Völker,

"A Correlation-Based Transition Model Using Local Variables—Part II: Test Cases and Industrial Applications,"

Journal of Turbomachinery, vol. 128, no. 3, pp. 423–434, Mar. 2004, doi: 10.1115/1.2184353.

26

R. B. Langtry and F. R. Menter

"Correlation-Based Transition Modeling for Unstructured Parallelized Computational Fluid Dynamics Codes"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AIAA Journal, vol. 47, no. 12, pp. 2894–2906, Dec.2009, doi: 10.2514/1.42362.

27

N. V. Nikitin, N. V. Popelenskaya, and A. Stroh,

"Prandtl's Secondary Flows of the Second Kind. Problems of Description, Prediction, and Simulation,"

Fluid Dyn, vol. 56, no. 4, pp. 513–538, Jul. 2021, doi: 10.1134/S0015462821040091.

28

B. Aupoix,

"Roughness Corrections for the k–ω Shear Stress Transport Model: Status and Proposals,"

Journal of Fluids Engineering, vol. 137, no. 2, Sep. 2014, doi: 10.1115/1.4028122.

29

K. Wieghardt and W. Tillmann,

"On the Turbulent Friction Layer for Rising Pressure,"

UNT Digital Library, Oct. 1951, https://digital.library.unt.edu/ark:/67531/metadc62980/ (accessed Jul. 23, 2021).

30

D. Driver, D.M.

"Reynolds shear stress measurements in a separated boundary layer flow,"

in 22nd Fluid Dynamics, Plasma Dynamics and Lasers Conference, American Institute of Aeronautics and Astronautics, 1991, doi: 10.2514/6.1991-1787.

31

D. M. Somers,

"Design and experimental results for the S805 airfoil,"

National Renewable Energy Lab., NREL/SR--440-6917, Jan. 1997

32

D. M. Somers,

"Design and Experimental Results for the S825 Airfoil,"

NREL/SR-500-36346, Jan. 2005.

33

D. M. Somers,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Design and Experimental Results for the S809 Airfoil, 1997."

NRELSR-440-6918, p. 103,

34

D. M. Somers,

"Design and experimental results for the S814 airfoil,"

National Renewable Energy Lab., NREL/SR--440-6919, Jan. 1997

35

W. A. Timmer and R. V. Rooij,

"Summary of the Delft University Wind Turbine Dedicated Airfoils,"

AIAA Paper 2003-0352, 2003, doi: 10.1115/1.1626129.

36

F. R. Menter, P. E. Smirnov, T. Liu, and R. Avancha,

"A One-Equation Local Correlation-Based Transition Model,"

Flow Turbulence Combust, vol. 95, no. 4, pp. 583–619, Dec. 2015, doi: 10.1007/s10494-015-9622-4.

37

G. Schewe,

"Reynolds-number effects in flow around more-or-less bluff bodies,"

Journal of Wind Engineering and Industrial Aerodynamics, vol. 89, no. 14–15, pp. 1267–1289, Dec. 2001, doi: 10.1016/S0167-6105(01)00158-1.

38

A. Wadcock,

"Investigation of low-speed turbulent separated flow around airfoils,"

Sep. 1987.

39

W. D. Bachalo and D. A. Johnson,

"Transonic, turbulent boundary-layer separation generated on an axisymmetric flow model,"

AIAA Journal, vol. 24, no. 3, pp. 437–443, Mar. 1986, doi: 10.2514/3.9286.

40

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

H. Raiesi, U. Piomelli, and A. Pollard,

"Evaluation of Turbulence Models Using Direct Numerical and Large-Eddy Simulation Data,"

Journal of Fluids Engineering, vol. 133, p. 021203, Feb. 2011, doi: 10.1115/1.4003425.

41

J. Klinner, M. Schroll, C. Morsbach, F. Möller, and C. Willert,

"Experimental and numerical investigation of 3-D corner separation in a channel flow with adverse pressure gradient,"

in 21st STAB/DGLR Symposium on New Results in Numerical and Experimental Fluid Mechanics, Darmstadt, Nov. 2018, vol. 12, no. 142, pp. 663–673.

42

E. M. Cherry, C. J. Elkins, and J. K. Eaton,

"Geometric sensitivity of three-dimensional separated flows,"

International Journal of Heat and Fluid Flow, vol. 29, no. 3, pp. 803–811, Jun. 2008, doi: 10.1016/j.ijheatfluidflow.2008.01.018.

43

E. M. Cherry, C. J. Elkins, and J. K. Eaton,

"Pressure measurements in a three-dimensional separated diffuser,"

International Journal of Heat and Fluid Flow, vol. 30, no. 1, pp. 1–2, doi: 10.1016/j.ijheatfluidflow.2008.10.003.

44

J. S. Chow, G. G. Zilliac, and P. Bradshaw,

"Mean and Turbulence Measurements in the Near Field of a Wingtip Vortex,"

AIAA Journal, vol. 35, no. 10, pp. 1561–1567, Oct. 1997, doi: 10.2514/2.1.

45

Hartley, C.D,

"Measurement of flow velocities within a hydrocyclone using laser doppler anemometry."

Technical Report FTN/X/82, AEA, Power Fluidics, BNFL, 1994.

46

D. Greenblatt, K. B. Paschal, C.-S. Yao, J. Harris, N. W. Schaeffler, and A. E. Washburn,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Experimental Investigation of Separation Control Part 1: Baseline and Steady Suction,"

AIAA Journal, vol. 44, no. 12, Dec. 2006, doi: 10.2514/1.13817.

47

D. Greenblatt, K. B. Paschal, C.-S. Yao, and J. Harris,

"Experimental Investigation of Separation Control Part 2: Zero Mass-Flux Oscillatory Blowing,"

AIAA Journal, vol. 44, no. 12, pp. 2831–2845, 2006, doi: 10.2514/1.19324.

48

J. W. Naughton, S. Viken, and D. Greenblatt,

"Skin Friction Measurements on the NASA Hump Model,"

AIAA Journal, vol. 44, no. 6, pp. 1255–1265, 2006, doi: 10.2514/1.14192.

49

C. L. Rumsey,

"Exploring a Method for Improving Turbulent Separated-Flow Predictions with k-ω Models,"

NASA/TM–2009–215952, 2009.

50

J. W. Baughn, A. E. Hechanova, and X. Yan,

"An experimental study of entrainment effects on the heat transfer from a flat surface to a heated circular impinging jet,"

Journal of Heat Transfer, vol. 113:4, Nov. 1991, doi: 10.1115/1.2911197.

51

R. E. Uittenbogaard,

"Stably Stratified Mixing Layer."

IAHR Working Group on Rfeined Flow Modelling, 1989.

52

F. R. Menter,

"Influence of freestream values on k-omega turbulence model predictions,"

AIAA Journal, vol. 30, no. 6, pp. 1657–1659, 1992, doi: 10.2514/3.11115.

53

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

P. R. Spalart,

"Strategies for turbulence modelling and simulations,"

International Journal of Heat and Fluid Flow, vol. 21, no. 3, pp. 252–263, Jun. 2000, doi: 10.1016/S0142-727X(00)00007-2.

54

S. Obi, K. Aoki, and S. Masuda,

"Experimental and Computational Study of Turbulent Separating Flow in an Asymmetric Plane Diffuser,"

Ninth Symposium on "Turbulent Shear Flows", Kyoto, Japan, p. 5.

55

H. Grotjans and F. R. Menter,

"Wall Functions for General Application CFD Codes."

56

B. E. Launder and B. I. Sharma,

"Application of the Energy-Dissipation Model of Turbulence to Calculation of Flow Near a Spinning Disc,"

Lett. Heat and Mass Transfer 1, vol. 1, no. 2, p. 8, 1974.

57

K.-Y. Chien,

"Predictions of Channel and Boundary-Layer Flows with a Low-Reynolds-Number Turbulence Model,"

AIAA Journal, vol. 20, no. 1, pp. 33–38, Jan. 1982, doi: 10.2514/3.51043.

58

M. Wolfshtein,

"The velocity and temperature distribution in one-dimensional flow with turbulence augmentation and pressure gradient,"

International Journal of Heat and Mass Transfer, vol. 12, no. 3, pp. 301–318, Mar. 1969, doi: 10.1016/0017-9310(69)90012-X.

59

P. A. Durbin,

"Near-wall turbulence closure modeling without 'damping functions,'"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Theoret. Comput. Fluid Dynamics, vol. 3, no. 1, pp. 1–13, Sep. 1991, doi: 10.1007/BF00271513.

12.2.8. Appendix A: Theory

The following topics are discussed:

12.2.8.1. The Closure Problem
12.2.8.2. Two-Equation Models
12.2.8.3. Near-Wall Treatment
12.2.8.4. Appendix A: Boundary Layer Parameters

12.2.8.1. The Closure Problem

For a detailed discussion of the derivation of turbulence equations please consult one of the available text books ([2 (p. 367)]–[7 (p. 367)]). In this section only a very basic model descriptions will be provided to allow the connection between the best Practice discussions and the turbulence model formulations.

The following topics are discussed:

12.2.8.1.1. Averaging
12.2.8.1.2. The Eddy-Viscosity Assumption
12.2.8.1.3. Reynolds Stress Modeling (RSM)
12.2.8.1.4. Explicit Algebraic Reynolds Stress Modeling (EARSM)
12.2.8.1.5. The Equation for the Turbulent Kinetic Energy
12.2.8.1.6. The Turbulent Scale-Equation

12.2.8.1.1. Averaging

Due to the high cost of DNS, the engineering approach to CFD lies in the Reynolds-Averaged Navier-Stokes (RANS) equations. These equations are obtained from the exact Navier-Stokes equations through time (or ensemble) averaging.

Time averaging is defined by:

$$\phi(x_i,t)=\overline{\phi}(x)+\phi'(x_i,t) \tag{12.42}$$

$$\overline{\phi}=\lim_{T\to\infty}\frac{1}{T}\int_0^T\phi(x_i,t)dt \tag{12.43}$$

In other words, the instantaneous quantity $\phi(x_i,t)$ is split into a time-mean value $\overline{\phi}(x)$ and a fluctuating part $\phi'(x_i,t)$. The simulation is then only concerned with the time-mean part. More generally, one can also perform an ensemble-averaging over many instances of an experiment:

Ensemble averaging is defined by:

$$\phi(x_i,t)=\overline{\phi}(x)+\varphi'(x_i,t) \tag{12.44}$$

$$\overline{\phi}=\lim_{T\to\infty}\frac{1}{N}\sum_1^N\phi(x_i,t) \tag{12.45}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The advantage of ensemble averaging is that it can also be applied to inherently unsteady flows (e.g. the flow in an internal combustion engine with moving parts). Both forms of averaging lead to the same equations and turbulence models, so the same overbar is used for simplicity.

When these averaging processes are applied to the Navier-Stokes equations, one obtains the RANS momentum equations:

$$\frac{\partial(\rho\overline{U_i})}{\partial t}+\frac{\partial(\rho\bar{U}_i\bar{U}_J)}{\partial x_j}=-\frac{\partial\overline{p}}{\partial x_i}+\frac{\partial}{\partial x_j}(\tau_{ij}-\rho\overline{u'_iu'_J}) \tag{12.46}$$

where ($\overline{U_i}$) ?is the vector of the averaged velocity field, ρ is the density (assumed constant in this brief overview), $\overline{p}$ is the averaged pressure and τ_{ij} is the Stokes (laminar) stress tensor. For incompressible flows:

$$\tau_{ij}=\mu(\frac{\partial\overline{U_i}}{\partial x_j}+\frac{\partial\overline{U_J}}{\partial x_i})\equiv\mu 2S_{ij} \tag{12.47}$$

with μ being the dynamic molecular viscosity.

The last term in the above equation ($-\rho\overline{u'_iu'_J}$) is so-called 'Reynolds Stress Tensor' and results from the averaging of the non-linear convection terms. This tensor represents the influence of the turbulent fluctuations on the mean velocity field. The above momentum equations are 'unclosed' as no equations are yet available for the Reynolds Stress Tensor. Turbulence models are needed to provide formulations for this tensor.

The RANS momentum equations above are derived under the assumption that there are no significant density variations due to turbulence – this is typically the case for flows with Mach numbers below 3. Note that density variations on the mean flow occur much earlier, starting as low as M~0.1.

12.2.8.1.2. The Eddy-Viscosity Assumption

The most widely applied assumption in turbulence modeling is the eddy-viscosity assumption. It is based on the concept that turbulent stresses can be represented in a similar fashion as the laminar stress tensor:

$$-\rho\overline{u'_iu'_j}=\mu_t(\frac{\partial\overline{U_i}}{\partial x_j}+\frac{\partial\overline{U_j}}{\partial x_i})-\frac{2}{3}\rho k\delta_{ij} \tag{12.48}$$

The last term on the right represents the turbulent kinetic energy, k, and accounts for the requirement that the sum of the diagonals of the Reynolds Stress Tensor must amount to $2k$. This term is not essential and can be avoided/ignored (e.g. in one-equation models where k is not available)

The original problem of providing closure equations for the six (due to symmetry of tensor) unknowns of the Reynolds Stress Tensor is thereby now reduced to providing a suitable eddy-viscosity μ_t (also called turbulent viscosity). It is important to note that μ_t is not a property of the fluid, but a property of the local turbulence. The eddy-viscosity has dimension:

$$\mu_t=\rho\nu_t\sim\rho\frac{L_t^2}{T_t} \tag{12.49}$$

where L_t and T_t are the length- and time-scales of turbulence, respectively.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.8.1.3. Reynolds Stress Modeling (RSM)

Contrary to the eddy-viscosity assumption, there are methods, which aim at computing the individual Reynolds Stresses individually. For this purpose, exact transport equations for each Reynolds Stress are derived (6 equations). These exact equations, however, contain again new unknown terms which need to be modeled (see e.g. [5 (p. 367)]). Different modeling assumptions for these terms then leads to a large variety of Reynolds Stress Models (RSM). The exact RSM equations read:

$$\frac{\partial \rho \overline{u'_i u'_j}}{\partial t} + \frac{\partial \rho \bar{U}_k u'_i u'_j}{\partial x_k} = -\overline{u'_i u'_k}\frac{\partial \rho \bar{U}_j}{\partial x_k} - \overline{u'_j u'_k}\frac{\partial \rho \bar{U}_i}{\partial x_k} + \overline{p'\left(\frac{\partial u'_i}{\partial x_j} + \frac{\partial u'_j}{\partial x_i}\right)} - 2\mu\overline{\frac{\partial u'_i}{\partial x_x}\frac{\partial u'_j}{\partial x_x}} - \frac{\partial}{\partial x_k}\left[\rho\overline{u'_i u'_j u'_k} + \overline{p'u'_i}\delta_{jk} + \overline{p'u'_j}\delta_{ik}\right] \quad (12.50)$$

The terms in these equations are: Line 1: time derivative, convection, production. Line 2: pressure strain (PS), dissipation. Line 3: turbulent diffusion and molecular diffusion. All terms in Line 1 are exact and need no modeling. This is true also for the last term in Line 3. All other terms require models to close the equations. In addition, there is a need for information on the turbulent scale required e.g. in the dissipation term. This information is typically obtained from an additional transport equation (e.g. ε-equation of ω-equation). All in all, RSM closure therefore requires the solution of seven additional equations. Proposed RSM differ mostly by the way they model the pressure-strain term.

12.2.8.1.4. Explicit Algebraic Reynolds Stress Modeling (EARSM)

Simplified versions of RSM models can be obtained where the Reynolds stresses are computed from algebraic formulations instead of transport equations. Such models are generically of the form:

$$-\rho\overline{u'_i u'_j} = F(S_{ij}, W_{ij}, k, \omega) \quad (12.51)$$

with

$$\bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{U}_i}{\partial x_j} + \frac{\partial \bar{U}_j}{\partial x_i}\right), \bar{W}_{ij} = \frac{1}{2}\left(\frac{\partial \bar{U}_i}{\partial x_j} - \frac{\partial \bar{U}_j}{\partial x_i}\right) \quad (12.52)$$

and k being the turbulent kinetic energy and ω a turbulent frequency scale. Note that the eddy-viscosity formulation (Eq. (9.7)) is the simplest form of such a model. The main model in Ansys CFD is the version of Wallin-Johansson (19 (p. 369)).

12.2.8.1.5. The Equation for the Turbulent Kinetic Energy

All industrial relevant two-equation models use the equation for the turbulent kinetic energy, k, to provide one of the two scales required. The k-equation can be derived by summing up half of the diagonal of the exact RSM equations (Eq.(9.9)).

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho \bar{U}_j k)}{\partial x_j} = P_k - \rho\varepsilon + D^{(k)} + \frac{\partial}{\partial x_j}\left(\mu\frac{\partial k}{\partial x_j}\right) \quad (12.53)$$

with:

$$P_k \equiv -\rho\overline{u_j' u_i'}\frac{\partial(\bar{U}_i)}{\partial x_j} \quad (12.54)$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\rho\varepsilon \equiv \mu \overline{\frac{\partial u_i'}{\partial x_j} \frac{\partial u_i'}{\partial x_j}} \tag{12.55}$$

$$D^{(k)} \equiv -\frac{\partial \left(\overline{u_i' p'} + 0.5\rho \overline{u_i' u_j' u_j'} \right)}{\partial x_i} \tag{12.56}$$

The term P_k is the production term. When inserting the eddy-viscosity formulation (Equation (9.7)) it can be formulated as:

$$P_k = \mu_t \tilde{S}^2 \tag{12.57}$$

with $\tilde{S}^2 = 2\tilde{S}_{ij}\tilde{S}_{ij}$

The term ε is the turbulence energy dissipation rate, which converts turbulent kinetic energy into heat (typically small enough to be neglected in the overall energy balance).

The term $D^{(k)}$ is called turbulent diffusion. In an integral sense, it does not contribute to the overall production or dissipation of turbulent kinetic energy, but does only smooths out the k -distribution. It is typically modeled by a simple gradient-diffusion hypothesis:

$$D^{(k)} \approx \frac{\partial}{\partial x_j} \left(\frac{\mu_t}{\sigma_k} \frac{\partial k}{\partial x_j} \right) \tag{12.58}$$

The modeled k -equation therefore reads:

$$\frac{\partial (\rho k)}{\partial t} + \frac{\partial (\rho \tilde{U}_j k)}{\partial x_j} = P_k - \rho\varepsilon + \frac{\partial}{\partial x_j} \left(\left(\mu + \frac{\mu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right) \tag{12.59}$$

With these formulations, the only term missing (unclosed) in the k -equation is the dissipation term ε. It needs to be provided from a separate scale-equation (e.g. the ε-equation or the ω-equation with $\varepsilon = C_\mu k\omega$).

12.2.8.1.6. The Turbulent Scale-Equation

What is meant by 'turbulent scales' in the context of RANS modeling? As discussed in the introduction, turbulence consists of a spectrum of scales – so identifying a single length- or time-scale is not trivial. However, the mixing processes which are of foremost engineering interest are driven by the largest scales of turbulence.

The derivation of the scale equation (typically ε or ω) is one of the weakest steps in RANS modeling. While exact transport equations can be derived e.g. for ε, such equations represent the small scales of turbulence (where dissipation takes place) and do not provide information on the large scales required in RANS closure. One could argue that in RANS ε is not trying to model 'dissipation' but rather the destruction of large scales into smaller scales – which are no longer relevant for the mixing processes.

To arrive at a suitable model equation, the assumption is made that the scale equation has a form similar to the k -equation:

$$\frac{\partial (\rho\varepsilon)}{\partial t} + \frac{\partial (\rho \tilde{U}_j \varepsilon)}{\partial x_j} = C_{\varepsilon 1} \frac{\varepsilon}{k} P_k - \rho C_{\varepsilon 2} \frac{\varepsilon}{k} \varepsilon + \frac{\partial}{\partial x_j} \left(\left(\mu + \frac{\mu_t}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_j} \right) \tag{12.60}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

with the ratio ε/k introduced for dimensional correctness. The coefficients $C_{\varepsilon 1}$,$C_{\varepsilon 2}$, σ_ε are model coefficients which need to be tuned by experimental data. The coefficients of the standard $k-\varepsilon$ model are:

Table 12.13: Coefficients for the standard $k-\varepsilon$ model (see e.g. [7 (p. 367)]).

$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	σ_k	σ_ε	C_μ
1.44	1.92	1.0	1.3	0.09

An equation for the turbulent frequency, ω can be derived with the same arguments:

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho\bar{U}_j\omega)}{\partial x_j}=C_{\omega 1}\frac{\omega}{k}P_k-\rho C_{\omega 2}\omega^2+\frac{\partial}{\partial x_j}\left((\mu+\frac{\mu_t}{\sigma_\omega})\frac{\partial\omega}{\partial x_j}\right) \tag{12.61}$$

With this set of equations, the eddy-viscosity can be computed:

$$\mu_t=C_\mu\rho\frac{k^2}{\varepsilon}=\rho\frac{k}{\omega} \tag{12.62}$$

using $\varepsilon=C_\mu k\omega$

Table 12.14: Coefficients standard $k-\omega$ model (see e.g. [7 (p. 367)])

$C_{\omega 1}$	$C_{\omega 2}$	σ_k	σ_ω	C_μ
5/9	3/40	2	2	0.09

Since ε and ω are related, one can investigate the differences between the $\varepsilon-$ and the $\omega-$ equations. This can be achieved e.g. by transforming the exact $\varepsilon-$ equation to an $\omega-$ equation. Assuming for simplicity $\sigma_k=\sigma_\varepsilon=\sigma_\omega$ one gets:

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho U_j\omega)}{\partial x_j}=C_{\omega 1}\frac{\omega}{k}P_k-\rho C_{\omega 2}\omega^2+CD+\frac{\partial}{\partial x_j}\left((\mu+\frac{\mu_t}{\sigma_\omega})\frac{\partial\omega}{\partial x_j}\right) \tag{12.63}$$

With the cross-diffusion term:

$$CD=\rho\frac{2}{\sigma_\omega}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j} \tag{12.64}$$

The transformation of the coefficients gives the relation:

$$C_{\omega 1}=C_{\varepsilon 1}-1;C_{\omega 2}=C_\mu(C_{\varepsilon 2}-1) \tag{12.65}$$

In other words, the ω-equation when derived on the same dimensional arguments as the ε-equation results in Equation ((9.20), whereas the transformation of the ε-equation into a ω-equation gives Equation (9.22). The difference lies mainly in the cross-diffusion term CD.

What is the effect of the CD term? There are two main effects, one positive and one negative. The negative effect is discussed in detail by Wilcox [7 (p. 367)] and is manifested in an improved performance of the ω-equation based models not utilizing the CD term in adverse pressure gradient boundary layers. Such models predict an improved response to the pressure gradient and a higher sensitivity to predict flow separation, compared to ε-equation based models. In other words, the inclusion of the CD term in the ω-equation in the near wall region would inhibit correct separation prediction.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The second effect of including the CD is desirable, as it avoids the strong sensitivity of the standard ω-equation (models without CD) based model to freestream values. As discussed in detail in [52 (p. 374)] the standard model produces a wide variation in results in case the level of ω changes outside the boundary or free shear layer (freestream). This is illustrated in Figure 12.161: Effect of freestream omega-levels on velocity profile for free mixing layer solution. Left: std. k-omega model without CD term. Right: GEKO model including CD term. (p. 381) for a mixing layer. The left part of the Figures shows the reaction of the Wilcox $k-\omega$ model to changes in freestream ω on the velocity profile and the right part shows the same test using the GEKO $k-\omega$ model (which includes the CD term). Figure 12.162: Effect of freestream ω-levels on non-dimensional eddy viscosity profiles for free mixing layer solution. Left: standard $k-\omega$ model without CD term. Right: GEKO model including CD term. (p. 381) shows the same effect on the eddy-viscosity profiles, where the difference in the standard $k-\omega$ model is more than a factor 2 between the low and the high freestream values. Since the level of freestream values cannot easily be controlled and is often based on roughly estimated inlet values, solution independence from these values is essential for reliable simulations.

Figure 12.161: Effect of freestream omega-levels on velocity profile for free mixing layer solution. Left: std. k-omega model without CD term. Right: GEKO model including CD term.

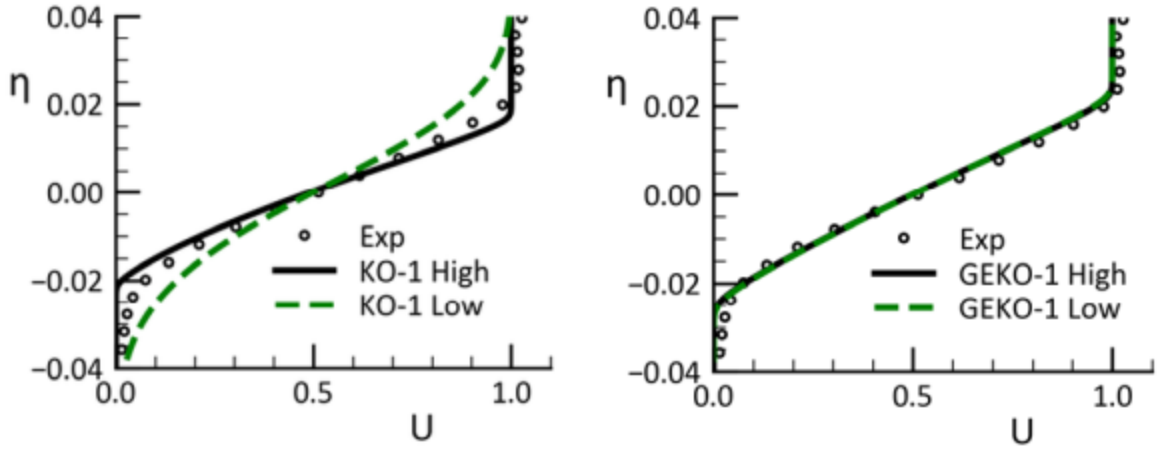
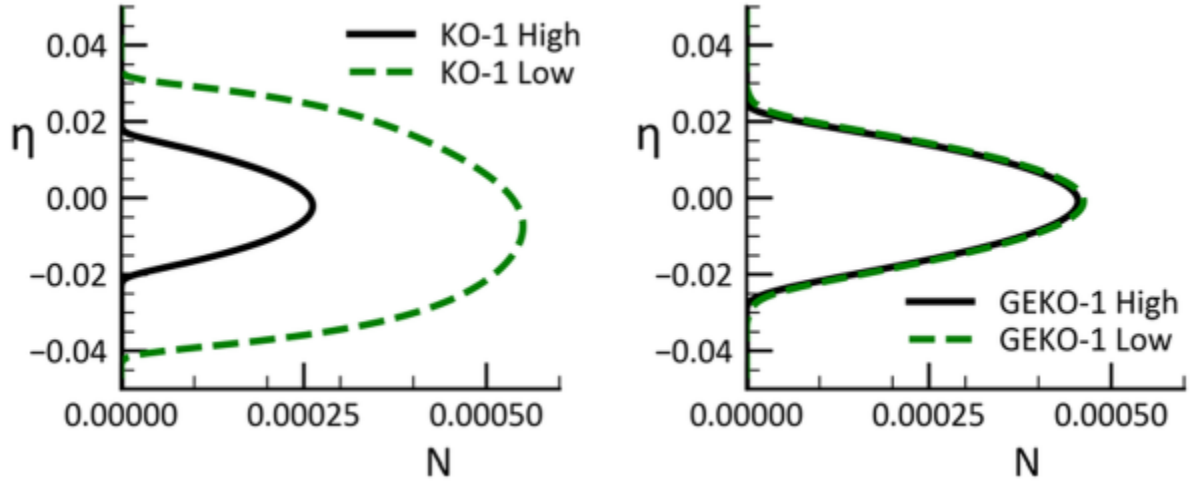


Figure 12.162: Effect of freestream ω-levels on non-dimensional eddy viscosity profiles for free mixing layer solution. Left: standard $k-\omega$ model without CD term. Right: GEKO model including CD term.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The equations discussed so far form the basis of all industrially $\varepsilon-$ and $\omega-$ based turbulence models.

12.2.8.2. Two-Equation Models

The motivation behind two-equation models originates from the need to obtain the two scales required to compute the eddy-viscosity. Based on dimensional arguments, a length-scale,L_t, and a time-scale, T_t, are required for that purpose. Note that any two other scales are equivalent as there are only two independent mechanical scales. These requirements naturally lead to two-equation models as a basis for providing these scales. However, two-equation models also form the basis for all further developments, like One-equation models, Reynolds Stress Models (RSM) or Explicit Algebraic Reynolds Stress models (EARSM). Two-equation models are also the optimal platform for including additional physics, like laminar-turbulent transition, rough walls, buoyancy, as well as hybrid RANS-LES concepts.

Two-equation models are composed of the following elements:

- Turbulent kinetic energy equation
- Scale equation (εor ω)
- Eddy-viscosity formulation
- Near-wall treatment
- Limiters

Each one of these elements can have a substantial influence of accuracy and robustness of the simulations.

The following topics are discussed:

12.2.8.2.1. The Models
12.2.8.2.2. The Models
12.2.8.2.3. Limiters

12.2.8.2.1. The $k-\omega$ Models

The BSL/SST Models

Based on the investigation of the freestream dependency of the Wilcox model [52 (p. 374)], Menter proposed model formulations which are composed of a blend of $k-\varepsilon$ and $k-\omega$ model elements (17 (p. 369)). As discussed in Section 9.2.2, the main difference between these models is the cross-diffusion term (CD) which appears when transforming the $k-\varepsilon$ model to a $k-\omega$ model. This term avoids the freestream sensitivity near the shear/boundary layer edge. However, while it is desirable to include the term near the shear/boundary layer edge, it is not desirable to activate it near the wall, as it negatively affects the adverse pressure gradient behavior of the model. The BSL model was therefore built on the blending function F_1, which is based on wall distance, y. The function has a value of F_1=1 inside the wall boundary layer and gradually blends to F_1=0 in the outer part of the boundary layer and beyond. This function blends the CD term into the ω-equation but also blends the coefficients from those of the $k-\omega$ to those of the $k-\varepsilon$ model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho U_j k)}{\partial x_j}=P_k-C_\mu\rho k\omega+\frac{\partial}{\partial x_j}\left[(\mu+\frac{\mu_t}{\sigma_k})\frac{\partial k}{\partial x_j}\right] \tag{12.66}$$

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho U_j \omega)}{\partial x_j}=C_{\omega1}\frac{\omega}{k}P_k-C_{\omega2}\rho\omega^2+\rho(1-F_1)CD+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\omega}\right)\frac{\partial\omega}{\partial x_j}\right] \tag{12.67}$$

$$\mu_t=\rho\nu_t=\rho\frac{k}{max\left(\omega,F_2S/a_1\right)} \tag{12.68}$$

$$CD=\frac{2}{\sigma_\omega}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j} \tag{12.69}$$

$$Coef=(1-F_1)Coef_{k-\omega}+F_1Coef_{k-\varepsilon} \tag{12.70}$$

Table 12.15: Coefficients of BSL model.

BSL	$C_{\omega1}$	$C_{\omega2}$	σ_k	σ_ω	C_μ	a_1
$\mathrm{Coef}_{k-\omega}$	5/9	3/40	2	2	0.09	∞
$\mathrm{Coef}_{k-\varepsilon}$	0.44	0.0828	1	1.17	0.09	∞

Table 12.16: Coefficients of SST model.

BSL	$C_{\omega1}$	$C_{\omega2}$	σ_k	σ_ω	C_μ	a_1
$\mathrm{Coef}_{k-\omega}$	5/9	3/40	1/0.85	2	0.09	0.31
$\mathrm{Coef}_{k-\varepsilon}$	0.44	0.0828	1	1.17	0.09	0.31

There are two model variants. The BSL (baseline model) and the SST (Shear Stress Transport) model. The SST model is based on the BSL model, but in addition limits the eddy-viscosity inside the boundary layer (using a second blending function F_2) to achieve improved model performance for flows under adverse pressure gradients and separation.

The two blending functions F_1 and F_2 read:

$$\begin{aligned}\arg_1&=\min\left[\max\left(\frac{\sqrt{k}}{0.09\omega y};\frac{500\nu}{\omega y^2}\right);\frac{4\rho\sigma_{\omega2}k}{\max(CD,10^{-20})y^2}\right]\\F_1&=\tanh(\arg_1^4)\end{aligned} \tag{12.71}$$

$$\begin{aligned}\arg_2&=\max\left(2\frac{\sqrt{k}}{0.09\omega y};\frac{500\nu}{\omega y^2}\right)\\F_2&=\tanh(\arg_2^2)\end{aligned} \tag{12.72}$$

The BSL model can also be used as the basis for combination with EARSM and RSM models - where the limiter on the eddy-viscosity is not required, as improved stress prediction is achieved through the EARSM/RSM formulation. The SST model on the other hand is calibrated for accurate prediction of separation and aerodynamic flows when used as an eddy-viscosity model.

There is a clear increase in complexity from the Wilcox model to the BSL/SST models, mainly through the need for blending functions, which in turn require the wall-distance. Wall-distance is not problematic if the mesh/geometry are fixed (as is the case in most CFD simulations). In case of varying mesh/geometry however, the wall distance computation can become expensive as it needs to be repeated for every time-step. Particularly for massively parallel simulations, the wall distance computation can dominate the CPU costs as the algebraic operation does not scale

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

well on such machines. The GEKO model (Section 3.3.2) has therefore been developed with an option avoiding the wall distance.

The SST model allows variation of the a_1 coefficient without affecting the calibration of the logarithmic layer. Its value can be increased, thereby reducing the sensitivity to adverse pressure gradient flows and delaying/reducing flow separation. The value cannot be decreased however, as that would interfere with the log-law behavior and would thereby negatively affect the flat plat calibration.

The Generalized $k-\omega$ model (GEKO)

The main characteristics of the GEKO model is that it has several free parameters for tuning the model to different flow scenarios. The starting point for the formulation is:

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho U_j k)}{\partial x_j}=P_k-C_\mu\rho k\omega+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{12.73}$$

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho U_j\omega)}{\partial x_j}=C_{\omega1}\frac{\omega}{k}P_k-C_{\omega2}\rho\omega^2+\rho C_{\omega3}CD+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\omega}\right)\frac{\partial\omega}{\partial x_j}\right] \tag{12.74}$$

$$\mu_t=\rho\nu_t=\rho\frac{k}{\max(\omega,S/C_{\text{Realize}})} \tag{12.75}$$

$$P_k=-\tau_{ij}\frac{\partial U_i}{\partial x_j} \tag{12.76}$$

$$\tau_{ij}=-\rho\overline{u_i'u_j'}=\mu_t 2S_{ij}-\frac{2}{3}\rho k\delta_{ij} \tag{12.77}$$

$$CD=\frac{2}{\sigma_\omega}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j} \tag{12.78}$$

There is a provision for corner flows which amounts to adding a quadratic term to the stress-strain relation. The terms is essentially the same as the Quadratic Closure Relation (QCR) proposed by Spalart [53 (p. 374)]:

$$\begin{aligned}&\tau_{ij}=-\rho\overline{u_i'u_j'}=\mu_t 2S_{ij}-\frac{2}{3}\rho k\delta_{ij}-C_{CORNER}\frac{1.2n_t}{\max(0.3\omega,\sqrt{(S^2+W^2)/2})}\left(S_{ik}W_{kj}-W_{ik}S_{kj}\right)\\&S_{i,j}=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}-\frac{2}{3}\frac{\partial U_k}{\partial x_k}\delta_{ij}\right);W_{i,j}=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}-\frac{\partial U_j}{\partial x_i}\right)\end{aligned} \tag{12.79}$$

The GEKO model formulation is otherwise conventional, it involves the cross-diffusion term which typically is derived from a transformation of the $k-\varepsilon$ model to a $k-\omega$. This term is included in most of the more recent $k-\omega$ models in various ways. It is generally understood that this term is required at the edge of turbulent layers to maintain the $k-\varepsilon$ model behavior and avoid freestream sensitivities.

The main difference of GEKO to existing models is the introduction of free parameters which can be tuned by the user to achieve different goals in different parts of the simulation domain. The details of the GEKO model formulation are not published at this point, but the effect of the free parameters will be described. Note that an entire 'Best Practice Guide' exists for the GEKO model: [1 (p. 367)].

Currently there are the following parameters included for tuning:

- C_{SEP} – Parameter to optimize flow separation from smooth surfaces

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- C_{NW} – Parameter to optimize flow in non-equilibrium near wall regions (heat transfer, C_f, ….)
- C_{MIX} – Parameter to optimize strength of mixing in free shear flows
- C_{JET} – Parameter to optimize free shear layer mixing (optimize free jets independent of mixing layer)
- C_{CC} – Parameter to optimize flows with streamline curvature and rotation
- C_{CORNER} – Parameter to optimize anisotropic flow in corners

All these parameters are designed so that they do not negatively affect the basic model calibration for flat plate flows. In other words, the logarithmic layer formulation is preserved despite the changes in these parameters. Note that the two last parameters can also be combined with other $k-\omega$ models.

All parameters (except C_{JET}) are available through User Defined Functions (UDF) access. This means that they can be defined either with a global value (in the GUI/TUI) or as zonal/local values via UDFs.

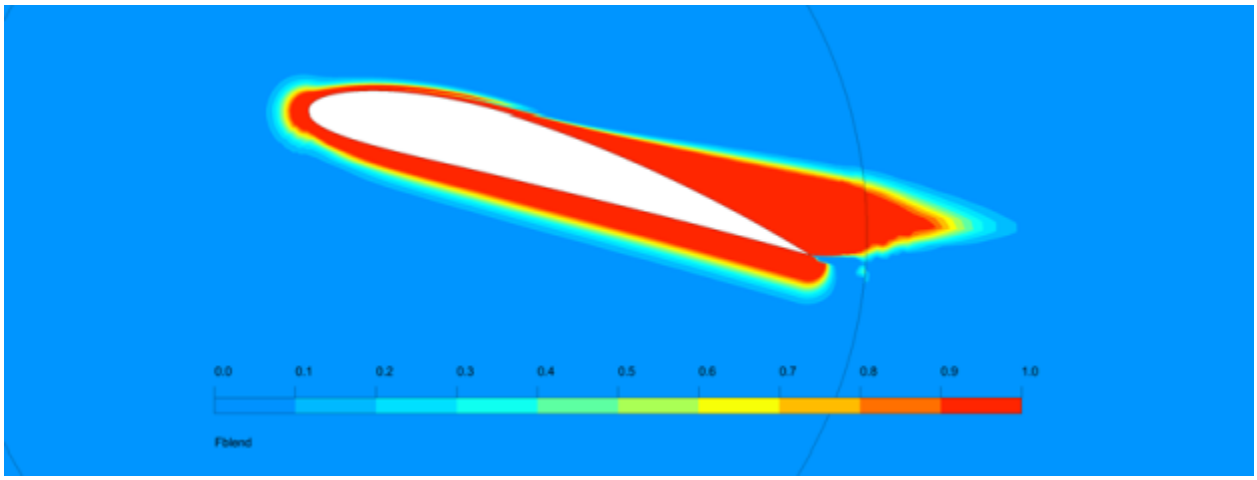
The function F_{Mix} is defined using the blending function F_{Blend} :

$$F_{\text{Mix}}=1+C_{\text{Mix}}F(C_{JET})(1-F_{\text{Blend}}) \tag{12.80}$$

Inside wall boundary layers, the term is de-activated (F_{Blend}=1) to ensure that the C_{MIX}/C_{JET} coefficient only affects free shear flows. It is also seen, that the coefficient C_{JET} only affects the impact of C_{MIX} (e.g. in case C_{MIX}=0 also C_{JET} has no effect). C_{JET} essentially reduced the impact of C_{MIX} on jet flows (where C_{MIX} would otherwise cause an over-prediction of jet spreading rates).

Figure 12.163: Blending function FBlend for introducing addtional mixing through coefficients C_{MIX}/C_{JET}. (p. 385) shows the function F_{Blend} for a NACA 4412 airfoil at α=12° angle of attack. The function is F_{Blend}=1 (red) inside the boundary layer and F_{Blend}=0 (blue) away from the wall. Especially at the trailing edge, where the wake (free shear flow) leaves the wall (boundary layer) the function switches swiftly to activate the C_{MIX}-term.

Figure 12.163: Blending function FBlend for introducing addtional mixing through coefficients C_{MIX}/C_{JET}.



The free coefficients should be in the range:

$$0.7 \le C_{SEP} \le 2.5$$

$$0.0 \le C_{MIX} \le 1.0...$$

$$-2.0 \le C_{NW} \le 2.0$$

$$0.0 \le C_{JET} \le 1.0$$

The coefficient C_{MIX} can in principle be increased beyond C_{MIX}=1, in cases with very strong mixing.

The main tuning parameter for the GEKO model is the coefficient C_{SEP}. When increasing C_{SEP}, boundary layer separation will be predicted more aggressively. However, in addition, the spreading rates of all free shear flows will be affected by changes in C_{SEP} as it generally decreases the eddy-viscosity level in the entire domain. This is not convenient for users, as they would have to tune 2 parameters simultaneously. To maintain the spreading rate for the most important free shear flow, namely the mixing layer, while changing C_{SEP}, a correlation was developed, which maintains the mixing layer spreading rates under changes of C_{SEP}:

$$\begin{aligned} \Delta C_{SEP} &= C_{SEP} - 1 \\ C_{\mathrm{MixCor}} &= 0.35 \cdot \mathrm{sign}(\Delta C_{SEP}) \cdot \sqrt{|\Delta C_{SEP}|} \end{aligned} \tag{12.81}$$

This correlation is set as default – however users can also optimize C_{MIX} by hand, independently of C_{SEP}.

The parameter C_{JET} is designed to reduce the spreading rate for round jets, which is over-predicted with all cnventional turbulence models. To achieve optimal performance for round jets, one needs to set C_{SEP} to values C_{SEP}~1.75-2.00 and C_{JET}=0.9 (default). With reduction in C_{SEP} and the corresponding reduction in C_{MIX}, the effect of C_{JET} vanishes (see [Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)].). There is another parameter CJET_AUX which also has an influence on the jet flow. It defines the limit between mixing layers and jet flows. The larger the value, the sharper the 'demarcation' and stronger the effect of C_{JET}. The default value is CJET_AUX=2.0. It is suitable for jet flow simulations to set the CJET_AUX=4.0. This is not done by default, as it can lead to oscillations on poor meshes. For more details and test cases, see the Best Practices Guidelines for the GEKO model [Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD (p. 401)].

At first, the versatility of the model seems to pose a challenge to the user as to the optimal selection of coefficients. However, there are strong defaults and there are certain combinations of coefficients, which can be applied to most flows, so that the user would only interfere on specific cases, where the model does not lead to satisfactory results. The selection of C_{SEP}=C_{NW}=1 results in an exact transformation to the $k-\varepsilon$ model, albeit with improved sub-layer treatment and limiters activated.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.8.2.2. The $k-\varepsilon$ Models

The standard $k-\varepsilon$ -Model

In order to explain the preference for ω-equation based models, it is important to critically discuss the ε-equation. It needs to be emphasized that the ε-equation has been used in many CFD simulations in the past which is a clear indication that the issues discussed below are not present in all or even the majority of ε-equation model simulations. However, these deficiencies are frequent enough to preclude the ε-equation as a turbulence model integration platform.

The standard $k-\varepsilon$ model without Viscous Sublayer Model (VSM) terms is given by:

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho U_j k)}{\partial x_j}=P_k-\rho\varepsilon+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{12.82}$$

$$\frac{\partial(\rho\varepsilon)}{\partial t}+\frac{\partial(\rho U_j \varepsilon)}{\partial x_j}=C_{\varepsilon 1}\frac{\varepsilon}{k}P_k-C_{\varepsilon 2}\rho\frac{\varepsilon}{k}\varepsilon+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] \tag{12.83}$$

$$\mu_t=C_\mu\rho\frac{k^2}{\varepsilon},P_k=\tau_{ij}\frac{\partial U_i}{\partial x_j},\tau_{ij}=-\rho\overline{u_i u_j}=2\mu_t S_{ij}-\frac{2}{3}\rho k\delta_{ij} \tag{12.84}$$

Table 12.17: Coefficients of standard $k-\varepsilon$ model.

$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	σ_k	σ_ε	C_μ
1.44	1.92	1.0	1.3	0.09

The $k-\varepsilon$ model provides a 'middle of the road' calibration which covers many flows with sensible accuracy – which might explain its popularity over the years.

A well-published deficiency of the ε-equation is the lack of response to adverse pressure gradient flows [7 (p. 367)], (17 (p. 369)). The model produces too large turbulence length scales near the wall and thereby delays or even suppresses flow separation when compared to experimental data. This in turn leads to overly optimistic design choices as the model predicts attached flow under conditions, where the real flow can already be severely separated.

This effect can be seen for a diffuser flow [54 (p. 375)]. It consists of a straight wall on one side and an inclined wall on the other. The flow in the experiment separates from the inclined wall. Figure 12.164: Flow topology for Obi [54] diffuser. Top - SST model. Bottom $k-\varepsilon$model. (p. 388) shows the different flow topologies predicted by the SST and the $k-\varepsilon$ model. Figure 12.165: Velocity profiles for Obi diffuser [54]. Comparison of SST, $k-\varepsilon$ and experimental data. (p. 388) provides a comparison of the model predictions with the exp. velocity profiles which clearly shows that the flow is separated as predicted by the SST model. In case where the design engineer would base the decision of the opening angle of the diffuser on the $k-\varepsilon$ model, the real performance (pressure loss) would be much more optimistic than later observed when the device is built.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.164: Flow topology for Obi [54 (p. 375)] diffuser. Top - SST model. Bottom $k-\varepsilon$model.

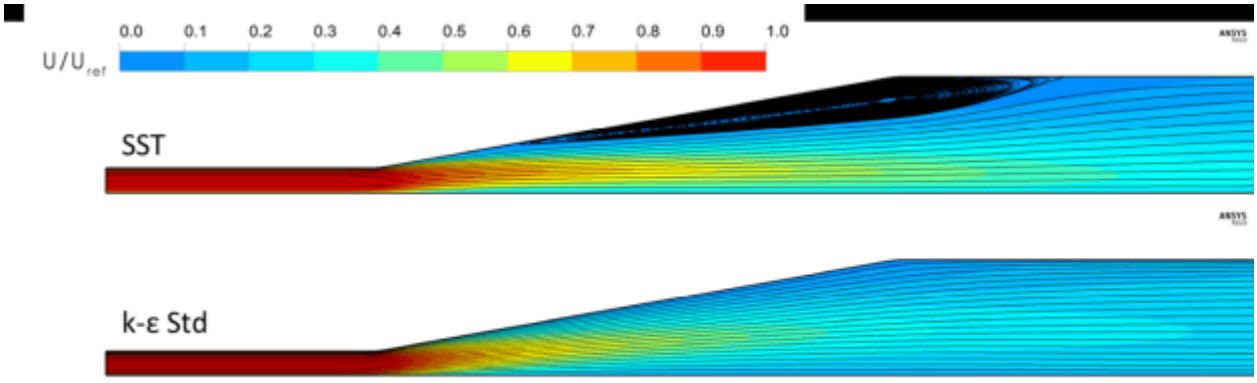
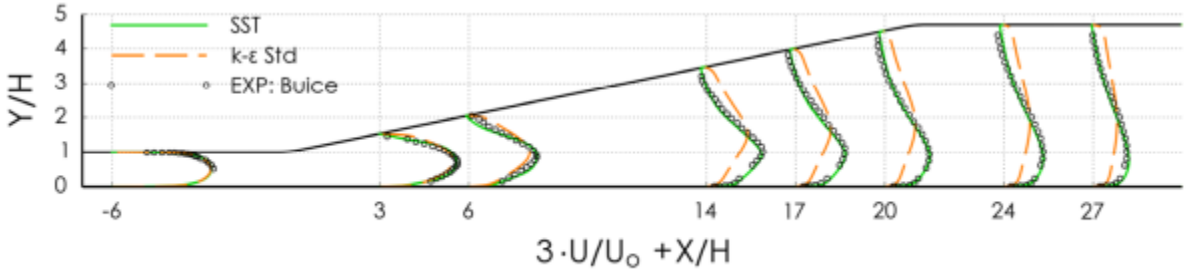


Figure 12.165: Velocity profiles for Obi diffuser [54 (p. 375)]. Comparison of SST, $k-\varepsilon$ and experimental data.



It is important to emphasize that similar pictures of failure could be produced for almost any turbulence model, as there are always flows where models struggle. However, the ability to accurately predict separation onset is essential for many applications, and the $k-\varepsilon$ based models have never been able to overcome their deficiencies in this respect.

The Realizable $k-\varepsilon$ Model (RKE)

The RKE model is a variation of the standard $k-\varepsilon$ model. Its derivation is somewhat curious, involving a mix of small and large scale arguments (15 (p. 369)), but this is of no relevance in the current document. The equations read:

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho U_j k)}{\partial x_j}=P_k-\rho\varepsilon+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{12.85}$$

$$\frac{\partial(\rho\varepsilon)}{\partial t}+\frac{\partial(\rho U_j \varepsilon)}{\partial x_j}=C_1\varepsilon S-C_{\varepsilon 2}\rho\frac{\varepsilon}{k+\sqrt{\nu\varepsilon}}\varepsilon+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] \tag{12.86}$$

$$\mu_t=C_\mu\rho\frac{k^2}{\varepsilon},P_k=\tau_{ij}\frac{\partial U_i}{\partial x_j},\tau_{ij}=-\rho\overline{u_i u_j}=2\mu_t S_{ij}-\frac{2}{3}\rho k\delta_{ij} \tag{12.87}$$

$$\begin{aligned}C_1&=\max\left[0.43,\frac{\eta}{\eta+5}\right],\eta=S\frac{k}{\varepsilon},S=\sqrt{2S_{ij}S_{ij}}\\C_\mu&=\frac{1}{A_0+A_S\frac{KU^*}{\varepsilon}},U^*=\sqrt{S_{ij}S_{ij}+\Omega_{ij}\Omega_{ij}}\end{aligned} \tag{12.88}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

$$A_0=4.04, A_S=\sqrt{6}\cos(\varphi)$$
$$\varphi=a\cos(\sqrt{6}\,W), W=\frac{S_{ij}S_{jk}S_{ki}}{\tilde{S}^3}, S^3=\sqrt{S_{ij}S_{ij}} \tag{12.89}$$

Table 12.18: Coefficients RKE $k-\varepsilon$ model

$C_{\varepsilon 2}$	σ_k	σ_ε
1.9	1.0	1.2

12.2.8.2.3. Limiters

One of the lesser appreciated deficiencies of two-equation models is their behavior in inviscid regions with non-zero strain rates. In such regions the vorticity is zero, but the strain-rate is not. A typical example is the stagnation region of an airfoil (outside the boundary layer). As the inviscid flow approaches the airfoil, there is an increasing level of shear, $S=\sqrt{2S_{ij}S_{ij}}$, but not due to a shear layer, but due to non-zero velocity gradients of the inviscid flow. Similar regions exist when inviscid flow is accelerated. Observations show that two-equation models can exhibit excessively high levels of eddy-viscosity there (Buoyancy Flows (p. 340)). From a physical standpoint, there is very little turbulence production in such areas, so that this is likely an artefact of the eddy-viscosity formulation. The introduction of the eddy-viscosity assumption into the production term of the turbulent kinetic energy equation changes the characteristics of that term. While in the exact k-equation it is a product of turbulent stresses times a velocity gradient (see Equation (9.9)) (and therefore linear in terms of the velocity gradients), it becomes a product of the eddy-viscosity times the strain rate square (and therefore quadratic in the velocity gradients). It is believed that this difference is responsible for the observation of high turbulence production.

To avoid such unphysical behavior, different types of limiters have been developed. Each has their pros and cons, but all are better than running the model without limiters.

Kato-Launder Limiter

The Kato-Launder limiter (21 (p. 370)) is based on the observation that in the inviscid region, the vorticity is zero, whereas in plane shear layers it is equal to the shear strain rate:

$$\begin{aligned}\Omega&=0 \quad \text{in inviscid flows}\\ \Omega&=S \quad \text{in shear flows}\end{aligned} \tag{12.90}$$

With

$$S_{ij}=\frac{1}{2}\left(\frac{\partial \bar{U}_i}{\partial x_j}+\frac{\partial \bar{U}_j}{\partial x_i}\right),\ W_{ij}=\frac{1}{2}\left(\frac{\partial \bar{U}_i}{\partial x_j}-\frac{\partial \bar{U}_j}{\partial x_i}\right) \tag{12.91}$$
$$S=\sqrt{2S_{ij}S_{ij}},\ \Omega=\sqrt{2W_{ij}W_{ij}} \tag{12.92}$$

The Kato-Launder limiter is formulated as a change to the (incompressible portion of the) production term P_k:

$$P_k=\mu_t S^2 \quad \rightarrow \quad P_k^{KL}=\mu_t S\Omega \tag{12.93}$$

Since most model calibrations are carried out for shear flows with $S=\Omega$ this does not change the model calibration for such flows. On the other hand, the modification turns off the production

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

term in inviscid flows where Ω=0. The downside of the Kato-Launder limiter is that it does affect non-trivial flows with 3D effects and or streamline curvature and rotation where $S \neq \Omega$.

Production Limiter

A limiter with much less impact on complex flows was proposed in (17 (p. 369)). The limiter is applied to the production term in relation to the dissipation.

$$P_k = \mu_t S^2 \quad \rightarrow \quad P_k^{KL} = \min(P_k, C_{PL}\rho\varepsilon) \tag{12.94}$$

Typically, the limiter is set to a value of C_{PL}=10, which is far away from typical calibration flows for which one has $P_k \approx \rho\varepsilon$.

Generic Realizability Limiter

The realizability limiter is based on the demand that the Reynolds stresses computed from the eddy-viscosity model should adhere to known restrictions to the Reynolds Stresses [2 (p. 367)] (e.g. non-negativity etc.). It reads:

$$\mu_t^{\text{Real}} = \mu_t \frac{1}{\max(1, S/(C_{\text{Real}}\omega))} \tag{12.95}$$

with $C_{\text{Real}} = 1/\sqrt{3}$. This limiter is similar to the production limiter but corresponds more closely to a $C_{PL} \approx 4$. It is therefore activated sooner than the production limiter.

Realizability Limiter of RKE Model

The realizability limiter of the RKE (Equation (9.47)) model is quite different from the classical realizability limiter. It depends not only on the strain rate but also on the vorticity and the third invariant of the strain rate tensor. Since A_s can become negative, it is not even ensured that the eddy-viscosity is limited. In addition, the production term in the k-equation is formulated as a conventional quadratic term in strain rate (S^2), whereas the production in the ε-equation is formulated as a linear term in S. This all makes it very difficult to judge the activation limit of this limiter and therefore its effectiveness.

12.2.8.3. Near-Wall Treatment

The wall treatment in a CFD method is necessary to allow the prediction of the wall shear-stress, required in the wall cells in the momentum equations as a wall-force. The wall shear-stress is computed from the known solution for the velocity $\overline{U}$ (and possibly k and other quantities) at point j=1 (known e.g. from the previous iteration). This is depicted in Figure 12.166: Near wall grid cells for a finite volume cell-centered method. (p. 391) for a cell-centered finite volume method. The task is therefore, given $\overline{U}$ at j=1, compute the wall shear-stress to close the discretization of the momentum equations.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.166: Near wall grid cells for a finite volume cell-centered method.

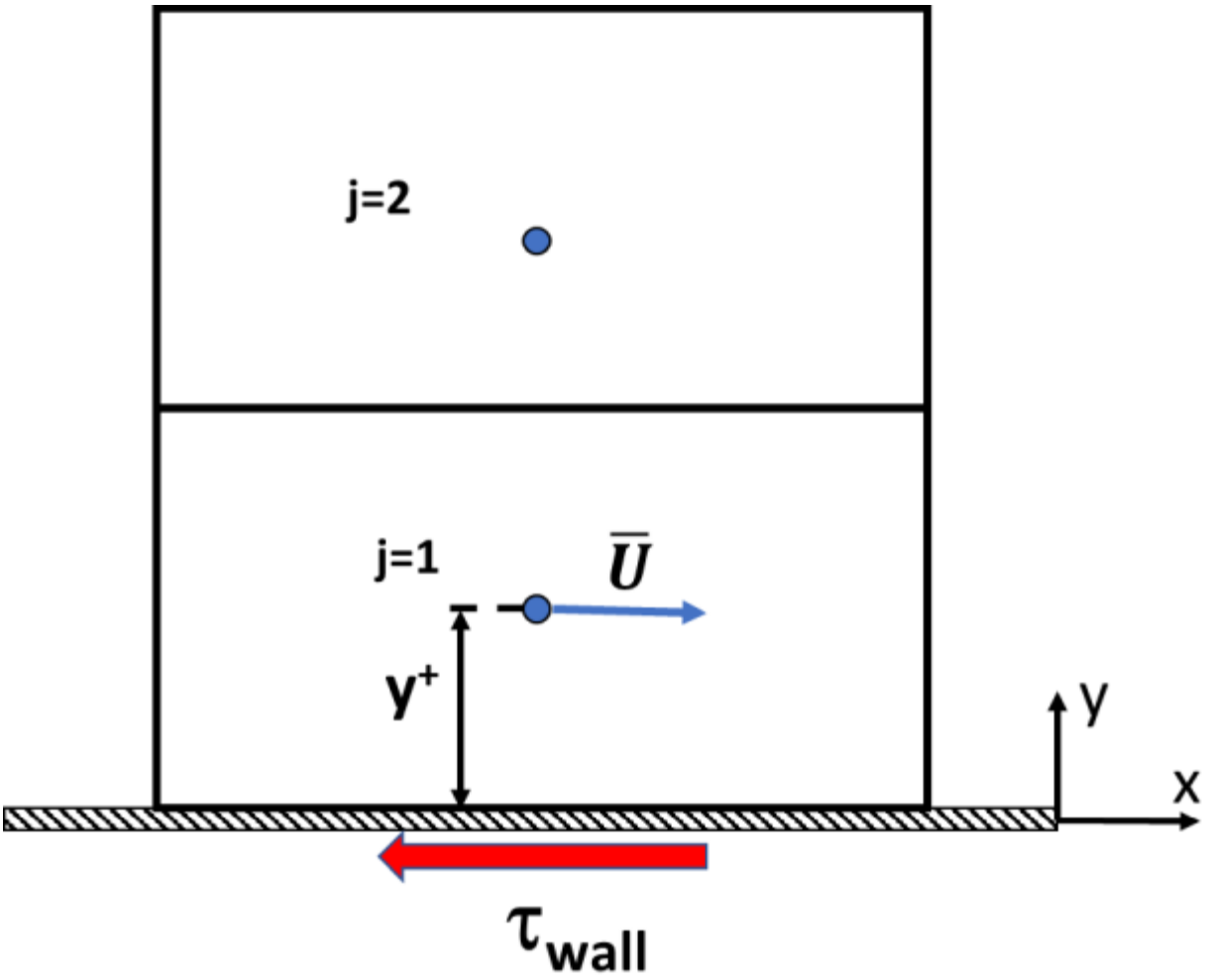


Figure 103: Near wall grid cells for a finite volume cell-centered method.

Near the wall of equilibrium flows (zero pressure gradient boundary layer, channel flow, pipe flow, Couette flow), the solution variables follow universal laws which scale with the wall-shear stress ?wall and the near wall molecular kinematic viscosity, ?. In other words, one can collapse all equilibrium near wall profiles onto a single curve if the near wall variables are non-dimensionalized by these quantities. The non-dimensionalization is based on the wall-friction velocity u_τ (computed from $u_\tau=\sqrt{\tau_{wall}}/\rho$), the turbulent kinetic energy $u^*=C_\mu^{0.25}\sqrt{k}$ (note that in the logarithmic layer one has $u_\tau=u^*$) and the kinematic viscosity ? in the following way:

$$U^+=\frac{\overline{U}}{u_\tau},\ y^+=\frac{u_\tau y}{\nu},\ y^*=\frac{u^* y}{\nu},\ k^+=\frac{k}{u_\tau^2},\ \omega^+=\frac{\omega\nu}{u_\tau^2},\ \varepsilon=\frac{\varepsilon\nu}{u_\tau^4} \tag{12.96}$$

Using these variables, one obtains a universal velocity profile of the form:

$$U^+=F(y^+)\ \text{or}\ U^+=F(y^*) \tag{12.97}$$

Such profiles are depicted in Figure 104 for two Reynolds number of a boundary layer. For both Reynolds numbers, the profiles are identical near the wall and differ in the outer part. The extent to which the universal profile is followed depends on the Reynolds number.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Assuming a universal velocity profile does exist, what is the benefit for CFD? Suppose the velocity is known in the near wall cell (j=1) e.g. from the last iteration. Since the location of the first cell center is also known and so are the viscosity and density, they can all be inserted into Equation (9.56). The only unknown is then the friction velocity u_τ, which can therefore be calculated (iteratively due to the non-linearity of the equation). From u_τ follows the wall-shear-stress $\tau_{wall}=\rho u_\tau u_\tau$, which is required to close the discretization of the near wall cell. Once this achieved, a new global iteration can be started until convergence. The details are slightly different for each model (involving e.g. u^*) and can be found in the user documentation. However, the principle is as outlined here.

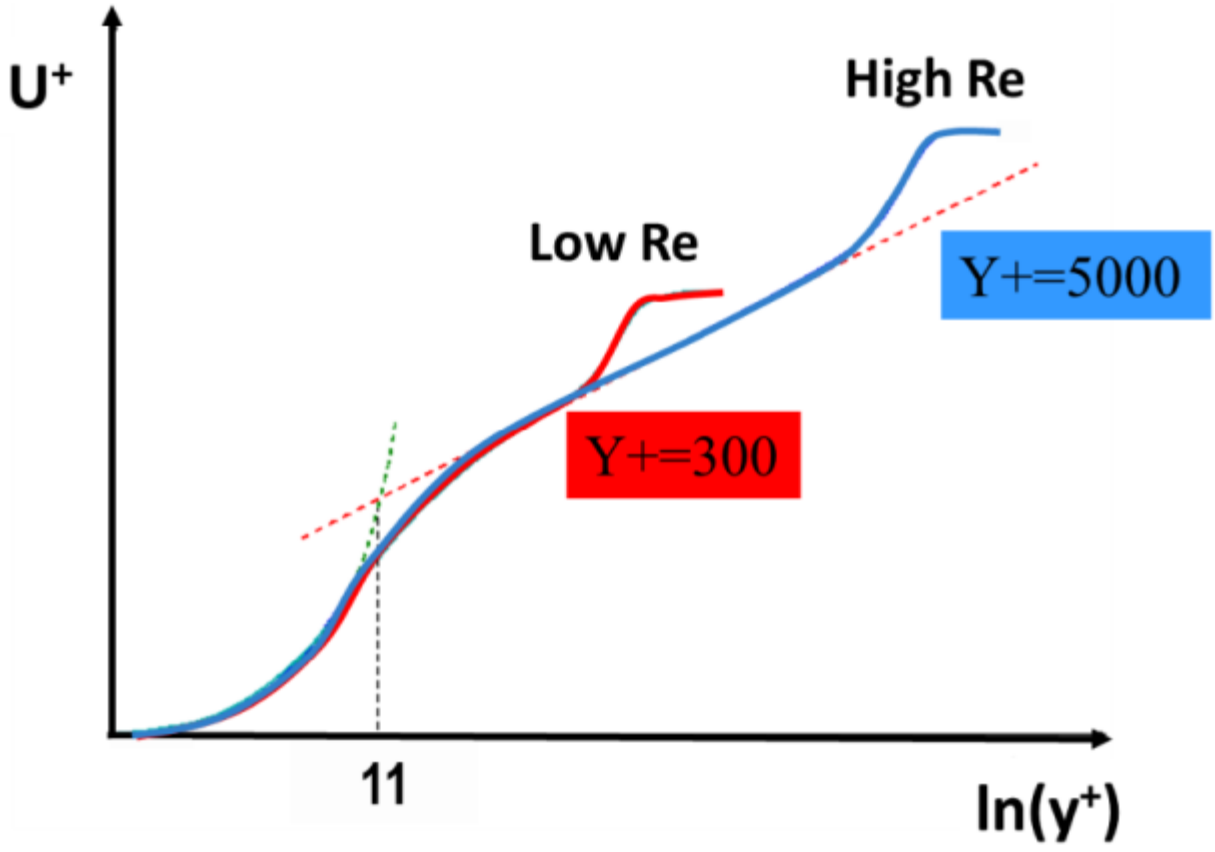
The universal profile consists of different parts. Very close to the wall, turbulence is damped, and the velocity profile is linear (viscous sublayer). In the fully-turbulent near wall layer, the profile follows a logarithmic form (log-law). These two regions are bridged by a 'buffer layer'. This is illustrated in Figure 12.167: Universal non-dimensional velocity-profile near the wall. (p. 393) which depicts two flat plate boundary layer profiles (at different Reynolds numbers) in a logarithmic plot. The velocity profile can roughly be split as follows:

Linear	$0 < y^+ < 5$	$U^+ = y^+$
Buffer	$5 \le y^+ \le 20$	
Logarithmic	$20 < y^+ < \cdots$	$U^+ = \frac{1}{\kappa} ln(y^+) + C$

The log-profile has two constants - the von Karman constant which is typically selected to be k=0.41 and an additive constant of approximately C=5.2 .

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.167: Universal non-dimensional velocity-profile near the wall.



Important: while the velocity profile follows a universal law near the wall, the thickness of the log-layer in terms of y^+ depends on the Reynolds number. A y^+ resolution of say y^+=50 will therefore be sufficient for a high Reynolds number flow, but not for a Low Reynolds number flow. In the high Re case, there are plenty of cells between the wall cell and the boundary layer edge (which is say at δ^+=5000), compared to a low-Re case where the boundary layer edge might be at say δ^+=200.

Important:

Even for high Re number flows, the inner layers (sub+ buffer+ log-law) cover only 10-20% of the entire boundary layer thickness.

The following topics are discussed:

12.2.8.3.1. Standard Wall Functions
12.2.8.3.2. Scalable Wall Functions
12.2.8.3.3. Viscous Sublayer Model (VSM)
12.2.8.3.4. Y+-Insensitive Wall Treatments

12.2.8.3.1. Standard Wall Functions

Historically, many turbulence models have been combined with a 'wall-function' (WF) approach. WF do not allow/require to integrate the equations through the viscous sublayer. Instead, the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

user needs to provide a grid, where the first grid point (cell center) lies in the log-layer. The wall shear stress is then computed from the log-layer formulation alone. Details of specific WF formulations can be more elaborate and can be found in the code documentation.

Wall function treatment has the advantage that simulations can be carried out on relatively coarse grids (in wall normal direction). However, the user must ensure that the first cell center lies in the log-layer. As depicted in Figure Figure 104, this is not trivial, as the upper limit of the log-layer depends on the Reynolds number of the boundary layer profile. One can therefore not define a universal upper limit, as to where the log layer is applicable. Especially for low to moderate Reynolds number (device Re numbers of 10^4-10^6) the thickness of the log-layer might be narrow, and the first grid point can easily be placed either too close or too far from the wall, missing the layer. In addition, at the lower Reynolds numbers, even if the first cell center is placed in the log-layer, there might be not enough resolution to cover the rest of the boundary layer with enough cells.

Standard wall functions are highly problematic, as the solution depends on the ability of the user to:

- Ensure that the cell center point is not too close to the wall so not to fall into the buffer/sub-layer.
- Ensure that the cell center point is not too far from the wall to not fall into the outer layer.
- Ensure that the boundary layer is resolved with enough cells.

The violation of any of these conditions can result in substantial errors. Most problematic is that mesh refinement beyond a certain level leads to a deterioration of results. This is counter to the principles of numerical methods, where grid refinement should eventually lead to more consistent and grid-converged solutions. The above requirements are hard to achieve, especially at low-moderate Reynolds numbers. One needs also not to forget that the mesh is created before a solution is available, so placing the first grid cell into the log-layer is a difficult job. The use of standard wall functions is therefore discouraged – it is not available when using $k-\omega$ based models for that reason.

Users are often under the false impression that wall functions 'take care of the boundary layer' – meaning that a grid with a properly placed y^+ near wall cell is all that is needed for an accurate boundary layer simulation. This is not the case. Wall functions are just a special type of boundary conditions – the boundary layer itself still needs to be resolved with a sufficient number of cells.

Important:

- With standard wall functions, one can easily create incorrect solutions, either by overly coarse or overly fine meshes. The usage of standard wall functions is therefore not recommended.
- Having a proper y^+ is not sufficient for accurate boundary layer simulations. One also needs a sufficient number of cells inside the boundary layer.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.2.8.3.2. Scalable Wall Functions

A simple trick can be employed to avoid the solution deterioration under mesh refinement with standard wall functions, by applying a limiter onto y^+ [55 (p. 375)]:

$$\tilde{y}^+ = \max(y^+, 11) \tag{12.98}$$

This limiter prevents the wall functions to move into the viscous sublayer and thereby guarantees a consistent wall shear-stress (and wall heat-transfer) even under mesh refinement. While the scalable wall function is a substantial improvement, it still entails significant assumptions, as the mesh is no longer refined towards the wall, but towards a fictitious location at y^+=11, thereby slightly altering the geometry. This change is more noticeable at low Reynolds numbers as the gap between y^+=0 and y^+=11 becomes a larger portion of the overall geometry (e.g. the height of the channel or boundary layer) than for high Reynolds numbers.

12.2.8.3.3. Viscous Sublayer Model (VSM)

A more consistent, albeit more expensive method is to integrate the equations all the way to the wall. This requires a near-wall grid with a resolution of y^+~1 or finer. However, such an integration can only be performed if the turbulence models is calibrated to represent the sublayer. Since turbulence is damped by viscosity in the buffer- and sublayer, most turbulence models require additional damping terms and/or specific boundary conditions to account for this effect.

Historically models which can be integrated to the wall have been termed 'Low-Reynolds number' models or 'low-Re' models. This is a highly confusing nomenclature, as the term 'low-Re' refers not to the device Reynolds number, but to the turbulent Reynolds number $\mathrm{Re}_t = \sqrt{k}\, y / \nu$ where k is the turbulent kinetic energy, y is the wall distance and ν is the kinematic viscosity. As k is damped by the presence of the wall, the turbulent kinetic energy k approaches zero for low y^+ levels and so does the turbulent Re number. This is the case for all flows close to the wall, at all device Reynolds numbers! In the following, the term 'low-Re model' will therefore be replaced by the terminology 'Viscous Sublayer Model – VSM'.

Developing VSMs for ε-equation based models has been a major challenge. Many such models have been developed over time (e.g. [56 (p. 375)]), but they typically suffer from one or more of the following deficiencies:

- Lack of robustness.
- Multiple solutions depending on initial conditions.
- Pseudo-transitional behavior not calibrated against data.
- Excessive wall shear and heat transfer at reattachment/stagnation points/lines.

There are some remedies to this. The first and most widely applied is the usage of a so-called 2-Layer formulation ([57 (p. 375)], [58 (p. 375)]). In this method, the epsilon equation is not solved through the buffer and sublayer but an algebraic formulation is used instead. The ε-equation is then dominated by a source term which enforces the algebraic relations.

A second approach is to add additional differential equations to the ε-equations based models which account for the near wall damping. These models are typically called 'elliptic-blending' or V2F models [59 (p. 375)]. The downside of this approach is the significantly increased complexity due to two additional equations and the associated complex boundary conditions. In addition,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

such models still allow for pseudo-transitional behavior, meaning an artificially non-calibrated laminar-turbulent transition process, which is undesirable. For these reasons V2F-like models are not perused in Ansys-CFD.

One of the strengths of the ω-equation based models is that they do not require additional damping terms. Integration to the wall can be obtained by a simple switch in boundary conditions. These models also avoid pseudo-transition for all practical purposes and do not lead to excessive wall shear and heat transfer at reattachment/stagnation points/lines. For this reason, ω-equation based models are the preferred option in Ansys CFD.

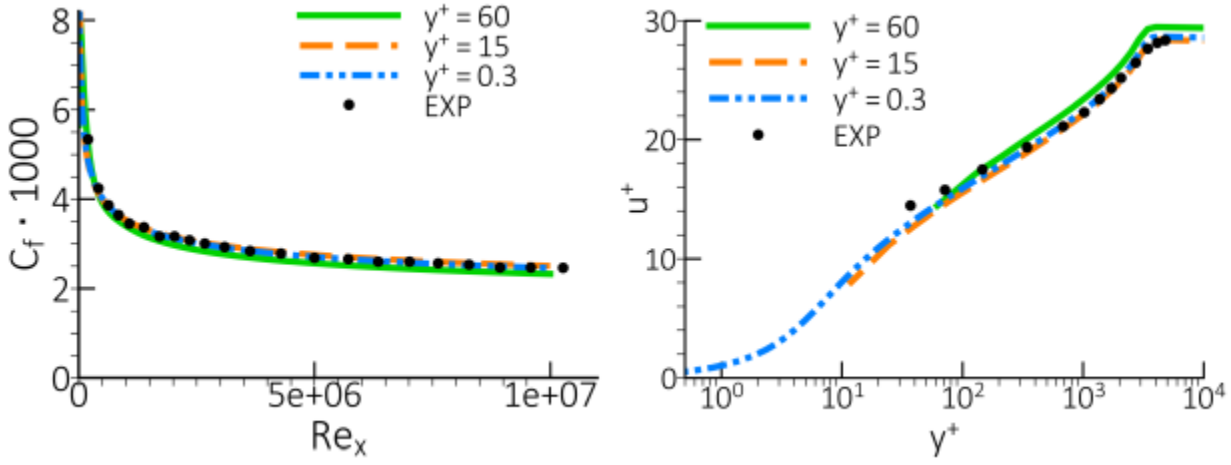
Important:

ε-equation based models are much harder to integrate to the wall than ?-based models.

12.2.8.3.4. Y+-Insensitive Wall Treatments

Classical VSM require always a fine near wall mesh with y^+~1 or smaller. This cannot be achieved for all walls in complex applications. It is therefore desirable to be able to provide formulations which maintain consistency for a variation in y^+. Such models are termed y^+-insensitive models (or all-y^+ models). When computing a flow on a series of grids with different y^+-resolution, such models provide wall shear-stress (and heat transfer) levels which are approximately independent of the y^+-value. Note that this statement is only correct in case of a sufficient number of cells inside the boundary layer. Simulations with such a formulation can be seen in Figure 12.168: Comparison of the skin friction coefficient and velocity profiles at $\mathrm{Re}_x=8.7\cdot10^6$ on different meshes for the flat plate boundary layer (28). (p. 396), where a flat plate zero pressure gradient boundary layer (28 (p. 371)) is computed on three different grids. The computed velocity profiles follow closely the fine grid solution. On meshes with y^+>20, the formulation essentially switches back to a wall function.

Figure 12.168: Comparison of the skin friction coefficient and velocity profiles at $\mathrm{Re}_x=8.7\cdot10^6$ on different meshes for the flat plate boundary layer (28 (p. 371)).



In Ansys Fluent, y^+-insensitive wall formulations are available for the Spalart-Allmaras one-equation model and $k-\varepsilon$ models (using a 2-Layer VSM formulation and the ML VSM). In both codes (Ansys Fluent and Ansys CFX) all ω-equations based models are based on a y^+-insensitive formulation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The details of such formulations are model and numerics dependent and are given in more detail in the Theory Documentation of the corresponding codes.

Y+-Insensitive Near Wall Treatment for ω-based Models

Near the wall, the ω-equation reduces to (in the viscous sublayer production and turbulent diffusion can also be neglected):

$$0=C_{\omega 1}S^2-C_{\omega 2}\rho\omega^2+\frac{\partial}{\partial x_j}\left(\left(\mu+\frac{\mu_t}{\sigma_\omega}\right)\frac{\partial\omega}{\partial x_j}\right) \tag{12.99}$$

This equation has the following near-wall solutions for ω in the viscous sub-layer and the log-layer:

$$\omega_{NW}=\frac{1}{\left(\sqrt{\frac{C_{\omega 2}}{6\nu}y^2}+\sqrt{\frac{1}{\omega_0}}\right)^2} \tag{12.100}$$

$$\omega_{Log}=\frac{1}{\sqrt{C_\mu}}\frac{\mu_\tau}{ky} \tag{12.101}$$

where ω_0 is the (typically high) value specified for ω at the wall. This formulation is sufficient to provide proper near wall damping and the correct logarithmic layer. The ω-equations y^+-insensitive formulation blends smoothly between these two solutions.

The simple and robust near wall formulation without the need for complex damping functions (or new additional transport equations like in the V2F model) makes the $k-\omega$ model the most attractive option as a basis for industrial turbulence models.

The $k-\varepsilon$ Two-Layer Formulation

To avoid the complication and robustness limitations of historic VSM (low-Re) $k-\varepsilon$ models, but still enable CFD users to integrate these models through the viscous sublayer, the so-called 'two-layer' (2L) formulation was introduced [57 (p. 375)], [58 (p. 375)]. The 2L formulation is today one of the more widely used turbulence near-wall formulations in industrial codes for the $k-\varepsilon$ model. Its principal idea is to avoid the complexities of the ε-equation near the wall entirely and to replace it with an algebraic formulation, using a mixing-length model there. This algebraic formulation is then blended with the ε-equation away from the wall.

In the 2L model the near-wall ε-distribution is computed from an algebraic equation:

$$\varepsilon=\frac{k^{1.5}}{l_\varepsilon} \tag{12.102}$$

where l_ε is the mixing-length based on wall-distance, y:

$$l_\varepsilon=yC_l^*\left(1-e^{-\mathrm{Re}_y/A_\varepsilon}\right)\ \text{with}\ \mathrm{Re}_y=\frac{y\sqrt{k}}{\nu} \tag{12.103}$$

In addition, the eddy-viscosity is also blended between the standard $k-\varepsilon$ formulation and an algebraic mixing length model:

$$\begin{aligned}\mu_{t,EWT}&=\lambda_\varepsilon\mu_t+(1-\lambda_\varepsilon)\mu_{t,2\mathrm{Layer}}\\ \mu_{t,2\mathrm{Layer}}&=\rho C_\mu l_\mu\sqrt{k}\\ l_\mu&=yC_l^*\left(1-e^{-\mathrm{Re}_y/A_\mu}\right)\end{aligned} \tag{12.104}$$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Using the blending function:

$$F_{2Layer}=\lambda_{\varepsilon}=\frac{1}{2}\left[1+\tanh\left(\frac{\mathrm{Re}_y-\mathrm{Re}_y^*}{A}\right)\right]'\ \mathrm{Re}_y^*=200$$
$$A=\frac{\Delta\mathrm{Re}_y}{a\tanh(0.98)}\qquad\Delta\mathrm{Re}_y=0.1\cdot\mathrm{Re}_y^* \tag{12.105}$$

The blending is required as the distributions resulting from the algebraic formulation and from the transport equation will not match at a pre-specified Re_y location. To prevent discontinuities and convergence problems, a smooth variation is required.

The constants are:

$$C_l^*=kC_{\mu}^{-3/4},\quad A_{\mu}=70,\quad A_{\varepsilon}=2C_l^* \tag{12.106}$$

The turbulent Reynolds number Re_y is also used to blend between the algebraic model and the transport equation for ε. The details of these blending are typically code-dependent and are therefore not provided here. Such blending between alg. and transport equations can lead to robustness problems and therefore needs to be performed gradually over a significant portion of the inner portion of the boundary layer (Re_y<200).

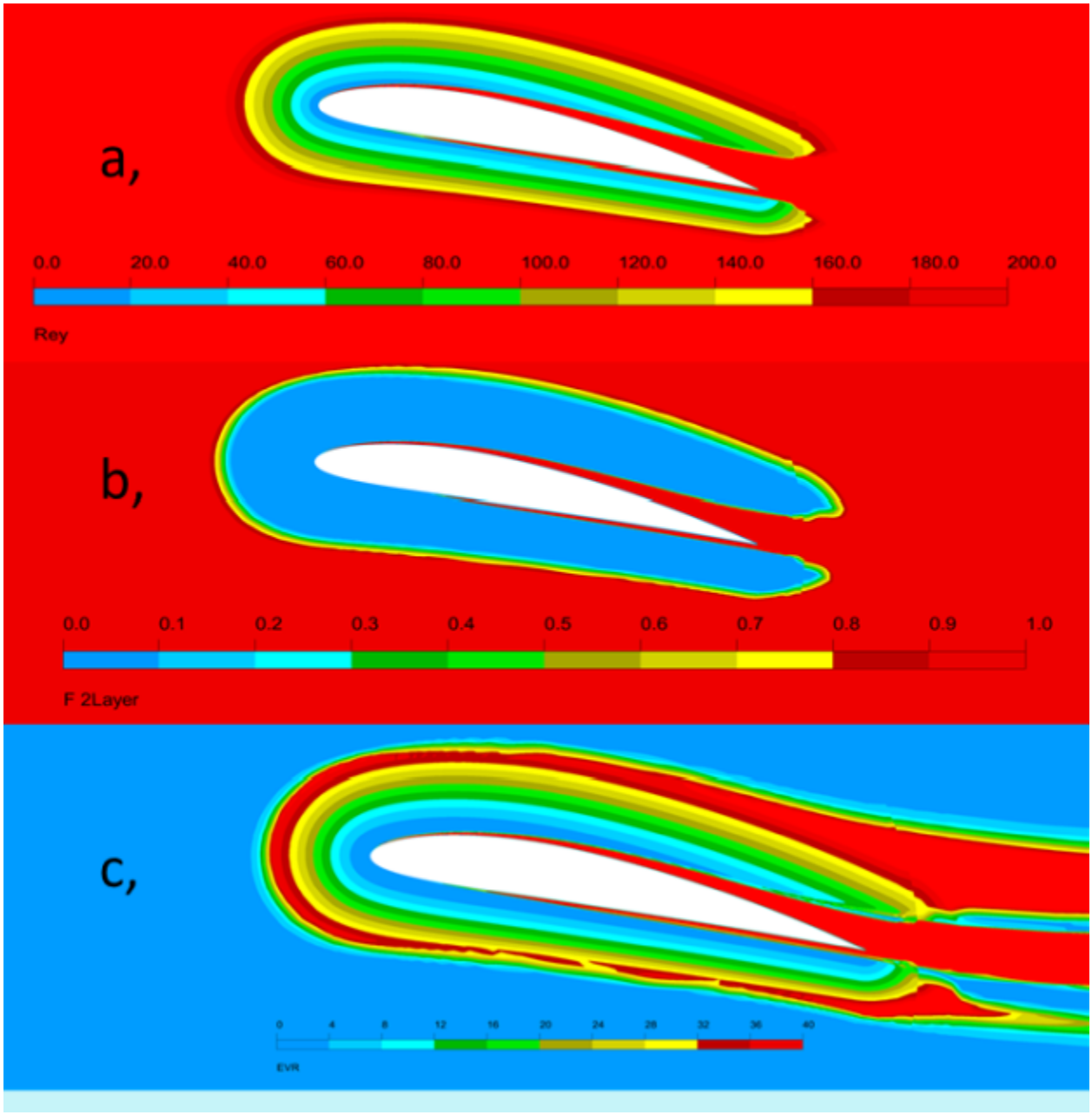
The 2L-blending can cause problems in low-Re number flows, where the model can be dominated by the algebraic formulation. There is also a considerable Reynolds-number effect for non-equilibrium flows, where the 2L mixing length model part and the ε-equation produce different results, which are then stitched together with the blending function. The relative portion across the boundary layer where the mixing length model is active depends on the Re number of the boundary layer. For low Re, a substantial part of the boundary layer will be covered by the alg. part of the model, whereas for high Re, the 2L model is active only in a small near-wall portion (relative to the boundary layer thickness). As both models behave differently for non-equilibrium flows, this mean that the combined model will lean towards a mixing length model for low Re_{θ} and towards a $k-\varepsilon$ model for high Re_{θ}.

Another problematic issue results from the observation that the blending function based on $\mathrm{Re}_y=\sqrt{k}\,y/n$ can switch back to 'near-wall mode' even outside the boundary layer in case of low freestream values for k, causing non-physical distributions of the turbulence variables outside the boundary layer. By this mechanism, a layer of high turbulence viscosity can be created artificially, as the eddy-viscosity is also switched to the mixing length formulation (with obviously a large mixing length due to the large wall distance). Figure 12.169: Contours of the Re_y (Top, a,) and the Blending Function (Middle, b,) and the ratio of eddy-viscosity/molecular viscosity $EVR=\mu_t/\mu$ (Bottom, c,) of the 2L formulation for flow around the NACA-4412 airfoil at AoA = 12° (p. 399) shows a simulation around a NACA 4412 airfoil. The Re_y distribution used inside the blending function is shown in part a, of the Figure. Re_y is zero at the wall, increases rapidly inside the boundary layer (hard to see in this scale), but then decreases again outside the boundary layer. The effect on the blending function is shown in the middle Figure 106. The function is F_{2Layer}=0 close to the wall (not seen on this scale) and then switches to F_{2Layer}=1 inside the boundary layer as desired. However, outside the boundary layer, the formulation switches back to the algebraic form (F_{2Layer}=0). This switch-back is problematic, as it also switches the eddy-viscosity back to the alg. formulation, which can be seen in the lower part of Figure 106c, where $EVR=\mu_t/\mu$ is plotted. This ratio reaches relatively high values outside the boundary layer and the switches back to the low freestream values further away, as the function switches back to F_{2Layer}=1.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

These examples demonstrate that the 2L model can cause issues due to its switching procedure. This is even more the case for flows with additional complexities like multi-phase flows and/or flows with large changes in physical properties, where the model has shown instabilities due to erratic switching between the mixing length and the transport equation, thereby preventing solver convergence. While the 2L formulation works well for many industrial flows, it is not strong enough to serve as a default turbulence model which needs to work for all combinations of physics.

Figure 12.169: Contours of the Re_y (Top, a,) and the Blending Function (Middle, b,) and the ratio of eddy-viscosity/molecular viscosity $EVR=\mu_t/\mu$ (Bottom, c,) of the 2L formulation for flow around the NACA-4412 airfoil at AoA = 12°



12.2.8.4. Appendix A: Boundary Layer Parameters

In order to create a high-quality mesh, it is helpful to have some approximate formulas for the development of boundary layers for zero pressure gradient boundary layers. The Boundary Layer develops in streamwise direction (x-direction) for a flat plate with origin at x=0. The boundary layer thickness is δ, the freestream velocity is U and the wall shear stress is τ_{wall}. The wall friction coefficient is $C_f = \tau_{\text{wall}} / (0.5\rho U^2)$ and $\text{Re}_x = \rho U x / \mu$

The following topics are discussed:

12.2.8.4.1. Laminar Flow:
12.2.8.4.2. Turbulent Flow:

12.2.8.4.1. Laminar Flow:

Wall shear stress:

$$C_f = 0.664 \text{Re}_x^{-1/2} \tag{12.107}$$

Boundary Layer thickness:

$$\frac{\delta}{x} = 5.0 \text{Re}_x^{-1/2} \tag{12.108}$$

12.2.8.4.2. Turbulent Flow:

Wall shear stress coefficient:

$$C_f = 0.058 \text{Re}_x^{-1/5} \tag{12.109}$$

Boundary Layer thickness:

$$\frac{\delta}{x} = 0.37 \text{Re}_x^{-1/5} \tag{12.110}$$

Viscous sublayer thickness – y^+-estimate:

$$\Delta y^+ = \frac{u_\tau \Delta y}{\nu} \rightarrow \Delta y = \frac{\Delta y^+ \nu}{u_\tau} \tag{12.111}$$

$$u_\tau = \sqrt{\frac{\tau_{wall}}{\rho}} = U\sqrt{0.5 C_f} = U\sqrt{0.5 \cdot 0.058 \cdot \text{Re}_x^{-1/5}} = 0.17 \cdot U \cdot \text{Re}_x^{-1/10} \tag{12.112}$$

$$\Delta y = \frac{\Delta y^+ \nu}{0.17 \cdot U \cdot \text{Re}_x^{-1/10}} = 5.8 \cdot \Delta y^+ \frac{\nu}{U} \text{Re}_x^{1/10} \tag{12.113}$$

This formula provides the required wall spacing Δy to achieve a desired Δy^+. For Re_x it is prudent to use half the running length of the boundary layer (e.g. half chord of an airfoil). The good news from this equation is that Δy^+ has only a very weak variation with Re_x. It is therefore possible to use a constant near wall grid spacing Δy for the entire body. Note that the above Δy^+ estimate is based on the grid distance between the surface and the first grid node off the wall. The code can internally use another Δy^+ definition – e.g. in a cell-centered code like Ansys Fluent it would use half of that value.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3. Generalized k-omega Two-Equation Turbulence Model (GEKO) in Ansys CFD

Two-equation turbulence models have matured to a point where a consolidation seems desirable. The main models used in industrial CFD codes today are $k-\varepsilon$ [14 (p. 443)] (standard and realizable/RKE), $k-\omega$[18 (p. 443),33 (p. 446),34 (p. 446)] (SST, BSL, Wilcox) and to a lesser degree V2f [12 (p. 443)] (different versions). The differences between the models are not fundamental, but can nevertheless have a strong impact on results. For boundary layers, the models differ mostly in their 'aggressiveness' to predict separation onset. Furthermore, in the very near wall region, models can predict vastly different results, especially for heat transfer simulations, due to their differences in wall-treatment. There are also noticeable differences for free shear flows, where each model tends to favor certain flows over others. Finally, different models feature different limiters, which typically do not affect the baseline flows, but can have substantial influence in complex applications.

ANSYS developed a new turbulence model family called Generalized $k-\omega$ (GEKO) model with the goal of turbulence model consolidation. GEKO is a two-equation model, based on the $k-\omega$ model formulation, but with the flexibility to tune the model over a wide range of flow scenarios. The key to such a strategy is the provision of free parameters which the user can adjust for specific types of applications without negative impact on the basic calibration of the model. In other words, instead of providing users flexibility through a multitude of different models, the current approach aims at providing one framework, using different coefficients to cover different application sectors. This will substantially simplify code usage for industrial CFD users. This approach also offers a much wider range of calibration capabilities than currently covered by switching between existing models. Finally, GEKO is (or will be made) compatible with all other options in the codes, so that there is no need to select any other model for compatibility or accuracy reasons.

Historically, the coefficients of turbulence models are exposed in the GUI to users (e.g. $C_{\varepsilon 1}$, $C_{\varepsilon 2}$, etc. in a $k-\varepsilon$ model). However, this exposure is of little value, as most coefficients are linked to the basic calibration of the model (namely the calibration for the logarithmic law affecting e.g. flat plate simulations). Users can therefore not freely change these values without affecting such flows. In the GEKO model, free coefficients are introduced, which do not affect the logarithmic layer calibration and can therefore be tuned to achieve the desired model behavior. The GEKO model offers six free parameters – two of them aiming at wall bounded flows, two for the calibration of free shear flows, one coefficient to improve corner flow simulations (corner separation) and finally a curvature correction term.

The generic idea behind the model will be discussed. Not all details can be provided as the model is at present unpublished. However, the variability of the model will be demonstrated for a variety of generic flows and Best Practice Guidelines for optimal usage will be provided.

In order to keep the document compact, only a sketch of the geometry is provided for the test cases as well as a reference to the publication. This is sufficient, as the test cases are typically simple and it is only required to understand the basic flow challenge.

The following topics are discussed:

12.3.1. The Generalized k-omega (GEKO) Model Formulation
12.3.2. The Influence of the Free GEKO Parameter
12.3.3. Strategies for Model Optimization
12.3.4. Summary
12.3.5. Example UDFs

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3.6.7 References

12.3.1. The Generalized k-omega (GEKO) Model Formulation

The following topics are discussed:

12.3.1.1. Basic Formulation
12.3.1.2. Limiters and Realizability
12.3.1.3. Near Wall Treatment
12.3.1.4. Terminology

12.3.1.1. Basic Formulation

The main characteristic of the GEKO model is that it has several free parameters for tuning the model to different flow scenarios. The starting point for the formulation is:

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial(\rho U_j k)}{\partial x_j}=P_k-C_\mu\rho k\omega+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] \tag{12.114}$$

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial(\rho U_j \omega)}{\partial x_j}=C_{\omega 1}F_1\frac{\omega}{k}P_k-C_{\omega 2}F_2\rho\omega^2+\rho F_3 CD+\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\omega}\right)\frac{\partial \omega}{\partial x_j}\right] \tag{12.115}$$

$$\mu_t=\rho\nu_t=\rho\frac{k}{max\left(\omega,S/C_{\text{Realize}}\right)} \tag{12.116}$$

$$P_k=-\tau_{ij}\frac{\partial U_i}{\partial x_j} \tag{12.117}$$

$$\tau_{ij}^{EV}=-\rho\overline{u_i u_j}=\mu_t 2S_{ij}-\frac{2}{3}\rho k\delta_{ij} \tag{12.118}$$

$$CD=\frac{2}{\sigma_\omega}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \tag{12.119}$$

$$\tau_{i,j}=\tau_{ij}^{EV}-C_{\text{CORNER}}\frac{1.2\mu_t}{\max\left(0.3\omega,\ \sqrt{0.5\left(S^2+\Omega^2\right)}\right)}\left(S_{ik}\Omega_{kj}-\Omega_{ik}S_{kj}\right) \tag{12.120}$$

with

$$S_{ij}=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_j}\right)\qquad \Omega_{ij}=\frac{1}{2}\left(\frac{\partial U_i}{\partial x_j}-\frac{\partial U_j}{\partial x_{ji}}\right)$$

$$S=\sqrt{2S_{ij}S_{ij}}\quad ;\ \Omega=\sqrt{2\Omega_{ij}\Omega_{ij}}$$

The free coefficients of the GEKO model are implemented through the functions (F_1,F_2, F_3) which can be tuned by the user to achieve different goals in different parts of the simulation domain. Currently there are six parameters included for that purpose:

- C_{SEP}
  - Main parameter for adjusting separation prediction for boundary layers

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

  - Affects all flows - Increasing C_{SEP} reduces eddy-viscosity leading to more sensitivity to adverse pressure gradients for boundary layers and to lower spreading rates for free shear flows (compensated by C_{MIX}).

- C_{NW}
  - Affects mostly the inner part of wall boundary layers (limited to no impact on free shear flows.
  - Increasing C_{NW} leads to higher wall shear stress and wall heat transfer rates in non-equilibrium flows.
  - Effect on non-generic flows (e.g. vortices) moderate but not systematically tested.
  - Users can mostly use C_{NW} = 0.5 (default).
- C_{MIX}
  - Affects only free shear flows (boundary layer shielded due to function Fblend).
  - Increasing C_{MIX} increases spreading rates of free shear flows.
  - For each value of C_{SEP} an optimal value of C_{MIX} exists, which maintains optimal free shear flows. This value is given by the correlation C_{MIX}=C_{MixCor} which is default C_{MixCor}=$0.35\mathrm{sign}(C_{SEP}-1)\sqrt{(|C_{SEP}-1|)}$
- C_{JET}
  - Is active in a sub-model of C_{MIX} (no impact for C_{MIX} = 0).
  - Affects mostly jet flows. Increasing C_{JET} while C_{MIX} is active, decreases spreading rate for jets.
  - Allows to adjust spreading rate of jet flows while maintaining spreading rate of mixing layer.
  - Users can mostly use C_{JET} = 0.9 (default).
  - Has no effect in case of C_{MIX} = 0
- C_{CORNER}
  - Non-linear stress-strain term to account for secondary flows in corners (e.g. wing-body junctions etc. [17 (p. 443)]).
- C_{CURV}
  - An existing model for curvature correction, which can be combined with the GEKO model [27 (p. 445),19 (p. 444)]

All coefficients (except of C_{JET} which is of minor importance) can be accessed globally or locally through User Defined Functions (UDFs), allowing a global or zonal model optimization.

The coefficients C_{SEP} and C_{NW} affect boundary layers, whereas C_{MIX} and C_{JET} are designed for free shear flows. In order to avoid any influence of C_{MIX} and C_{JET} onto boundary layers, a blending

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

function is introduced, which de-activates C_{MIX} and C_{JET} in the boundary layer. The function is similar to the blending function used in the BSL/SST model formulation and given by:

$$\begin{aligned} L_T &= \frac{\sqrt{\widetilde{k}}}{C_\mu \omega} \\ \widetilde{k} &= \max(k, CFb_{Lam} \cdot \nu\omega) \\ x_{blend} &= CFb_{Turb} \frac{L_T}{y} \\ F_{GEKO} &= \tanh(x_{blend}^4) \end{aligned} \tag{12.121}$$

This function activates the free shear flow parameters as follows:

$$F_{Free} = C_{MIX} F_{Jet}(C_{JET})(1 - F_{Blend}) \tag{12.122}$$

There are two important aspects. Firstly, the function $F_{\mathrm{Blend}}=1$ inside boundary layers and $F_{\mathrm{Blend}}=0$ for free shear flows. Secondly, the parameter C_{JET} is a sub-parameter of C_{MIX}. It only affects the simulation in case $C_{\mathrm{MIX}} \neq 0$.

The free coefficients should be in the range (defaults in parenthesis):

MIN		Parameter	MAX			Default
0.7	≤	$C_{\mathrm{SEP},}$	≤	2.5		1.75
-2.0	≤	C_{NW}	≤	2.0		0.50
0.0	≤	C_{MIX}	≤	1.0...		C_{MixCor}
0.0	≤	C_{JET}	≤	1.0		0.90
0.0	≤	C_{CORNER}	≤	1.5		1.00
0.0	≤	C_{CURV}	≤	1.5		1.00

The greyed values for C_{MIX} are only suggestions. There might be situations where values higher than 1.0 can be appropriate.

As mentioned above, for the coefficient C_{MIX} a correlation is provided as default, which ensures that changes in C_{SEP} do not negatively affect free mixing layers:

$$C_{\mathrm{MixCor}} = 0.35 \mathrm{sign}(C_{\mathrm{SEP}} - 1) \sqrt{(|C_{\mathrm{SEP}} - 1|)} \tag{12.123}$$

12.3.1.2. Limiters and Realizability

It is well-known that any conventional two-equation model exhibits build-up of turbulence in stagnation regions of bluff bodies (like leading edges of airfoils etc.). The reason lies in the substitution of the eddy-viscosity assumption into the production term P_k of the two-equation model. The eddy-viscosity assumption is not representing the physics correctly in such regions and leads to an over-production of turbulence kinetic energy, which in turn can lead to excessive eddy-viscosity levels in these areas. This can have a detrimental effect on the flow over the rest of the geometry, as the boundary layers starting from the leading edge stagnation point would be predicted incorrectly. This can cause large errors and even false separation. Such situations do not only appear in classical stagnation zones of airfoils, but in many technical flows, where flows hit on a surface or different flow streams collide to form a stagnation region.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

There are numerous remedies for that problem. The most widely known is the use of the so-called Kato-Launder correction [15 (p. 443)] whereby the square of the shear strain in P_k is replaced by strain times vorticity:

$$P_k=\mu_t S^2 \quad \rightarrow \quad P_k=\mu_t S\Omega \tag{12.124}$$

This option is available, albeit typically not by default. It is however activated when using the GEKO model in combination with a model for laminar-turbulent transition, as it prevents even small production rates which can have a significant effect on transitional flows. It should be noted that the Kato-Launder modification can have an effect on flows with rotation and swirl relative to the original model calibration. It also leads to un-physical production in rotating systems as pointed out by Durbin and Reif [11 (p. 442)] and should therefore be applied with caution.

An alternative was proposed by Menter [17 (p. 443)] in form of a production limiter:

$$\tilde{P}_k=\min\left(P_k, C_{PK_{\lim}}\rho\varepsilon\right) \tag{12.125}$$

The limiting coefficient can be chosen fairly large (typically $C_{PK_{\lim}}=10$), relative to the equilibrium relation $P_k/(\rho\varepsilon)=1$. It will therefore not affect any calibrated flow and still avoids the stagnation build-up and is used as a default option in GEKO (as in all other $k-\omega$ based models in Ansys CFD).

A more theoretical concept can also be applied by imposing a realizability constraint. Realizability demands e.g. that all normal Reynolds Stress components need to always remain positive (e.g. [11 (p. 442)]). This is clearly true from a physical standpoint, but can be violated by eddy-viscosity models (as well as EARSM and RSM). One can argue about how important this constraint is from a practical standpoint, as eddy-viscosity models do not attempt to accurately describe each single Reynolds Stress, but model essentially the principal shear stress. Still, the realizability constraint can, as a side-effect, help to avoid stagnation build up. For eddy-viscosity models it reads:

$$\nu_t=\min\left(\frac{k}{\omega}, C_{\text{Realize}}\frac{k}{S}\right)=\frac{k}{\max\left(\omega, S/C_{\text{Realize}}\right)};\quad C_{\text{Realize}}=\frac{1}{\sqrt{3}}\approx 0.577 \tag{12.126}$$

The realizability limiter has the additional benefit to prevent numerical break-down in case ω approaches zero at any point in the domain. Without the constraint in the denominator, this would lead to arbitrarily high eddy-viscosities, whereas with the limiter, such points are typically handled gracefully.

In the GEKO model the realizability limiter is utilized in addition to the production limiter by default. Users can change the values of both coefficients.

12.3.1.3. Near Wall Treatment

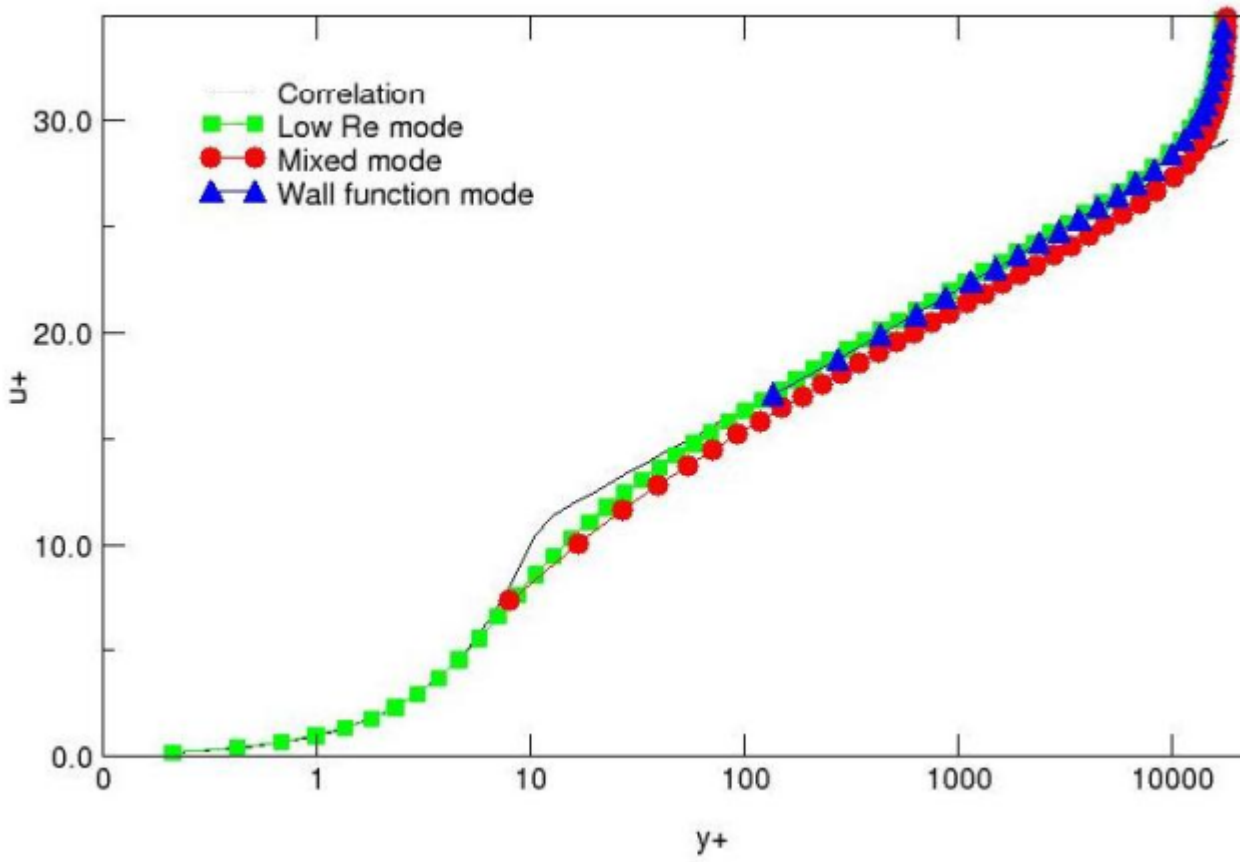
The near wall formulation of a turbulence model has a substantial effect on its accuracy and its robustness. In addition, modern CFD codes typically feature so-called y^+-insensitive wall formulations, which allow the user to obtain sensible simulations over a wide range of grids with different y^+ near wall resolutions.

The GEKO model family was designed by sticking closely to the original $k-\omega$ model formulation – following the argument that the modeled k is proportional to $\overline{v'v'}$ and not to the physical turbulence kinetic energy. The peak observed in the turbulence kinetic energy in the buffer layer is therefore not modelled, as it consists mostly of passive motion (no effect on shear stress).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A y^+-insensitive wall formulation has been developed for the GEKO model (similar to what is used in the BSL and SST models). It allows the use of the model, on meshes of arbitrary y^+ values, as long as the y^+-value lies in the logarithmic layer of the boundary layer, and as long as the rest of the boundary layer is resolved with a sufficient number of cells.

Figure 12.170: Near wall y^+ insensitive wall treatment for GEKO model.



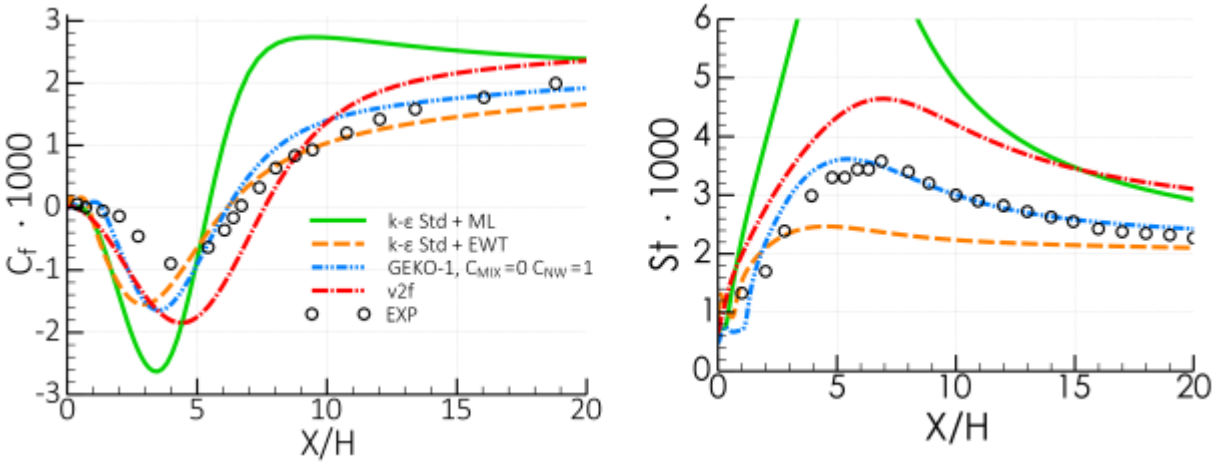
The y^+-insensitive wall formulation has the advantage that users do not have to select a wall treatment. The optimal formulation is selected by the formulation based on the grid provided.

It is important to counter a widely held belief that $k-\omega$ based models require a finer near wall resolution than say a $k-\varepsilon$ model with wall functions. This is not correct, as the y^+-insensitive wall formulation blends to the exact same wall function once the grid is coarsened.

In order to demonstrate the superior behavior of $k-\omega$ based models compared with other models, a backstep was computed on a y+~1 mesh. The wall shear stress and Stanton number (heat transfer) distribution downstream of the step are shown in Figure 12.171: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow [31] (p. 407). All curves are based on the same high Re number $k-\varepsilon$ model (the GEKO model is set to an exact transformation of $k-\varepsilon$). The ML is a low Re number $k-\varepsilon$ model, EWT is a 2-layer formulation [35 (p. 446)] and the V2f model is an extension of $k-\varepsilon$ with elliptic blending [12 (p. 443)]. While all baseline models are essentially identical, the differences in near wall formulation results in very large differences between the results. It is obvious that the GEKO model is closest to the experimental data.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.171: Wall shear stress coefficient, C_f (left) and wall heat transfer coefficient, St, (right) for backward-facing step flow [31]



12.3.1.4. Terminology

In order to clearly characterize the model variant used in an application, it is important to have a unified terminology for the model. It is proposed to just name the coefficients which are not default.

- GEKO with C_{SEP}=1.2, C_{NW}=1.0, C_{MIX}=1.0, C_{JET}=0.9 would be termed
  - GEKO:(C_{SEP}=1.2, C_{NW}=1.0, C_{MIX}=1.0).
- GEKO with C_{SEP}=1.5, C_{NW}=0.0, C_{MIX}=CMixCor, C_{JET}=1.0 would be termed
  - GEKO:(C_{SEP}=1.5, C_{NW}=0.0, C_{JET}=1.0).
- Situations where only C_{SEP} is changed (most frequent case) will just be characterized in short notion:
  - GEKO with C_{SEP}=1.5 will be termed GEKO-1.5

12.3.2. The Influence of the Free GEKO Parameter

The following topics are discussed:

12.3.2.1. The 'Separation' Parameter CSEP
12.3.2.2. 3.2 The 'Near Wall' Parameter CNW
12.3.2.3. The 'Mixing' Parameter CMIX
12.3.2.4. The 'Jet' Parameter CJET
12.3.2.5. The 'Corner' Parameter CCORNER
12.3.2.6. The 'Curvature' Parameter CCURV
12.3.2.7. The Blending Function
12.3.2.8. Other Special Coefficients

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3.2.1. The 'Separation' Parameter C_{SEP}

The ability to predict separation depends mostly on the level of the eddy-viscosity in the boundary layer. A suitable approach for tuning the model to adverse pressure gradients and separation is to allow a re-calibration of the basic model constants, while at the same time maintaining the calibration for the slope of the logarithmic layer and the proper near wall viscous damping required for achieving the correct shift in the log-layer (and thereby the correct wall shear stress). This is achieved by using the free parameter C_{SEP}.

Figure 12.172: Change of the eddy-viscosity ration (EVR) under changes of $C_{\mathrm{SEP},}$ for a flat plate. Top:$C_{\mathrm{SEP},}$=1.0 (all else default). Bottom $C_{\mathrm{SEP},}$=1.75 (all else default) (p. 408) shows the effect of C_{SEP} on a flat plate boundary layer computation for the ratio $EVR=\mu_t/\mu$. Increasing C_{SEP} from C_{SEP}=1.00 to C_{SEP}=1.75 leads to a significant reduction in the ER levels. (Note that the effect is even more pronounced for flows with adverse pressure gradients and separation).

Figure 12.172: Change of the eddy-viscosity ration (EVR) under changes of $C_{\mathrm{SEP},}$ for a flat plate. Top:$C_{\mathrm{SEP},}$=1.0 (all else default). Bottom $C_{\mathrm{SEP},}$=1.75 (all else default)

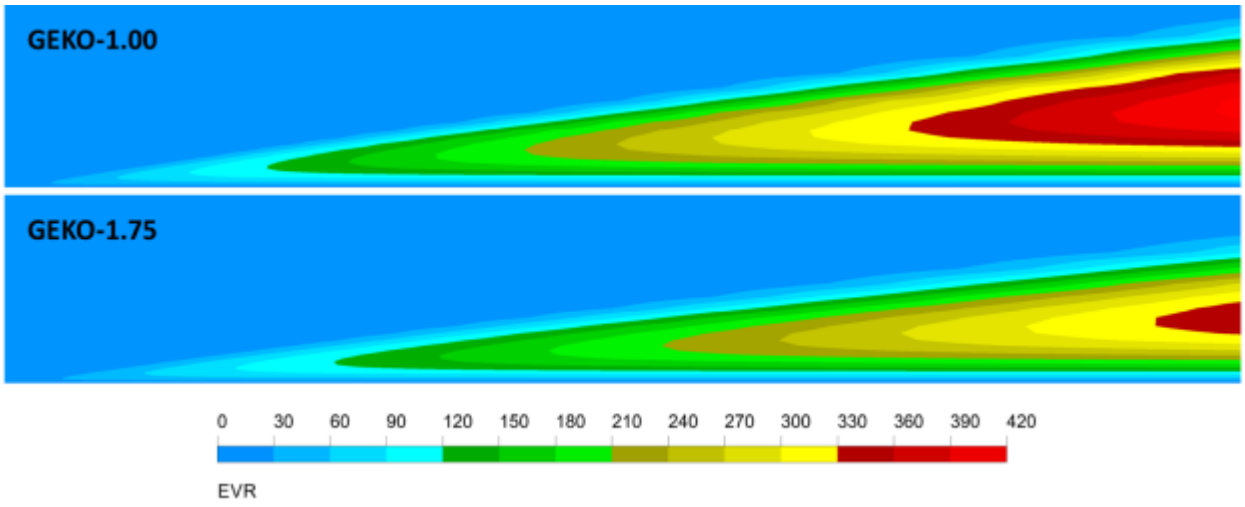
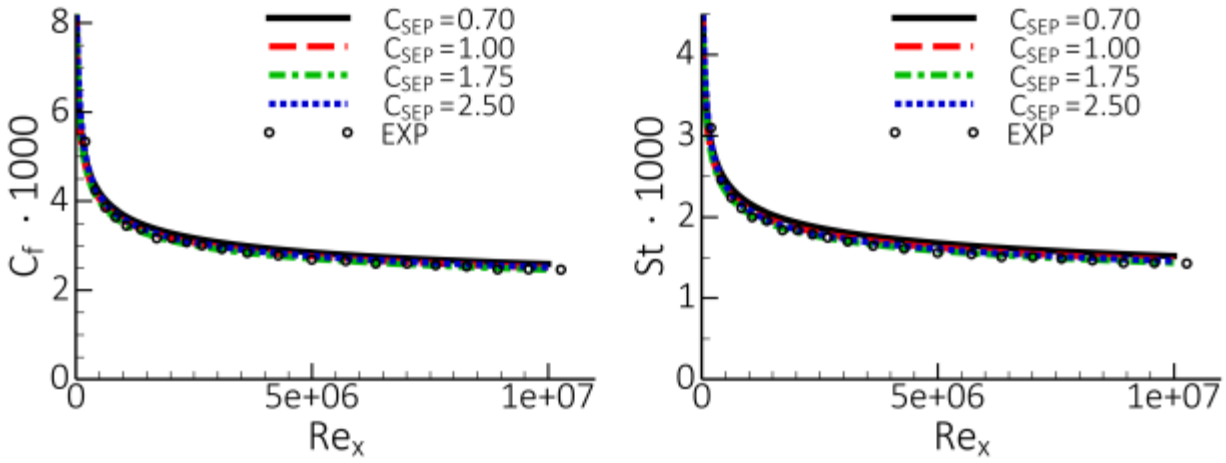


Figure 12.173: Flat plate boundary layer under variation of C_{SEP} (C_{NW}=0.5). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St (p. 409) shows that even large changes in C_{SEP} do not have any effect on the wall shear stress (C_f) and heat transfer coefficients (St). This is the basic design criterion for the GEKO model. It ensures that the user can adjust coefficients like C_{SEP} freely (within range).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

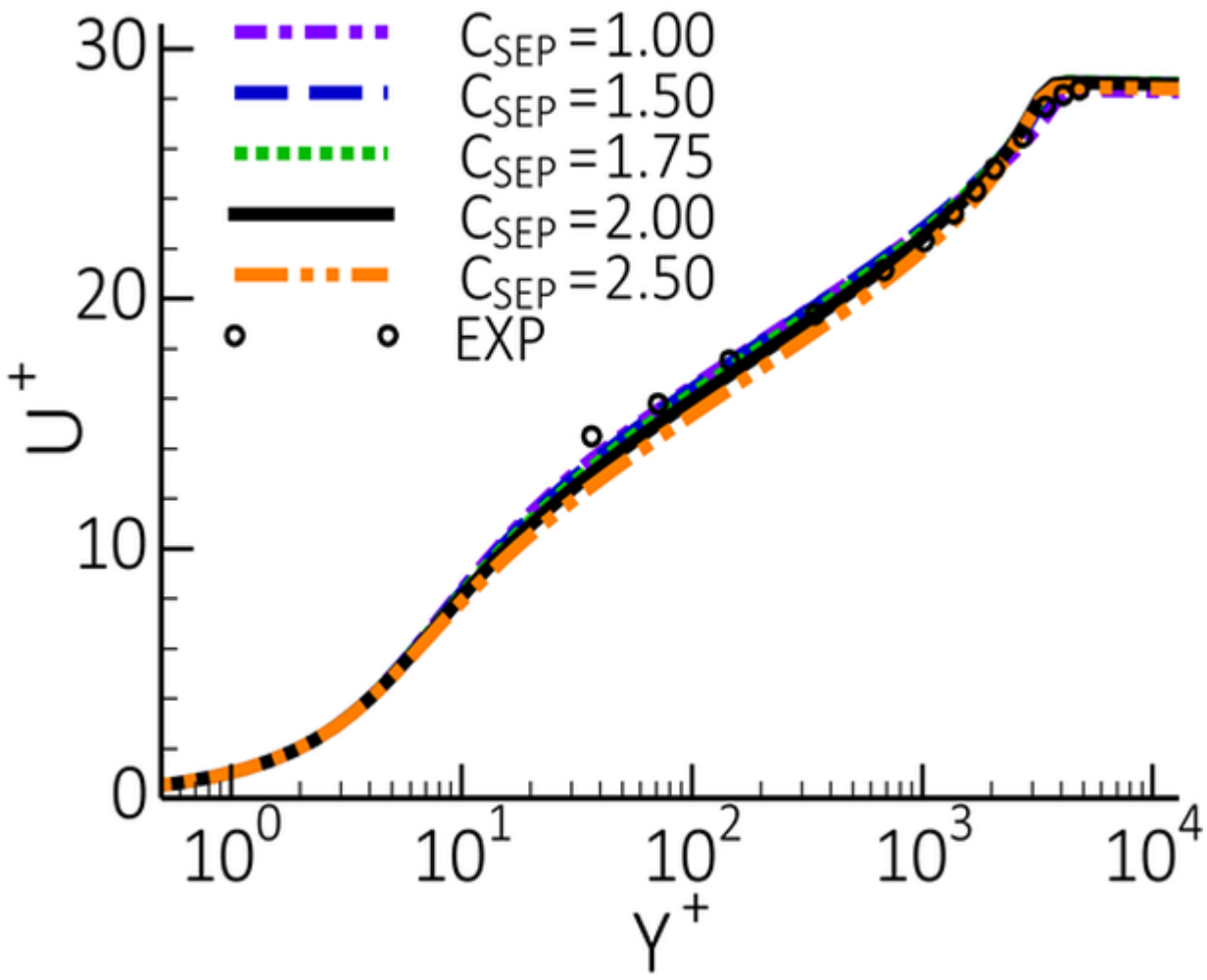
Figure 12.173: Flat plate boundary layer under variation of C_{SEP} (C_{NW}=0.5). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St



The independence of the mean flow velocity is also illustrated by the logarithmic velocity profiles shown in Figure 12.174: Velocity profiles in logarithmic plot under variation of C_{SEP} for flat plate (p. 410). Again, the logarithmic velocity profile is maintained over a wide range of C_{SEP} coefficients.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

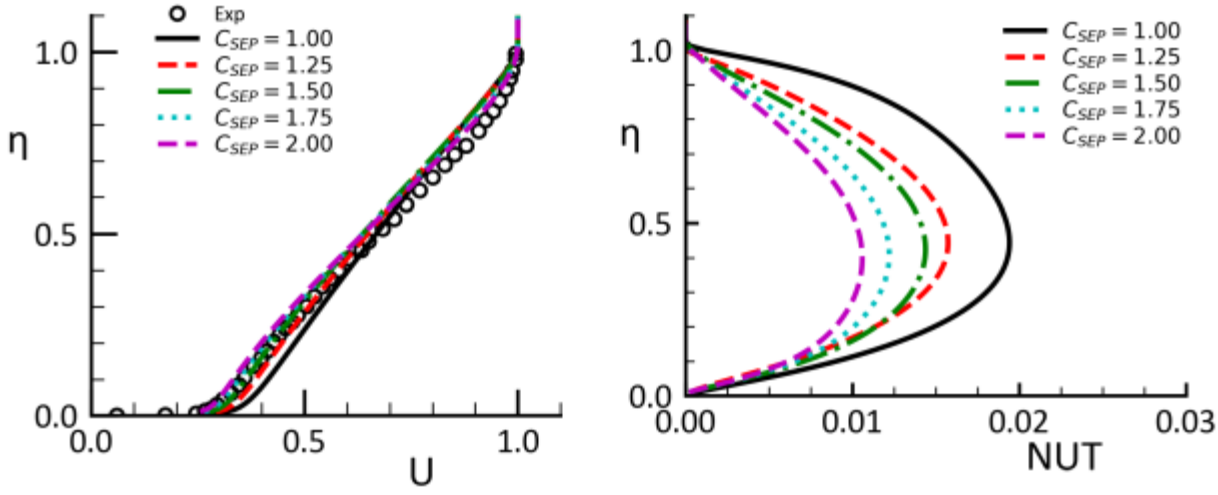
Figure 12.174: Velocity profiles in logarithmic plot under variation of C_{SEP} for flat plate



The effect of C_{SEP} on an equilibrium adverse pressure-gradient boundary layer flows is demonstrated for the experiment of Skare and Krogstad (β_T=20, Re_{δ^*}=1.0×10^5) [21 (p. 444)]. The simulations are based on the equilibrium boundary layer equations as given by Wilcox [31 (p. 445)]. The non-dimensional pressure gradient β_T=20 drives the flow close to separation, a regime, which is very sensitive to turbulence modeling. The left part of Figure 12.175: Impact of variation in C_{SEP} on boundary layer with adverse pressure gradient. Left: Velocity profile. Right Eddy-viscosity profiles (p. 411) shows the influence of C_{SEP} on the velocity profiles whereas the right part shows the impact on the non-dimensional eddy-viscosity, $NUT=\nu_t/\left(U_\delta\delta^*\right)$ (U_δ is the velocity at the edge of the boundary layer and δ^* is the displacement thickness). The velocity profile exhibits moderately but visibly more decelerated with increasing C_{SEP}. The eddy-viscosity, however, is drastically reduced by more than a factor of two by the variation of C_{SEP} between 1-2. The most accurate solution is the one with C_{SEP}=1.75. This reduction in eddy-viscosity is desirable for adverse pressure-gradient boundary layers, as low values increase the sensitivity of the model to adverse pressure gradients and separation. Note that the coefficients C_{MIX}, C_{JET} have little effect on the boundary layer flows, due to the blending function being mostly equal to one inside the boundary layer. The parameter C_{NW} was set to its default value C_{NW}=0.5 (as will be detailed later). However, variations of C_{NW} would have little effect on the current flow.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.175: Impact of variation in C_{SEP} on boundary layer with adverse pressure gradient. Left: Velocity profile. Right Eddy-viscosity profiles



The effect of variation in C_{SEP} is shown in Figure 12.176: Impact of variation in C_{SEP} on CS0 diffuser flow [10]. Left wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p (p. 411) for the axi-symmetric diffuser flow of Driver et al [10 (p. 442)] (C_{NW}=0.5, C_{MIX}=CMixCor, C_{JET}=0.9). As expected from the results for the equilibrium boundary layer, with increasing C_{SEP}, the model becomes more sensitive to the adverse pressure gradient in the diffuser and improves its separation predict up to a value of C_{SEP}~1.75-2.00. Higher values of C_{SEP} lead to over-separation. Note that an optimal calibration of a turbulence model for the CS0 diffuser, does not necessarily guarantee an optimal solution for other similar flows. As will be shown below, for 2D airfoils more 'aggressive' settings are required to match the exp. data. It is therefore desirable that the GEKO model can be pushed to over-separation for the CS0 case.

Figure 12.176: Impact of variation in C_{SEP} on CS0 diffuser flow [10 (p. 442)]. Left wall shear stress coefficient, C_f. Right: Wall pressure coefficient C_p

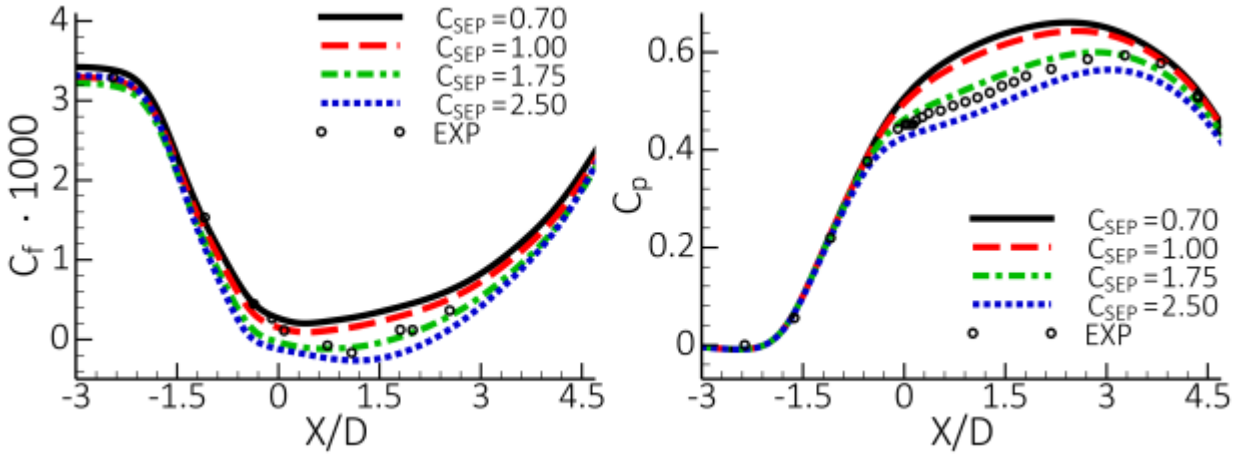
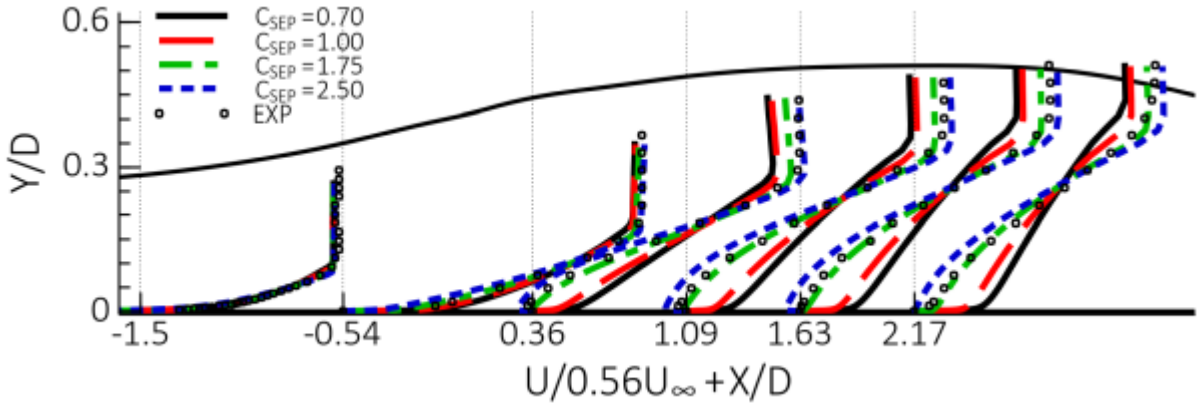


Figure 12.177: Impact of variation in C_{SEP} on velocity profiles for CS0 diffuser flow [10 (p. 442)]



A more realistic application for tuning the C_{SEP} coefficient is the simulation of 2D airfoil profiles under variation of the angle of attack, α. Figure 12.178: Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 (Left) and DU-97-W-300 (Right) airfoil at $\mathrm{Re}=3\cdot10^{6}$ [28] (p. 413) to Figure 12.182: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=18°$ (Right) with GEKO model for S814 airfoil at $\mathrm{Re}=1.5\cdot10^{6}$ [26] (p. 415) compare lift curves for numerous airfoils computed with the GEKO model. In these comparisons, it should be kept in mind that 2D simulations are not entirely correct in the stall and post-stall region (around and past maximum lift, C_{Lmax}), due to the formation of 3D structures in the experiments. Despite of this, it is possible to adjust the GEKO model for the prediction of such flows for angles of attack in 2D simulations. Increasing C_{SEP} predicts earlier separation onset, which improves agreement of the predicted pressure coefficient and results in an improved agreement of the lift coefficient with the experimental data[1] near stall. The optimal value for C_{SEP} for such flows is therefore somewhere between C_{SEP}=2.00-2.50.

The current test cases demonstrate the advantage of the GEKO model over e.g. the SST model. The SST model is tuned to match adverse pressure gradients flows and flows with separation well on average. However, for the 2D airfoils the SST model is clearly too conservative resulting in overly optimistic CLmax levels. This would be hard to correct within the SST model and in any case would require expert knowledge in turbulence modeling. Within the GEKO model, separation prediction can easily be adjusted by changing C_{SEP} even by a non-expert.

[1] There are no experimental data for the pressure coefficients for the DU-96-W-180 and DU-97-W-300 airfoils. Only distribution of lift coefficient in wide range of angles of attack are available.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.178: Prediction of lift and pressure coefficients with GEKO model for DU-96-W-180 (Left) and DU-97-W-300 (Right) airfoil at $\mathrm{Re}=3\cdot10^6$ **[28 (p. 445)]**

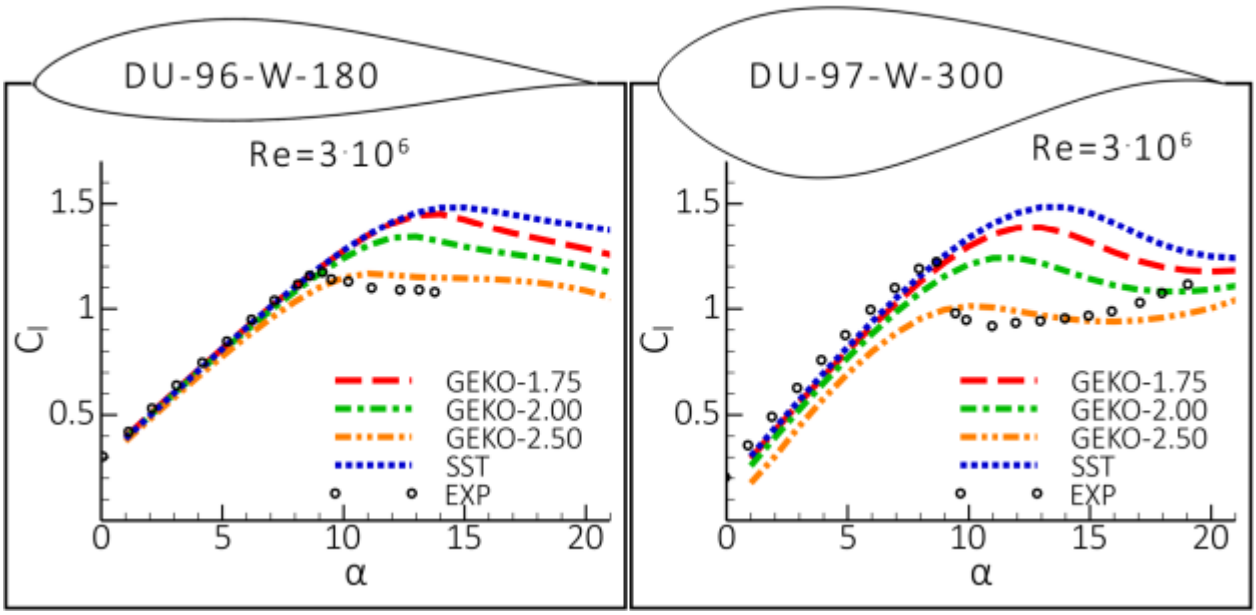
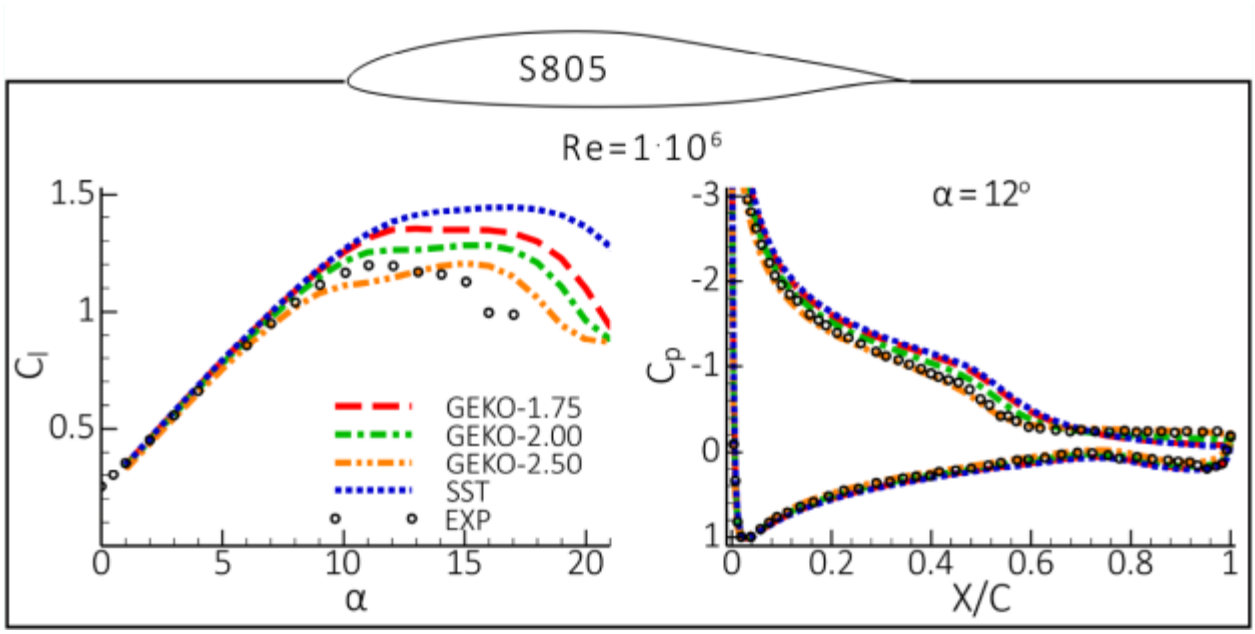


Figure 12.179: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=12°$ **(Right) with GEKO model for S805 airfoil at** $\mathrm{Re}=1\cdot10^6$ **[23 (p. 444)]**



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.180: Prediction of lift coefficient wide range of angle of attacks (Left) and pressure coefficient for $\alpha=17°$ (Right) with GEKO model for S825 airfoil at $\mathrm{Re}=2\cdot10^6$ [24 (p. 444)]

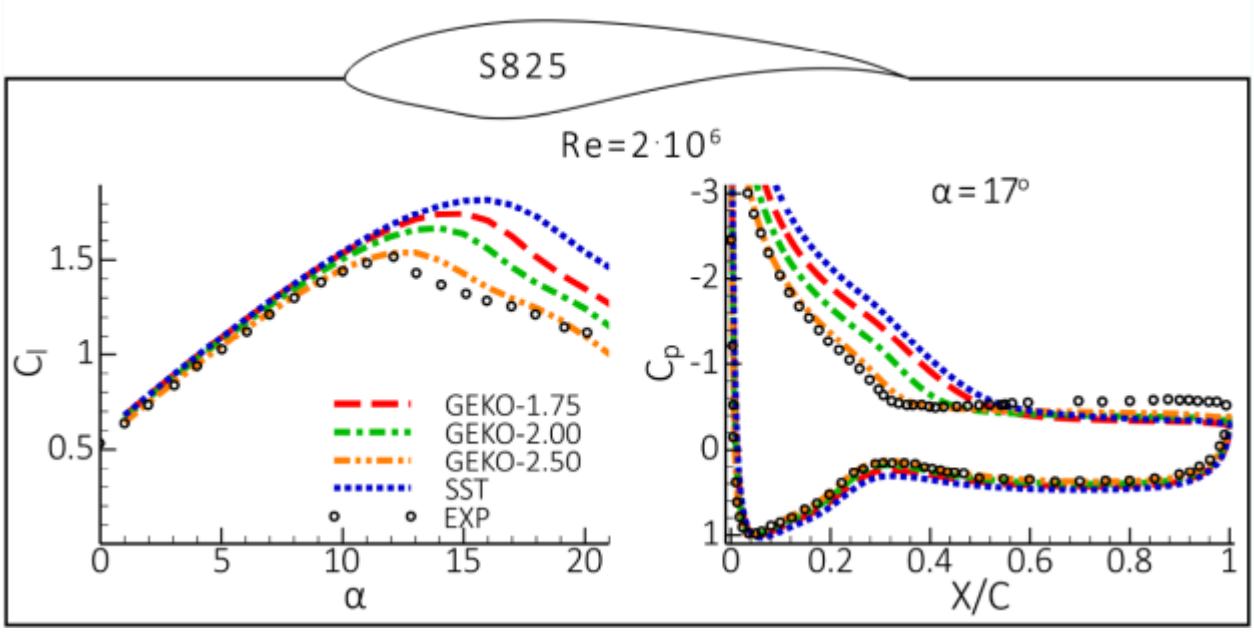
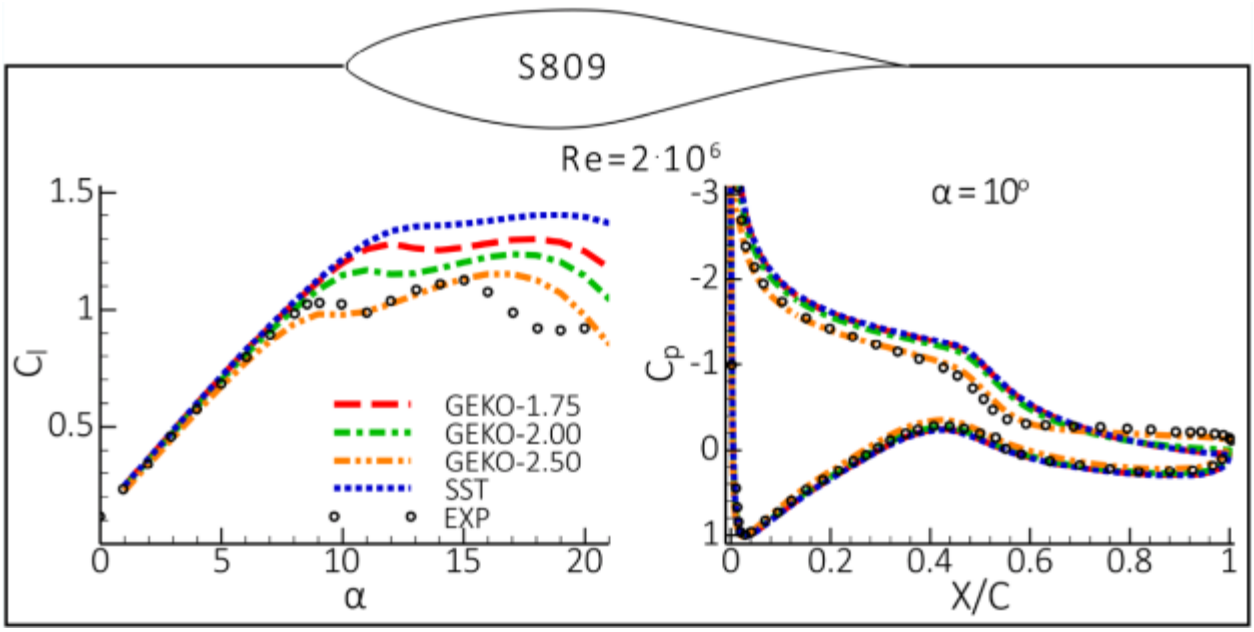
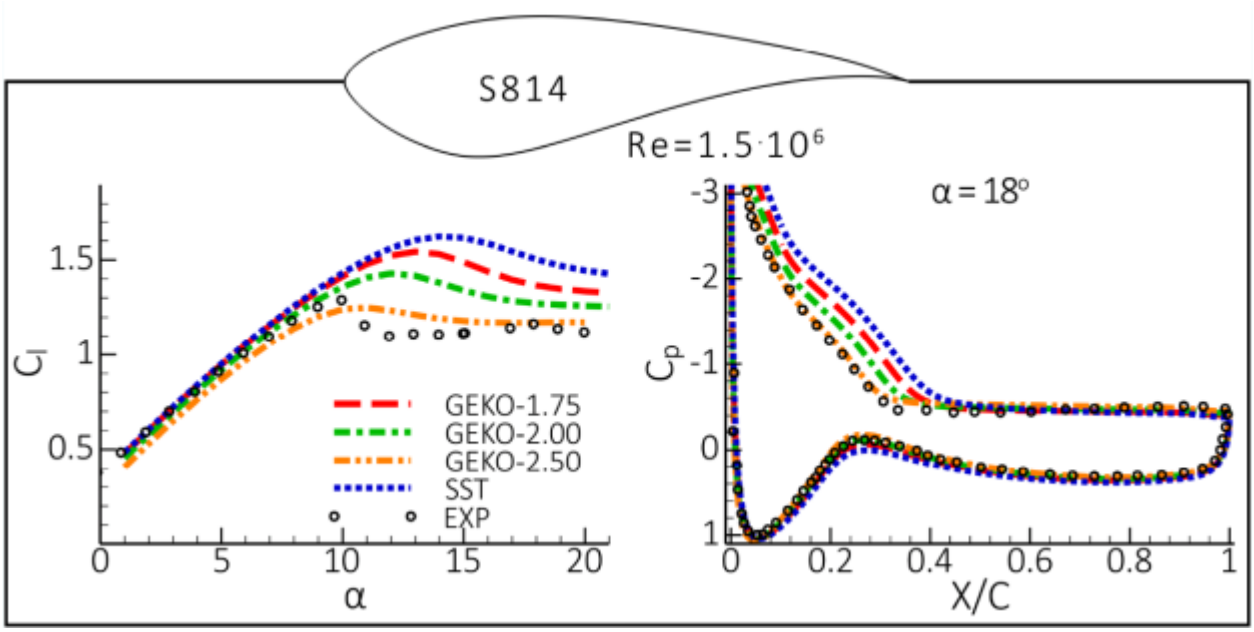


Figure 12.181: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for $\alpha=10°$ (Right) with GEKO model for S809 airfoil at $\mathrm{Re}=2\cdot10^6$ [25 (p. 444)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.182: Prediction of lift coefficient in wide range of angle of attacks (Left) and pressure coefficient for α=18° (Right) with GEKO model for S814 airfoil at $\mathrm{Re}=1.5\cdot10^6$ [26 (p. 445)]



It is important to understand that changes of C_{SEP} affect not only boundary layers, but the entire flow field. Increasing C_{SEP} reduces the eddy-viscosity in all parts of the domain, including for free shear flows (e.g. mixing layers etc.). Frequently, the adjustment of boundary layer separation is the main optimization requirement and it is desirable to maintain the performance and calibration for free shear flows (e.g. spreading rates for mixing layers and jets etc.). In order to avoid any negative impact of C_{SEP} on free shear flows, the reduction in eddy-viscosity is corrected by modifying C_{MIX} accordingly. This is achieved through the correlation C_{MixCor} given in Equation (2.11). This correlation increases C_{MIX} with increasing C_{SEP} to maintain the spreading rates for mixing layers. As the correlation C_{MIX} =CMixCor is default, users do not have to adjust C_{MIX} when changing C_{SEP}. This effect will be demonstrated in the Section 3.3 dealing with the impact of C_{MIX}.

The Bachalo-Johnson NASA Bump flow experiment [2 (p. 441)], as depicted in Figure 12.183: Schematic of transonic axi-symmetric bump flow (p. 416), features a subsonic inflow with Ma=0.875. The flow is then accelerated to supersonic speed over the bump and then reverts to subsonic speed through a shock wave. The shock causes the boundary layer behind the shock to separate, which in turn interacts with the shock by pushing it forward. The ability to predict the shock location is therefore directly linked to a model's ability to predict boundary layer separation. As expected, again, the models group as GEKO-1.75/SST and GEKO-1.00/RKE as seen from Figure 12.184: Pressure coefficient,C_p, along wall of bump. Comparison of different GEKO settings with RKE and SST model and experimental data [2]. (p. 416). Both, the GEKO-1.75 and the SST model can predict the shock location and the post-shock separation zone properly. The GEKO-1.00/RKE models fail due to their lack of separation sensitivity.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.183: Schematic of transonic axi-symmetric bump flow

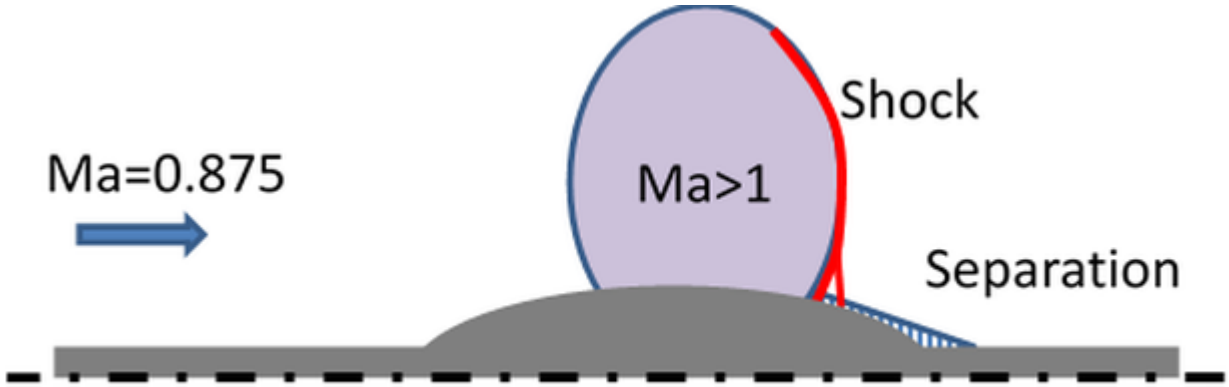
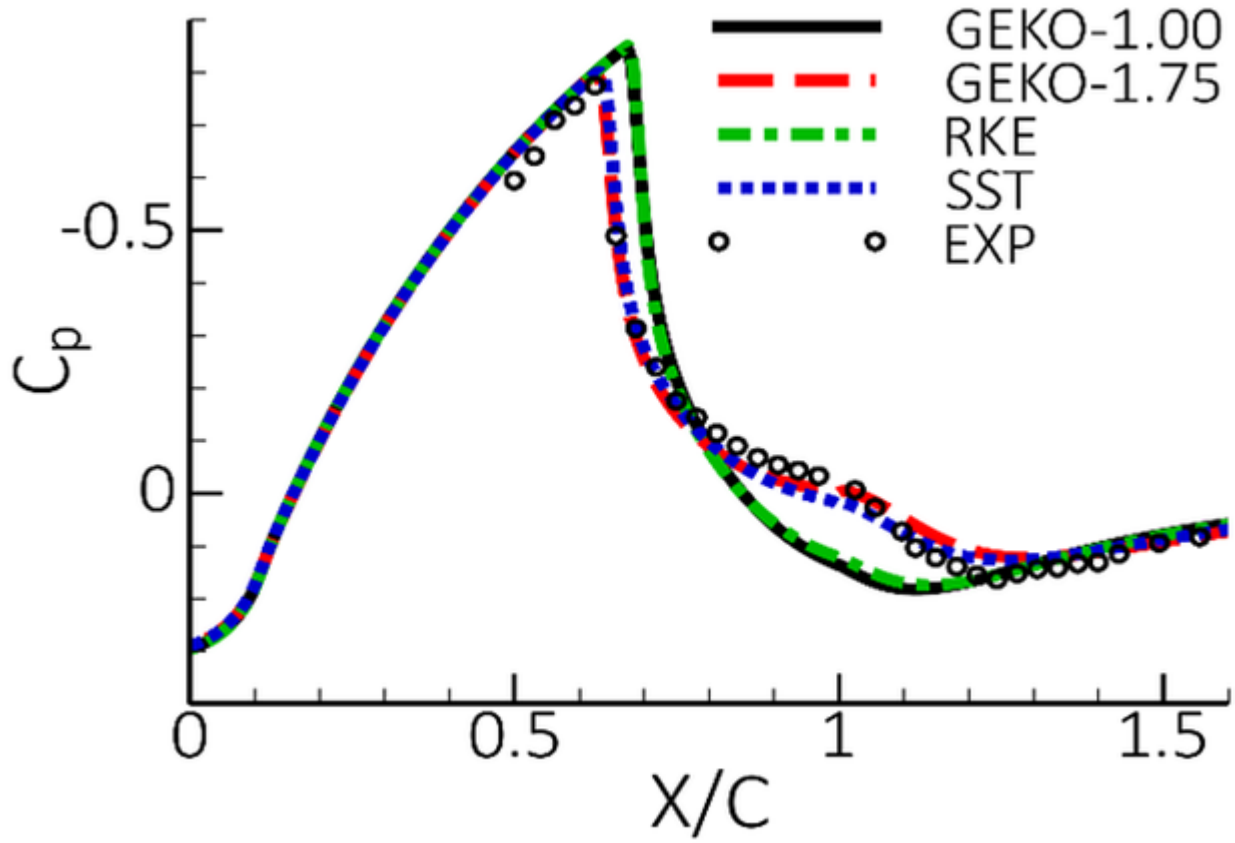


Figure 12.184: Pressure coefficient, C_p, along wall of bump. Comparison of different GEKO settings with RKE and SST model and experimental data [2 (p. 441)].



12.3.2.2. 3.2 The 'Near Wall' Parameter C_{NW}

The coefficient C_{NW} is introduced to allow a modification of the model characteristics in the near wall region under non-equilibrium conditions. It has a strong effect on heat transfer predictions in reattachment and stagnation regions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The first task is to show that variations in C_{NW} (like C_{SEP}) do not affect flat plate boundary layer behavior. This can be seen from Figure 12.185: Flat plate boundary layer under variation of C_{NW}(C_{SEP}=1.75). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St. (p. 417) where both, the wall shear stress coefficient,C_f, and the wall heat transfer coefficient St are unaffected by variations in C_{NW}. In addition, the velocity profile in log-scale is maintained as shown in Figure 12.186: Velocity profiles in logarithmic coordinates - variation of C_{NW}(C_{SEP}=1.75) (p. 418). Many more variations of parameters C_{SEP} and C_{NW} have been tested and the agreement is like that shown in Figure 12.186: Velocity profiles in logarithmic coordinates - variation of C_{NW}(C_{SEP}=1.75) (p. 418).

Figure 12.185: Flat plate boundary layer under variation of C_{NW}(C_{SEP}=1.75). Left: Wall-shear stress coefficient, C_f, Right: Wall heat transfer coefficient, St.

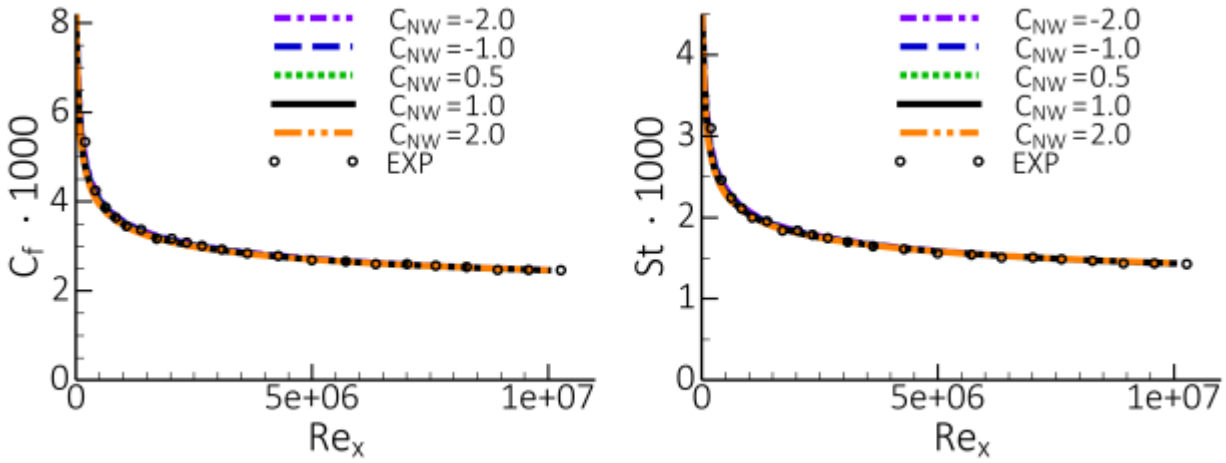
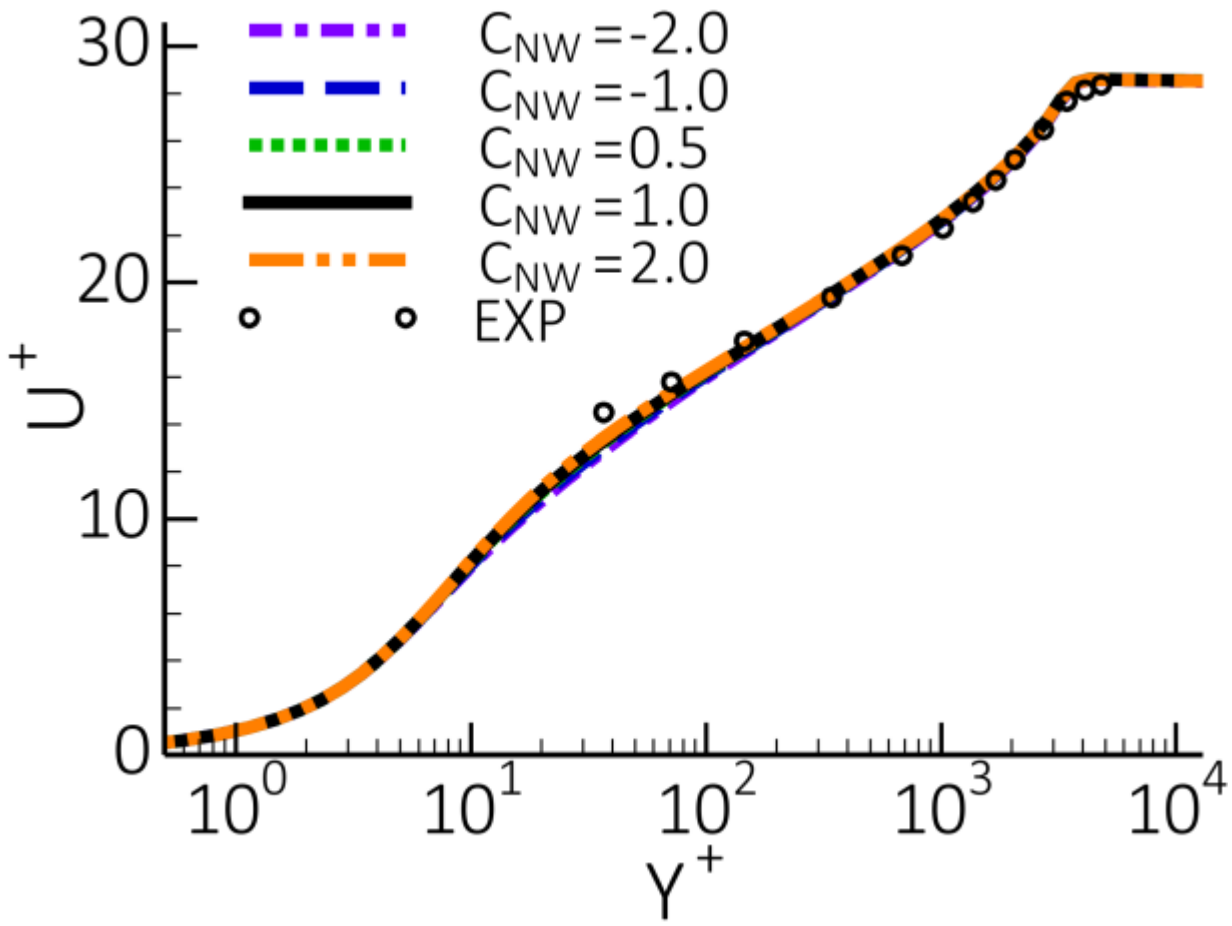


Figure 12.186: Velocity profiles in logarithmic coordinates - variation of C_{NW}(C_{SEP}=1.75)



The effect of changing C_{NW} for the axisymmetric diffuser test case CS0 [10 (p. 442)] is shown in Figure 12.187: Impact of variation in C_{NW} on CS0 diffuser flow [10]. Left: Wall shear-stress coefficient, C_f. Right: Wall pressure coefficient C_p (C_{SEP}=1.0, C_{MIX}=0.0, C_{JET}=0.9) (p. 419). The coefficient C_{NW} is designed to have an influence mostly near the wall. Therefore, the wall shear-stress distribution shows a much larger sensitivity to this variation (Figure 12.187: Impact of variation in C_{NW} on CS0 diffuser flow [10]. Left: Wall shear-stress coefficient, C_f. Right: Wall pressure coefficient C_p (C_{SEP}=1.0, C_{MIX}=0.0, C_{JET}=0.9) (p. 419) Left) than the Cp-distribution (Figure 12.187: Impact of variation in C_{NW} on CS0 diffuser flow [10]. Left: Wall shear-stress coefficient, C_f. Right: Wall pressure coefficient C_p (C_{SEP}=1.0, C_{MIX}=0.0, C_{JET}=0.9) (p. 419) Right). The velocity profiles shown in Figure 12.188: Impact of variation in C_{NW} on velocity profiles for CS0 diffuser flow [10] (p. 419) illustrate the effect more clearly. Contrary to changes of C_{SEP}, where the entire boundary layer is affected, variations in C_{NW} change only the inner part of the velocity profiles.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.187: Impact of variation in C_{NW} on CS0 diffuser flow [10 (p. 442)]. Left: Wall shear-stress coefficient, C_f. Right: Wall pressure coefficient C_p (C_{SEP}=1.0, C_{MIX}=0.0, C_{JET}=0.9)

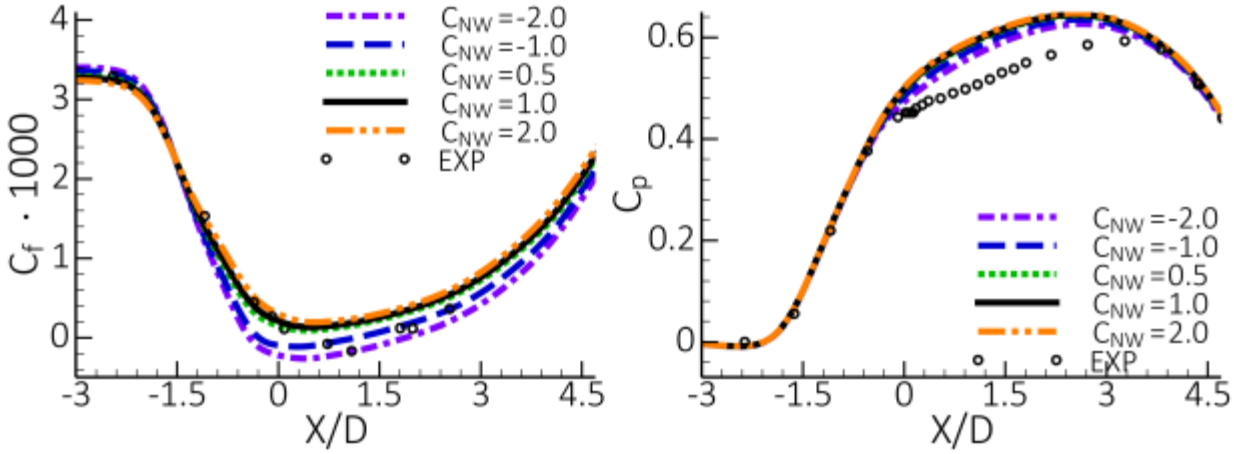
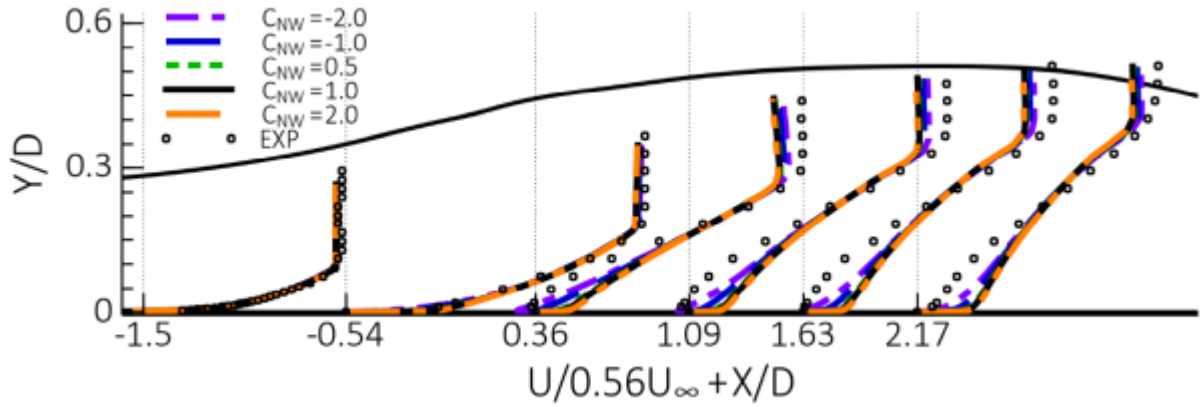


Figure 12.188: Impact of variation in C_{NW} on velocity profiles for CS0 diffuser flow [10 (p. 442)]

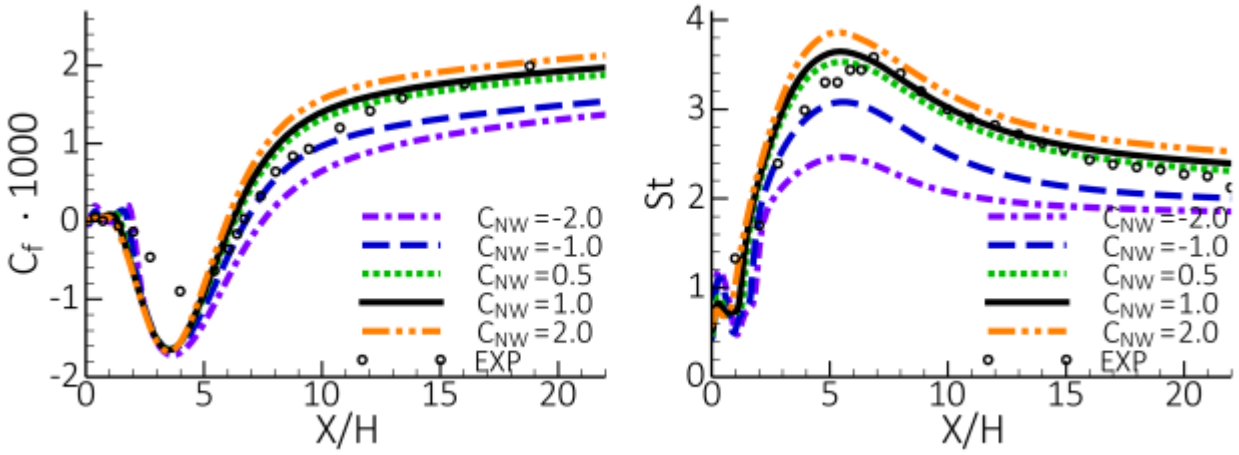


For non-equilibrium flows, the wall shear stress and more importantly, the heat transfer to a wall, depend mostly on the model details close to the wall. The coefficient C_{NW} can be tuned to allow fine-tuning for such flows. An example is backward-facing step with Cf and heat transfer measurements, St, downstream of the step [31 (p. 445)]. The effect is shown in Figure 12.189: Backward-facing step with heat transfer under variation of C_{NW} (Top:C_{SEP}=1.00, Bottom: C_{SEP}=1.75, Both: C_{JET}=0.9, C_{MIX}=C_{MixCor}). Left: Wall shear stress coefficient, C_f, Right: Heat transfer Stanton number (p. 420).
It is computed with both, C_{SEP}=1.0 and C_{SEP}=1.75, C_{MIX}=CMixCor , C_{JET}=0.9 and a variation in C_{NW}. The heat transfer is strongly affected (and can therefore be optimized) by changes in C_{NW}. In the current application, the optimal value is C_{NW}=0.5 for both values of C_{SEP}. C_{NW}=0.5 is also the default setting. It is important to note that C_{NW} is a minor parameter and will likely not have to be tuned for most flows. Wider validation studies indicate that the default value is suitable for most applications.

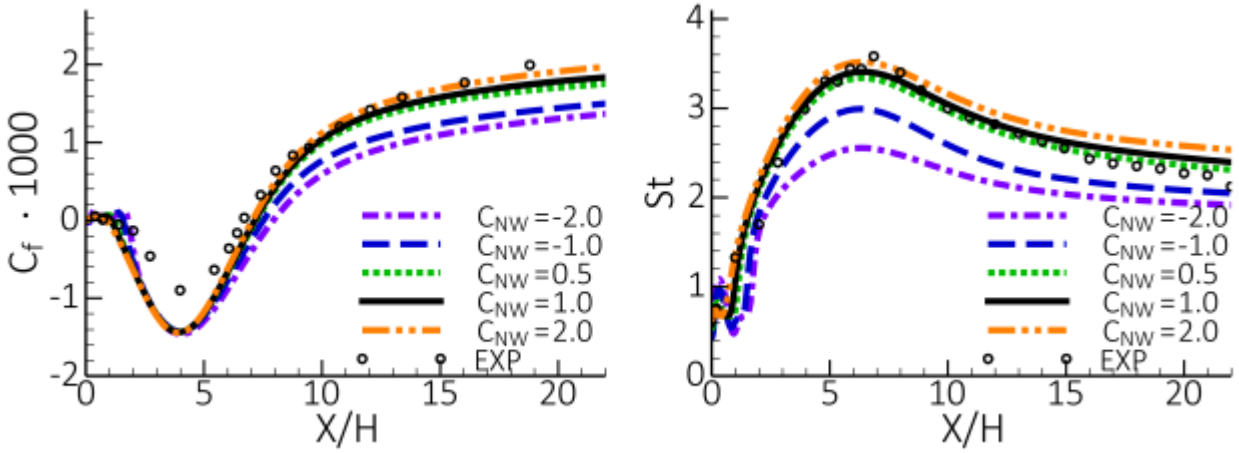
C_{SEP}=1

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.189: Backward-facing step with heat transfer under variation of C_{NW} (Top:C_{SEP}=1.00, Bottom: C_{SEP}=1.75, Both: C_{JET}=0.9, C_{MIX}=C_{MixCor}). Left: Wall shear stress coefficient, C_f, Right: Heat transfer Stanton number



C_{SEP}=1.75



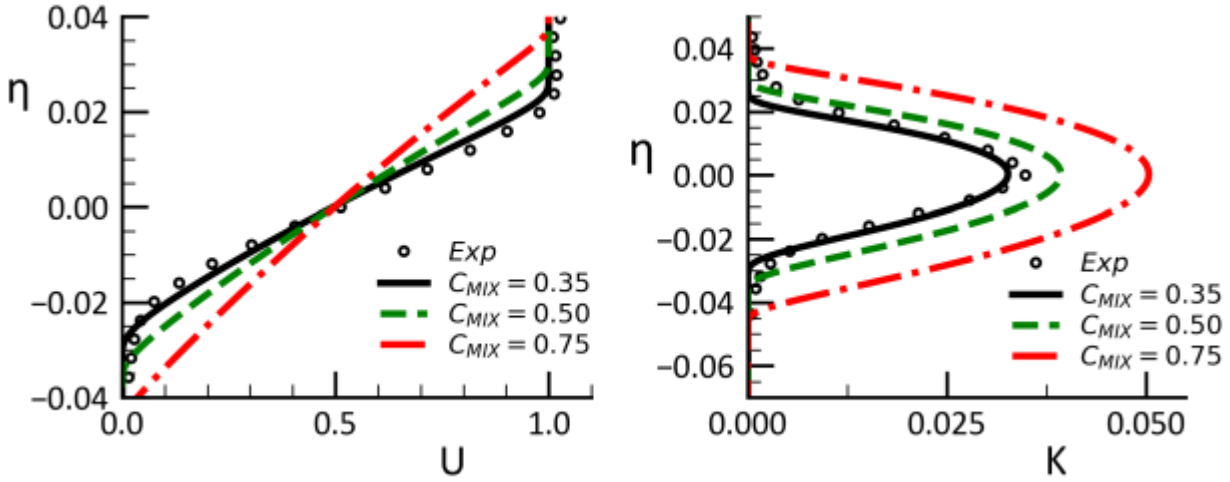
12.3.2.3. The 'Mixing' Parameter C_{MIX}

The parameter C_{MIX} affects only free shear flows. It has no impact on boundary layers due to the blending function FBlend. Increasing C_{MIX} leads to larger eddy-viscosity (higher turbulence levels) in free shear flows. This is illustrated in Figure 12.190: Impact of variation in C_{MIX} on mixing layer when using the C_{SEP}=2(C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles (p. 421) which shows results for a mixing layer simulation compared with experimental data [3 (p. 441)]. As expected, increasing C_{MIX} leads to larger spreading rates of the velocity profile, associated with higher levels of turbulence kinetic energy. In most applications, it is desirable to calibrate the coefficient such that a good agreement with mixing layer flows is achieved (in the current example with C_{SEP}=2 this means C_{MIX}=0.35). However, there can be cases, where stronger mixing is desired and where the calibration for a classical mixing layer is not sufficient. Examples are flows with strong mixing characteristics, like flows past bluff bodies etc. It should be emphasized that the physical reason for increased mixing in such cases is often a result of flow unsteadiness (e.g. vortex shedding) augmenting the conventional turbulence mixing. In such cases it will not be

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

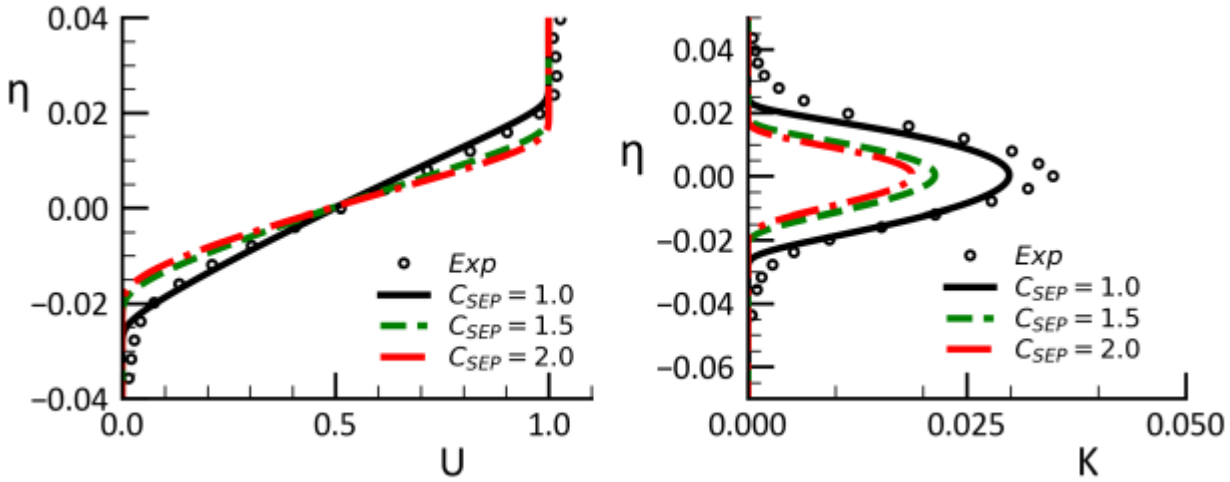
possible to obtain a perfect agreement against data, as such unsteady effects are typically stronger than that which a turbulence model can provide. Still, increasing C_{MIX} can at least compensate for some of the missing effects, in case that unsteady (scale resolving) simulations are not feasible.

Figure 12.190: Impact of variation in C_{MIX} on mixing layer when using the C_{SEP}=2(C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles



As already mentioned in the section on C_{SEP} (section 3.1), there is a subtle interaction of C_{MIX} and C_{SEP} which users have to understand. Variations in C_{SEP} also affect free shear flows and increases in C_{SEP} result in lower spreading rates. This can be seen in Figure 12.191: Impact of variation in C_{SEP} on mixing layer (C_{MIX}=0, C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles (p. 421) which shows the results of a C_{SEP} variation for the mixing layer [3 (p. 441)] (C_{MIX} = 0, C_{JET} = 0.9, C_{NW} = 0.5). Here the solution with C_{SEP} = 1 is closest to the data whereas the solution with C_{SEP}=2 predicts far too low spreading rates and turbulence levels. The effect on the eddy-viscosity is even stronger than for the boundary layer resulting in a reduction of more than a factor 3 with increasing C_{SEP} (not shown).

Figure 12.191: Impact of variation in C_{SEP} on mixing layer (C_{MIX}=0, C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles

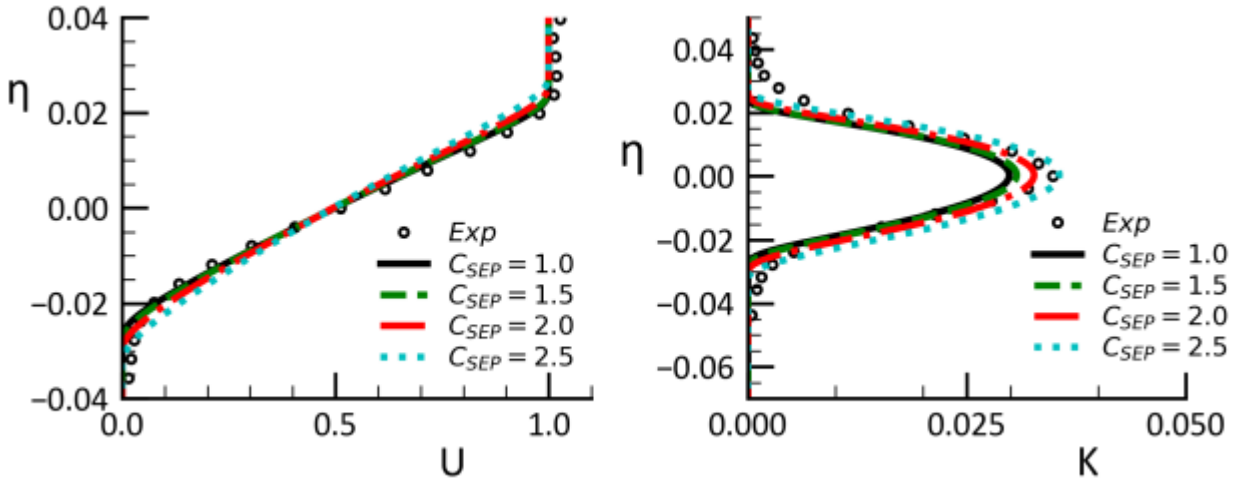


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In order to compensate the effect of C_{SEP} on free shear flows, the coefficient C_{MIX} has to be increased with increasing C_{SEP}. This is achieved by the correlation CMixCor given in Equation (2.11) which is the model default. The correlation is provided so that users know which value is optimal for a given C_{SEP} setting. Increasing C_{MIX} above this value will lead to higher and decreasing it to lower turbulence levels and spreading rates for free shear flows.

The results for mixing layer simulations with C_{MIX}= CMixCor are shown in Figure 12.192: Impact of variation in C_{SEP} on mixing layer when using the C_{MIX}=C_{MixCor} correlation (C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles (p. 422). For all selected values of C_{SEP}, the mixing layer is maintained, and correct spreading rates are achieved.

Figure 12.192: Impact of variation in C_{SEP} on mixing layer when using the C_{MIX}=C_{MixCor} correlation (C_{JET}=0.9, C_{NW}=0.5). Left: Velocity profile. Right: Turbulence kinetic energy profiles



12.3.2.4. The 'Jet' Parameter C_{JET}

The impact of the C_{JET} parameter is subtle and in most applications it can be maintained at its default value. It is important in cases where jet flows need to be computed with high accuracy. Remember that conventional models like k–ε or SST will over-predict spreading rates of round jets substantially while giving reasonable results for plane jets. For this so-called round-jet plane-jet 'anomaly' see e.g. [19 (p. 444),33 (p. 446)].

The GEKO model is designed to provide parameter settings which allow an accurate representation of jet flow. This is achieved through the parameter C_{JET}. The way this is achieved is by reducing the influence of C_{MIX} (which increases mixing layer spreading rates) on jet flows (especially round jets). It should also be noted that the coefficient C_{JET} is a sub-function of the coefficient C_{MIX}. In case C_{MIX}=0, the coefficient C_{JET} has no impact.

Simulations are again based on self-similar jet-flow equations as given by Wilcox [33 (p. 446)]. The two test cases are Wygnanski and Fielder [36 (p. 446)] for the plane jet and Bradbury [4 (p. 441)] for the round jet. Figure 12.193: Effect of C_{JET} for plane (left) and round (right) jet flow (C_{SEP}=2.0, C_{MIX}=0.35, C_{NW}=0.5) (p. 423) shows simulations with C_{SEP}=2, C_{MIX}=0.35 (Correlation) and C_{NW}=0.5 (default). It is clearly seen that the model overpredicts the spreading rates especially for the round jet with C_{JET}=0. The effect of C_{JET} is to reduce the spreading rate of both jet flows, with C_{JET}=0.9

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

being close to both experimental data-sets. With C_{SEP}=2 and C_{JET}=0.9, the model avoids the round jet/plane jet anomaly and predicts lower spreading rates for the round than for the plane jet. Note again, that the changes of C_{JET} discussed here do not affect the mixing layer.

In contrast, Figure 12.194: Comparison of GEKO-1 and GEKO-2 model (with variation of C_{JET}=0.9) (p. 423) shows a comparison between GEKO-1 (C_{SEP}=1.0) and GEKO-2 (C_{SEP}=1.0) with C_{JET}=0.9. As GEKO-1 is a close cousin of the standard k–ε model it behaves just like that model. It gives a correct spreading rate for the plane jet but over-predicts the round jet. Note again, that changes to C_{JET} in GEKO-1 would have no effect, as for that model C_{MIX}=0 (so that the sub-model C_{JET} is de-activated).

Figure 12.193: Effect of C_{JET} for plane (left) and round (right) jet flow (C_{SEP}=2.0, C_{MIX}=0.35, C_{NW}=0.5)

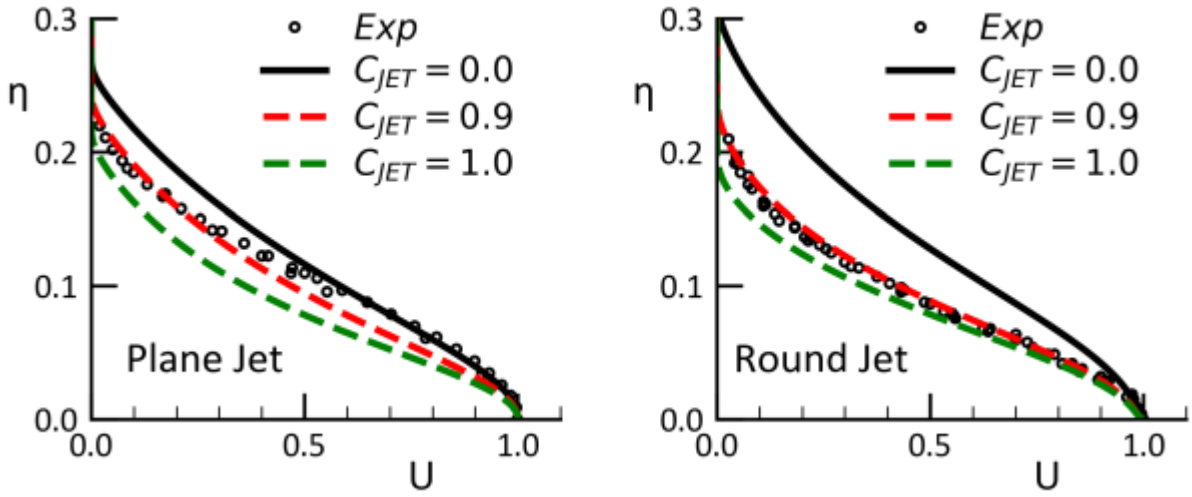
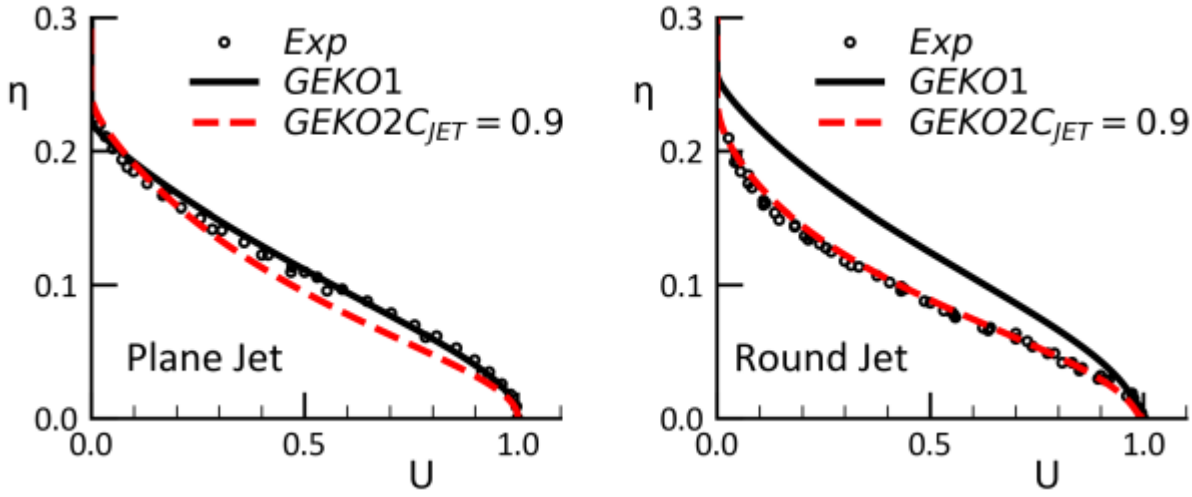


Figure 12.194: Comparison of GEKO-1 and GEKO-2 model (with variation of C_{JET}=0.9)



In summary, in order to achieve optimal performance for round jets, one needs to set C_{SEP} to values C_{SEP}~1.75-2.00 and C_{JET}=0.9 (default). With reduction in C_{SEP} and the corresponding reduction in C_{MIX}, the effect of C_{JET} vanishes.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

There is another parameter CJET_AUX which also has an influence on the jet flow. It defines the limit between mixing layers and jet flows. The larger the value, the sharper the 'demarcation' and stronger the effect of C_{JET}. The default value is CJET_AUX =2.0. It is suitable for jet flow simulations to set the value to CJET_AUX =4.0. This is not done by default, as it can lead to oscillations on poor meshes.

Users who do not have a need for accurate predictions of jets can use the default settings for C_{JET}.

12.3.2.5. The 'Corner' Parameter CCORNER

It is well known that for rectangular turbulent channel flows, secondary flows develop in the plane normal to the mean flow. Such secondary flow is not present in laminar flows and can also not be represented at all by eddy-viscosity models. The situation is depicted in Figure 12.195: Comparison of streamline velocity contours and secondary motion predicted by linear (Left) and non-linear (Center) GEKO-1.00 model with the DNS (Right) data in the fully developed square duct flow [19] (p. 424) for a square channel. The main flow direction is normal to the plotting plane. The velocity contour colors on the left picture show the main flow as computed by a linear eddy-viscosity model. There are no streamlines when projected into this plane.

The practical effect of this effect is that the secondary flow transports flow (and thereby momentum) into the corner. If the corner flow experiences an adverse pressure gradient, this additional momentum can help the flow to avoid/delay separation. When computing such flows with an eddy-viscosity model, such separations can occur earlier than in the experiments, which in turn can have a substantial effect on the overall performance. The right picture shows the results from a Direct Numerical Simulation (DNS, e.g. [19 (p. 444)]) of this flow. There are clearly recognizable secondary streamlines pushing flow into the corners. The center picture shows the GEKO model in combination with an existing non-linear algebraic stress-strain model (Wallin-Johansson Explicit Reynolds Stress Models WJ-EARSM), which can mimic the effect. The WJ-EARSM is fairly complex and the GEKO model was used as a reference point for calibration of the much simpler quadratic stress-strain relationship which can also account for this effect (Equation (2.7)). This term has a free parameter, CCORNER, which can be tuned by the users.

Figure 12.195: Comparison of streamline velocity contours and secondary motion predicted by linear (Left) and non-linear (Center) GEKO-1.00 model with the DNS (Right) data in the fully developed square duct flow [19 (p. 444)]

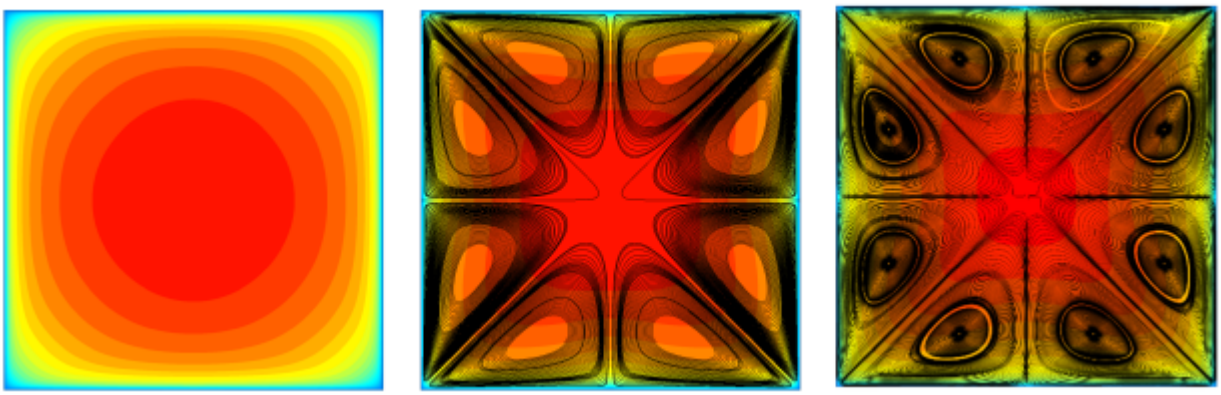


Figure 12.196: Comparison of turbulence models for flow in square channel. Left: GEKO with WJ-EARSM. Middle, GEKO with CCORER=0.9. Right GEKO linear (CCORNER=0.0). (p. 425) shows a comparison for the rectangular channel for GEKO-175 (C_{SEP}=1.75 all else default). The left picture shows

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

the combination of GEKO with the WJ-EARSM, the middle picture the combination with the quadratic term and CCORNER=0.9 and the left for CCORNER=0.0. The simple quadratic model gives a good approximation of the flow, similar to the WJ-EARSM.

Figure 12.196: Comparison of turbulence models for flow in square channel. Left: GEKO with WJ-EARSM. Middle, GEKO with CCORER=0.9. Right GEKO linear (CCORNER=0.0).

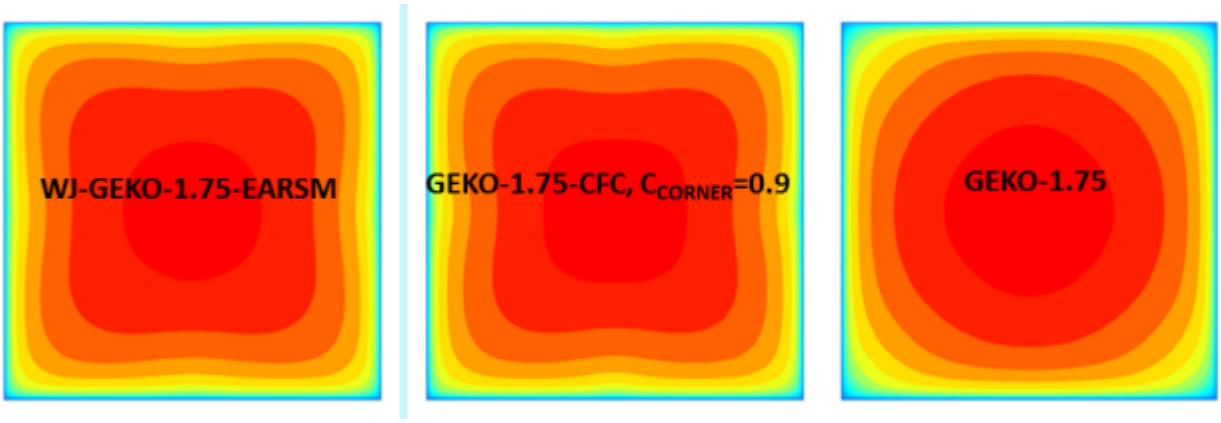
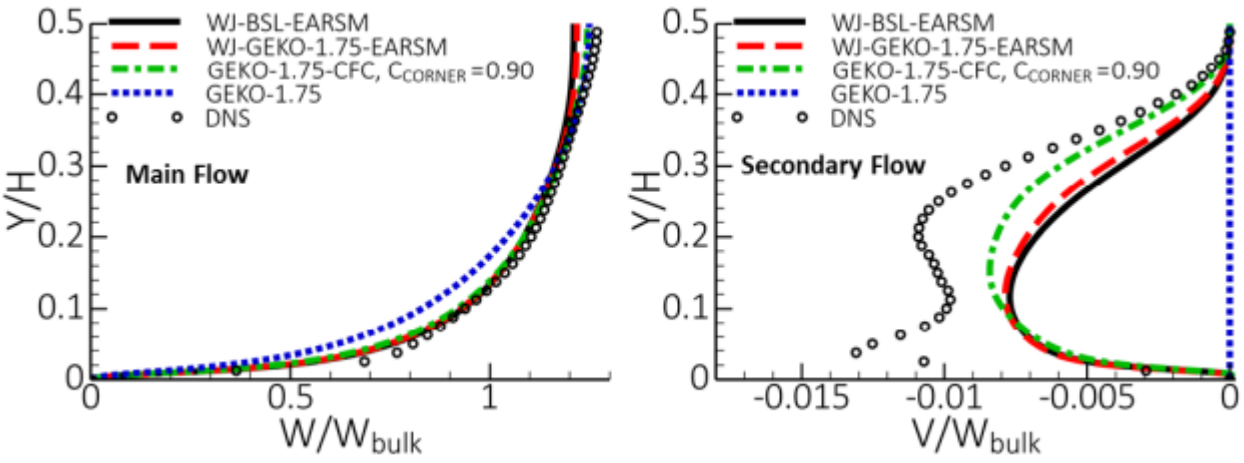


Figure 12.197: Velocity profiles plotted along diagonal of channel for different turbulence models against DNS data. Right: Main flow, Left Secondary Flow (p. 425) shows a comparison of velocity profiles for different turbulence models for the square channel. The right part of the picture shows the secondary flow into the corner along a diagonal of the channel cross-section. Most obviously, the linear model (GEKO-1.75) gives zero velocity along the diagonal. The other models provide fairly similar strength and distributions of the same order as the reference DNS data, but clearly also not in perfect agreement, especially very close to the wall. Note that further increases in C_{CORNER} for this flow would break the symmetry of the crossflow pattern. The left part of the figure shows the mean flow profile also along the diagonal. The most important effect is the higher velocity close to the wall ('fuller profile') which allows the flow to overcome a stronger pressure gradient before separating.

Figure 12.197: Velocity profiles plotted along diagonal of channel for different turbulence models against DNS data. Right: Main flow, Left Secondary Flow



A more challenging example is the flow in a 3D diffuser [5 (p. 441)] shown in Figure 12.198: Geometry of 3D diffuser flow [4] (p. 426). The diffuser has two opening angles, a large one on the upper wall and a small one at the side walls. In the experiments, the flow separates from one of the corners

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

and then attaches to the lower wall. In the simulations, the flow topology depends strongly on the model and the corner flow representation.

Figure 12.198: Geometry of 3D diffuser flow [4 (p. 441)]

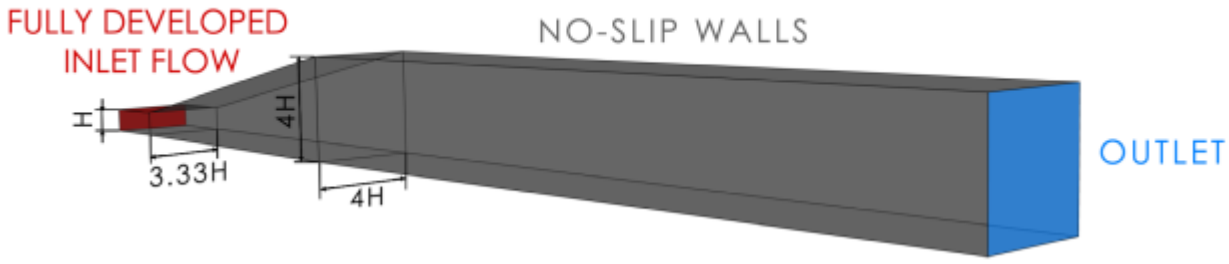
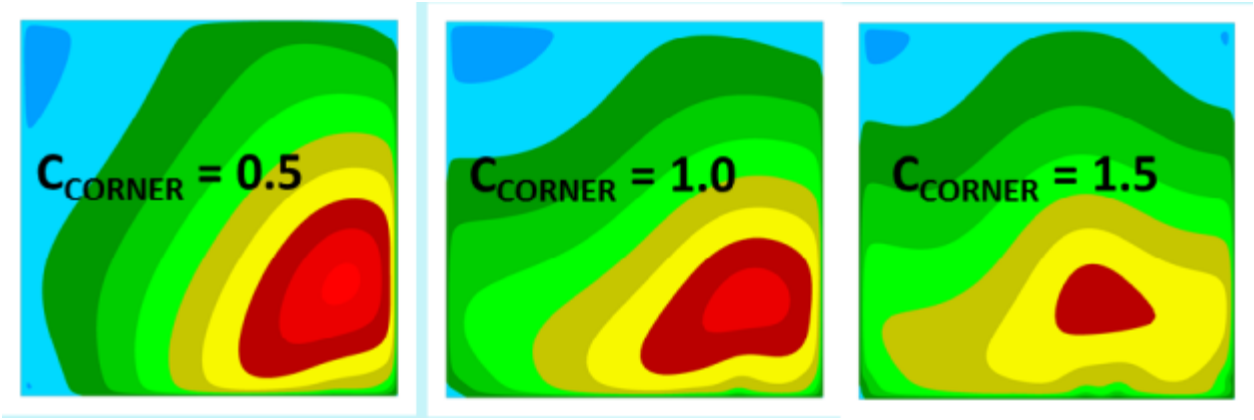


Figure 12.199: Flow topology for 3D diffuser flow shown through streamwise velocity contour at downstream end of diffuser opening section for GEKO-1.00 with different C_{CORNER} values. (p. 426) shows the mean velocity contours at the end of the expansion section of the diffuser. Clearly the flow topology depends strongly on the selected values of the corner flow correction term.

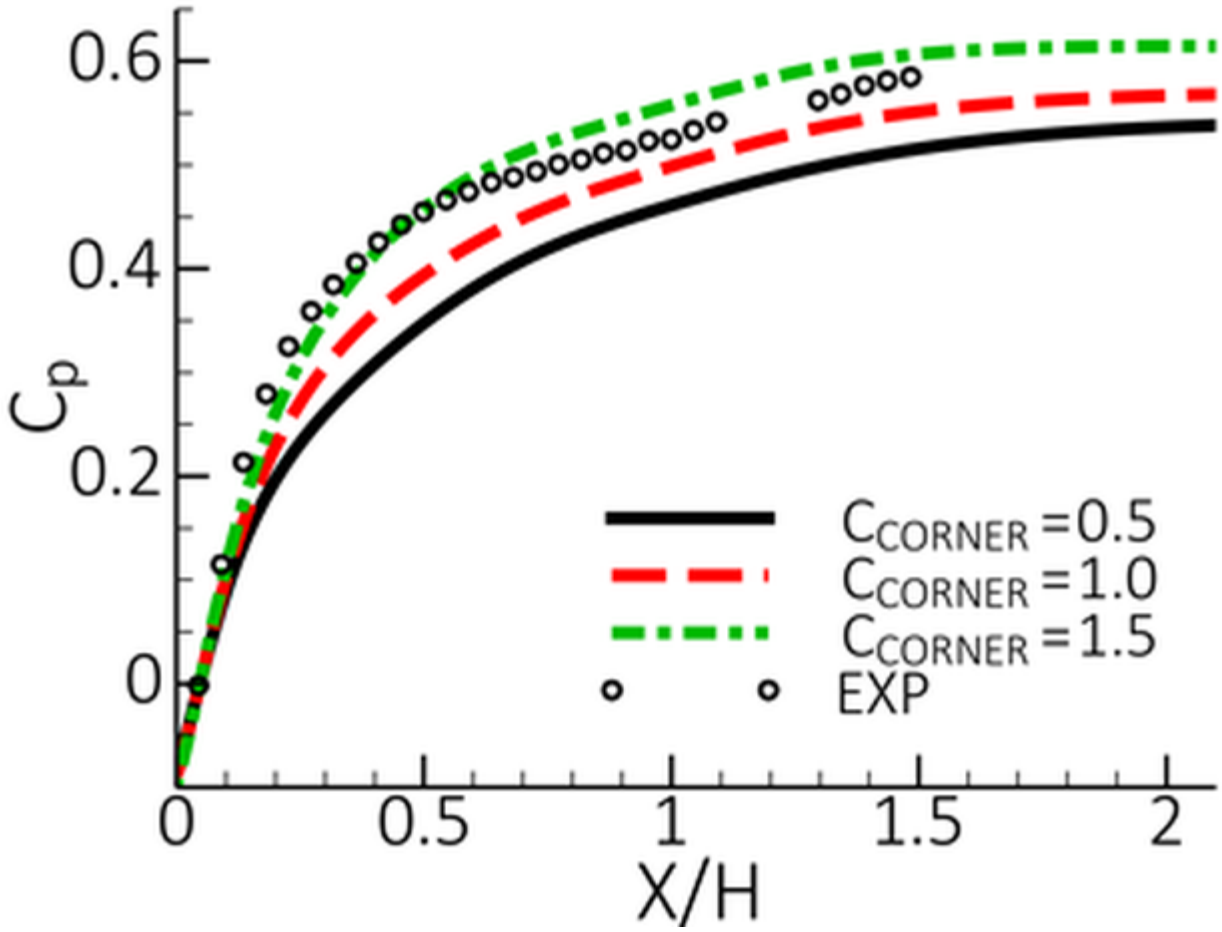
The influence of the model changes on the pressure coefficient, Cp, can be seen in Figure 12.200: Wall pressure coefficient,C_p, for 3D diffuser for GEKO-1.00 with different C_{CORNER} values. (p. 427). Note that the current flow is very sensitive and hard to compute, but it does demonstrate the importance of corner flow separation for technical flows.

Figure 12.199: Flow topology for 3D diffuser flow shown through streamwise velocity contour at downstream end of diffuser opening section for GEKO-1.00 with different C_{CORNER} values.



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.200: Wall pressure coefficient, C_p, for 3D diffuser for GEKO-1.00 with different C_{CORNER} values.



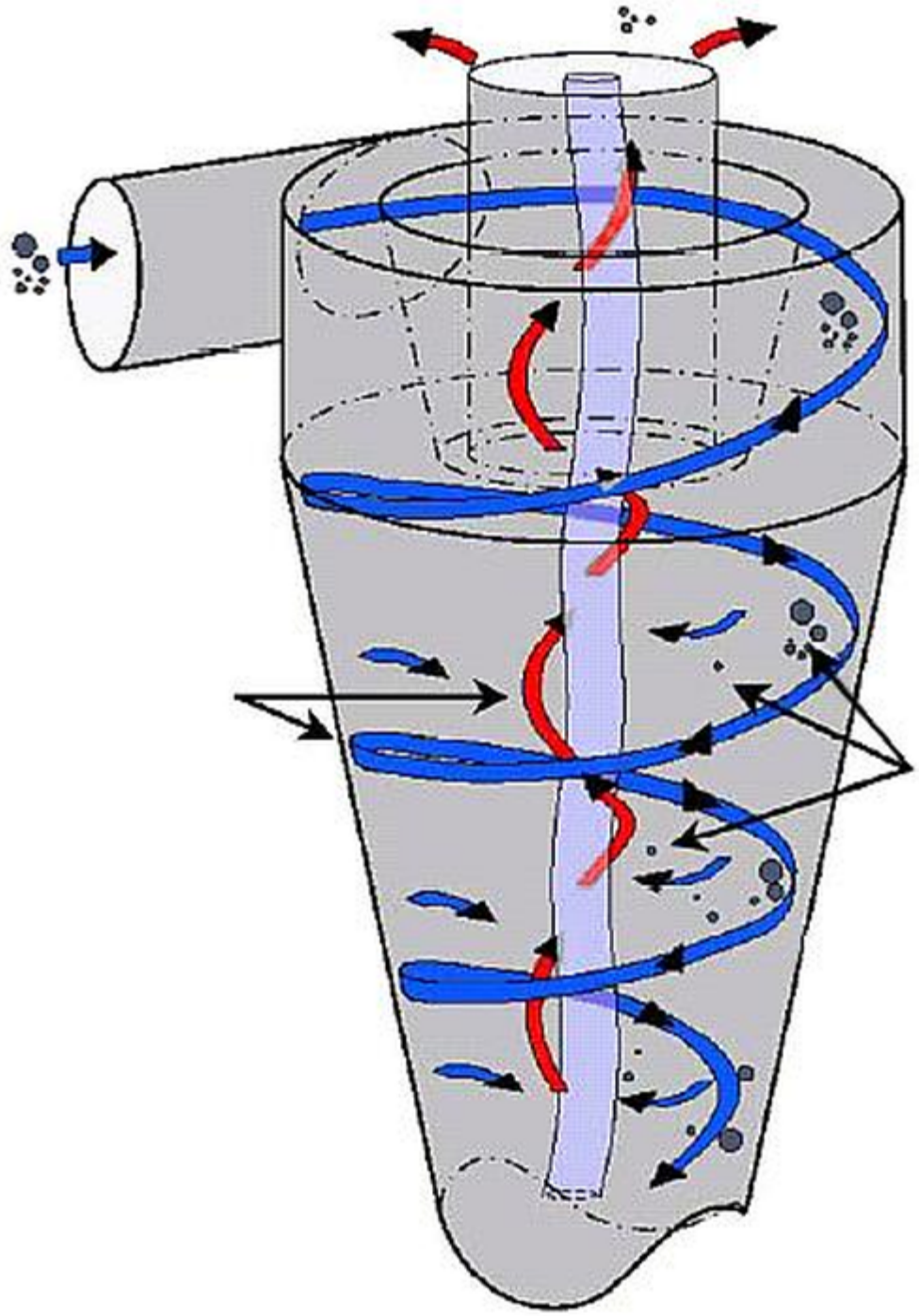
12.3.2.6. The 'Curvature' Parameter CCURV

The curvature correction (CC) is an already existing model, which is accessible to all eddy-viscosity models in Fluent. The only change when integrating it with GEKO was that the coefficient CCURV is now also accessible through a User Defined Function (UDF) and can thus be optimized zonally, or even locally.

No detailed description of the CC formulation is given, as it is described elsewhere [22 (p. 444)]. The effect of the CC can be shown for a hydro-cyclone (see Figure 12.201: Hydrocyclone typical flow structure. Reproduced from Cullivan et al [9]. (p. 428)) The flow (typically with particles) enters the domain tangentially through the feed. A strong swirl is generated pushing the particles to the wall and out through the underflow, whereas the cleaned fluid leaves the domain through the overflow. The effect of swirl and rotation cannot be handled by eddy-viscosity models without corrections. The current CC serves this purpose, as can be seen by a comparison of the circumferential velocity profiles for the SST model, the SST-CC and a full Reynolds Stress model (RSM-SSG) in Figure 12.202: Time-averaged profiles of the tangentional velocity in the hydrocyclone. Comparison with the experiments of Hartley [13]. (p. 429). While the model without CC produces mostly a solid-body rotation, the SST-CC model gives results much closer to the experiments and similar to a full Reynolds Stress model (note that at the time of writing, the GEKO model has not been run for this test case. However, the effect of the CC is mostly independent of the underlying model formulation).

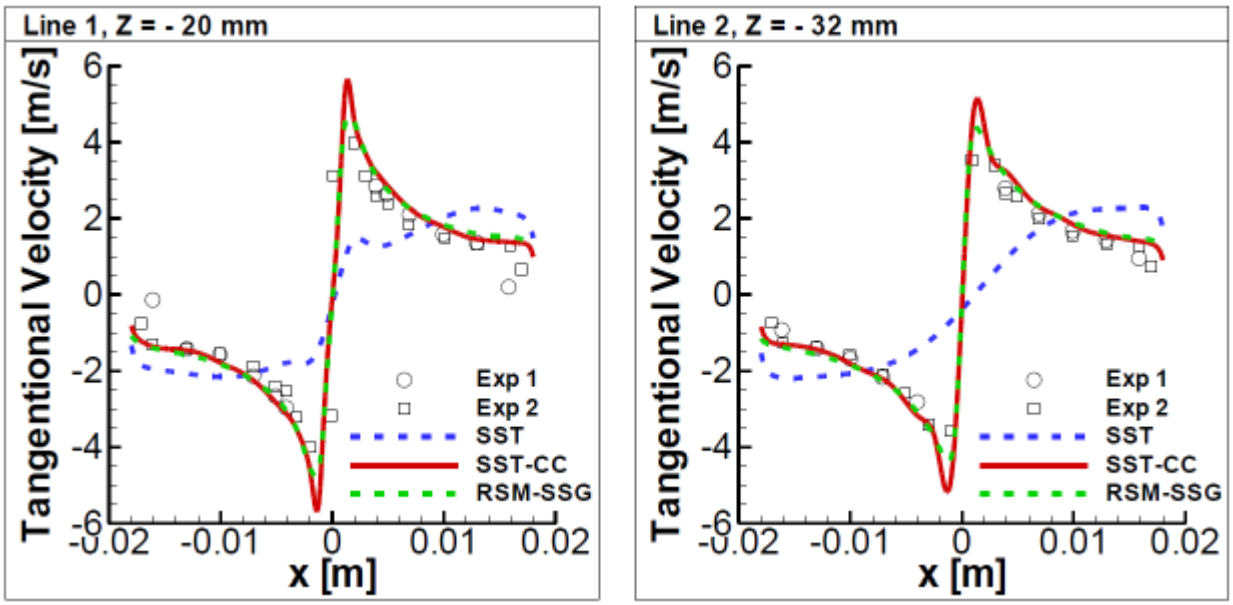
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

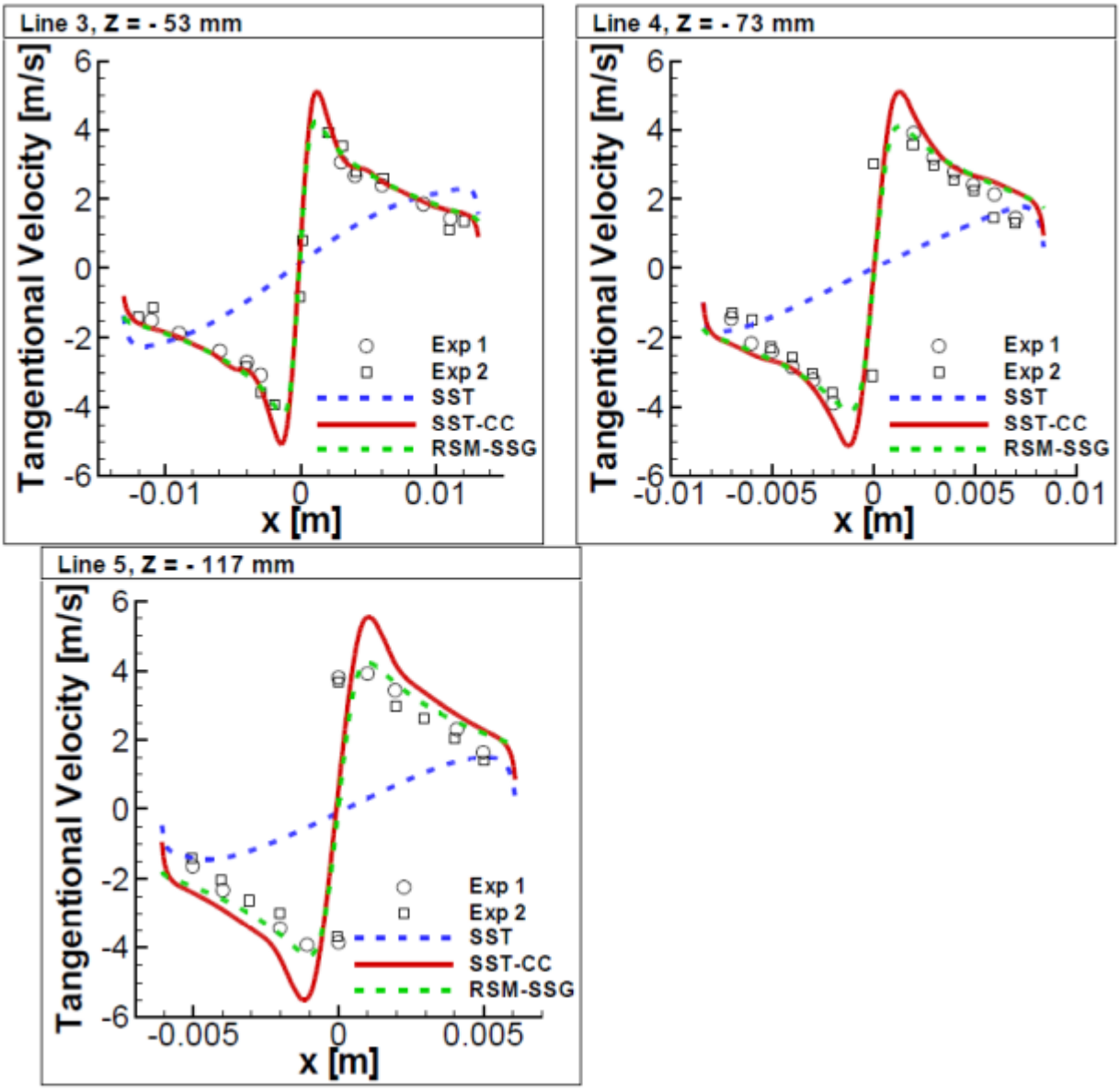
Figure 12.201: Hydrocyclone typical flow structure. Reproduced from Cullivan et al [9 (p. 442)].



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.202: Time-averaged profiles of the tangentional velocity in the hydrocyclone. Comparison with the experiments of Hartley [13 (p. 443)].





12.3.2.7. The Blending Function

The blending function activates the coefficients $C_{\mathrm{MIX}}/C_{\mathrm{JET}}$. In order to avoid any impact on boundary layers, the blending function is designed such that these parameters have only a minor effect there. In most cases the users will not have to change the function FGEKO (in Fluent it is called 'Blending Function for GEKO'). However, there are several ways to modify the function in case it is needed.

The easiest way is to adjust the coefficients exposed in the GUI. For fully turbulent flows (no transition model) there is only one coefficient called CBF_TURB (GEKO). Increasing it will increase the thickness of the near wall 'shielding' and decreasing it will decrease the 'shielding' from $C_{\mathrm{MIX}}/C_{\mathrm{JET}}$ inside the boundary layer. This effect can be seen in Figure 12.203: Effect of CBF_TURB on F_{GEKO} for flow around airfoil. Left CBF_TURB=2.0. Right: CBF_TURB=4.0. (p. 431) for the flow around a NACA 4412 airfoil. The lefts side of the figure shows the default setting (CBF_TURB=2) and the right side the function for CBF_TURB=4. The thickness of the boundary layer 'shielding' is clearly increased on the right part of the figure.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

There is a second coefficient for this function, which is a bit more involved. It is introduced to allow protection of the laminar boundary layer from the $C_{\mathrm{MIX}}/C_{\mathrm{JET}}$ term. This is desirable in case of transition predictions [16 (p. 443)], where otherwise the $C_{\mathrm{MIX}}/C_{\mathrm{JET}}$ term can affect the transition location. For this purpose, a second coefficient, CBF_LAM (GEKO) is used in case a transition model is selected. The effect is shown in Figure 12.204: Effect of CBF_LAM on F_{GEKO} for flow around airfoil. Left CBF_LAM=1.0 (default for turbulent simulations). Right: CBF_LAM=25 (default for transitional simulations). (p. 431) . The left figure shows the turbulent default (CBF_TURB=2.0, CBF_LAM=1.0 same as Figure 12.203: Effect of CBF_TURB on F_{GEKO} for flow around airfoil. Left CBF_TURB=2.0. Right: CBF_TURB=4.0. (p. 431) Left) and the right part shows the increase to its default for transitional simulations (CBF_TURB=2.0 CBF_LAM=25.0). As can be seen from the right part of Figure 12.203: Effect of CBF_TURB on F_{GEKO} for flow around airfoil. Left CBF_TURB=2.0. Right: CBF_TURB=4.0. (p. 431), the coefficient CBF_LAM provides increased shielding, especially in the leading edge region. In case of fully turbulent settings (no transition model), users can only access CBF_TURB (CBF_LAM=1.0 is fixed by default). The parameter CBF_LAM becomes available once a transition model is selected. The coefficient CBF_LAM needs to always satisfy $CBF_LAM \geq 1$.

Figure 12.203: Effect of CBF_TURB on F_{GEKO} for flow around airfoil. Left CBF_TURB=2.0. Right: CBF_TURB=4.0.

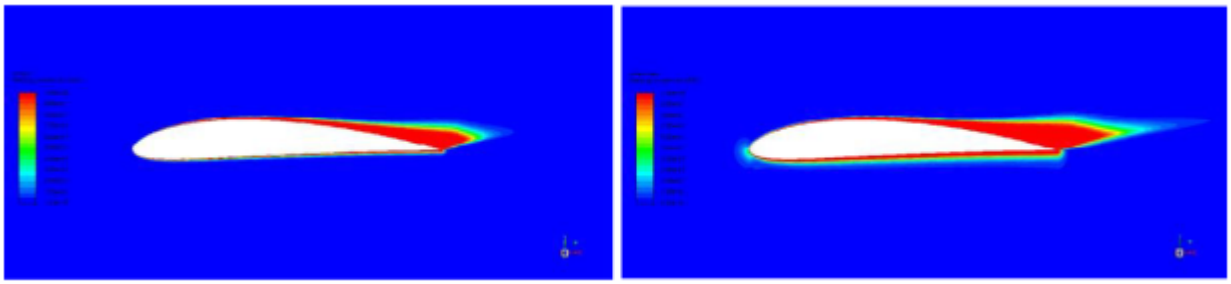
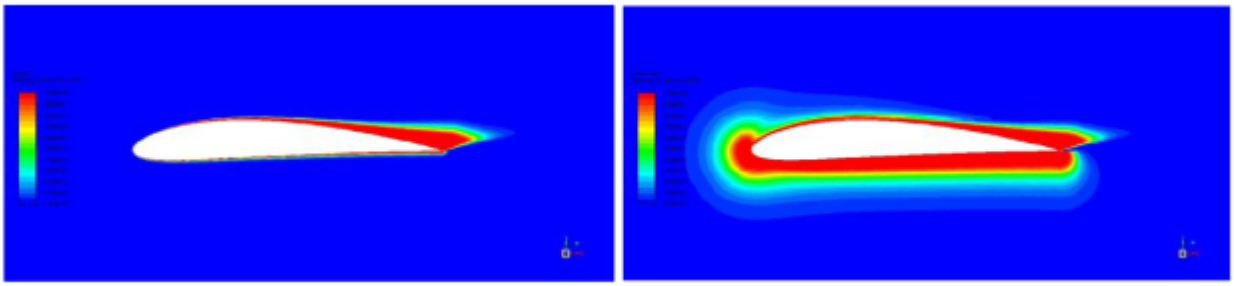


Figure 12.204: Effect of CBF_LAM on F_{GEKO} for flow around airfoil. Left CBF_LAM=1.0 (default for turbulent simulations). Right: CBF_LAM=25 (default for transitional simulations).



Finally, the function FGEKO can be adjusted through a User Defined Function (UDF). Users can either over-write the function in the entire domain, or only in certain parts, as the original function is accessible in the UDF. This could make sense in areas where free shear flows and boundary layers cannot be easily discerned by the function itself, especially when C_{MIX} is increased to obtain more mixing in the free shear flow region.

An example of such a flow is shown in Figure 12.205: Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 433) for a multi-element airfoil. Assume, the wake of the slat (upstream airfoil) over the main airfoil is of interest. Assume that the default settings provide not enough mixing in this region, hence an increased mixing through increases in C_{MIX} is desired. This works for moderate

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

increases in C_{MIX}, but large increases (like C_{MIX}=3) lead to decreasing efficiency of the C_{MIX} term. The reason is that through increases in the eddy-viscosity, the blending function becomes activated also in the wake, due to the proximity of the main airfoil wall. This is shown in the upper right of Figure 12.205: Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 433), where one can clearly see the activation of the blending function in the wake of the slat. To counter such an effect, one can over-write the blending function. In the current case, a function based on wall-distance was used as shown in the bottom left of Figure 12.205: Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 433). Note that this is just a generic example to demonstrate the issues and no effort was made to optimize the bending function. The effect of the change in the function F_{GEKO} is seen in Figure 12.206: Ratio of turbulence to molecular viscosity for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 434) which shows the ratio of turbulence to molecular viscosity for the blending functions and settings shown in Figure 12.205: Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 433). Clearly, the ratio increase from the left (default settings) to the middle picture. However, the increase is moderate. When restricting the function FGEKO to the immediate boundary layer around the main wing, the effect of increasing C_{MIX} becomes much stronger, resulting in a substantial increase in the viscosity ratio (Figure 12.206: Ratio of turbulence to molecular viscosity for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF (p. 434) – right).

It needs to be stressed again, that such modifications of F_{GEKO} are typically not required and the flexibility of provided by simply changing coefficients is sufficient in most cases. However, the discussion shows the versatility of the current GEKO models implementation.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.205: Blending function F_{GEKO} for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF

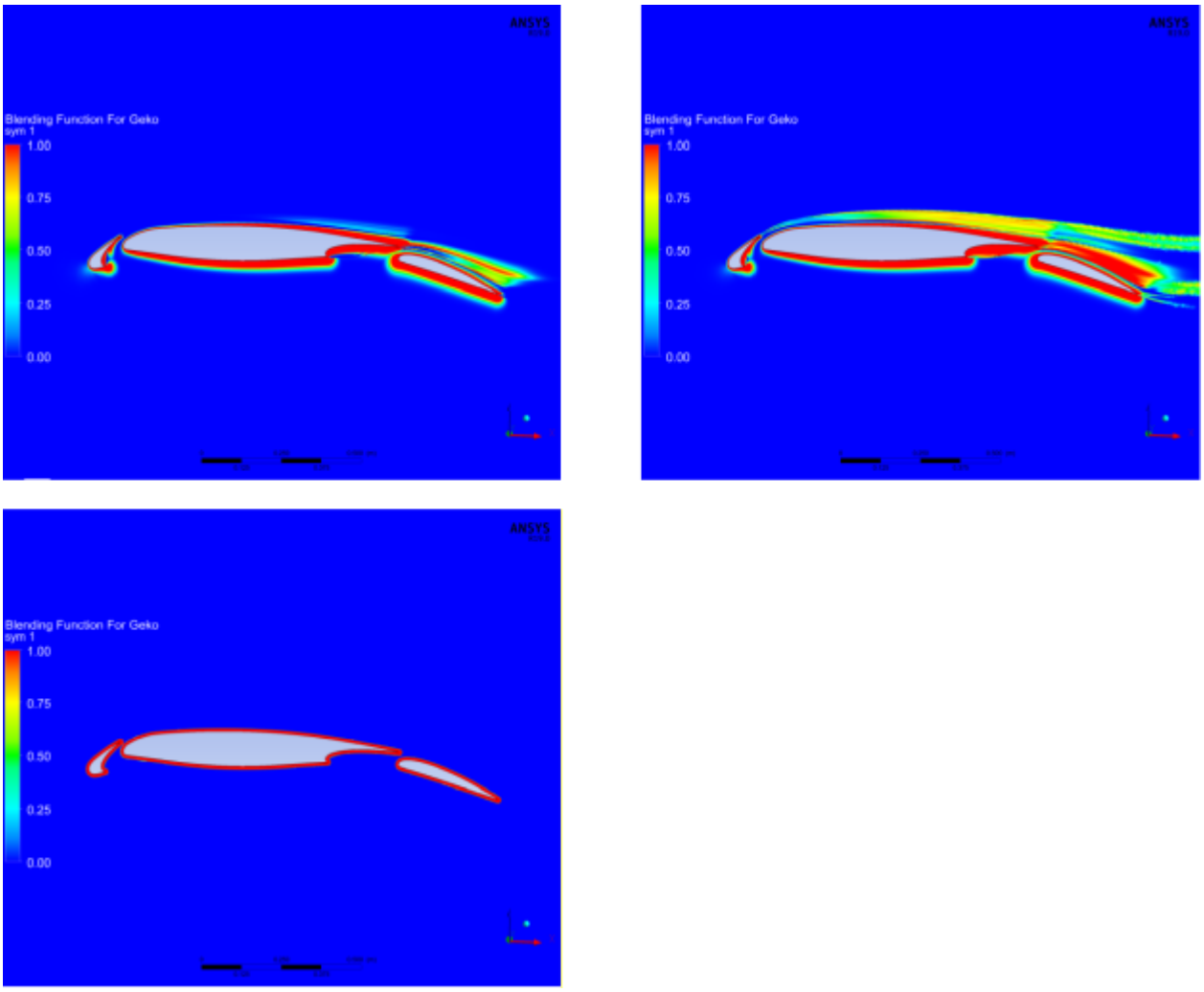
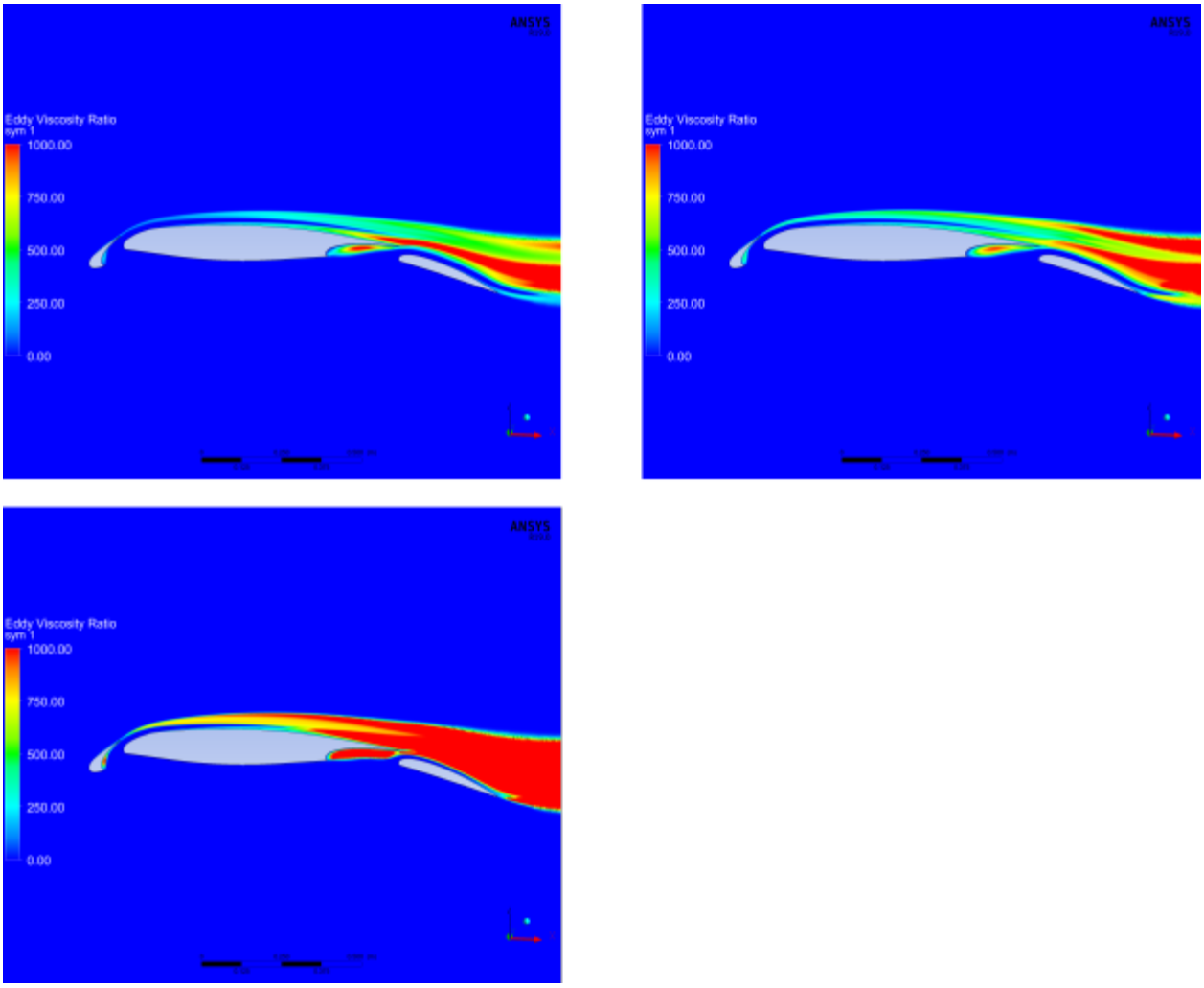


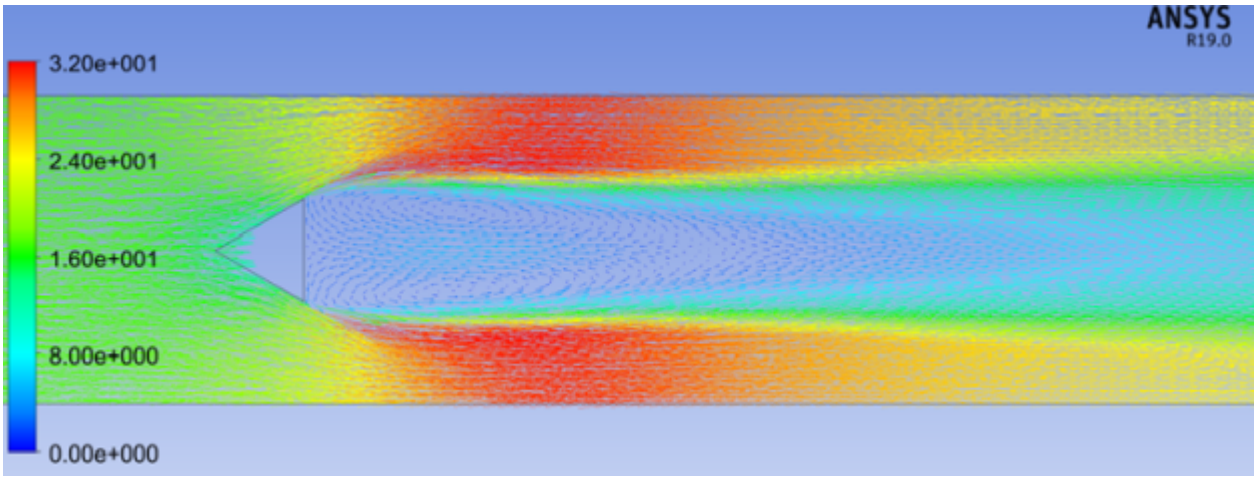
Figure 12.206: Ratio of turbulence to molecular viscosity for multi-element airfoil. Upper left: Default C_{SEP}=1.75, C_{MIX}=C_{MixCor}=0.303. Upper right: C_{MIX}=3. Bottom left: F_{GEKO} through UDF



A more dramatic example of how the change in blending function can affect results is given by the flow around a triangular cylinder. The set-up is shown in Figure 12.207: Flow around triangular cylinder [37] (p. 435).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.207: Flow around triangular cylinder [37]

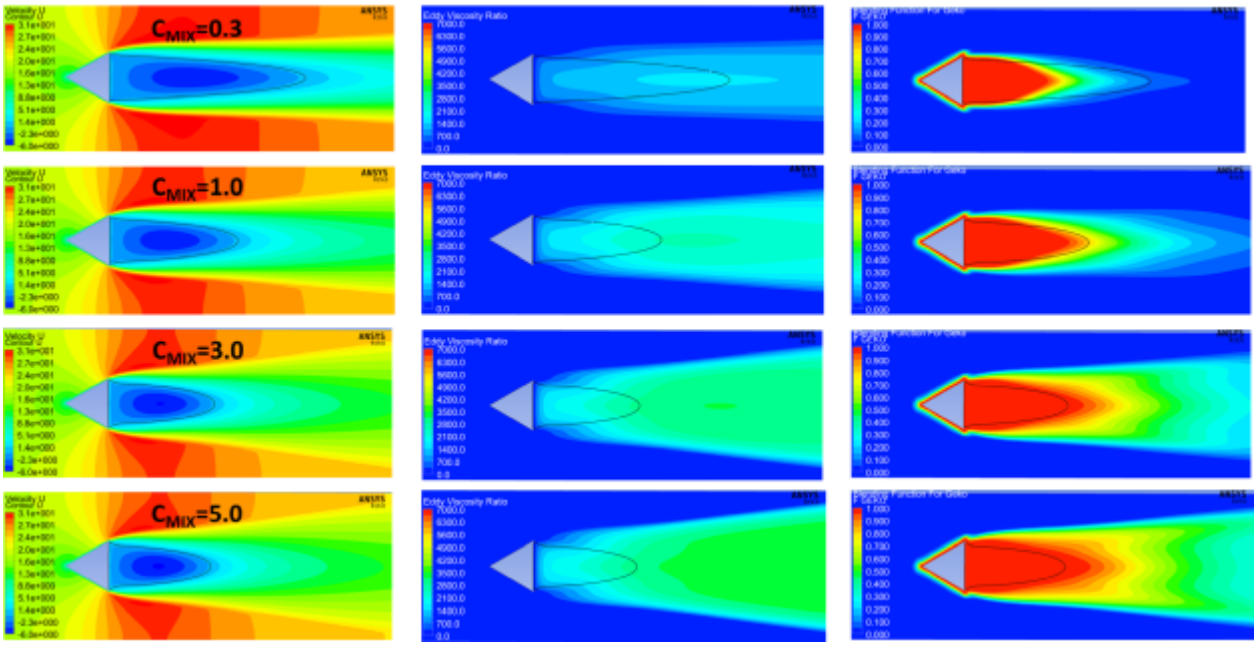


The cylinder has an edge length of L=$0.04m$, the height of the channel is H=$0.12m$. The inlet velocity is $U \sim 16\, m/s$. The simulation is carried out in steady state mode – and converges to a steady state solution (which is not always possible with bluff body flows). The real flow is of course unsteady with strong vortex shedding in addition to turbulence created unsteadiness. The vector field depicted shows a very large separation zone downstream of the cylinder, as expected. The simulation was carried out with GEKO-1.75 (default) under steady-state conditions.

In order to test by how much, the re-circulation zone can be reduced, the C_{MIX} coefficient was increased. The results are shown in Figure 12.208: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U. Middle: Eddy-viscosity ratio. Right: F_{GEKO} using built-in function [28] (p. 436) when using the built-in version of the F_{GEKO} function. The recirculation (line in middle figure) decreases with increasing C_{MIX}. However, eventually, the entire separation zone lies within the F_{GEKO}=1 region and no further effect can be achieved by increasing C_{MIX}.

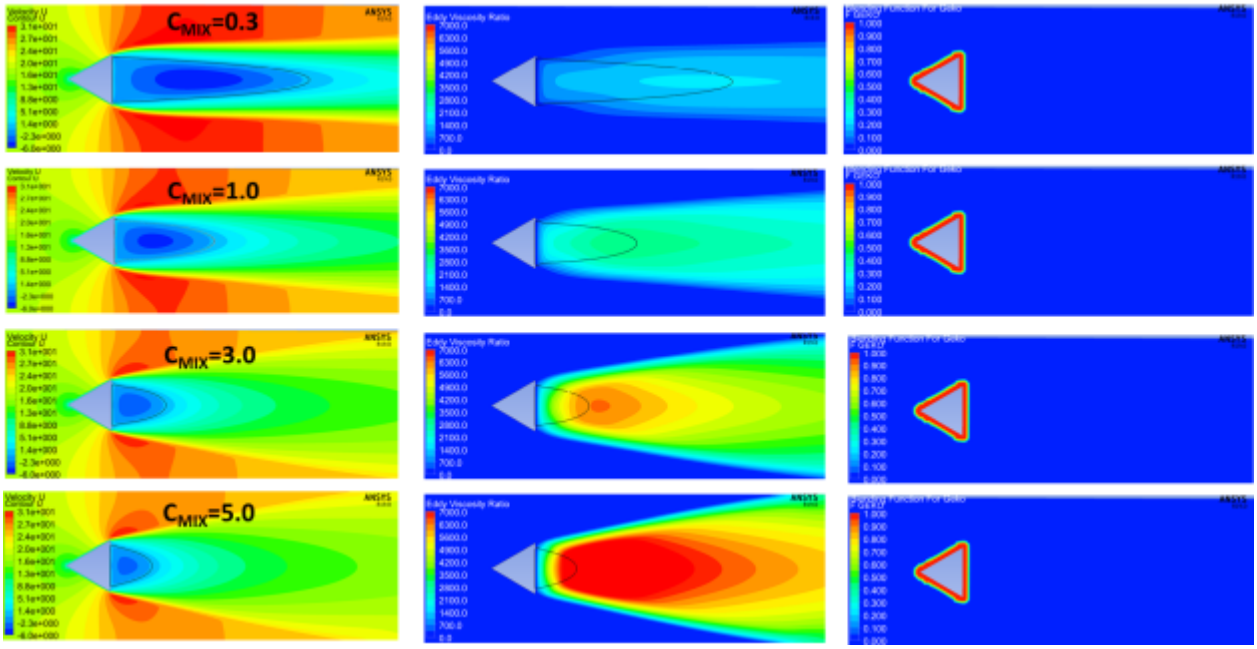
Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 12.208: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U. Middle: Eddy-viscosity ratio. Right: F_{GEKO} using built-in function [28 (p. 445)]



The same tests are shown in Figure 12.209: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U Middle: Eddy-viscosity ratio. Right: F_{GEKO} using UDF-based function [28] (p. 436) but with the help of a UDF-based F_{GEKO} function. As shown in the right part of the figure, F_{GEKO} is defined only within a given wall distance as F_{GEKO}=1 and is switched to free shear flow status outside. This avoids the restriction of the impact of increasing C_{MIX} and allows a further reduction in the re-circulation zone due to increased mixing.

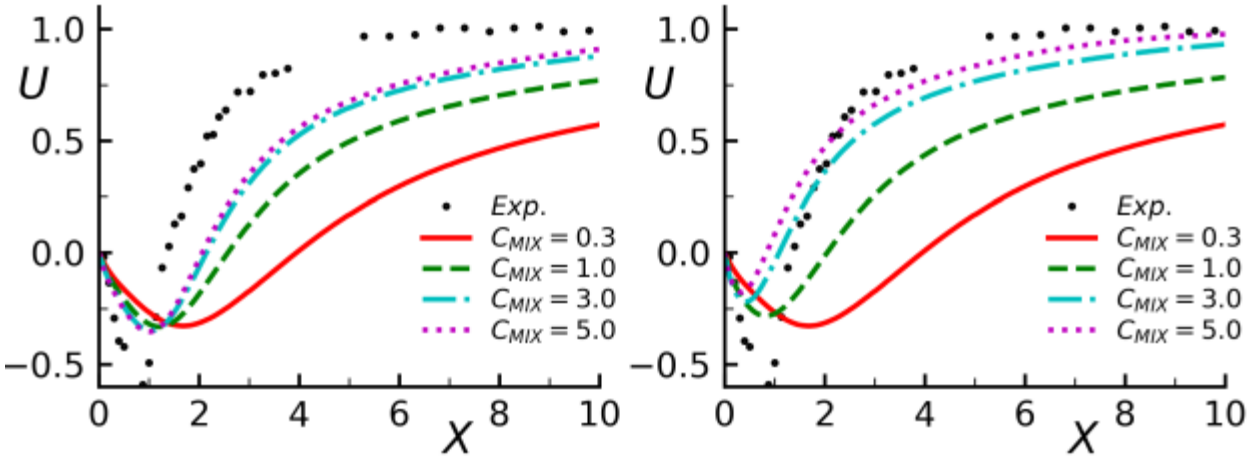
Figure 12.209: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U Middle: Eddy-viscosity ratio. Right: F_{GEKO} using UDF-based function [28 (p. 445)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The comparison with experimental data along the center-line downstream of the cylinder is shown for both F_{GEKO} variants in Figure 12.210: Center line velocity for triangular cylinder. Left: GEKO-1.75 with built-in F_{GEKO} function. Right: GEKO-1.75 with UDF-based F_{GEKO} function [28] (p. 437). The left part of the figure shows the solution using the built-in function for F_{GEKO} and the right part shows the solution using a wall-distance-based UDF variant (as shown in right part of Figure 12.209: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U Middle: Eddy-viscosity ratio. Right: F_{GEKO} using UDF-based function [28] (p. 436)). Clearly, the UDF based variant allows a much stronger reduction in separation size. It should be noted that none of the simulations is able to produce the fairly strong backflow velocity in the re-circulation zone. This should not come as a surprise, as it is mostly a result of very strong backflow events in the unsteady vortex shedding cycle of the experiment, which cannot be captured by a steady state solution. Nevertheless, it is important to observe the much better agreement in flow recovery downstream of the recirculation zone. In case of more complex arrangements, any additional parts of the geometry downstream of the triangular cylinder would see a much more realistic approaching flow field than with the default settings.

Figure 12.210: Center line velocity for triangular cylinder. Left: GEKO-1.75 with built-in F_{GEKO} function. Right: GEKO-1.75 with UDF-based F_{GEKO} function [28 (p. 445)]



12.3.2.8. Other Special Coefficients

The realizability coefficient is set to its standard value of C_{REAL}=0.577. The value should typically not be changed but is accessible for special situations. However, it was found that limiters can have a strong effect for flows which are severely under-resolved. An example would be an inlet condition for the velocity field, where the velocity in the lower half of the inlet has one value and in the upper half another. At the jump between these two values, the flow is under-resolved and the coefficient C_{REAL} could potentially delay the growth of the mixing layer. Note that for such flows, the production limiter could have a similar effect. If this is the case, it would be an option to set both limiters to very large values and instead activate the Kato-Launder limiter.

The coefficient CNW_SUB allows for slight re-tuning of the log-layer shift which will also affect the wall shear stress distribution for boundary layers. Increasing its value from CNW_SUB=1.7 will shift the log-layer more into the laminar direction (up) and decrease the wall shear stress.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3.3. Strategies for Model Optimization

Most users are intimidated/scared of the prospect to modify turbulence model coefficients. Part of the reason for this lies in the interconnectedness of coefficients in conventional turbulence models, where any change to any coefficient can have detrimental effects even on the simplest flows, like flat plate boundary layers (which users typically do not want to modify). Still, a conservative attitude is also commendable for the GEKO model, even though the effects of coefficient changes are much more predictable. The model should therefore only be adjusted if other sources of error have been minimized (it is not always the turbulence model's fault if things do not match). In addition, modifications should be guided by experimental data as far as possible.

The following topics are discussed:

12.3.3.1. GEKO Defaults

12.3.3.2. Optimizing Coefficients

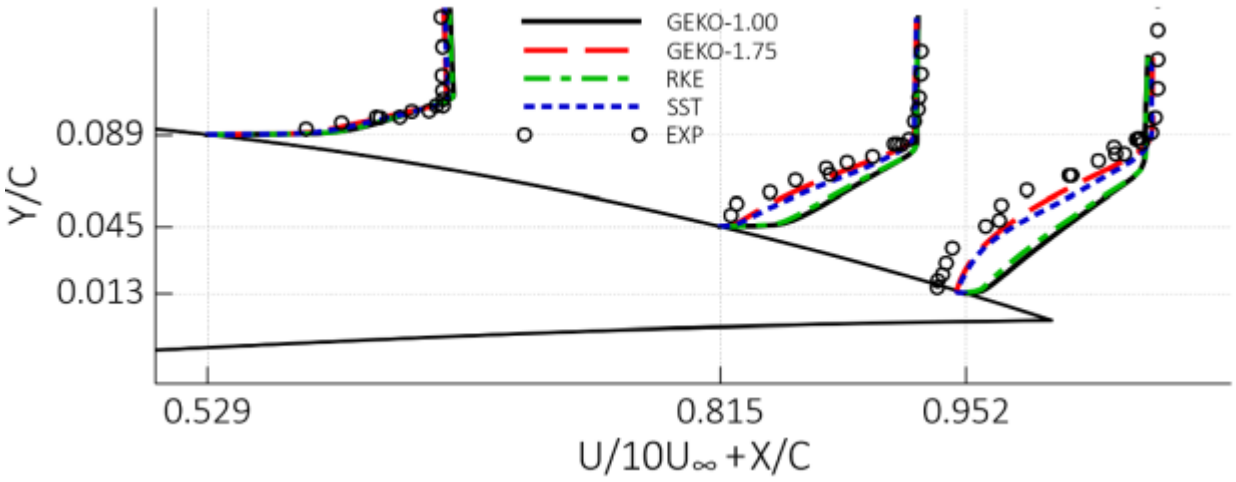
12.3.3.1. GEKO Defaults

The defaults for the GEKO model have been selected to match the SST models performance as closely as possible for the building block flows. Especially for boundary layers, the defaults predict very close results to those of the SST model. The SST model is used in many industrial CFD simulations already, so the default selected for the GEKO model provides a fairly save conversion from SST to GEKO.

There is another 'fix point' in coefficient settings. With the combination C_{SEP}=1.0, C_{NW}=1.0, which automatically sets C_{MIX}=0.0 from the correlation CMixCor (note again that C_{MIX}=0.0 renders C_{JET} passive, as it is a sub-model to C_{MIX}). This setting is an exact transformation of the standard $k-\varepsilon$ model (except for the wall treatment and the realizability limiter). Users who have used the $k-\varepsilon$ model successfully in the past, are therefore advised to use these settings.

Figure 12.211: Comparison of velocity profiles for flow around the NACA-4412 airfoil [32] (p. 438) shows a comparison of GEKO model settings for the NACA 4412 airfoil [32 (p. 445)] against their 'reference' model. Two model pairs are clearly visible (GEKO-1, RKE) and (GEKO-175, SST) – each pair giving almost identical results.

Figure 12.211: Comparison of velocity profiles for flow around the NACA-4412 airfoil [32 (p. 445)]



Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3.3.2. Optimizing Coefficients

C_{SEP}

The most important coefficients for most applications is C_{SEP}. It controls the separation points/lines from smooth body-separation. In case the flow is dominated by boundary layers, users should only modify this coefficient and explore if values within the range given by Equation (2.10) are sufficient for obtaining improved results. Again, increasing C_{SEP} will lead to stronger/earlier separation. When changing C_{SEP} one should in a first step keep all other coefficients at their default values.

C_{NW}

The coefficient C_{NW} should only be changed if detailed near wall or surface information needs to be matched and if this cannot be achieved by optimizing C_{SEP} alone. The most prominent example would be optimizations with respect to heat transfer coefficients or oil-flow pictures from experiments. Increasing C_{NW} will increase heat transfer and wall shear stress levels in non-equilibrium regions.

C_{MIX}

In some cases, standard settings (or models) under-estimate the turbulent mixing in free shear flows. The coefficients C_{MIX} will allow an adjustment under such scenarios. Increasing C_{MIX} will increase eddy-viscosity levels in such zones. It should be noted that this is only possible within physical limits. E.g. in some cases, strong mixing is observed behind bluff bodies. Such effects often result from vortex shedding and cannot be covered fully by a steady state turbulence model run. However, increasing C_{MIX}, can improve such situations relative to default settings (see Figure 12.209: GEKO-1.75 solution under variation of C_{MIX}. Left: Velocity U Middle: Eddy-viscosity ratio. Right: F_{GEKO} using UDF-based function [28] (p. 436)). In case changes to C_{MIX} do not show the desired influence, it is advisable to check the blending function F_{GEKO}. It should be recalled that C_{MIX} is only effective in regions where $F_{GEKO}\neq 1$. In case, C_{MIX} is de-activated by F_{GEKO} in the region of interest, modifications to the F_{GEKO} function might be required.

C_{JET}

The coefficient C_{JET} is subtle. As the name implies it should only be considered when jets are present in the domain. Regions with round jets should best be computed with C_{SEP}=1.75-2.00 as otherwise the effect of C_{JET} is not strong enough to achieve the desired effect. In case these settings for C_{JET} are not suitable in the entire domain, one can set these values also locally through UDF access. For highly accurate jet simulations also set CJET_AUX =4.0 (it activates the C_{JET} function more aggressively).

In summary – for the free shear flows, the GEKO-1 model behaves like the underlying $k-\varepsilon$ model, with good spreading rates for the mixing layer and plane jet but an over-prediction of spreading for the round jet. When increasing C_{SEP} one also needs to adjust C_{MIX} to maintain proper spreading for the mixing layer. This is automatically achieved through a correlation relating these two coefficients (C_{MIX}=C_{MixCor}). To avoid over-prediction of spreading rates for jets, with increasing C_{MIX}, the coefficient C_{JET} is introduced. A value of C_{SEP}=2 and C_{JET}=0.9 provides correct asymptotic spreading for both, the round and the plane jet. In case stronger mixing is required, the coefficient C_{MIX} can

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

be increased relative to its correlation value. In some cases, it might even be desirable to modify the blending function F_{GEKO} to obtain an even stronger effect for mixing layers near walls (see 3.7).

C_{CORNER}, C_{CURV}

The coefficients C_{CORNER} and C_{CURV} are also available for other $k-\omega$ models. They can be activated in combination with GEKO as required by the simulation.

Wall Distance Free

The GEKO model can also be run without a need for computing the wall distance. This is desirable for cases with moving grids/geometries. If the 'wall-distance-free' option is selected the coefficients $C_{\mathrm{MIX}}/C_{\mathrm{JET}}$ are de-activated. In order to still achieve proper mixing layer growth, set C_{SEP}=1.

12.3.4. Summary

A new RANS concept has been introduced. The new model is termed Generalize $k-\omega$ model (GEKO). It is based on a $k-\omega$ model platform and is designed to consolidate RANS turbulence models in Ansys CFD. Instead of offering a wide range of different models, it is the goal to provide a single model, with the flexibility to adjust it to a wide range of generic flow conditions and applications.

Flexibility within GEKO is achieved by augmenting the model with free coefficients, which can be adjusted by the user without the danger of violating the basic model calibration for conventional free shear flows and boundary layers. This allows the user to tune the model in a safe parameter space without the need for expert knowledge in turbulence modeling. In other words, instead of switching between a large number of existing turbulence model to find the optimal one for a given application, the user can now stay within one model framework and simple adjust the free coefficients. The model is designed such that the modification of the coefficients allows coverage of a wide solution space (actually a wider space than by switching between existing models).

In addition to being able to adjust the model coefficients, GEKO offers the advantage that the variation of coefficients is much more transparent than the change between different models. By changing from one model to another (say from SST to $k-\varepsilon$) one does not only switch the model but also numerous other settings (e.g. different limiters, different wall treatment, …). Such changes might have an additional large effect on the solution without being transparent to the user. Finally, not all options/extensions are compatible with all turbulence models. Case in point is that models for laminar-turbulent transition are not compatible with existing $k-\varepsilon$ models. A $k-\varepsilon$ model user who wants to add transition physics, will therefore need to first switch to a $k-\omega$ model and then activate the transition model. It will then be difficult to attribute solutions changes to any one of the two changes made in the settings. Within GEKO, the user can make changes step by step and observe their impact on the solution separately, as all options are (or will be made) available within this model framework.

It is to be emphasized that users do not have to adjust coefficients. There are strong default settings (similar to the SST model) and there are further recommendations for settings to mimic other existing models like $k-\varepsilon$. It is however assumed that many simulation results can significantly be improved by very few changes to the GEKO coefficients. For simple geometries, a global optimization of the coefficients is typically sufficient. For more complex application, where several different turbulence-related phenomena are included in one set-up, the coefficients can be set via UDFs zonally. In the longer run, it can also be anticipated that the free coefficients can be used for automatic optimization and Machine Learning.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12.3.5. Example UDFs

This UDF overwrites the blending function F_{GEKO} in regions where x≤2:

```
#include "udf.h"
DEFINE_KW_GEKO_BF(user_geko_bf, c, t)
 {
    real bf_value;
    real xc[ND_ND];
    C_CENTROID(xc,c,t);
    if (xc[0]>--> 2.0)
      bf_value = Get_Geko_Blending_Function (c, t,
                 C_R(c,t), C_MU_L(c,t), C_WALL_DIST(c,t),
                 C_K(c,t), C_O(c,t));
    else
      bf_value = 1.0;
    return bf_value;
 }
```

12.3.6. 7 References

1

Albertson, M.L., Dai Y.B., Iensen, R.A. and Rouse H. (1948),

"Diffusion of submerged jets",

Proc. of ASME, v.74.

2

Bachalo, W. D. and Johnson, D. A. (1986),

"Transonic, Turbulent Boundary-Layer Separation Generated on an Axisymmetric Flow Model",

AIAA Journal, Vol. 24, No. 3, , pp. 437-443.

3

Bell, J. H. and Mehta, R. D. (1990),

"Development of a two-stream mixing layer from tripped and untripped boundary layers",

AIAA J., vol. 28, no.12, pp. 2034-2042.

4

Bradbury, L.J.S. (1965),

"The Structure of a Self-Preserving Plane Jet",

Journal of Fluid Mechanics Vol. 23, pp.31-64.

5

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Buice, C.U. and Eaton, J.K. (2000),

"Experimental Investigation of Flow Through an Asymmetric Plane Diffuser",

Journal of Fluids Engineering, Vol. 122, pp. 433-435. 96.

6

Cebeci, T. and Bradshaw, P. (1988),

"Physical and Computational Aspects of Convective Heat Transfer",

1st ed. 1984/2nd ed. 1988, Springer-Verlag.

7

Coles, D.E., and Hirst, E.A., Eds (1969),

"Computation of turbulent boundary layers- 1968 AFOSR-IFP-Stanford Conference",

Vol. 2, Stanford University, CA.

8

Cook, P.H., M.A. McDonald and M.C.P. Firmin (1979),

"Aerofoil RAE 2822 - Pressure Distributions, and Boundary Layer and Wake Measurements",

Experimental Data Base for Computer Program Assessment, AGARD Report AR 138.

9

Cullivan, J.C. Williams. R.A., Dyakowski T. and Cross C.R. (2004),

"New understanding of a hydrocyclone flow field and separation mechanism from computational fluid dynamics",

Minerals Engineering, Vol.17 No.5, 2004.

10

Driver, D.M. (1991),

"Reynolds Shear Stress Measurements in a Separated Boundary Layer Flow",

AIAA 22nd Fluid Dynamics, Plasma Dynamics and Laser Conference, AIAA-91-1787.

11

Durbin, P.A. and Reif, B.A. (2001),

"Statistical Theory and Modeling for Turbulent Flows,"

John Wiley & Sons.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

12

Durbin, P.A. (1991),

"Near-wall turbulence closure modeling without 'damping functions'",

Theor. Comput. Fluid Dyn. 3(1).

13

Hartley, C.D. (1994),

"Measurement of flow velocities within a hydrocyclone using laser doppler anemometry",

Technical Report FTN/X/82, AEA, Power Fluidics, BNFL

14

Jones W. P. and Launder B. E. (1972),

"The Prediction of Laminarization with a Two-Equation Model of Turbulence",

Int. J. Heat Mass Transfer., 15:301–314.

15

Kato, M., and Launder, B.E. (1993),

"The modelling of turbulent flow around stationary and vibrating square cylinders".

In Proc. Ninth Symposium on Turbulent Shear Flows, volume 9, page 10.4.1, Kyoto, Japan.

16

Langtry, R.B. and Menter, F.R. (2009),

"Correlation-Based Transition Modeling for Unstructured Parallelized Computational Fluid Dynamics Codes",

AIAA J. 47(12), 2984-2906.

17

Mani, M., Babcock, D. A., Winkler, C. M., and Spalart, P. R. (2014),

"Predictions of a Supersonic Turbulent Flow in a Square Duct",

AIAA Paper 2013-0860, January 2013.

18

Menter, F.R. (1994),

"Two-equation eddy-viscosity turbulence models for engineering applications",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AIAA Journal, Vol. 32, No. 8, pp. 1598-1605.

19

Pope, S.B. (1978),

"An explanation of the turbulent round-jet/plane-jet anomaly",

AIAA Journal, Vol. 16, No. 3, pp. 279-281.

20

Raiesi, H., Piomelli and U., Pollard, A. (2011),

"Evaluation of Turbulence Models Using Direct Numerical and Large-Eddy Simulation Data",

Journal of Fluids Engineering, v.133, No.2, 021203 (10 pp.).

21

Skare, P. E., and Krogstad, P. (1994),

"A Turbulent Equilibrium Boundary Layer Near Separation",

J. Fluid Mech., 272, pp. 319–348.

22

Smirnov P. E. and Menter F. R. (2008),

"Sensitization of the SST Turbulence Model to Rotation and Curvature by Applying the Spalart-Shur Correction Term",

ASME Paper GT 2008-50480. Berlin, Germany.

23

Somers, D. M. (1997),

"Design and Experimental Results for the S805 airfoil",

NREL/SR-440-6917

24

Somers, D. M. (2005),

"Design and Experimental Results for the S825 Airfoil",

NREL/SR-500-36346

25

Somers, D. M. (1997),

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Design and Experimental Results for the S809 Airfoil",

NRELSR-440-6918.

26

Somers, D. M. (1997),

"Design and Experimental Results for the S814 airfoil",

NREL/SR-440-6919.

27

Spalart P. R. and Shur M. L (1997),

"On the Sensitization of Turbulence Models to Rotation and Curvature",

Aerospace Sci. Tech. 1(5). 297–302.

28

Sjunnesson, A. Nelsons C. and Max E. (1991),

"LDA measurement of velocities and turbulence in a bluff body stabilized flame",

Volvo Flygmotor AB, Trollhattan

29

Timmer W. A. and van Rooij R. P. J. O. M. (2003),

"Summary of the Delft University Wind Turbine Dedicated Airfoils",

AIAA Paper, 2003-0352

30

Van der Hegge Zijnen B.G. (1959),

"Measurement of turbulence in a plane jet of air by the diffusion method and by hot-wire method",

Appl. Sci. Res., S.A., vol.7, No. 4.

31

Vogel, J.C. and Eaton, J.K. (1985),

"Combined Heat Transfer and Fluid Dynamic Measurements Downstream of a Backward-Facing Step",

Vol. 107, Journal of Heat Transfer, pp. 922 – 929.

32

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Wadcock, A. J. (1987),

"Investigation of Low-Speed Turbulent Separated Flow Around Airfoils",

NASA Contractor Report 177450

33

Wilcox, D.C. (2006),

"Turbulence Modeling for CFD",

DCW Industries Inc., 3rd. Edition.

34

Wilcox, D.C (2007),

"Formulation of the k-? Turbulence Model Revisited",

AIAA Paper 2007-1408.

35

Wolfshtein. M. (1969),

"The Velocity and Temperature Distribution of One-Dimensional Flow with Turbulence Augmentation and Pressure Gradient",

Int. J. Heat Mass Transfer. 12. 301–318.

36

Wygnanski, I. and Fiedler, H.E. (1968),

"Some Measurements in the Self-Preserving Jet",

Boeing Scientific Research Labs, Document D1-82-0712.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 13: CFX Command Language (CCL)

The CFX Command Language (CCL) is the internal communication and command language of Ansys CFX. It is a simple language that can be used to create objects or perform actions in the post-processor. All CCL statements can be classified into one of three categories:

- Object and parameter definitions, which are described in Object Creation and Deletion.
- CCL actions, which are commands that perform a specific task (such as reading a session file) and which are described in Command Actions in the *CFD-Post User's Guide*.
- Power Syntax programming, which uses the Perl programming language to allow loops, logic, and custom macros (subroutines). Power Syntax enables you to embed Perl commands into CCL to achieve powerful quantitative postprocessing. For details, see Power Syntax in Ansys CFX (p. 559).

State files and session files contain object definitions in CCL. In addition, session files can also contain CCL action commands. You can view and modify the CCL in these files by using a text editor.

For more information, see Object Creation and Deletion.

13.1. CFX Command Language (CCL) Syntax

The following topics will be discussed:

- Basic Terminology (p. 448)
- The Data Hierarchy (p. 448)
- Simple Syntax Details (p. 448)
  - Case Sensitivity (p. 448)
  - CCL Names Definition (p. 449)
  - Indentation (p. 449)
  - End of Line Comment Character (p. 449)
  - Continuation Character (p. 449)
  - Named Objects (p. 450)
  - Singleton Objects (p. 450)
  - Parameters (p. 450)
  - Lists (p. 450)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

– Parameter Values (p. 450)

– Escape Character (p. 452)

13.1.1. Basic Terminology

The following is an example of a CCL object that defines an isosurface.

```
ISOSURFACE: Iso1
   Variable = Pressure
   Value = 15000 [Pa]
   Color = 1,0,0
   Transparency = 0.5
END
```

- `ISOSURFACE` is an object type
- `Iso1` is an object name
- `Variable = Pressure` is a parameter
- `Variable` is a parameter name
- `Pressure` is a parameter value
- If the object type does not need a name, it is called a singleton object. Only one object of a given singleton type can exist.

13.1.2. The Data Hierarchy

Data is entered via parameters. These are grouped into objects that are stored in a tree structure.

```
OBJECT1: object name
   name1 = value
   name2 = value
END
```

Objects and parameters may be placed in any order, provided that the information is set prior to being used further down the file. If data is set in one place and modified in another, the latter definition overrides the first.

In CFD-Post, all object definitions are only one object level deep (that is, objects contain parameters, but not other objects).

13.1.3. Simple Syntax Details

The following applies to any line that is not a Power Syntax or action line (that is, the line does not start with a `!` or `>`).

13.1.3.1. Case Sensitivity

Everything in the file is sensitive to case.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Case sensitivity is not ideal for typing in many long parameter names, but it is essential for bringing the CFX Expression Language (CEL) into CCL. This is because some names used to define CCL objects (such as `Fluids`, `Materials` and `Additional Variables`) are used to construct corresponding CEL names.

For simplicity and consistency, the following is implemented:

- Singletons and object types use upper case only.
- Parameter names, and predefined object names, are mixed case. The CFX Expression Language tries to follow the following conventions:
  1. Major words start with an upper case letter, while minor words such as prepositions and conjunctions are left in lower case (for example, `Mass Flow in`).
  2. Case is preserved for familiar names (for variables `k` or `r`), or for abbreviation RNG.
- User object names conventions can be chosen arbitrarily by you.

13.1.3.2. CCL Names Definition

Names of singletons, types of object, names of objects, and names of parameters all follow the same rules:

- In simple syntax, a CCL name must be at least one character. This first character must be alphabetic; there may be any number of subsequent characters and these can be alphabetic, numeric, space or tab.
- The effects of spaces in CCL names are:
  - Spaces appearing before or after a name are not considered to be part of the name.
  - Single spaces appearing inside a name are significant.
  - Multiple spaces and tabs appearing inside a name are treated as a single space.

13.1.3.3. Indentation

Nothing in the file is sensitive to indentation, but indentation can be used for easier reading.

13.1.3.4. End of Line Comment Character

The # character is used for this. Any text to the right of this character will be treated as comments. Any characters may be used within comments.

13.1.3.5. Continuation Character

If a line ends with the character \, the following line will be linked to the existing line. There is no restriction on the number of continuation lines.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

13.1.3.6. Named Objects

A named object consists of an object type at the start of a line, followed by a `:` and an object name. Subsequent lines may define parameters and child objects associated with this object. The object definition is terminated by the string `END` on a line by itself.

Object names must be unique within the given scope, and the name must not contain an underscore.

13.1.3.7. Singleton Objects

A singleton object consists of an object type at the start of a line, followed by a `:`. Subsequent lines may define parameters and child objects associated with this object. The object definition is terminated by the string `END` on a line by itself.

The difference between a singleton object and a named object is that (after the data has been processed), a singleton can appear just once as the child of a parent object. However, there may be several instances of a named object of the same type defined with different names.

13.1.3.8. Parameters

A parameter consists of a parameter name at the start of a line followed by an = character followed by a parameter value. A parameter may belong to many different object types. For example, `U Velocity = 1.0 [m/s]` may belong to an initial value object and `U Velocity = 2.0 [m/s]` may belong to a boundary condition object. Both refer to the same definition of U velocity in the rules file.

13.1.3.9. Lists

Lists are used within the context of parameter values and are comma separated.

13.1.3.10. Parameter Values

All parameter values are initially handled as data of type String, and should first of all conform to the following definition of allowed String values:

13.1.3.10.1. String

- Any characters can be used in a parameter value.
- String values or other parameter type values are normally unquoted. If any quotes are present, they are considered part of the value. Leading and trailing spaces are ignored. Internal spaces in parameter values are preserved as given, although a given application is free to subsequently assume a space condensation rule when using the data.
- The characters $ and # have a special meaning. A string beginning with $ is evaluated as a Power Syntax variable, even if it occurs within a simple syntax statement. This is useful for performing more complex Power Syntax variable manipulation, and then using the result as part of a parameter or object definition. The appearance of # anywhere in the CCL file denotes the start of a comment.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The characters such as [,], {, and } are special only if used in conjunction with $. Following a $, such characters terminate the preceding Perl variable name.
- Other characters that might be special elsewhere in power syntax are escaped automatically when they appear in parameter values. For example, @, % and & are escaped automatically (that is, you do not need to precede these characters with the escape character \ when using them in parameter values).
- Parameter values can contain commas, but if the string is processed as a List or part of a List then the commas may be interpreted as separators (see below under List data types).

Some examples of valid parameter values using special characters in power syntax are:

```
Estimated cost = \$500
Title = Run\#1
Sys Command = "echo 'Starting up Stress solver' ; fred.exe &"
Pressure = $myArray[4]
Option = $myHash{"foo"}
Fuel = C${numberCatoms}H${numberHatoms}
```

Parameter values for data types other than String will additionally conform to one of the following definitions.

13.1.3.10.2. String List

A list of string items separated by commas. Items in a String List should *not* contain a comma unless contained between parentheses. One exception can be made if the String List to be is interpreted as a Real List (see below). Otherwise, each item in the String List follows the same rules as String data.

```
names = one, two, three, four
```

13.1.3.10.3. Integer

Sequence of digits containing no spaces or commas. If a real is specified when an integer is needed, the real is rounded to the nearest integer.

13.1.3.10.4. Integer List

List of integers, separated by commas.

13.1.3.10.5. Real

A single-precision real number that may be specified in integer, floating point, or scientific format, followed optionally by a dimension. Units use the same syntax as CEL.

Expressions are allowed to include commas inside function call argument lists.

Example usage:

```
a = 12.24
a = 1.224E01
a = 12.24 [m s^-1]
```

A real may also be specified as an expression such as:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
a = myvel^2 + b
a = max(b,2.0)
```

13.1.3.10.6. Real List

List of reals, comma separated. Note that all items in the list must have the same dimensions. Items that are expressions may include commas inside function call argument lists, and the enclosed commas will be ignored when the list is parsed into individual items. Example usage:

```
a = 1.0 [m/s], 2.0 [m/s], 3.0 [m/s], 2.0*myvel, 4.0 [cm/s]
```

The list syntax `5*2.0` to represent 5 entries of the value 2.0 is not supported within CCL and hence within CFD-Post.

13.1.3.10.7. Logical

Several forms are acceptable: YES, TRUE, `1` or `ON` are all equivalent; `NO` or FALSE or `0` or `OFF` are all equivalent; initial letter variants Y, T, N, F are accepted (`O` is not accepted for `On`/`Off`); all case variants are accepted. Logical strings are also case insensitive (`YeS`, `nO`).

13.1.3.10.8. Logical List

List of logicals, separated by commas.

13.1.3.11. Escape Character

The \ character to be used as an escape character, for example, to allow $ or # to be used in strings.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 14: CFX Expression Language (CEL)

CFX Expression Language (CEL) is an interpreted, declarative language that has been developed to enable CFX users to enhance their simulations without recourse to writing and linking separate external Fortran routines.

You can use CEL expressions anywhere a value is required for input in Ansys CFX.

CEL can be used to:

- Define material properties that depend on other variables.
- Specify complex boundary conditions.
- Add terms to the solved equations.

You can also monitor the value of an expression during the solution using monitor points.

Note:

CFX-Pre and CFD-Post evaluate CEL expressions with single (not double) precision.

Important:

There is some CEL that works elsewhere in Ansys CFX, but not in CFD-Post. Any expression created in CFX-Pre and used as a Design Exploration output parameter could potentially cause fatal errors during the Design Exploration run, so you should create all expressions for Design Exploration output parameters in CFD-Post.

This chapter describes:

14.1. CEL Fundamentals
14.2. CEL Operators, Constants, and Expressions
14.3. CEL Examples
14.4. CEL Technical Details

14.1. CEL Fundamentals

The following topics will be discussed:

- Values and Expressions (p. 454)
- CFX Expression Language Statements (p. 455)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

14.1.1. Values and Expressions

CEL can be used to generate both values and expressions. Values are dimensional (that is, with units) or dimensionless constants. The simplest type of definition is the dimensionless value, for example:

b = 3.743

You can also specify a value with units, for example:

g = 9.81 [m s^-2]

The dimensions of the quantities of interest for CFD calculations can be written in terms of mass, length, time, temperature and angle. The concept of units is fundamental to the behavior of values and expressions.

Values can be used directly, or they can be used as part of an expression. For example, you can use an expression to add two values together:

```
<Expr_1> = <Value_1> + <Value_2>
```

In this example, you may want to predefine <Value_1> and <Value_2>, but this is not required. However, in order to add two quantities together, *they must have the same dimension*; that is, it is meaningful to add a quantity in inches to one expressed in meters, but it is not meaningful to add one expressed in kilograms to one in square feet.

Expressions can also be functions of other (predefined) expressions:

```
<Expr_2> = <Expr_1> + <Value_3>
```

Units follow the conventions in the rest of CFX, in that a calculation has a set of solution units (by default, SI units), and that any quantity can be defined either in terms of the solution units, or any other set of units with the correct form.

An expression does not have its own units string, but if it references quantities that have dimensions, these will determine the resulting units for the expression. For example, if an expression depends inversely on the square of the x coordinate, then it has *implied dimensions* of length to the power -2.

14.1.1.1. Using Locators in Expressions

A CFX simulation has physics locators and mesh locators; physics locators are boundaries while mesh locators are regions. These two types of locator can occupy completely different spaces in a simulation; however, there is no requirement that locator names be unique between physics and mesh. This can lead to ambiguities when you use these names in expressions.

To avoid these ambiguities, Ansys CFX first checks to see if "@<locator>" is a physics name; if this is not found, the name is checked in the list of mesh names. Thus if "in1" is both the name of a physics locator and the name of a mesh locator, "@in1" is taken to indicate the physics locator.

Ansys CFX also has @REGION CEL syntax so that you can identify a named locator as being a mesh locator. Thus to identify the mesh locator in1, you would use the syntax:

```
@REGION:in1
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note that if <locator> does not appear as a physics name or a mesh name, the expression fails.

Note:

For any CEL expression that is defined for a locator (such as a domain, boundary, subdomain, injection region), the input coordinates (x, y, z, r, theta) are interpreted in the local frame but variables are normally interpreted in the global frame.

14.1.2. CFX Expression Language Statements

The CFX Expression Language is declarative. You declare the name and definition of the expression using expression language statements. The statements must conform to a predefined syntax that is similar to Fortran mathematical statements and to C statements for logical expressions.

The statement must consist of the following:

- a number, optionally with associated units. This defines a *constant*. Constants without units are termed *dimensionless*.
- for mathematical expressions, one or more references to mathematical constants, system variables, or existing user variables, separated by + (addition), - (subtraction), * (multiplication), / (division) and ^ (exponentiation), with optional grouping of these by parentheses. The syntax rules for these expressions are the same as those for conventional arithmetic.
- for logical expressions involving relational operators, one or more references to mathematical constants or results from mathematical expressions, separated by <= (is less than or equal to), < (is less than), == (is equal to), != (is not equal to), > (is greater than) and >= (is greater than or equal to) with optional grouping of these by parentheses.
- for logical expressions involving logical operators, one or more references to logical constants or results from relational operations separated by ! (negation), && (logical AND) and || (logical OR), with optional grouping by parentheses.

14.1.2.1. Use of Constants

Constants do not need to be defined prior to being used in an expression. For example, you could choose to evaluate the expression `x + 5 [m]`. Or, you could define a constant, `b = 5 [m]` and then create an expression `x + b`.

The logical constants are `false` and `true`. Results of logical expressions are either false or true, which are evaluated as `0` and `1` (corresponding to `false` and `true`, respectively) when a numerical representation is required.

The use of constants may be of benefit in generating complicated expressions or if you have several expressions that use the same constants.

14.1.2.2. Expression Syntax

All numbers are treated as real numbers.

The precedence of mathematical operators is as follows (from highest to lowest):

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- The power operator ^ as in x^y.
- The unary minus or negation operator - as in -x.
- Multiplication and division as in x*y/z.
- Addition and subtraction as in x+y-z.

The precedence of logical and relational operators is as follows (from highest to lowest):

- The negation operator ! as in !x.
- The relational operators involving less than or greater than (<=, <, > and >=) as in x >= y.
- The relational operator is equal to and is not equal to (== and !=) as in x != y.
- The logical AND operator (&&) as in x && y.
- The logical OR operator (||) as in x || y.

14.1.2.3. Multiple-Line Expressions

It is often useful, particularly with complex expressions, to use more than one line when creating your expression. CFX allows you to use multiple lines to generate an expression, provided each line is separated by an appropriate operator.

For example, you may have an equation, A + B/C, that consists of three complex terms, A, B, and C. In this case, you could use three lines to simplify creating the expression:

```
A +
B
/ C
```

Note that the operator may be used at the end of a line (A +) or at the beginning of a line (/ C). You do not need to enter the operator twice.

Once the expression has been created, it will appear in the Existing Definitions list box as if it were generated on a single line (A + B/C).

14.2. CEL Operators, Constants, and Expressions

The following topics are discussed:

- CEL Operators (p. 457)
- Conditional if Statement (p. 458)
- CEL Constants (p. 458)
- Using Expressions (p. 459)

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

14.2.1. CEL Operators

CFX provides a range of mathematical, logical and operational operators as built-in functions to help you create complex expressions using the **Expression** details view.

Table 14.1: CEL Operators

Operator	First Operand's Dimensions [x]	Second Operand's Dimensions [y]	Operands' Values (Approx)[a]	Result's Dimensions
-x	Any		Any	[x]
x+y	Any	[x]	Any	[x]
x-y	Any	[x]	Any	[x]
x*y	Any	Any	Any	[x]*[y]
x/y	Any	Any	$y \neq 0$	[x]/[y]
x^y (if y is a simple, constant, integer expression)	Any	Dimensionless	Any[b]	[x]^y
x^y (if y is any simple, constant, expression)	Any	Dimensionless	x > 0	[x]^y
x^y (if y is not simple and constant)	Dimensionless	Dimensionless	x > 0	Dimensionless
!x	Dimensionless		false or true	Dimensionless
x <= y	Any	[x]	false or true	Dimensionless
x < y	Any	[x]	false or true	Dimensionless
x > y	Any	[x]	false or true	Dimensionless
x >= y	Any	[x]	false or true	Dimensionless
x == y	Any	[x]	false or true	Dimensionless
x != y	Any	[x]	false or true	Dimensionless
x && y	Dimensionless	Dimensionless	false or true	Dimensionless

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Operator	First Operand's Dimensions [x]	Second Operand's Dimensions [y]	Operands' Values (Approx)[a]	Result's Dimensions
x || y	Dimensionless	Dimensionless	false or true	Dimensionless

[a] The logical constants true and false are represented by "1" and "0" when a numerical representation is required.

[b] For y < 0, x must be non-zero.

14.2.2. Conditional if Statement

CEL supports the conditional if statement using the following syntax:

```
if( cond_expr, true_expr, false_expr )
```

where:

- `cond_expr`: is the logical expression used as the conditional test
- `true_expr`: is the mathematical expression used to determine the result if the conditional test is `true`.
- `false_expr` : is the mathematical expression used to determine the result if the conditional test is `false`.

Note:

The expressions `true_expr` and `false_expr` are always evaluated independent of whether the evaluation of `cond_expr` is `true` or `false`. As a consequence, a conditional statement cannot be used to avoid division by zero as in `if( x>0, 1/x, 1.0)`. In this case, when `x=0.0`, a division by zero will still occur because the expression `1/x` is evaluated independent of whether `x>0` is satisfied or not.

14.2.3. CEL Constants

Right-click in the **Expression** details view to access the following useful constants when developing expressions:

Table 14.2: CEL Constants

Constant	Units	Description
R	J K^-1 mol^-1	Universal Gas Constant: 8.314472
avogadro	mol^-1	6.02214199E+23
boltzmann	J K^-1	1.3806503E-23
clight	m s^-1	2.99792458E+08
e	Dimensionless	Constant: 2.7182818284590452
echarge	A s	Constant: 1.60217653E-19

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Constant	Units	Description
epspermo		1./(clight*clight*mupermo)
g	m s^-2	Acceleration due to gravity: 9.8066502
mupermo	N A^-2	4*pi*1.E-07
pi	Dimensionless	Constant: 3.1415926535897931
planck	J s	6.62606876E-34
stefan	W m^-2 K^-4	5.670400E-08

14.2.4. Using Expressions

The interaction with CEL consists of two phases:

- a definition phase, and,
- a use phase.

The definition phase consists of creating a set of values and expressions of valid syntax. The purpose of the **Expression** details view is to help you to do this.

14.2.4.1. Use of Offset Temperature

When using temperature values in expressions, it is generally safer to use units of [K] only. When units are used that have an offset (for example, [C]), they are converted internally to [K]. For terms that have the dimensions of temperature, any unit conversion will include the offset between temperature scales. However, in all other cases the offset is ignored because this is usually the most appropriate behavior. You should therefore take care when specifying an expression involving non-unit powers of temperature. For example, each of the expressions below is equivalent:

```
Temperature = 30 [C]
Temperature = 303.15 [K]
Temperature = 0 [C] + 30 [K]
Temperature = 273.15 [K] + 30 [K]
```

These are only equivalent because all units are to the power of unity and units other than [K] appear no more than once in each expression. The following expression will not produce the expected result:

```
Temperature = 0 [C] + 30 [C]
```

This is equivalent to `576.30 [K]` because each value is converted to [K] and then summed. The two expressions below are equivalent (as expected) because the offset in scales is ignored for units that are not temperatures:

```
Specific Heat = 4200 [J kg^-1 C^-1]
Specific Heat = 4200 [J kg^-1 K^-1]
```

14.3. CEL Examples

The following examples are included in this section:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Example: Reynolds Number Dependent Viscosity (p. 460)
- Example: Feedback to Control Inlet Temperature (p. 460)

14.3.1. Example: Reynolds Number Dependent Viscosity

In this example it is assumed that some of the fluid properties, including the dynamic viscosity, are not known. However the Reynolds number, inlet velocity and a length scale are known. The flow is compressible and therefore the density is variable.

Given this information it is possible to calculate the fluid dynamic viscosity based on the Reynolds number. The Reynolds number is given by:

$$Re=\frac{\rho U L}{\mu}$$

where ρ is density, U a velocity scale, L a length scale and μ the dynamic viscosity. The velocity scale is taken as the inlet velocity, the length scale as the inlet width and the density is calculated as the average density over the inlet area.

The LIBRARY section of the CCL (CFX Command Language) file appears as follows:

```
LIBRARY :
   CEL :
      EXPRESSIONS :
         Re = 4.29E6 [  ]
         Vel = 60 [m s^-1]
         L=1.044[m]
         Visc=areaAve(density)@in*Vel*L/Re
      END
   END
   MATERIAL : Air Ideal Gas
      Option = Pure Substance
      PROPERTIES :
         Option = Ideal Gas
         Molar Mass = 2.896E1 [kg kmol^-1]
         Dynamic Viscosity = Visc
         Specific Heat Capacity = 1.E3 [J kg^-1 K^-1]
         Thermal Conductivity = 2.52E-2 [W m^-1 K^-1]
      END
   END
END
```

This shows that four CEL expressions have been created. The first three expressions define constant values that are used in the `Visc` expression. The `Visc` expression calculates the dynamic viscosity based on the equation for Reynolds number given above. Within the expression the function `areaAve(density)@in` is used to evaluate the average density at the inlet.

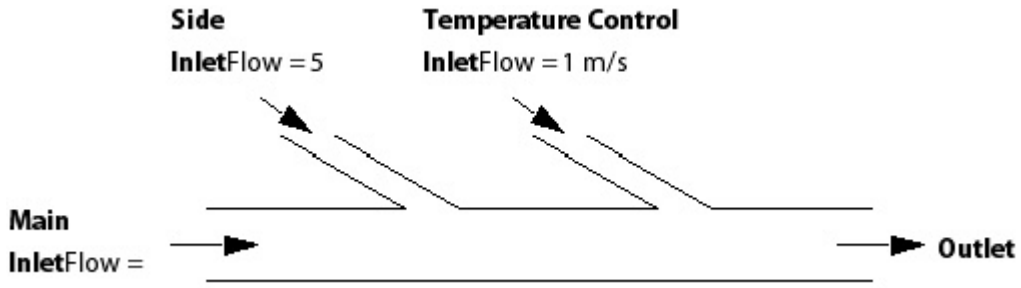
The `Visc` expression can now be used to replace the value of `Dynamic Viscosity` in the `MATERIAL` > `PROPERTIES` section.

14.3.2. Example: Feedback to Control Inlet Temperature

In this example a feedback loop is used to control the outlet temperature by varying the temperature at an inlet. To illustrate the example consider the geometry shown below:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 14.1: Temperature Feedback Loop



Fluid from a main and a side inlet enter at temperatures of 275 K and 375 K respectively. The temperature of the fluid entering from the third inlet depends on the outlet temperature. When the outlet temperature is greater than 325 K, the fluid from the third inlet is set to 275 K. When the outlet temperature is less than 325 K, the fluid from the third inlet is set to 375 K. In addition an expression is used to set the dynamic viscosity to be a linear function of temperature.

The LIBRARY section of the `.ccl` (CFX Command Language) file appears as follows. Note that the "\" character indicates a line continuation in CCL.

```
LIBRARY:
   MATERIAL: Water at STP Modified
      Option = Pure Substance
      PROPERTIES:
         Option = General Fluid
         Density = 9.999E2 [kg m^-3]
         Dynamic Viscosity = VisT
         Specific Heat Capacity = 4.21E3 [J kg^-1 K^-1]
         Thermal Conductivity = 5.69E-1 [W m^-1 K^-1]
      END # PROPERTIES
   END # MATERIAL Water at STP Modified
   CEL:
      EXPRESSIONS:
         Tupper = 375.0 [ K ]  # Upper temp.
         Tlower = 275.0 [ K ]  # Lower temp.
         Visupper = 0.000545 [ N s m^-2 ]  # Vis. at Tupper
         Vislower = 0.0018 [ N s m^-2 ]  # Vis. at Tlower
         VisT = Vislower+(Visupper-Vislower)*(T-Tlower)/ \
            (Tupper-Tlower)
         # Vis.-Temp. relationship
         Tm=(Tupper+Tlower)/2
         Tout=areaAve(Water at STP Modified.T)@outlet
         Tcontrol=Tlower*step((Tout-Tm)/1[K]) \
            +Tupper*step((Tm-Tout)/1[K])
      END # EXPRESSIONS
   END # CEL
END # LIBRARY
```

The first four expressions, `Tupper`, `Tlower`, `Visupper` and `Vislower` are simply constant values to define temperature and viscosity values. The expression `VisT` produces a linear function for the dynamic viscosity taking a value of `Visupper` at `Tupper` and a value of `Vislower` at `Tlower`. The expression `Tm` sets the desired value of the outlet temperature. In this case it is set to a mean value of the two inlet temperatures.

`Tout` calculates the outlet temperature using the `areaAve` function.

Finally the expression `Tcontrol` is used to set the temperature of the third inlet. Two step functions are used so that the temperature is equal to `Tlower` when `Tout-Tm` is positive (that is, the outlet temperature is greater than `Tm`), and is equal to `Tupper` when `Tout-Tm` is positive.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

14.3.3. Examples: Using Expressions in CFD-Post

The first example is a single-valued expression that calculates the pressure drop through a pipe. The names of inlet and outlet boundaries are "inlet" and "outlet".

Create a new expression named "dp":

```
dp = massFlowAve(Pressure)@inlet – massFlowAve(Pressure)@outlet
```

When you click **Apply**, the value is shown below the editor.

Tip:

Alternatively, type the expression in a table cell and prefix with '=' sign. The cell displays the result when you click outside of the cell.

The second example is a variable expression that plots the pressure coefficient variation on a surface or a line:

1. Click the **Expressions** tab, then right-click in the **Expressions** area and select **New**.
2. Create these three expressions:

   ```
   RefPressure = 100000 [Pa]
   dynHead = 0.5 * areaAve(Density)@inlet * areaAve(Velocity)@inlet^2
   cpExp = (Pressure - RefPressure)/dynHead
   ```

3. Click the **Variables** tab, then right-click and select **New**.
4. Create a user variable defined by cpExp.
5. Select **Insert** > **Location** > **Line** and use the details view to position the line in the simulation.

   From the details view **Color** tab, plot the user variable on a surface or a line (just as you would with any other variable).

14.4. CEL Technical Details

CEL is a byte code compiled language. Compiled languages, such as Fortran, rely on a translation program to convert them into the native machine language of the host platform. Interpreted languages are of two types: the fully interpreted languages such as the UNIX C shell, and the byte code compiled languages such as CEL. With byte codes, host machines are loaded with a client program (written in a compiled language and compiled for that machine architecture) that interprets the byte stream. The advantage of the byte code is that they can be the same on all host platforms, obviating the need for platform dependent codes.

Because the byte codes are interpreted, there is no need to re-link executable programs to perform a different calculation. Furthermore, many of the problems encountered by writing and linking in separate routines, for instance in C or Fortran, are averted, and the time taken to set up and debug complicated problems reduced considerably.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The link between CEL and the CFX-Solver is accomplished through an internal program called *genicode*. Genicode generates intermediate code from your CEL definitions and writes to a file that is then interpreted by the CFX-Solver during the solution process.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 15: Functions in Ansys CFX

This chapter describes predefined functions in Ansys CFX:

15.1. CEL Mathematical Functions

15.2. Quantitative CEL Functions in Ansys CFX

15.3. Functions Involving Coordinates

15.4. CEL Functions with Multiphase Flow

15.5. Quantitative Function List

15.1. CEL Mathematical Functions

The following mathematical functions are available for use with all CEL expressions.

Note:

In the **Function** column in the table below, [a] denotes any dimension of the first operand.

Table 15.1: Standard Mathematical CEL Functions

Function	Operand's Values	Result's Dimensions
abs([a])	Any	[a]
acos([ ])	$-1 \le x \le 1$	Radians
asin([ ])	$-1 \le x \le 1$	Radians
atan([ ])[a]	Any	Radians
atan2([a], [a])[a]	Any	Radians
besselJ([ ], [ ])[b]	$0 \le n$	Dimensionless
besselY([ ], [ ])[b]	$0 \le n$	Dimensionless
cos([radians])	Any	Dimensionless
cosh([ ])	Any	Dimensionless
exp([ ])	Any	Dimensionless
int([ ])[c]	Dimensionless	Dimensionless
loge([ ])[d]	$0 < x$	Dimensionless
log10([ ])[e]	$0 < x$	Dimensionless
min([a], [a])	Any	[a]
max([a], [a])	Any	[a]

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function	Operand's Values	Result's Dimensions
mod([a], [a])[f]	Any	[a]
nint([ ])[g]	Dimensionless	Dimensionless
sin([radians])	Any	Dimensionless
sinh([ ])	Any	Dimensionless
sqrt([a])	$0 \le x$	[a]^0.5
step([ ]) [h]	Any	Dimensionless
tan([radians])[i]	Any	Dimensionless
tanh([ ])	Any	Dimensionless

[a] atan does not determine the quadrant of the result, but atan2 does.

[b] The value of the first dimensionless operand n, also referred to as the order of the Bessel function, must be an integer (n=0, 1, 2, ...). The second argument is a dimensionless real number.

[c] The int() function truncates the dimensionless argument to its integer part.

Examples:

int(1) = 1

int(2.5) = 2

int(-3.1) = -3

int(-4.8) = -4

The int() function requires a dimensionless argument but will not report an error if the argument of the function has a dimension of radians or degrees.

[d] ln(x) is valid as an alias for loge(x)

[e] log(x) is valid as an alias for log10(x)

[f] mod(x, y) returns the remainder on dividing x by y; the function is not defined for y = 0.

[g] The nint function requires a dimensionless argument and is defined as:

int(x + 0.5) if x >= 0

int(x - 0.5) if x < 0

See the implementation of int( ) function in the table above.

Examples:

nint(2.6) = 3

nint(2.5) = 3

nint(2.4) = 2

nint(1) = 1

nint(-1) = -1

nint(-2.4) = -2

nint(-2.5) = -3

nint(-2.6) = -3

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Note that the nint() function will not report an error if the argument of the function has a dimension of radians or degrees.

[h] step(x) is 0 for negative x, 1 for positive x and 0.5 for x=0. x must be dimensionless.

[i] tan(x) is undefined for x=nπ/2, where n=1, 3, 5, ...

15.2. Quantitative CEL Functions in Ansys CFX

CEL expressions can incorporate specialized functions that are useful in CFD calculations. All CEL functions are described in Quantitative Function List (p. 472). For a description of the full CFX Expression Language, see CFX Expression Language (CEL) (p. 453).

Important:

You must use consistent units when adding, subtracting, or comparing values.

There are some differences between CEL functions in CFX-Pre and CFX-Solver and those in CFD-Post. For details, see below.

The syntax used for calling these functions when used within CEL expressions is:

```
[<Phase_Name>.][<Component_Name>.]<Function>([<Operand>])@<Location>
```

where:

- Terms enclosed in square brackets [ ] are optional and terms in angle brackets < > should be replaced with the required entry.
- <Phase_Name>: specifies a valid name of a phase. The phase can be fluid, particle, solid, fluid pair, or polydispersed fluid. For multi-phase cases in CFX-Pre, if the phase name is not specified in the <Operand>, then the phase name associated with the domain, subdomain, domain boundary, initialization or function in which the operand is being evaluated will be used. For multi-phase cases in CFX-Pre, a discussion of the handling of the phase name when it is not used to qualify (prepended to) <Function> and/or <Operand> can be found in CEL Functions with Multiphase Flow (p. 471). For multi-phase cases in CFD-Post, if the phase name is not specified then the bulk quantity (if available for the CFX-Solver Results file) is used.
- <Component_Name>: specifies a valid name of a component material, size group, or reaction. For multi-component cases in CFX-Pre, if the component name is not specified in the <Operand>, then the component name associated with the domain, subdomain, domain boundary, initialization or function in which the operand is being evaluated is used.
- <Function>: specifies the CEL function to evaluate. See Quantitative Function List (p. 472). The function can be further qualified by appending _Coordinate_Direction. In CFX-Pre, if the coordinate frame is not specified (in _Coordinate_Direction) then the function will use the coordinate frame associated with the object (such as for a material, domain, subdomain, domain boundary, source point, monitor point, initialization, reference location or spark ignition object) in which it is being invoked.
- <Coordinate_Direction>: specifies a particular coordinate direction. The syntax of the coordinate direction is [x|y|z][_<Coordinate_Frame>] where the coordinate frame can be the global coordinate frame or any user defined coordinate frame. In CFD-Post, if the coordinate frame is not

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

specified then the global frame is used. See Coordinate Frame Command in the *CFD-Post User's Guide*, for discussion of creating a coordinate frame in CFD-Post.

Note:

For a vector CEL function, you can only specify a stationary local coordinate frame in `<Coordinate_Direction>`.

- `<Operand>`: specifies the argument of the function (if required). The operand can be either a valid mathematical CEL expression (only in CFD-Post) or specified using the following general variable syntax:

```
[<Phase_Name>.][<Component_Name>.]<Variable_Name>[.<Variable_Operator>][.Difference]
```

  In CFX-Pre the operand cannot be a CEL expression or any operand qualified by `<Variable_Operator>`. However, you can create an Additional Variable based on any expression and then use the Additional Variable as the operand. In this case, it is important to qualify component variables with their respective component names; any unqualified variables are assumed to be in the context of the mixture rather than any particular mixture component.

  The operand always uses the conservative values unless the `Boundcon` variable operator is specified (for details, see Data Acquisition Routines in the *CFX-Solver Modeling Guide*). For primitive or composite mesh regions, conservative values will be used even if the `Boundcon` variable operator is specified.

  The operand must be valid for the physical models being used over the entire location. For example, if the location spans fluid and solid domains, then the operand cannot be `Pressure`.

  For some functions, the operand must be left blank as in `area()@Inlet`.

  In CFD-Post, difference variables created during case comparison are appended by `.Difference`.

- `<Variable_Name>`: specifies the base name of the variable. You can use the short or long form for variable names. The variable name can be further qualified by appending `_<Coordinate_Direction>`. This is useful for specifying a particular component of a vector or tensor, for example `Velocity_y_myLocalFrame`. If the variable name corresponds to that of a component of a vector or a tensor, and a coordinate frame is not prescribed (as part of the coordinate direction), then the global coordinate frame is used. An exception applies for the position vector `x`, `y`, `z` (or `r`, `theta`, `z`) components, which are always local, see Functions Involving Coordinates (p. 470).

  Note:

  – For a vector variable, you can only specify a stationary local coordinate frame in `<Coordinate_Direction>`.

    For a tensor variable, the only coordinate frame supported is the global coordinate frame.

  – A local coordinate frame cannot be appended to a variable that is used as an argument to a quantitative CEL function, unless that function is `RBState` for rigid bodies. A

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

workaround that involves the use of an Additional Variable is given in [Monitor Name]: Coordinate Frame.

- `<Variable_Operator>` specifies the name of the variable operator. The variable operators are:

Long Name	Short Name
Magnitude	magnitude
Gradient	grad
Curl	curl
Laplacian	laplacian
Time Derivative	dt
Transient Minimum	Trnmin
Transient Maximum	Trnmax
Transient Std Deviation	Trnsdv
Transient RMS	Trnrms
Transient Average	Trnavg
Boundcon	hybrid

Note:

– Some variable operators might not be available in CFD-Post. Other variable operators might require the existence of certain derived variables in the solution results before they can be used in CFD-Post.

– The `Magnitude` and `Curl` operators may only be applied to vector variables. All other operators may be applied to scalar variables and vector components.

- `<Location>`: specifies the location over which the function is to be applied. The syntax of location is:

```
[Case:<Case_Name>.][REGION:]<Location_Name>
```

The case syntax `[Case:<Case_Name>.]` is only available in CFD-Post. When multiple cases are loaded, this syntax is used to narrow down the locator to the desired case.

In CFX-Pre `[<Location_Name>]` must be a domain boundary, domain, subdomain, injection region, or primitive or composite mesh region. If the location name of a mesh region is the same as the name of a named boundary, domain, subdomain, or injection region, then the mesh location name must be prepended by `REGION:`. For primitive or composite mesh regions, conservative values will be used even if the name of the mesh region is the same as that of a named boundary.

In CFD-Post `[<Location_Name>]` can be any loaded or user-defined location (for example, a point, domain boundary, plane, mesh region, or injection region). The syntax `REGION:<Region Name>` can also be used in CFD-Post to refer to any mesh region. If a mesh region is present with the same name as, for example, a domain boundary, then the mesh region is imported into CFD-Post with a

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Region suffix. For example, if there is both a domain boundary and a mesh region called in1 in the CFX-Solver Results file, then in CFD-Post the mesh region will appear in CFD-Post as in1 Region. The syntax in1 will refer to the domain boundary, and either of in1 Region or REGION:in1 can be used to refer to the mesh region as desired.

Note:

You cannot use a composite region that consists of a mixture of regions from more than one of the following categories: 2D regions, 3D regions, injection regions.

Table 15.2: Examples of the Calling Syntax for an Expression

areaAve(p)@Inlet	This results in the area-weighted average of pressure on the boundary named Inlet.
area()@REGION:myCompositeMeshRegion	This results in the area of a 2D mesh region named myCompositeMeshRegion.
areaAve(Pressure - 10000 [Pa])@outlet	This syntax is appropriate only for CFD-Post.
area_x()@inlet	
Water at RTP.force_z()@Default	
area()@CASE:newcase.myplane	This syntax is appropriate only for CFD-Post and is used when multiple files are loaded. It follows the general form function()@CASE:case name.location. This example results in the area of the part of myplane that is located within the case newcase.
probe(Pressure)@CASE:1.Point 1 probe(Pressure)@CASE:2.Point 1	This syntax is used only for CFD-Post, when two cases are loaded in the comparison mode. This example results in the difference in pressure between cases 1 and 2, where the pressure for Case 1 and the pressure for Case 2 are evaluated at the same point. The general syntax, function()@CASE:[1|2].location, is used when performing file comparisons.

15.3. Functions Involving Coordinates

The CEL variables x, y, z, r and theta, representing the local coordinates, cannot be used as the variable. However, the variables xGlobal, yGlobal and zGlobal can be used. For example, the following is a valid expression definition:

```
z*areaAve(xGlobal)@inlet
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.4. CEL Functions with Multiphase Flow

Note:

These functions are available in CFX-Pre and CFX-Solver without restrictions, and in CFD-Post with the restriction that you cannot use short names.

If the function is fluid-specific, various behaviors are possible depending on the function type:

- For `massFlow` and `massFlowAve`, if the phase name is not specified for the function, then the bulk mass flows will be used. See cases 1 to 7 in the table below.
- For other fluid-specific functions:
  - if a fluid-specific operand is specified and no fluid is specified for the function, then the fluid specified for the operand will be assumed for the function as well. See case 8 in the table below.
  - if the function is specified and no fluid is specified for the operand, then the fluid specified for the function will be assumed for the operand as well. See cases 7 and 9 in the table below.
- If both the function or operand are fluid-specific, and a phase name is not given for either, the solver will stop with an error. See case 10 in the table below.

Table 15.3: CEL Multiphase Examples

Case	CEL Function - Multiphase	Behavior
1	`massFlow()@inlet`	Bulk mass flow rate through inlet
2	`Air.massFlow()@inlet`	Air mass flow rate through inlet
3	`massFlowAve(Pressure)@inlet`	Bulk mass flow averaged pressure on inlet
4	`Air.massFlowAve(Pressure)@inlet`	Air mass flow averaged pressure on inlet
5	`massFlowAve(Air.Volume Fraction)@inlet`	Bulk mass flow averaged air volume fraction on inlet
6	`Air.massFlowAve(Air.Volume Fraction)@inlet`	Air mass flow averaged air volume fraction on inlet
7	`Air.massFlowAve(Volume Fraction)@inlet`	Same as `Air.massFlowAve(Air.Volume Fraction)@ inlet`
8	`massInt(Air.Volume Fraction)@domain1`	Same as `Air.massInt(Air.Volume Fraction)@ domain1`
9	`Air.massInt(Volume Fraction)@domain1`	Same as `Air.massInt(Air.Volume Fraction)@ domain1`
10	`massFlowAve(Volume Fraction)@inlet`	Error because no fluid specified

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5. Quantitative Function List

The available quantitative functions are outlined in the sections that follow.

In the table that follows, <Expression> applies to CFD-Post only. CFX-Pre and CFX-Solver can only accept variables as arguments to quantitative functions. Note that for CFX-Pre and CFX-Solver, an Additional Variable can be used to pass an expression indirectly into a quantitative function.

The behavior of the functions in the table below depends on the type of <Location>. Typically:

- On domains and subdomains, the functions use vertex (node) values for the operand.
- On a boundary, the functions use conservative values for the operand unless this is overridden by the Boundcon variable operator in CFX-Pre.
- On user locations in CFD-Post, the functions use values interpolated from nodal values.

Table 15.4: CEL Functions in CFX-Pre/CFX-Solver and in CFD-Post

Function Name and Syntax <required> [<optional>]	Operation	Availability
area( )	Area of a 2D region, such as boundary, interface, or injection region. For an injection region, only the cumulative geometry of the injection positions is considered, not the location of the entire injection region. Supports "@<Location>". See area (p. 479).	All
area_x[_<Coord Frame>]( ) area_y[_<Coord Frame>]( ) area_z[_<Coord Frame>]( )	The (signed) component of the normal area vector in the local X, Y, or Z direction. The normal area vectors are always directed out of the domain, therefore you may obtain positive or negative areas depending on the orientation of your domain and the boundary you are operating on. The area of a closed surface will always be zero. Supports "@<Location>".	All[a]
areaAve(<Variable|Expression>)	Area-weighted average of <Variable|Expression> on a 2D region, such as boundary, interface, or injection region. For an injection region, only the cumulative geometry of the injection positions is considered,	All

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	Operation	Availability
	not the location of the entire injection region. Supports "@<Location>". See areaAve (p. 480).	
areaAve_x[_<Coord Frame>]( ) areaAve_y[_<Coord Frame>]( ) areaAve_z[_<Coord Frame>]( )	The (signed) component of the normal area vector *weighted average* in the local X, Y or Z direction. The normal area vectors are always directed out of the domain, therefore you may obtain positive or negative areas depending on the orientation of your domain and the boundary you are operating on. The area of a closed surface will always be zero. Supports "@<Location>".	CFD-Post
areaInt(<Variable|Expression>)	Area-weighted integral of <Variable|Expression> on a 2D region, such as boundary, interface, or injection region. For an injection region, only the cumulative geometry of the injection positions is considered, not the location of the entire injection region. The `areaInt` function projects the location onto a plane normal to the specified direction (if the direction is not set to `None`) and then performs the calculation on the projected location (the direction specification can also be `None`). The direction of the normal vectors for the location is important and will cancel out for surfaces such as closed surfaces. Supports "@<Location>". See areaInt (p. 481).	All
areaInt_x[_<Coord Frame>]( ) areaInt_y[_<Coord Frame>]( )	The (signed) component of the normal area vector *weighted integral* in the local X, Y, or Z	All[a]

Function Name and Syntax <required> [<optional>]	Operation	Availability
areaInt_z[_<Coord Frame>]( )	direction. The normal area vectors are always directed out of the domain, therefore you may obtain positive or negative areas depending on the orientation of your domain and the boundary you are operating on. The area of a closed surface will always be zero. Supports "@<Location>".	
ave(<Variable|Expression>)	Arithmetic average of <Variable|Expression> over nodes within a domain, subdomain, or injection region. For an injection region, an average value is calculated at each position and then these are averaged over all positions. Supports "@<Location>". See ave (p. 482).	All
count( )	Counts the number of evaluation points (for example, nodes) on the named region. For an injection region, this function returns the number of unique vertices affected by the injection region positions. See count (p. 483).	All
countTrue(<Variable|Expression>)	Counts the number of nodes at which the Variable or logical expression evaluates to true. Supports "@<Location>". See countTrue (p. 484).	All
force( )	The magnitude of the force vector on a boundary. Supports "[<Phase>.]" and "@<Location>". See force (p. 484).	All

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	Operation	Availability
forceNorm [_<Axis>[_<Coord Frame>]]( )	The length of the normalized force on a curve in the specified direction. Supports "[<Phase>.]" and "@<Location>". See forceNorm (p. 486).	CFD-Post
force_x[_<Coord Frame>]( ) force_y[_<Coord Frame>]( ) force_z[_<Coord Frame>]( )	The (signed) component of the force vector in the local X, Y, or Z direction. Supports "[<Phase>.]" and "@<Location>".	All[a]
inside()	Similar to the `subdomain` variable, but allows a specific 2D, 3D, or injection region location to be given. Supports "@<Location>". See inside (p. 486).	CFX-Pre, CFX-Solver
length()	Length of a curve. Supports "@<Location>". See length (p. 487).	CFD-Post
lengthAve(<Variable|Expression>)	Length-weighted average. Supports "@<Location>". See lengthAve (p. 488).	CFD-Post
lengthInt(<Variable|Expression>)	Length-weighted integration. Supports "@<Location>". See lengthInt (p. 488).	CFD-Post
lineCloudAve(<Variable>)@<Location>	Arithmetic average of <Variable> over a line cloud. An average value is calculated for each line in a field of lines. The result is a field of values that have been averaged in the direction of the lines. `<Location>` is any user line cloud. See lineCloudAve (p. 488).	CFX-Pre, CFX-Solver

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	**Operation**	**Availability**
mass()	The total mass within a domain or subdomain. This is fluid-dependent. Supports "@<Location>". See mass (p. 489).	CFX-Pre, CFX-Solver
massAve(<Variable|Expression>)	Mass-weighted average of <Variable|Expression> on a domain or subdomain. Supports "@<Location>". See massAve (p. 489).	CFX-Pre, CFX-Solver
massFlow()	Mass flow through a boundary, interface, or injection region. For an injection region, the cumulative mass flow from all positions is considered. Supports "[<Phase>.]" and "@<Location>". See massFlow (p. 490).	All
massFlowAve(<Variable|Expression>)	Mass flow weighted average of <Variable|Expression> on a boundary, interface, or injection region. For an injection region, the mass flow at each affected vertex is used. Supports "[<Phase>.]" and "@<Location>". See massFlowAve (p. 491).	All
massFlowAveAbs(<Variable|Expression>)	Absolute mass flow weighted average of <Variable|Expression> on a boundary, interface, or injection region. For an injection region, the mass flow at each affected vertex is used. Supports "[<Phase>.]" and "@<Location>". See massFlowAveAbs (p. 492).	All
massFlowInt(<Variable|Expression>)	Mass flow weighted integration of <Variable|Expression> on a boundary, interface, or injection	All

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	Operation	Availability
	region. For an injection region, the mass flow at each affected vertex is used. Supports "[<Phase>.]" and "@<Location>". See massFlowInt (p. 494).	
massInt(<Variable|Expression>)	The mass-weighted integration of <Variable|Expression> within a domain or subdomain. Supports "@<Location>". See massInt (p. 494).	CFX-Pre, CFX-Solver
maxVal(<Variable|Expression>)	Maximum value of <Variable|Expression> within a domain or subdomain. For an injection region, the maximum over the affected vertices is used. Supports "@<Location>". See maxVal (p. 494).	All
minVal(<Variable|Expression>)	Minimum value of <Variable|Expression> within a domain or subdomain. For an injection region, the minimum over the affected vertices is used. Supports "@<Location>". See minVal (p. 495).	All
probe(<Variable|Expression>)	Returns the value of the specified variable or expression on the specified Point object. Supports "@<Location>". See probe (p. 495).	All
rbstate(<rbvar>[<axis>])	Returns the position/velocity/acceleration or orientation/angular velocity/angular acceleration (or axis components of these) of a rigid body object or immersed solid that is governed by a rigid body solution. These quantities	CFX-Pre, CFX-Solver

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	Operation	Availability
	are with respect to the global coordinate frame. See rbstate (p. 496).	
rmsAve(<Variable|Expression>)	RMS average of <Variable|Expression> on a boundary, interface, injection region, or within a 3D domain or subdomain. For an injection region, an average value is calculated at each position, and then these are RMS-averaged over all positions. Supports "@<Location>". See rmsAve (p. 497).	CFX-Pre, CFX-Solver
sum(<Variable|Expression>)	Sum of <Variable|Expression> over all nodes of a 2D or 3D region (such as a boundary, interface, injection region, domain, or subdomain). For an injection region, an average value is calculated at each position, and then these are summed over all positions. Supports "@<Location>". See sum (p. 497).	All
torque( )	Magnitude of the torque vector on a boundary. Supports "[<Phase>.]" and "@<Location>". See torque (p. 498).	All
torque_x[_<Coord Frame>]() torque_y[_<Coord Frame>]() torque_z[_<Coord Frame>]()	The (signed) components of the torque vector about the local x, y, or z coordinate axis. Supports "[<Phase>.]" and "@<Location>".	All[a]
volume( )	The total volume of a domain or subdomain. Supports "@<Location>". See volume (p. 498).	All

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Function Name and Syntax <required> [<optional>]	Operation	Availability
volumeAve(<Variable|Expression>)	Volume-weighted average of <Variable|Expression> on a domain. Supports "@<Location>". See volumeAve (p. 499).	All
volumeInt(<Variable|Expression>)	Volume-weighted integration of <Variable|Expression> within a domain or subdomain. Supports "@<Location>". See volumeInt (p. 499).	All

[a] See the definitions for <Function> and <Coordinate_Direction> in Quantitative CEL Functions in Ansys CFX (p. 467).

15.5.1. area

The area function is used to calculate the area of a 2D locator.

```
area[_<Axis>[_<Coord Frame>] ]()@<Location>
```

where:

- <Axis> is x, y, or z.
- <Coord Frame> is the coordinate frame. See the definitions for <Function> and <Coordinate_Direction> in Quantitative CEL Functions in Ansys CFX (p. 467).
- <Location> is any 2D region (such as a boundary or interface) or injection region. For an injection region, only the cumulative geometry of the injection positions is considered, not the location of the entire injection region.

An error is raised if the location specified is not a 2D object. If an axis is not specified, the total area of the location is calculated.

area()@Isosurface1 calculates the total area of the location, and Isosurface1.area_y()@Isosurface1 calculates the projected area of Isosurface1 onto a plane normal to the Y axis.

15.5.1.1. Tools > Command Editor Example

```
>calculate area, <Location>, [<Axis>]
```

The specification of an axis is optional. If an axis is not specified, the value held in the object will be used. To calculate the total area of the location, the axis specification should be left blank (that is, type a comma after the location specification).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

`>calculate area, myplane` calculates the area of the locator `myplane` projected onto a plane normal to the axis specification in the `CALCULATOR` object.

`>calculate area, myplane,` calculates the area of the locator `myplane`. Note that adding the comma after `myplane` removes the axis specification.

15.5.1.2. Tools > Function Calculator Example

The following example will calculate the total area of the locator `Plane1`:

Function: area, **Location:** `Plane1`.

15.5.2. areaAve

The `areaAve` function calculates the area-weighted average of a variable or expression on a 2D location. The area-weighted average of a variable or expression is the average value of the variable or expression on a location with the mesh element sizes taken into account. Without the area weighting function, the average of all the nodal variable or expression values would be biased towards values in regions of high mesh density.

```
areaAve[_<Axis>[_<Coord Frame>] ](<Variable|Expression>)@<Location>
```

where:

- `<Axis>` is x, y, or z.
- `<Coord Frame>` is available in CFD-Post only.
- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any 2D region (such as a boundary or interface) or injection region. For an injection region, only the cumulative geometry of the injection positions is considered, not the location of the entire injection region. An error is raised if the location specified is not a 2D object.

To calculate the pressure coefficient C_p, use:

```
(Pressure - 1[bar])/(0.5*Density*(areaAve(Velocity)@inlet)^2)
```

You can create an expression using this, and then create a user variable using the expression. The user variable can then be plotted on objects like any other variable.

Note:

Projected areaAve (for example, areaAve_x) works as expected only for surfaces that do not fold in the selected direction. In extreme case, if the surface is fully closed, the projected average will result in a randomly large number, as the projected area will be zero.

15.5.2.1. Tools > Command Editor Example

```
>calculate areaAve, <Expression>, <Location>, <Axis>
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.2.2. Tools > Function Calculator Examples

- This example will calculate the average magnitude of `Velocity` on `outlet`.

  Function: `areaAve`, **Location:** `outlet`, **Variable:** `Velocity`.

  Note that flow direction is not considered because the magnitude of a vector quantity at each node is calculated.

- You can use the scalar components of `Velocity` (such as `Velocity u`) to include a directional sign. This example will calculate the area-weighted average value of `Velocity u`, with negative values of `Velocity u` replaced by zero. Note that this is not the average positive value because zero values will contribute to the average.

  Function: `areaAve`, **Location:** `outlet`, **Variable:** `max(Velocity u, 0.0[m s^-1])`.

15.5.3. areaInt

The `areaInt` function integrates a variable over the specified 2D location. To perform the integration over the total face area, select the `None` option from the **Axis** drop-down menu. If a direction is selected, the result is an integration over the projected area of each face onto a plane normal to that direction. Each point on a location has an associated area that is stored as a vector and therefore has direction. By selecting a direction in the function calculator, you are using only a single component of the vector in the area-weighting function. Because these components can be positive or negative, depending on the direction of the normal on the location, it is possible for areas to cancel out. An example of this would be on a closed surface where the projected area will always be zero (the results returned will not in general be exactly zero because the variable values differ over the closed surface). On a flat surface, the normal vectors always point in the same direction and never cancel out.

```
areaInt[_<Axis>[_<Coord Frame>] ](<Variable|Expression>)@<Location>
```

where:

- `<Axis>` is x, y, or z.

  Axis is optional; if not specified the integration is performed over the total face area. If axis is specified, then the integration is performed over the projected face area. A function description is available.

- `<Coord Frame>` is the coordinate frame. See the definitions for `<Function>` and `<Coordinate_Direction>` in Quantitative CEL Functions in Ansys CFX (p. 467).

- `<Variable|Expression>` is a variable or expression.

- `<Location>` is any 2D region (such as a boundary or interface) or injection region. For an injection region, only the cumulative geometry of the injection positions is considered, not the location of the entire injection region. An error is raised if the location specified is not a 2D object.

`areaInt_y_Frame2(Pressure)@boundary1` calculates the pressure force acting in the Y direction of the coordinate frame `Frame2` on the locator `boundary1`. This differs from a calculation using the force function, which calculates the total force on a wall boundary (that is, viscous forces on the boundary are included).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.3.1. Tools > Command Editor Example

```
>calculate areaInt, <Expression>, <Location>, [<Axis>]
```

Axis is optional. If it is not specified, the value held in the object will be used. To perform the integration over the total face area, the axis specification should be blank (that is, type a comma after the location name). A function description is available in areaInt (p. 481).

15.5.3.2. Tools > Function Calculator Examples

- This example integrates `Pressure` over `Plane 1`. The returned result is the total pressure force acting on `Plane 1`. The magnitude of each area vector is used and so the direction of the vectors is not considered.

  Function: `areaInt`, **Location:** `Plane 1`, **Variable:** `Pressure`, **Direction:** `None`

- This example integrates `Pressure` over the projected area of `Plane 1` onto a plane normal to the X axis. The result is the pressure force acting in the X direction on `Plane 1`. This differs slightly from using the force function to calculate the X-directional force on `Plane 1`. The force function includes forces due to the advection of momentum when calculating the force on an internal arbitrary plane or a non-wall boundary (inlets, and so on).

  Function: areaInt, **Location:** `Plane 1`, **Variable:** `Pressure`, **Direction:** `Global X`.

15.5.4. ave

The `ave` function calculates the arithmetic average of a variable or expression on the specified location.

Note:

CFD-Post and CFX-Solver may compute slightly different average values due to differences in how the average is computed:

- CFD-Post computes the arithmetically averaged vertex value for the location. Specifically, the average is computed by summing the vertex values and then dividing by the number of vertices.
- CFX-Solver computes the arithmetically averaged face value for the location. Specifically, the average is computed by summing the face values and then dividing by the number of faces. The value for a particular face is computed by arithmetically averaging the vertex values of the face.
- Computation in CFX-Solver and CFD-Post can be made more consistent by using expert parameter `bcp arithmetic aver sum option`. For details, see Discretization Parameters in the *CFX-Solver Modeling Guide*.

Results will be biased towards areas of high nodal density on the location. To obtain a mesh independent result, you should use the `lengthAve`, `areaAve`, `volumeAve` or `massFlowAve` functions.

```
ave(<Variable|Expression>)@<Location>
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where:

- <Variable|Expression> is a variable or expression.
- <Location> is any 3D region (such as a domain or subdomain) or injection region. For an injection region, an average value is calculated at each position and then these are averaged over all positions.

The ave function can be used on point, 1D, 2D, or 3D locations, or injection regions.

ave(Yplus)@Default calculates the mean Yplus values from each node on the default walls.

15.5.4.1. Tools > Command Editor Example

```
>calculate ave, <Variable|Expression>, <Location>
```

Note:

To obtain a mesh-independent result, you should use the lengthAve, areaAve, volumeAve or massFlowAve functions.

The average of a vector value is calculated as an average of its magnitudes, not the magnitude of component averages. As an example, for velocity:

$$|v|_{\text{ave}} = \frac{|v_1| + |v_2|}{2} \tag{15.1}$$

where

$$|v_i| = \sqrt{\left(v_{x,i}^2 + v_{y,i}^2 + v_{z,i}^2\right)} \tag{15.2}$$

15.5.4.2. Tools > Function Calculator Example

This example calculates the mean temperature at all nodes in the selected domain.

Function: ave, **Location:** MainDomain, **Variable:** Temperature.

15.5.5. count

The count function returns the number of nodes on the specified location.

```
count()@<Location>
```

where:

- <Location> is valid for point, 1D, 2D, or 3D locations, and injection regions. For an injection region, this function returns the number of unique vertices affected by the injection region positions.

count()@Polyline1 returns the number of points on the specified polyline locator.

15.5.5.1. Tools > Command Editor Example

```
>calculate count, <Location>
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.5.2. Tools > Function Calculator Example

This example returns the number of nodes in the specified domain.

Function: `count`, **Location:** `MainDomain`.

15.5.6. countTrue

The `countTrue` function returns the number of mesh nodes on the specified region that evaluate to "true", where true means greater than or equal to 0.5. The `countTrue` function is valid for 1D, 2D, and 3D locations.

```
countTrue(<Expression>)@<Location>
```

where `<Expression>` is:

- In CFD-Post, an expression that contains the logical operators =, >, <, <=, or >=.
- In CFX-Solver, an Additional Variable that you define. For example:

  ```
  TemperatureLE = Temperature > 300[K]
  ```

`countTrue(TemperatureLE)@Polyline1` returns the number of nodes on the specified polyline locator that evaluate to true.

15.5.6.1. Tools > Command Editor Examples

In CFD-Post:

```
>calculate countTrue(Temperature > 300[K]), Domain1
```

In CFX-Solver:

```
>calculate countTrue(TemperatureLE), Domain1
```

15.5.6.2. Tools > Function Calculator Example

This example returns the number of nodes that evaluate to "true" in the specified domain.

Function: `countTrue`, **Location:** `MainDomain`, **Expression:** `Temperature > 300[K]`.

15.5.7. force

This function returns the force exerted by the fluid on the specified 2D locator in the specified direction.

```
[<Phase>.]force[_<Axis>[_<Coord Frame>] ]()@<Location>
```

where:

- `[<Phase>.]` is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Axis> is x, y, or z
- <Coord Frame> is the coordinate frame. See the definitions for <Function> and <Coordinate_Direction> in Quantitative CEL Functions in Ansys CFX (p. 467).
- <Location> is any 2D region (such as a boundary or interface).

Force calculations on boundaries require additional momentum flow data.

Water at RTP.force_x()@wall1 returns the total force in the X direction acting on wall1 due to the fluid Water at RTP.

The force on a boundary is calculated using momentum flow data from the results file, if it is available. The result can be positive or negative, indicating the direction of the force. For non-boundary locators, an approximate force is always calculated.

CFD-Post calculates the approximate force as follows:

- If the locator is a wall boundary, the force is equal to the pressure force.
- For all other locators, the force is equal to the pressure force plus the mass flow force (due to the advection of momentum).
- In all cases, if wall shear data exists in the results file, the viscous force is added to the calculated force.

The force function enables you to select the fluids to use when performing your calculation. The result returned is the force on the locator due to that fluid/those fluids. Because the pressure force is the same at each node irrespective of the choice of fluids, the only difference is in the viscous forces (on wall boundaries) or the mass flow forces.

It is important to note that forces arising as a result of the reference pressure are not included in the force calculation. You can include reference pressure effects in the force calculation in the CFX-Solver by setting the expert parameter include pref in forces = t

When performing transient simulations with rotating reference frames, forces may be reported in either the stationary frame or the rotating frame. The forces reported by the CFX-Solver Output file are always given in the rotating frame. The forces reported by CFD-Post are reported in the stationary frame if the 'Angular Shift for Rotating Locations' option is active, as discussed in General in the *CFD-Post User's Guide*.

15.5.7.1. Tools > Command Editor Example

```
>calculate force, <Location>, <Axis>, [<Phase>]
```

15.5.7.2. Tools > Function Calculator Examples

- This calculates the total force on the default wall boundaries in the X direction. Pressure and viscous forces are included.

  Function: force, **Location:** Default, **Direction:** Global X, **Phase:** All Fluids.

- This calculates the forces on inlet1 due to pressure and the advection of momentum.

Function: force, **Location:** inlet1, **Direction:** `Global X`, **Phase:** `Water at RTP`.

15.5.8. forceNorm

Returns the per unit width force on a line in the direction of the specified axis. It is available only for a polyline created by intersecting a locator on a boundary. Momentum data must also be available. The magnitude of the value returned can be thought of as the force in the specified direction on a polyline, if the polyline were 2D with a width of one unit.

```
[<Phase>.]forceNorm[_<Axis>[_<Coord Frame>] ]()@<Location>
```

where:

- `[<Phase>.]` is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).
- `<Axis>` is x, y, or z
- `<Coord Frame>` is available in CFD-Post only
- `<Location>` is any 1D location. An error will be raised if the location specified is not one-dimensional.

`forceNorm_y()@Polyline1` calculates the per unit width force in the Y direction on the selected polyline.

15.5.8.1. Tools > Command Editor Example

```
>calculate forceNorm, <Location>, <Axis>, [<Phase>]
```

15.5.8.2. Tools > Function Calculator Example

The result from this calculation is force per unit width on `Polyline1` in the X direction.

Function: `forceNorm`, **Location:** `Polyline1`, **Direction:** `Global X`, **Phase:** `All Fluids`.

15.5.9. inside

The `inside` CEL function is essentially a step function variable, defined to be unity within a subdomain and zero elsewhere. This is useful for describing different initial values or fluid properties in different regions of the domain. It is similar to the CEL subdomain variable, but allows a specific 2D or 3D location to be given. For example, `273 [K] * inside()@Subdomain 1` has a value of 273 [K] at points in Subdomain 1 and 0 [K] elsewhere. The location does not need to be a subdomain, but can be any 2D or 3D named sub-region of the physical location on which the expression is evaluated. For immersed solids simulations, the location can also be a specific immersed solid domain, and the inside function will be updated automatically at the beginning of each time step.

```
inside()@<Location>
```

where:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Location> is any 2D or 3D named sub-region of the physical location on which the expression is evaluated.
- <Location> can also be an injection region.

  <Location> can also be an immersed solid domain on which the expression is evaluated dynamically.

Note:

The inside CEL function is not available in CFD-Post.

15.5.9.1. Tools > Command Editor Example

```
>calculate inside, <Location>
```

15.5.10. length

Computes the length of the specified line as the sum of the distances between the points making up the line.

```
length()@<Location>
```

where:

- <Location> is any 1D location. Specifying a 2D location will not produce an error; the sum of the edge lengths from the elements in the locator will be returned.

length()@Polyline1 returns the length of the polyline.

15.5.10.1. Tools > Command Editor Example

```
>calculate length, <Location>
```

Note:

While using this function in Power Syntax, the leading character is capitalized to avoid confusion with the Perl internal command "length".

15.5.10.2. Tools > Function Calculator Example

This example calculates the length of a polyline.

Function: length, **Location:** Polyline1.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.11. lengthAve

Computes the length-based average of the variable or expression on the specified line. This is the 1D equivalent of the `areaAve` function. The results is independent of the nodal distribution along the line because a weighting function assigns a higher weighting to areas of sparse nodal density.

```
lengthAve(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any 1D or 2D location.

`lengthAve(T)@Polyline1` calculates the average temperature on `Polyline1` weighted by the distance between each point (T is the system variable for temperature).

15.5.11.1. Tools > Command Editor Example

```
>calculate lengthAve, <Expression>, <Location>
```

15.5.11.2. Tools > Function Calculator Example

This calculates the average velocity on the location `Polyline1` using a length-based weighting function to account for the distribution of points along the line.

Function: `lengthAve`, **Location:** `Polyline1`, **Variable:** `Velocity`.

15.5.12. lengthInt

Computes the length-based integral of the variable or expression on the specified line. This is the 1D equivalent of the areaInt function.

```
lengthInt(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any 1D location.

15.5.12.1. Tools > Command Editor Example

```
>calculate lengthInt, <Expression>, <Location>.
```

15.5.13. lineCloudAve

Computes the average of a variable along each individual line of the specified user line cloud, and returns a location-dependent result.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

For example, the expression `lineCloudAve(phi)@MyCloud` would cause the following to happen during its evaluation:

- The mesh based variable `phi` would be interpolated onto the line points in user line cloud `MyCloud`.
- For each line in the line cloud, the values of phi on the line points are averaged, and the average value is stored at each of the line points.
- The line-averaged values of phi are then scattered outward from each line to the surrounding mesh, to produce a field of line averaged values, accessible as the function result.

In this example, the `MyCloud` user line cloud contains all of the settings that control the above process. For example, if the user line cloud is defined using the `From Boundaries` option, the result of the `lineCloudAve` function is a field of values available on or near the specified boundaries. The averaging is controlled by the **Line Averaging** settings in the user line cloud definition. The resulting field is interpolated onto the mesh vertices using the **Line Interpolation** settings in the user line cloud definition. See User Line Cloud in the *CFX-Pre User's Guide*.

```
lineCloudAve(<Variable>)@<Location>
```

where:

- `<Variable>` is a variable.
- `<Location>` is any user line cloud.

15.5.13.1. Tools > Command Editor Example

```
>calculate lineCloudAve, <Variable>, <Location>
```

15.5.13.2. Tools > Function Calculator Example

This calculates the average velocity on the location `LineCloud1`.

Function: `lineCloudAve`, **Location:** `LineCloud1`, **Variable:** `Velocity`.

15.5.14. mass

```
mass()@<Location>
```

where:

- `<Location>` is any 3D region (such as a domain or subdomain).

15.5.15. massAve

```
massAve(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Location> is any 3D region (such as a domain or subdomain).

15.5.16. massFlow

Computes the mass flow through the specified 2D location.

```
[<Phase>.]massFlow()@<Location>
```

where:

- [<Phase>.] is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).
- <Location> is any fluid surface (such as Inlets, Outlets, Openings and fluid-fluid interfaces) or injection region. For an injection region, the cumulative mass flow from all positions is considered.

`Air at STP.massFlow()@DegassingOutlet` calculates the mass flow of `Air at STP` through the selected location.

For boundary locators:

- The mass flow is calculated using mass flow data from the results file, if it is available[1]. Otherwise, an approximate mass flow is calculated.
- For multiphase cases, the mass flow through a boundary on a GGI interface evaluated in CFD-Post is an approximation to the 'exact' mass flow evaluated by the solver. This approximation vanishes as the mesh is refined or as the volume fraction on the interface becomes uniform.

For non-boundary locators (that is, internal locators):

- If the locator is an edge based locator (such as a slice plane or isosurface), the domain mass flow data from the results file will be used.
- In all other cases, an approximate mass flow is calculated.
- The `massFlow` function (and related functions `massFlowAve` and `massFlowAveAbs`) in CFD-Post can be inaccurate when evaluated on a physical locator (cut plane, isosurface, etc.) that cuts through a GGI interface, due to inaccuracies in evaluating the conserved mass flow (from integration point data) along the intersection of the physical locator and the GGI interface. In this situation, calculating the approximate mass flow would be more accurate. The approximate mass flow can be calculated with an expression of the form: `sum(Approximated Mass Flow)@[locator]`. For example, `sum(Approximated Mass Flow)@Plane 1`. Alternatively, the approximate mass flow can be calculated using an expression of the form: `areaInt_x(Velocity u * Density)@myLoc + areaInt_y(Velocity v * Density)@myLoc + areaInt_z(Velocity w * Density)@myLoc`.

[1] The availability depends on the setting for **Output Boundary Flows** for the file (see Output Boundary Flows Check Box in the *CFX-Pre User's Guide* for details).

In addition to this, mass flow information written to a backup file at time zero is not accurate. To calculate accurate mass flow from a backup file, use information from the backup file written after the first timestep/iteration of the run.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The `massFlow` function enables you to select the fluids to use when performing your calculation. The result returned is the mass flow of the selected fluids through the locator.

If a user specifies a boundary mass source, this mass source is not included in the massFlow calculator. This will affect all massFlow related calculations, such as massFlowInt().

15.5.16.1. Mass Flow Sign Convention

The mass flow through a surface is defined by $-\rho\mathbf{V}\cdot\mathbf{n}$ where $\mathbf{V}$ is the velocity vector and $\mathbf{n}$ is the surface normal vector. By convention, the surface normal at a domain boundary is directed out of the domain. Therefore, the mass flow is positive at an inlet boundary with the velocity directed into the domain. For planes and surfaces that cut through a domain, the normal of the plane or surface is determined by from the right-hand rule and the manner in which the plane or surface is constructed. For example, the surface normal for a Z-X plane has the same sense and direction as the Y axis.

15.5.16.2. Tools > Command Editor Example

```
>calculate massFlow, <Location>, [<Phase>]
```

15.5.16.3. Tools > Function Calculator Example

This calculates the mass flow for all fluids in the domains through the location outlet2:

Function: `massFlow`, **Location:** `outlet2`, **Phase:** `All Fluids`.

15.5.17. massFlowAve

Computes the average of a variable or expression on the specified 2D location. The `massFlowAve` function enables you to select the fluids to use for the calculation. The returned result is the average variable or expression value, evaluated according to the formula:

$$\text{massFlowAve}\ (\Phi) = \frac{\Sigma\ (m\,\Phi)}{\Sigma\ m} \tag{15.3}$$

where Φ represents the variable or expression being averaged and m represents the local mass flow (net local mass flow if more than one fluid is selected). Each summation term is evaluated on, and corresponds to, a node on the 2D locator. The mass flow for each term is derived from summing contributions from the surrounding solver integration points. As a result, the denominator evaluates to the net mass flow through the 2D locator.

In cases where there is significant flow, but little or no net flow through the 2D locator (as can happen with recirculation), the denominator of the averaging formula becomes small, and the resulting average value may become adversely affected. In such cases, the `massFlowAveAbs` (see massFlowAveAbs (p. 492)) function is a viable alternative to the `massFlowAve` function.

```
[<Phase>.]massFlowAve(<Variable|Expression>)@<Location>
```

where:

- `[<Phase>.]` is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any fluid surface (such as Inlets, Outlets, Openings and fluid-fluid interfaces) or injection region. For an injection region, the mass flow at each affected vertex is used. An error is raised if the location specified is not 2D.

`massFlowAve(Density)@Plane1` calculates the average density on `Plane1` weighted by the mass flow at each point on the location.

Examples:

- Tools > Function Calculator Example

  This example calculates the average velocity on `Plane1` weighted by the mass flow for all fluids assigned to each point on `Plane1`:

  Function: `massFlowAve`, **Location:** `Plane1`, **Variable:** `Velocity`, **Phase:** `All Fluids`

- Tools > Command Editor Example

```
>calculate massFlowAve, <Variable|Expression>, <Location>, [<Phase>]
```

See Details on Mass Flow Related Functions (p. 493).

15.5.18. massFlowAveAbs

This function is similar to the `massFlowAve` function (see massFlowAve (p. 491)), except that each local mass flow value used in the averaging formula has the absolute function applied. That is:

$$\text{massFlowAveAbs}\ (\Phi) = \frac{\Sigma\ (|m|\ \Phi)}{\Sigma\ |m|} \tag{15.4}$$

```
[<Phase>.]massFlowAveAbs(<Variable|Expression>)@<Location>
```

where:

- `[<Phase>.]` is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).
- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any fluid surface (such as Inlets, Outlets, Openings and fluid-fluid interfaces) or injection region. For an injection region, the mass flow at each affected vertex is used. An error is raised if the location specified is not 2D.

`massFlowAve(Density)@Plane1` calculates the average density on `Plane1` weighted by the mass flow at each point on the location.

In cases where there is significant flow, but little or no net flow through the 2D locator (as can happen with recirculation), the `massFlowAveAbs` function is a viable alternative to the `massFlowAve` function (see massFlowAve (p. 491)).

See Details on Mass Flow Related Functions (p. 493).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.19. Details on Mass Flow Related Functions

Note that the `massFlowAve` and `massFlowAveAbs` functions provide the same result, and that the denominator evaluates to the net mass flow through the 2D locator, only when all of the flow passes through the 2D locator in the same general direction (that is, when there is no backflow). If there is any backflow through the 2D locator, the denominator in the function for `massFlowAveAbs` evaluates to a value of greater magnitude than the net mass flow through the 2D locator (although this is not necessarily harmful to the resulting average value).

Note:

The values of variables other than mass flow are stored at the mesh nodes and are applied to the locator nodes by linear interpolation. For the mass flow variable, CFD-Post uses either:

- Integration point mass flow data, or
- Approximate mass flow values based on mesh nodal values of velocity (and density, if available)

according to Table 15.5: Source of mass flow data (p. 493).

Table 15.5: Source of mass flow data

Flow condition	massFlowAve	massFlowAveAbs
No recirculation (backflow) present	Integration point data if available (CFX results only), otherwise approximate mass flow values	Integration point data if available (CFX results only), otherwise approximate mass flow values
recirculation (backflow) present	Integration point data if available (CFX results only), otherwise approximate mass flow values	Approximate mass flow values

Note:

The `massFlow`, `massFlowAve`, and `massFlowAveAbs` functions in CFD-Post can be inaccurate when evaluated on a physical locator (cut plane, isosurface, etc.) that cuts through a GGI interface, due to inaccuracies in evaluating the conserved mass flow (from integration point data) along the intersection of the physical locator and the GGI interface. In this situation, calculating the approximate mass flow would be more accurate. The approximate mass flow can be calculated with an expression of the form: `sum(Approximated Mass Flow)@[locator]`. For example, `sum(Approximated Mass Flow)@Plane 1`.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

15.5.20. massFlowInt

Integrates a variable or expression over the specified 2D location. A weighting function is applied to the variable or expression value at each point based on the mass flow assigned to that point. You can also specify the fluid(s) used to calculate the mass flow at each locator point.

```
[<Phase>.]massFlowInt(<Variable|Expression>)@<Location>
```

where:

- `[<Phase>.]` is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).
- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any fluid surface (such as Inlets, Outlets, Openings and fluid-fluid interfaces) or injection region. For an injection region, the mass flow at each affected vertex is used. An error is raised if the location specified is not 2D.

15.5.20.1. Tools > Command Editor Example

```
>calculate massFlowInt, <Variable|Expression>, <Location>, [<Phase>]
```

15.5.20.2. Tools > Function Calculator Example

This example integrates pressure over `Plane1`. The result is the pressure force acting on Plane1 weighted by the mass flow assigned to each point on `Plane1`:

Function: `massFlowInt`, **Location:** `Plane1`, **Variable:** `Pressure`, **Phase:** `All Fluids`

15.5.21. massInt

The mass-weighted integration of a variable or expression within a domain or subdomain.

```
massInt(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any 3D region (such as a domain or subdomain).

15.5.22. maxVal

Returns the maximum value of the specified variable or expression on the specified locator. You should create a user variable if you want to find the maximum value of an expression.

```
maxVal(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Location> is, in the context of CFX-Solver, a 2D region or 3D region (such as a domain or sub-domain) or, in the context of CFD-Post, a Point object or a 1D, 2D, or 3D locator, or injection region. For an injection region, the maximum over the affected vertices is used.

15.5.22.1. Tools > Command Editor Example

```
>calculate maxVal, <Variable|Expression>, <Location>
```

15.5.22.2. Tools > Function Calculator Example

This will return the maximum Yplus value on the default wall boundaries:

Function: maxVal, **Location:** Default, **Variable:** Yplus

15.5.23. minVal

Returns the minimum value of the specified variable or expression on the specified locator. You should create a user variable if you want to find the minimum value of an expression.

```
minVal(<Variable|Expression>)@<Location>
```

where:

- <Variable|Expression> is a variable or expression.
- <Location> is, in the context of CFX-Solver, a 2D region or 3D region (such as a domain or sub-domain) or, in the context of CFD-Post, a Point object or a 1D, 2D, or 3D locator, or injection region. For an injection region, the minimum over the affected vertices is used

15.5.23.1. Tools > Command Editor Example

```
>calculate minVal, <Variable|Expression>, <Location>
```

15.5.23.2. Tools > Function Calculator Example

These settings will return the minimum temperature in the domain:

Function: minVal, **Location:** MainDomain, **Variable:** Temperature

15.5.24. probe

Returns the value of the specified variable or expression on the specified Point object.

```
probe(<Variable|Expression>)@<Location>
```

where:

- <Variable|Expression> is a variable or expression.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Location> is any point object (such as a Source Point or Cartesian Monitor Point).

Important:

This calculation should be performed only for point locators described by single points. Incorrect solutions will be produced for multiple point locators.

15.5.24.1. Tools > Command Editor Example

```
>calculate probe, <Expression>, <Location>
```

15.5.24.2. Tools > Function Calculator Example

This example returns the density value at Point1:

Function: probe, **Location:** Point1, **Variable:** Density

15.5.25. rbstate

Returns the value of the specified rigid body state variable, or axis component thereof, on the specified:

- Rigid Body object (for a standard rigid body definition), or
- Immersed Solid domain that is governed by a rigid body solution.

The rigid body state variables are:

- Position
- Linear Velocity
- Linear Acceleration
- Euler Angle
- Angular Velocity
- Angular Acceleration

Syntax:

```
rbstate(<rbvar>[<Axis>])@<Location>
```

where:

- <rbvar> is a rigid body state variable.
- <Axis> is X, Y, or Z.

  For the Euler Angle rigid body state variable, an axis must be specified. For all of the other rigid body state variables, the axis is optional. If you do not specify an axis, the magnitude of the vector

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

is returned. For example, if the variable is `Position` and you do not specify an axis, the distance from the origin will be returned.

- `<Location>` is any rigid body object or any immersed solid domain that is governed by a rigid body solution.

Results are given with respect to the global coordinate frame (Coord 0).

15.5.25.1. Expressions Details View Example

```
rbstate(Linear Velocity Z)@Buoy
```

15.5.26. rmsAve

The `rmsAve` function calculates the RMS average of a variable or expression on the specified location.

- CFX-Solver computes the RMS average of the vertex values for a 3D location.
- CFX-Solver computes the RMS average of the face values for a 2D location. The value for a particular face is computed by arithmetically averaging the vertex values of the face.

```
rmsAve(<Variable|Expression>)@<Location>
```

where:

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is a 2D region or 3D region (such as a domain or subdomain) or injection region. For an injection region, an average value is calculated at each position, and then these are RMS-averaged over all positions.

15.5.27. sum

The `sum` function calculates the sum of a variable or expression on the specified location.

Note:

CFD-Post and CFX-Solver compute the sum in different ways:

- CFD-Post computes the sum of the vertex values for the location.
- CFX-Solver computes the sum of the vertex values for a 3D location.
- CFX-Solver computes the sum of the face values for a 2D location. The value for a particular face is computed by arithmetically averaging the vertex values of the face.

```
sum(<Variable|Expression>)@<Location>
```

where:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- <Variable|Expression> is a variable or expression.
- <Location> is, in the context of CFX-Solver, a 2D region or 3D region (such as a domain or sub-domain) or, in the context of CFD-Post, a Point object or a 1D, 2D, or 3D locator, or injection region. For an injection region, an average value is calculated at each position, and then these are summed over all positions.

15.5.27.1. Tools > Command Editor Example

```
>calculate sum, <Variable|Expression>, <Location>
```

15.5.27.2. Tools > Function Calculator Example

This example returns the sum of the finite volumes assigned to each node in the location SubDomain1. In this case, this sums to the volume of the subdomain:

Function: sum, **Location:** SubDomain1, **Variable:** Volume of Finite Volume

15.5.28. torque

Returns the torque on a 2D locator about the specified axis. The force calculated during evaluation of the torque function has the same behavior as the force function. For details, see force (p. 484). You can select the fluids involved in the calculation.

```
[<Phase>.]torque_[<Axis>[_<Coord Frame>] ]()@<Location>
```

where:

- [<Phase>.] is an optional prefix that is not required for single-phase flows. For details, see CEL Functions with Multiphase Flow (p. 471).
- <Axis> is x, y, or z.
- <Coord Frame> is the coordinate frame. See the definitions for <Function> and <Coordinate_Direction> in Quantitative CEL Functions in Ansys CFX (p. 467).
- <Location> is any 2D region (such as a wall). If the location specified is not 2D, an error is raised.

15.5.28.1. Tools > Command Editor Example

```
>calculate torque, <Location>, <Axis>, [<Phase>]
```

15.5.28.2. Tools > Function Calculator Example

This example calculates the torque on Plane1 about the Z axis due to all fluids in the domain.

Function: torque, **Location:** Plane1, **Axis:** Global Z, **Phase:** All Fluids

15.5.29. volume

Calculates the volume of a 3D location.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
volume()@<Location>
```

where:

- <Location> is any 3D region (such as a domain or subdomain). An error is raised if the location specified is not a 3D object. For details, see volume (p. 498).

15.5.29.1. Tools > Command Editor Example

```
>calculate volume, <Location>
```

15.5.29.2. Tools > Function Calculator Example

This example returns the sum of the volumes of each mesh element included in the location Volume1.

Function: volume, **Location:** Volume1

15.5.30. volumeAve

Calculates the volume-weighted average of a variable or expression on a 3D location. This is the 3D equivalent of the areaAve function. The volume-weighted average is the average value on a location weighted by the volume assigned to each vertex on a location. Without the volume weighting function, the average of all the nodal values would be biased towards values in regions of high mesh density. The following example demonstrates use of the function.

```
volumeAve(<Variable|Expression>)@<Location>
```

where:

- <Variable|Expression> is a variable or expression.
- <Location> is any 3D region (such as a domain or subdomain).

15.5.30.1. Tools > Command Editor Example

```
>calculate volumeAve, <Variable|Expression>, <Location>
```

15.5.30.2. Tools > Function Calculator Example

This example calculates the volume-weighted average value of density in the region enclosed by the location Volume1:

Function: volumeAve, **Location:** Volume1, **Variable:** Density

15.5.31. volumeInt

Integrates the specified variable or expression over the volume location. This is the 3D equivalent of the areaInt function.

```
volumeInt(<Variable|Expression>)@<Location>
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

where:

- `<Variable|Expression>` is a variable or expression.
- `<Location>` is any 3D region (such as a domain or subdomain). An error is raised if the location specified is not a 3D object.

For example, `volumeInt(Density)@StaticMixer` will calculate the total fluid mass in the domain `StaticMixer`.

Note:

Because the Density variable represents the average density during the timestep rather than the density at the end of the timestep, the `volumeInt(Density)` does not accurately give the mass of fluid at the end of a timestep. Use the `mass()` function instead.

The volumeInt function does not take into account the porosity of the location specified for porous domains. To include the porosity effect in your calculation, you need to manually multiply your argument by:

- The Volume Porosity if you want to evaluate the integral on the fluid side
- (1-Volume Porosity) if you want to evaluate the integral on the solid side

---

15.5.31.1. Tools > Command Editor Example

```
>calculate volumeInt, <Variable|Expression>, <Location>
```

15.5.31.2. Tools > Function Calculator Example

This example calculates the integral of density (the total mass) in `Volume1`.

Function: `volumeInt`, **Location:** `Volume1`, **Variable:** `Density`

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 16: Variables in Ansys CFX

This chapter describes the variables available in Ansys CFX:

16.1. Hybrid and Conservative Variable Values
16.2. List of Field Variables
16.3. Particle Variables Generated by the Solver
16.4. Miscellaneous Variables

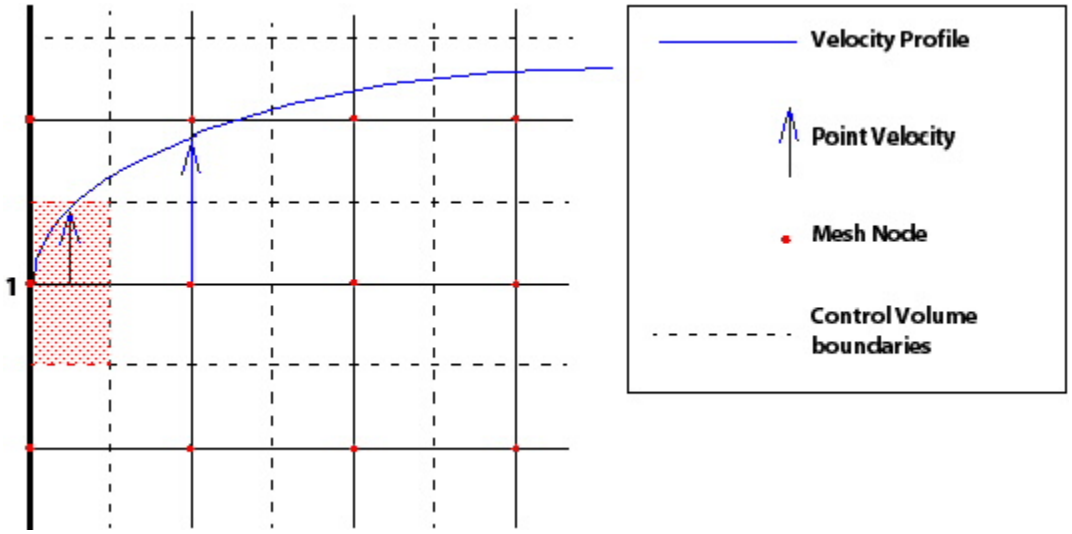
16.1. Hybrid and Conservative Variable Values

The CFX-Solver calculates the solution to your CFD problem using polyhedral finite volumes surrounding the vertices of the underlying mesh elements (hexahedrons, tetrahedrons, prisms, pyramids). Analytical solutions to the Navier-Stokes equations exist for only the simplest of flows under ideal conditions. To obtain solutions for real flows, a numerical approach must be adopted whereby the equations are replaced by algebraic approximations that may be solved using a numerical method.

The solution values on the boundary vertices, called *conservative values*, are the values obtained from solving the conservation equations for the boundary control volumes. These values are not necessarily the same as the specified boundary condition values, although the specified boundary value is used to close boundary fluxes for the boundary control volume. For example, on a no-slip wall, the wall velocity is used to compute the viscous force for the boundary face of the boundary control volume, but the resulting control volume equation solution will not necessarily be the wall velocity. The conservative values are representative of the boundary control volume, not the boundary itself. For visualization purposes, it is often useful to view the specified boundary condition value for the boundary vertices rather than the conservative values. This is especially true when the value of a conservative solution variable (such as pressure or temperature, for instance) is specified at a particular boundary condition. The specified boundary values are called *hybrid values*. CFD-Post uses hybrid values by default for most variables. Hybrid values are obtained by overwriting the conservative results on the boundary nodes produced by the CFX-Solver with values based on the specified boundary conditions. This ensures, for example, that the velocity is displayed as zero on no-slip walls. For quantitative calculations, the conservative values should normally be used because they are consistent with the discrete solutions obtained by the solver. If you want to use these values in CFD-Post, you can select them from the **Variables Editor** dialog box as described above. By default, CFD-Post uses conservative values when the **Calculate** command is used.

The difference between hybrid and conservative values at wall boundaries can be demonstrated using the following figure:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Using velocity as an example, the velocity value calculated at a mesh node is based upon the 'average' in the control volume surrounding that node. For calculation purposes, the entire control volume is then assumed to possess that velocity. At a boundary node, its surrounding control volume includes an area in the bulk of the fluid (this area is highlighted around the boundary node marked **1**). Hence, the conservative velocity calculated at the wall node is not zero, but an 'average' over the control volume adjacent to the boundary. At a wall boundary node the difference between conservative and hybrid values can be illustrated by considering the case of the mass flow rate through the wall-adjacent control volume. If a zero velocity was enforced at the boundary node, then this would produce zero mass flow through the control volume, which is clearly not correct.

Note:

Of the files that CFD-Post can load, only CFX-Solver Results files with extension `.res` contain both hybrid and conservative values.

16.1.1. Solid-Fluid Interface Variable Values

16.1.1.1. Conservative Values at 1:1 Interface

At a solid-fluid 1:1 interface, duplicate nodes exist. The conservative value for the solid-side node is the variable values averaged over the half on the control volume that lies inside the solid. The conservative value for the fluid-side node is the variable values averaged over the half of the control volume that lies in the fluid.

Consider the example of heat transfer from a hot solid to a cool fluid when advection dominates within the fluid. If you create a plot across the solid-fluid interface using conservative values of temperature, then you will see a sharp change in temperature across the interface. This is because values are interpolated from the interface into the bulk of the solid domain using the value for the solid-side node at the interface, while values are interpolated from the interface into the bulk of the fluid domain using the value for the fluid-side node at the interface. This results in a temperature discontinuity at the interface.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.1.1.2. Hybrid Values at 1:1 Interface

When creating plots using hybrid variable values (the default in CFD-Post), the 1:1 interface is single valued and takes the solid-side conservative value. You can therefore expect to see the same plot within the solid, but the temperature profile between the interface and the first node in the fluid interpolates between the solid-side interface value and the first fluid node value. In this case, a discontinuity does not exist because all nodes are single valued.

Conservative values should be used for all quantitative calculations.

16.1.1.3. Conservative Values on a GGI Interface

At a GGI interface, the CFX Solver calculates both fluid-side and solid-side temperatures based on heat flux conservation. These values are representative of the temperature within the half-control volumes around the vertices on the interface. The fluid-side and solid-side temperatures are generally not equal. As a result, a plot of conservative values of temperature will generally show a discontinuity across a GGI interface.

16.1.1.4. Hybrid Values on a GGI Interface

At a GGI interface, the CFX Solver calculates a "surface temperature" based on a flux-conservation equation for the 'control surfaces' that lie between the fluid side and the solid side. The surface temperature is usually between the fluid-side and solid-side temperatures. Hybrid values of temperature on a GGI interface are set equal to the surface temperature. As a result, there is no discontinuity in hybrid values of temperature across a GGI interface.

16.2. List of Field Variables

Each of the subsections of this section contains one or more tables that list field variables. In general, variables may be defined in CFX-Pre, viewed in CFD-Post, and exported. Many variables are relevant only for specific physical models.

The information given in the tables includes:

- **Long Variable Name**: The name that you see in the user interface. This name may be used in CEL expressions.
- **Short Variable Name**: A shorter name that may be used in CEL expressions.
- **Units**: The default units for the variable. An empty entry [ ] indicates a dimensionless variable.

  Note:

  The entries in the Units columns are SI but could as easily be any other system of units.

- In the **Availability** column:
  - A number represents the user level (1 indicates that the variable appears in default lists, 2 and 3 indicate that the variable appears in extended lists that you see when you click). This number

is useful when using the CFX Export facility. For details, see File Export Utility in the *CFX-Solver Manager User's Guide*. Note that the CFX-Solver may sometimes override the user-level setting depending on the physics of the problem. In these cases, the User Level may be different from that shown in the tables that follow.

– **Boundary (B)**: A **B** in this column indicates that the variable contains only non-zero values on the boundary of the model. See Boundary-Value-Only Variables in the *CFD-Post User's Guide* for more details.

  Boundary-Value-Only Variables in the *CFD-Post User's Guide* describes the useful things that you can do with variables that are defined only on the boundaries of the model.

– A indicates the variable is available for mesh adaption

– C indicates the variable is available in CEL

– DT indicates the variable is available for data transfer to Ansys

– M indicates the variable is available for monitoring

– P indicates the variable is available for particle user-routine argument lists

– PR indicates the variable is available for particle results

– R indicates the variable is available to be output to the results, transient results, and backup files

– RA indicates the variable is available for radiation results

– TS indicates the variable is available for transient statistics

- **Definition**: Defines the variable.

This is not a complete list of variables. Information on obtaining details on all variables is available in RULES and VARIABLES Files in the *CFX-Solver Manager User's Guide*.

16.2.1. Common Variables Relevant for Most CFD Calculations

The following table contains a list of variables (with both long and short variable names) that can be used when working with CFD calculations. For an explanation of the column headings, see List of Field Variables (p. 503).

Note:

Variables with names shown in **bold text** are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from **Extra Output Variables List** on the **Results** tab of the **Solver** > **Output Control** details view of CFX-Pre.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Density	density	[kg m^-3]	1	Mass per unit volume.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
			A, C, M, P, R, TS	Note that for fixed composition, variable composition, and reacting mixtures, when density is governed by the `Ideal Mixture` option, the density is determined by a mass-fraction-weighted harmonic average: $\frac{Y_A}{\rho_A}+\frac{Y_B}{\rho_B}+...+\frac{Y_N}{\rho_N}=\frac{1}{\rho_{mix}}$
Dynamic Viscosity	viscosity	[kg m^-1 s^-1]	2 A, C, M, P, R, TS	Dynamic viscosity (μ), also called *absolute viscosity*, is a measure of the resistance of a fluid to shearing forces, and appears in the momentum equations. Using an expression to set the dynamic viscosity is possible. For details, see Non-Newtonian Flow in the *CFX-Solver Modeling Guide*.
Velocity[a]	vel	[m s^-1]	1 A, C, M, P, R, TS	Velocity vector.
Velocity u Velocity v Velocity w	u v w	[m s^-1]	1 A, C, M, P, R, TS	Components of velocity.
Pressure	p	[kg m^-1 s^-2]	1 A, C, M, P, R, TS	Both `Pressure` and `Total Pressure` are measured relative to the reference pressure that you specified on the **Domains** panel in CFX-Pre. Additionally, `Pressure` is the total normal stress, which means that when using the k-e turbulence

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				model, `Pressure` is the thermodynamic pressure plus the turbulent normal stress. `Static Pressure` is the thermodynamic pressure, in most cases this is the same as `Pressure`. For details, see Modified Pressure in the *CFX-Solver Theory Guide*.
Static Pressure	pstat	[kg m^-1 s^-2]	3	CFX solves for the relative `Static Pressure` (thermodynamic pressure) p_{stat} in the flow field, and is related to `Absolute Pressure` $p_{\text{abs}}=p_{\text{stat}}+p_{\text{ref}}$.
Total Pressure	ptot	[kg m^-1 s^-2]	2 A, C, M, P, R, TS	The total pressure, p_{tot}, is defined as the pressure that would exist at a point if the fluid was brought instantaneously to rest such that the dynamic energy of the flow converted to pressure without losses. The following three sections describe how total pressure is computed for a pure component material with constant density, ideal gas equation of state and a general equation of state (CEL expression or RGP table). For details, see Total Pressure in the *CFX-Solver Theory Guide*.
Wall Shear	wall shear	Pa	3,**B**	For details, see Scalable Wall Functions in the *CFX-Solver Theory Guide*.
Volume of Finite Volume			3 C, DT, R, TS	Volume of finite volume. For details, see Discretization of the Governing Equations in the *CFX-Solver Theory Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	**Short Variable Name**	**Units**	**Availability**	**Definition**
X coordinate	x	[m]	2 C	Cartesian coordinate components.
Y coordinate	y	[m]	2 C	
Z coordinate	z	[m]	2 C	
Kinematic Diffusivity	visckin		2 C, M, P, R, TS	*Kinematic diffusivity* describes how rapidly a scalar quantity would move through the fluid in the absence of convection. For convection-dominated flows, the kinematic diffusivity can have little effect because convection processes dominate over diffusion processes.
Shear Strain Rate	sstrnr	[s^-1]	2 A, C, M, R, TS	For details see Non-Newtonian Flow in the *CFX-Solver Modeling Guide*.
Specific Heat Capacity at Constant Pressure	Cp	[m^2 s^-2 K^-1]	2 A, C, M, R, TS	For details, see Specific Heat Capacity in the *CFX-Solver Modeling Guide*.
Specific Heat Capacity at Constant Volume	**Cv**	[m^2 s^-2 K^-1]	2 A, C, M, P, R, TS	
Thermal Conductivity	cond	[kg m s^-3 K^-1]	2 A, C, M, R, TS	Thermal conductivity, λ, is the property of a fluid that characterizes its ability to transfer heat by conduction. For details, see Thermal Conductivity in the *CFX-Solver Modeling Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Temperature	T	[K]	1 A, C, DT, M, P, R, TS	The static temperature, T_{stat}, is the thermodynamic temperature, and depends on the internal energy of the fluid. In CFX, depending on the heat transfer model you select, the flow solver calculates either total or static enthalpy (corresponding to the total or thermal energy equations). For details, see Static Temperature in the *CFX-Solver Theory Guide*.
Total Temperature	Ttot	[K]	1 A, C, M, P, R, TS	The total temperature is derived from the concept of total enthalpy and is computed exactly the same way as static temperature, except that total enthalpy is used in the property relationships. For details, see Total Temperature in the *CFX-Solver Theory Guide*.
Wall Heat Flux	Qwall	[W m^-2]	2,B C, DT, R, TS	A heat flux is specified across the wall boundary. A positive value indicates heat flux into the domain. For multiphase cases, when the bulk heat flux into both phases is set, this option is labeled Wall Heat Flux instead of Heat Flux. When set on a per fluid basis, this option is labelled Heat Flux.
Wall Heat Transfer Coefficient	htc	[W m^-2 K^-1]	2,B C, R, TS	For details, see Heat Transfer Coefficient and Wall Heat Transfer Coefficient in the *CFX-Solver Modeling Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Total Enthalpy	htot	[m^2 s^-2]	A, C, M, R, TS	h_{tot} For details, see Transport Equations in the *CFX-Solver Theory Guide*.
Static Enthalpy	enthalpy	[m^2 s^-2]	2 A, C, M, P, R, TS	For details, see Static Enthalpy in the *CFX-Solver Theory Guide*.

[a] When a rotating frame of reference is used, all variables in the CFX-5 results file are relative to the rotating frame, unless specified as a `Stn Frame` variable.

16.2.2. Variables Relevant for Turbulent Flows

The following table contains a list of variables (with both long and short variable names) that can be used when working with turbulent flows. For an explanation of the column headings, see List of Field Variables (p. 503).

A **B** in the **Type** column indicates that the variable contains only non-zero values on the boundary of the model.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Blending Function for DES model	desbf	[ ]	2 C, M, R, TS	Controls blending between RANS and LES regimes for the DES model
Turbulence Kinetic Energy	ke	[m^2 s^-2]	1 A, C, M, P, R, TS	For details, see The k-epsilon Model in the *CFX-Solver Modeling Guide*.
Turbulence Eddy Dissipation	ed	[m^2 s^-3]	1 A, C, M, P, R, TS	The rate at which the velocity fluctuations dissipate. For details, see The k-epsilon Model in the *CFX-Solver Modeling Guide*.
Turbulent Eddy Frequency	tef	[s^-1]	1 A, C, M, P, R, TS	
Eddy Viscosity	eddy viscosity	[kg m^-1 s-1]	2 A, C, M, P, R, TS	The "eddy viscosity model" proposes that turbulence consists of small eddies that are continuously

Long Variable Name	Short Variable Name	Units	Availability	Definition
				forming and dissipating, and in which the Reynolds stresses are assumed to be proportional to mean velocity gradients. For details, see Eddy Viscosity Turbulence Models in the *CFX-Solver Theory Guide*.
Reynolds Stress	rs	[m^2 s^-2]	2 A, C, M, P, R, TS	This is a tensor quantity with six components. For details, see Statistical Reynolds Stresses in the *CFX-Solver Modeling Guide* and Reynolds Stress Turbulence Models in the *CFX-Solver Theory Guide* in the Ansys CFX documentation.
Statistical Reynolds Stress uu	rsstat uu	[m^2 s^-2]	3 M, R	In LES runs, Reynolds Stress components are automatically generated using running statistics of the instantaneous, transient velocity field. For details, see Statistical Reynolds Stresses in the *CFX-Solver Modeling Guide*.
Statistical Reynolds Stress vv	rsstat vv	[m^2 s^-2]	3 M, R	
Statistical Reynolds Stress ww	rsstat ww	[m^2 s^-2]	3 M, R	
Statistical Reynolds Stress uv	rsstat uv	[m^2 s^-2]	3 M, R	
Statistical Reynolds Stress uw	rsstat uw	[m^2 s^-2]	3 M, R	
Statistical Reynolds Stress vw	rsstat vw	[m^2 s^-2]	3 M, R	
Velocity Correlation uu	uu	[m^2 s^-2]	3 C, M, R	These variables represent the instantaneous velocity correlation and are used in the first term on the right hand side of Equation 4.10
Velocity Correlation vv	vv	[m^2 s^-2]	3	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
			C, M, R	in the *CFX-Solver Modeling Guide*.
Velocity Correlation ww	ww	[m^2 s^-2]	3 C, M, R	
Velocity Correlation uv	uv	[m^2 s^-2]	3 C, M, R	
Velocity Correlation uw	uw	[m^2 s^-2]	3 C, M, R	
Velocity Correlation vw	vw	[m^2 s^-2]	3 C, M, R	
Yplus	yplusstd	[ ]	2,**B** C, R, TS	A variable based on the distance from the wall to the first node and the wall shear stress. For details, see Solver Yplus and Yplus in the *CFX-Solver Modeling Guide*.
Solver Yplus	yplus	[ ]	2,**B** C, R, TS	A deprecated internal variable. For details, see Solver Yplus and Yplus in the *CFX-Solver Modeling Guide*.

16.2.3. Variables Relevant for Buoyant Flow

The following table contains a list of variables (with both long and short variable names) that can be used when working with buoyant flows. For an explanation of the column headings, see List of Field Variables (p. 503).

Long Variable Name	Short Variable Name	Units	Availability	Definition
Thermal Expansivity	beta	[K^ -1]	2 C	For details, see Buoyancy in the *CFX-Solver Modeling Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.2.4. Variables Relevant for Compressible Flow

The following table contains a list of variables (with both long and short variable names) that can be used when working with compressible flows.

Note:

Variables with names shown in **bold text** are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from **Extra Output Variables List** on the **Results** tab of the **Solver** > **Output Control** details view of CFX-Pre.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Isobaric Compressibility	compisoP	[K^-1]	2 C, M, R	$-\frac{1}{\rho}\frac{\partial \rho}{\partial T}\Big\vert_{p}$
Isothermal Compressibility	compisoT	[m s^2 kg^-1]	2 C, M, R	Defines the rate of change of the system volume with pressure. $\frac{1}{\rho}\frac{\partial \rho}{\partial p}\Big\vert_{T}$
Mach Number	Mach	[ ]	1 A, C, M, R, TS	For details, see List of Symbols in the *CFX-Solver Theory Guide*.
Shock Indicator	shock indicator	[ ]	2 A, C, M, R, TS	The variable takes a value of 0 away from a shock and a value of 1 in the vicinity of a shock. This variable is available only if **Solver Control** > **Advanced Options** > **Compressibility Control** > **High Speed Numerics** is selected.
Isentropic Compressibility	compisoS	[m s^2 kg^-1]	2 C, M, R	The extent to which a material reduces its volume when it is subjected to compressive stresses at a constant value of entropy. $\left(\frac{1}{\rho}\right)\left(\frac{\partial \rho}{\partial p}\right)_{s}$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.2.5. Variables Relevant for Particle Tracking

The following table contains a list of variables (with both long and short variable names) that can be used when working with particle tracking.

Note:

Variables with names shown in **bold text** are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from **Extra Output Variables List** on the **Results** tab of the **Solver** > **Output Control** details view of CFX-Pre.

Long Variable Name	Short Variable Name	Units	User Level	Definition
Latent Heat	lheat	[ ]	2 C, R, M	User-specified latent heat for phase pairs involving a particle phase.
Particle Momentum Source	ptmomsrc	[ ]	2 A, C, M, P, R	Momentum source from particle phase to continuous phase.
Particle Diameter	particle diameter	[ ]	3 A, C, M, R	Diameter of a particle phase.

16.2.6. Variables Relevant for Calculations with a Rotating Frame of Reference

The following table contains a list of variables (with both long and short variable names) that can be used when working with a rotating frame of reference. For an explanation of the column headings, see List of Field Variables (p. 503).

Long Variable Name	Short Variable Name	Units	Availability	Definition
Total Pressure in Stn Frame	ptotstn	[kg m^-1 s^-2]	2 A, C, M, P, R, TS	The velocity in the rotating frame of reference is defined as: $$\boldsymbol{U}_{\mathrm{rel}}=\boldsymbol{U}_{\mathrm{stn}}-\boldsymbol{\omega}\times\boldsymbol{R}$$ where ω is the angular velocity, $\boldsymbol{R}$ is the local radius vector, and $\boldsymbol{U}_{\mathrm{stn}}$ is velocity in the stationary frame of reference. For details, see Rotating Frame
Total Temperature in Stn Frame	Ttotstn	[K]	2 A, C, DT, M, P, R, TS	
Total Enthalpy in Stn Frame	htotstn	[kg m^2 s^-2]	2 A, C, M, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Mach Number in Stn Frame	Machstn	[ ]	1 A, C, M, R, TS	Quantities in the *CFX-Solver Theory Guide*.
Velocity in Stn Frame	velstn	[m s^-1]	1 A, C, M, R, TS	

16.2.7. Variables Relevant for Parallel Calculations

The following table contains a list of variables (with both long and short variable names) that can be used when working with parallel calculations. For an explanation of the column headings, see List of Field Variables (p. 503).

Long Variable Name	Short Variable Name	Units	Availability	Definition
Real Partition Number		[ ]	2 C, M, R	The partition that the node was in for the parallel run.

16.2.8. Variables Relevant for Multicomponent Calculations

The following table contains a list of variables (with both long and short variable names) that can be used when working with multicomponent calculations. For an explanation of the column headings, see List of Field Variables (p. 503).

Long Variable Name	Short Variable Name	Units	Availability	Definition
Mass Fraction	mf	[ ]	1 A, C, M, P, R, TS	The fraction of a component in a multicomponent fluid by mass.
Mass Concentration	mconc	[kg m^-3]	2 A, C, M, P, R, TS	The concentration of a component.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.2.9. Variables Relevant for Multiphase Calculations

The following table contains a list of variables (with both long and short variable names) that can be used when working with multiphase calculations. For an explanation of the column headings, see List of Field Variables (p. 503).

Note:

Variables with names shown in **bold text** are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from **Extra Output Variables List** on the **Results** tab of the **Solver** > **Output Control** details view of CFX-Pre.

Long Variable Name	**Short Variable Name**	**Units**	**Availability**	**Definition**
Interfacial Area Density	area density	[m^-1]	3 C	Interface area per unit volume for Eulerian multiphase fluid pairs.
Interphase Mass Transfer Rate	ipmt rate	[ ]	3 C	Interface mass transfer rate for Eulerian multiphase fluid pairs.
Mean Particle Diameter	mean particle diameter	[m]	3 A, C, M	Mean particle diameter for an Eulerian dispersed phase (including Polydispersed Fluids).
Volume Fraction	vf	[ ]	1 A, C, M, P, R, TS	For details, see Volume Fraction in the *CFX-Solver Modeling Guide*.
Conservative Volume Fraction	vfc	[ ]	2 A, C, M, R, TS	For details, see Volume Fraction in the *CFX-Solver Modeling Guide*.
Drift Velocity	drift velocity	[ ]	2 C, M, R, TS	Velocity of an algebraic slip component relative to the mixture.
Slip Reyn-olds Number	slip Re	[ ]	3 C	Reynolds number for Eulerian multiphase fluid pairs.
Slip Velocity	slipvel	[ ]	1 C, M, R, TS	Velocity of an algebraic slip component relative to the continuous component.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Surface Tension Coefficient	surface tension coefficient	[N m^-1]	2 C	Surface tension coefficient between fluids in a fluid pair.
Un-clipped Interfa-cial Area Density	unclipped area density	[m^-1]	3 C	Similar to area density, but values are not clipped to be non-zero.
Superficial Velocity	volflx	[m s^-1]	1 A, C, M, R, TS	The Fluid.Volume Fraction multiplied by the Fluid.Velocity.
Interphase Lift Force	MPFlift	[Kg m^-2 s^-1]	3	Force on the continuous phase due to MPF lift
Interphase Drag Force	MPFdrag	[Kg m^-2 s^-1]	3	Force on the continuous phase due to MPF drag
Interphase Turbulent Dispersion Force	MPFtdisp	[Kg m^-2 s^-1]	3	Force on the continuous phase due to MPF turbulent dispersion
Interphase Wall Lubrication Force	MPFlubr	[Kg m^-2 s^-1]	3	Force on the continuous phase due to MPF wall lubrication
Lubrication Force Direction	wlubdir	[]	3	Direction in which the wall lubrication force acts on the dispersed phase

16.2.10. Variables Relevant for Radiation Calculations

The following table contains a list of variables (with both long and short variable names) that can be used when working with radiation calculations. For an explanation of the column headings, see List of Field Variables (p. 503).

A **B** in the **Type** column indicates that the variable contains only non-zero values on the boundary of the model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Wall Radiative Heat Flux	Qrad	[W m^-2]	2,**B** DT, R, TS	Wall Radiative Heat Flux represents the net radiative energy flux leaving the boundary. It is computed as the difference between the radiative emission and the incoming radiative flux (Wall Irradiation Flux).
Wall Heat Flux	Qwall	[W m^-2]	2,**B** C, DT, R, TS	Wall Heat Flux is sum of the Wall Radiative Heat Flux and the Wall Convective Heat Flux. For an adiabatic wall, the sum should be zero.
Wall Irradiation Flux	irrad	[W m^-2]	2,**B** C, DT, R, TS	Wall Irradiation Flux represents the incoming radiative flux. It is computed as the solid angle integral of the incoming Radiative Intensity over a hemisphere on the boundary. For simulations using the multiband model, the Wall Irradiation Flux for each spectral band is also available for postprocessing.
Radiation Intensity	radint	[W m^-2 sr^-1]	RA, R, A, M, C, P, TS	Radiation Intensity represents the radiative energy flow (in units of energy per time) per unit solid angle (solid angle of the outgoing or incoming radiation, measured in steradians) and per unit area (area of a surface that is normal to the ray direction). **Note:** This quantity should not be used for quantitative calculations when computed using surface-to-surface models, since it has been agglomerated and it represents an average value over the whole domain.
Wall Absorbed Radiation Flux	sabsor	[W m^-2]	2,**B** DT, R, TS	Wall Absorbed Radiation Flux represents the absorbed heat flux due to radiation, and is computed

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				as Wall Radiative Heat Flux minus reflection and emission.
Incident Radiation	radinc	[kg s^-3]	1 C, DT, M, R, TS	The integral of `Radiation Intensity` over a full sphere of unit radius. For isotropic radiation, it is a factor of 4π greater than the `Radiation Intensity`.
Absorption Coefficient	absorp	[m^-1]	1 C, M, R, TS	For details, see Table 16.2: Common CEL Field Variables and Predefined Expressions (p. 521)
Scattering Coefficient	scatter	[m^-1]	1 C, M, R, TS	
Refractive Index	refrac	[ ]	1 C, R, TS	
Radiative Emission	rademis	[kg s^-3]	1 RA	
Extinction Coefficient	extinct	[m^-1]	1 C	
Emissivity	emis	[ ]	1 C	

16.2.11. Variables for Total Enthalpies, Temperatures, and Pressures

The following table lists the names of the various total enthalpies, temperatures, and pressures when visualizing results in CFD-Post or for use in CEL expressions. For an explanation of the column headings, see List of Field Variables (p. 503).

Long Variable Name	Short Variable Name	Units	Availability	Definition
Total Enthalpy	htot	[m^2 s^-2]	A, C, M, R, TS	h_{tot} For details, see Transport Equations in the *CFX-Solver Theory Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Rothalpy	rothalpy	[m^2 s^-2]	A, C, M, R, TS	I
Total Enthalpy in Stn Frame	htotstn	[m^2 s^-2]	A, C, M, R, TS	$h_{\mathrm{tot,stn}}$
Total Temperature in Rel Frame	Ttotrel	[K]	A, C, DT, M, P, R, TS	$T_{\mathrm{tot,rel}}$
Total Temperature	Ttot	[K]	A, C, DT, M, P, R, TS	T_{tot} For rotating domains, this is based on rothalpy; otherwise, this is based on total enthalpy.
Total Temperature in Stn Frame	Ttotstn	[K]	A, C, DT, M, P, R, TS	$T_{\mathrm{tot,stn}}$
Total Pressure in Rel Frame	ptotrel	[kg m^-1 s^-2]	A, C, M, P, R, TS	$P_{\mathrm{tot,rel}}$
Total Pressure	ptot	[kg m^-1 s^-2]	A, C, M, P, R, TS	P_{tot} For rotating domains, this is based on rothalpy; otherwise, this is based on total enthalpy.
Total Pressure in Stn Frame	ptotstn	[kg m^-1 s^-2]	A, C, M, P, R, TS	$P_{\mathrm{tot,stn}}$

16.2.12. Variables and Predefined Expressions Available in CEL Expressions

The following is a table of the more common variables and predefined expressions that are available for use with CEL when defining expressions. To view a complete list, open the **Expressions** workspace. For an explanation of the column headings, see List of Field Variables (p. 503).

Many variables and expressions have a long and a short form (for example, *Pressure* or *p*).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Additional Variables and expressions are available in CFD-Post. For details, see CFX Expression Language (CEL) in CFD-Post in the *CFD-Post User's Guide*.

Table 16.1: Common CEL Single-Value Variables and Predefined Expressions

Long Variable Name	Short Variable Name	Units	Availability	Definition
Accumulated Coupling Step	acplgstep	[ ]	2 C	These single-value variables enable access to timestep, timestep interval, and iteration number in CEL expressions. They may be useful in setting parameters such as the Physical Timescale via CEL expressions. For details, see Timestep, Timestep Interval, and Iteration Number Variables (p. 532).
Accumulated Iteration Number	aitern	[ ]	2 C	
Accumulated Time Step	atstep	[ ]	2 C	
Current Iteration Number	citern	[ ]	2 C	
Current Stagger Iteration	cstagger	[ ]	2 C	
Current Time Step	ctstep	[ ]	2 C	
Sequence Step	sstep	[ ]	2 C	
Time Step Size	dtstep	[s]	2 C	
Time	t	[s]	2 C	

Note:

Variables with names shown in **bold text** in the tables that follow are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Extra Output Variables List on the **Results** tab of the **Solver** > **Output Control** details view in CFX-Pre.

Table 16.2: Common CEL Field Variables and Predefined Expressions

Long Variable Name	Short Variable Name	Units	Availability	Definition
Axial Distance	aaxis	[m]	2 C	Axial spatial location measured along the locally-defined axis from the origin of the latter. When the locally-defined axis happens to be the Z axis, z and aaxis are identical.
Absorption Coefficient	absorp	[m^-1]	1 C, M, R, TS	The property of a medium that describes the amount of absorption of thermal radiation per unit path length within the medium. It can be interpreted as the inverse of the mean free path that a photon will travel before being absorbed (if the absorption coefficient does not vary along the path).
Boundary Distance	bnd distance	[m]	2 A, C, M, R, TS	
Boundary Scale	**bnd scale**	[m^-2]	3 C, M, R, TS	
Contact Area Fraction	**af**	[ ]	3 M	
[AV name]	[AV name]			Additional Variable name
Thermal Expansivity	beta	[K^-1]	2 C	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Effective Density	**deneff**	[kg m^-3]	3 A, C, M, R, TS	
Density	density	[kg m^-3]	2 A, C, M, P, R, TS	
Turbulence Eddy Dissipation	ed	[m^2 s^-3]	1 A, C, M, P, R, TS	
Eddy Viscosity	eddy viscosity	[kg m^-1 s^-1]	1 A, C, M, P, R, TS	
Emissivity	emis	[ ]	1 C	A characteristic of a surface that describes the fraction of emitted radiation with respect to the blackbody emission at the same temperature.
Extinction Coeffi-cient	**extinct**	[m^-1]	1 C	The property of a medium that describes the amount of absorption and scattering of thermal radiation per unit path length for propagation in the medium.
Initial Cartesian Coordinates	initcartcrd	[m]	2 C	The position of each node as it was at the start of the simulation (that is, the current position with Total Mesh Displacement subtracted). The individual components are referred to as "Initial X", "Initial Y" and "Initial Z".
Turbulence Kinetic Energy	ke	[m^2 s^-2]	1 A, C, M, P, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Mach Number	Mach	[ ]	1 A, C, M, R, TS	
Mach Number in Stn Frame	Machstn	[ ]	1 A, C, M, R, TS	Mach Number in Stationary Frame
Mass Concentration	mconc	[m^-3 kg]	2 A, C, M, P, R, TS	Mass concentration of a component
Mass Fraction	mf	[ ]	1 A, C, M, P, R, TS	
Conservat-ive Mass Fraction	**mfc**	[ ]	2 A, C, M, R, TS	
Mean Particle Diameter	mean particle diameter	[m]	3 C, P	
Mesh Displacement	meshdisp	[m]	3 C, M, R, TS	The displacement relative to the previous mesh
Mesh Expansion Factor	mesh exp fact	[ ]	2 C, M, R, TS	Ratio of largest to smallest sector volumes for each control volume.
Mesh Initialisation Time	meshinittime	[s]	2 C	Simulation time at which the mesh was last re-initialized (most often due to interpolation that occurs as part of remeshing)
Mixture Fraction	mixfrc	[ ]	1 A, C, M, R, TS	Mixture Fraction Mean
Mixture Model Length Scale	**mixture length scale**	[m]	3 M	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Mixture Fraction Variance	mixvar	[ ]	1 A, C, M, R, TS	
Molar Concentration	molconc	[m^-3 mol]	2 A, C, M, P, R, TS	
Molar Fraction	molf	[ ]	2 A, C, M, P, R, TS	
Molar Mass	**mw**	[kg mol^-1]	3 C, P	
Orthogonality Angle	orthangle	[rad]	2 C, M, R, TS	A measure of the average mesh orthogonality angle
Orthogonality Angle Minimum	orthanglemin	[rad]	2 C, M, R, TS	A measure of the worst mesh orthogonality angle
Orthogonality Factor	orthfact		2 C, M, R, TS	A non-dimensional measure of the average mesh orthogonality
Orthogonality Factor Minimum	orthfactmin		2 C, M, R, TS	A measure of the worst mesh orthogonality angle
Pressure	p	[kg m^-1 s^-2]	1 A, C, M, P, R, TS	
Absolute Pressure	pabs	[kg m^-1 s^-2]	2 A, C, M, R, TS	
Reference Pressure	pref	[kg m^-1 s^-2]	2 C	The `Reference Pressure` is the absolute pressure datum from which all other pressure values are taken. All relative pressure specifications

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				in CFX are relative to the `Reference Pressure`. For details, see Setting a Reference Pressure in the *CFX-Solver Modeling Guide*.
Distance from local Z axis	**r**	[m]	2 C	Radial spatial location. $r=\sqrt{x^2+y^2}$. For details, see CEL Variables r and theta (p. 531).
Radius	raxis	[m]	2 C	Radial spatial location measured normal to the locally-defined axis. When the locally-defined axis happens to be the Z axis, r and raxis are identical.
Radiative Emission	rademis	[kg s^-3]	1 RA	Blackbody radiative emission is defined as σ * Refractive Index^2 * Temperature^4, where σ represents the Stefan-Boltzmann constant.
Incident Radiation	radinc	[kg s^-3]	1 C, DT, M, R, TS	This is a volumetric quantity and has no relevance at boundaries. For relevant radiation quantities at boundaries, see the definitions for `Wall Radiative Heat Flux` and `Wall Ir radiation Flux` in Variables Relevant for Radiation Calculations (p. 516).
Radiation Intensity	radint	[kg s^-3]	1 A, C, M, P, R, TS	This is written to the results file for all radiation models. For Monte Carlo simulations, the `Radi`

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				`ation Intens ity.Normalized Std Deviation` is also written. This variable represents the statistical deviation with respect to the mean for the Monte Carlo model.
Refractive Index	refrac	[ ]	1 C, R, TS	A non-dimensional parameter defined as the ratio of the speed of light in a vacuum to that in a material.
Non dimensional radius	**rNoDim**	[ ]	2 C	Non-dimensional radius (available only when a rotating domain exists). For details, see CEL Variable rNoDim (p. 532).
Reynolds Stress	rs uu, rs vv, rs ww, rs uv, rs uw, rs vw	[m^2 s^-2]	2 A, C, M, P, R, TS	The six Reynolds Stress components
Statistical Reynolds Stress	rsstat uu, rsstat vv, rsstat ww, rsstat uv, rsstat uw, rsstat vw	[m^2 s^-2]	3 M, R	The six Statistical Reynolds Stress components
Scattering Coefficient	scatter	[m^-1]	1 C, M, R, TS	The property of a medium that describes the amount of scattering of thermal radiation per unit path length for propagation in the medium. It can be interpreted as the inverse of the mean free path that a photon will travel before undergoing scattering (if the

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				scattering coefficient does not vary along the path).
Soot Mass Fraction	sootmf	[ ]	1 A, C, M, R, TS	
Soot Nuclei Specific Concentration	sootncl	[m^-3]	1 A, C, M, R, TS	
Specific Volume	**specvol**	[m^3 kg^-1]	3 A, C, M, R, TS	
Local Speed of Sound	**speedof-sound**	[m s^-1]	2 C, M, R, TS	
Subdo-main	**subdo-main**	[ ]	2 C	Subdomain variable (1.0 in subdomain, 0.0 elsewhere). For details, see CEL Variable "subdomain" and CEL Function "inside" (p. 532).
inside() @<Loca-tions>	**inside() @<Loca-tions>**			inside variable (1.0 in subdomain, 0.0 elsewhere). For details, see CEL Variable "subdomain" and CEL Function "inside" (p. 532).
Theta	taxis	[rad]	2 C	taxis is the angular spatial location measured around the locally-defined axis, when the latter is defined by the Coordinate Axis option. When the locally defined axis is the z(/x/y)-axis, taxis is measured from the x(/y/z)-axis, positive

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				direction as per right-hand rule.
Turbulence Eddy Frequency	tef	[s^-1]	1 A, C, M, P, R, TS	
Angle around local Z axis	**theta**	[rad]	2 C	Angle, arctan(y/x). For details, see CEL Variables r and theta (p. 531).
Total Mesh Displacement	meshdisptot	[m]	1 C, DT, M, R, TS	The total displacement relative to the initial mesh
Velocity u Velocity v Velocity w	u v w	[m s^-1]	1 A, C, M, P, R, TS	Velocity in the x, y, and z coordinate directions
Velocity in Stn Frame u Velocity in Stn Frame v Velocity in Stn Frame w	velstn u velstn v velstn w	[m s^-1]	1 A, C, M, R, TS	Velocity in Stationary Frame in the x, y, and z coordinate directions
Volume Fraction	vf	[ ]	1 A, C, M, P, R, TS	
Conservative Volume Fraction	vfc	[ ]	2 A, C, M, R, TS	The variable `<fluid>.Conservative Volume Fraction` should not usually be used for postprocessing.
Kinematic Viscosity	**visckin**	[m^2 s^-1]	2 A, C, M, P, R, TS	
Wall Distance	wall distance	[m]	2	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
			A, C, M, P, R, TS	
Wall Power Density	wall powerdens	[W m^-2]	2 C, DT, R, TS	Time rate of work done by a wall moving against the contacting fluid, per unit of wall area This quantity is intended primarily for use in blade flutter analysis. The wall motion considered is measured in the same frame of reference as the blade (whether rotating or stationary). Only the component of wall motion normal to the wall surface is considered. The component of wall motion tangential to the wall surface is neglected. This variable is 0 at stationary walls.
Wall Work Density	wall workdens	[J m^-2]	2 C, DT, R, TS	Work done by a wall moving against the contacting fluid, per unit wall area, over the current time step This quantity is intended primarily for use in blade flutter analysis. The wall motion considered is measured in the same frame of reference as the blade (whether rotating or stationary). Only the component of wall motion normal to the wall surface is

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
				considered. The component of wall motion tangential to the wall surface is neglected. This variable is 0 at stationary walls.
Wall Periodic Work Density	wall perworkdens	[J m^-2]	2 C, DT, R, TS	Work done by a wall moving against the contacting fluid, per unit wall area, over one vibration period This quantity is intended primarily for use in blade flutter analysis. The wall motion considered is measured in the same frame of reference as the blade (whether rotating or stationary). Only the component of wall motion normal to the wall surface is considered. The component of wall motion tangential to the wall surface is neglected. This quantity is computed based on Fourier coefficient data. This variable is 0 at stationary walls.
Wall Scale	**wall scale**	[m^2]	3 M, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.2.12.1. System Variable Prefixes

In order to distinguish system variables of the different components and fluids in your CFX model, prefixes are used. For example, if carbon dioxide is a material used in the fluid `air`, then some of the system variables that you might expect to see are:

- `air.density` - the density of air
- `air.viscosity` - the viscosity of air
- `air.carbondioxide.mf` - the mass fraction of carbon dioxide in air
- `air | water.surface tension coefficient` – the surface tension coefficient between air and water
- `air | water.area density` – the interfacial area density between air and water.

In a single phase simulation the fluid prefix may be omitted.

For multiphase cases, a fluid prefix indicates either a specific fluid, or a specific fluid pair. The absence of a prefix indicates a bulk or fluid independent variable, such as pressure.

For porous solids, those variables that exist in the solid are prefixed by the name of the solid phase.

16.2.12.2. CEL Variables r and theta

`r` is defined as the normal distance from the third axis with respect to the reference coordinate frame. `theta` is defined as the angular rotation about the third axis with respect to the reference coordinate frame. For details, see Coordinate Frames in the *CFX-Solver Modeling Guide*.

The variables `Radius` and `theta` are available only when the rotational axis has been defined. The rotational axis can either be defined in the results file or in CFD-Post through the **Initialization** panel in the **Turbo** workspace.

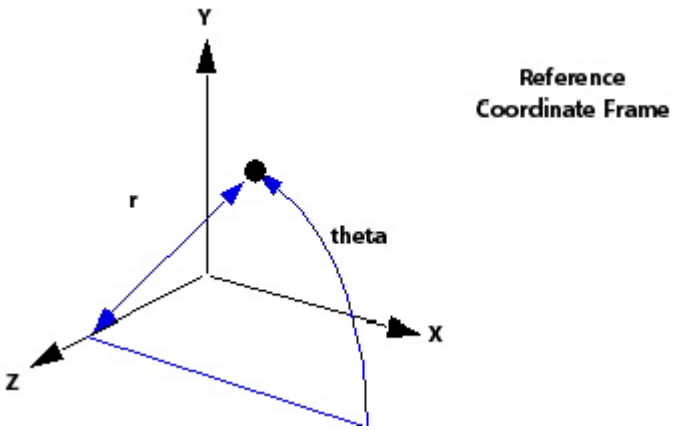
Note:

`theta` is expressed in radians and will have values between $-\pi$ and π.

`r` and `theta` are particularly useful for describing radial distributions, for instance the velocity profile at the inlet to a pipe.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Figure 16.1: r and theta with Respect to the Reference Coordinate Frame



16.2.12.3. CEL Variable rNoDim

`rNoDim` is a dimensionless system variable that can be useful for rotating machinery applications. It is a ratio of radii, defined to be zero at the minimum radius and unity at the maximum radius, so that in general:

$$rNoDim = \frac{R - R_{\min}}{R_{\max} - R_{\min}}$$

where R is the radius of any point in the domain from the axis of rotation. `rNoDim` is only available for domains defined with a rotating frame of reference.

16.2.12.4. CEL Variable "subdomain" and CEL Function "inside"

`subdomain` is essentially a step function variable, defined to be unity within a subdomain and zero elsewhere. This is useful for describing different initial values or fluid properties in different regions of the domain. It works in all subdomains but cannot be applied to specific subdomains (for example, an expression for temperature in a subdomain could be `373*subdomain [K]`).

The `inside` CEL function can be used in a similar way to the `subdomain` variable, but allows a specific 2D or 3D location to be given. For example, `273 [K] * inside()@Subdomain 1` has a value of 273 [K] at points in `Subdomain 1` and 0 [K] elsewhere. Furthermore, the location can be any 2D or 3D named sub-region of the physical location on which the expression is evaluated. The location can also be an immersed solid domain.

16.2.12.5. Timestep, Timestep Interval, and Iteration Number Variables

These variables enable access to timestep, timestep interval, and iteration number in CEL expressions. They may be useful in setting parameters such as the Physical Timescale via CEL expressions.

16.2.12.5.1. Steady-State Runs

In steady-state runs, only Accumulated Iteration Number (or, equivalently Accumulated Time Step) and Current Iteration Number (or, equivalently Current Time Step) are of use. Current Iteration Number gives the outer iteration number of the current run. The outer iteration number begins

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

at 1 for each run, irrespective of whether it is a restarted run. Accumulated Iteration Number gives the accumulated outer iteration number, which accumulates across a restarted run.

16.2.12.5.2. Transient Runs

In transient runs, Accumulated Time Step and Current Time Step are used for the accumulated and current timestep numbers of the outer timestep loop. Current Iteration Number gives the current coefficient loop number within the current timestep. Thus, Current Iteration Number will cycle between 1 and n for each timestep during a transient run, where n is the number of coefficient loops. Accumulated Iteration Number is equivalent to Current Iteration Number for transient runs.

16.2.12.5.3. Timestep Variables in CFD-Post

In CFD-Post, Sequence Step (`sstep`) is the 'global' sequence timestep. It is equivalent to the **Step** value in Timestep Selector in the *CFD-Post User's Guide*.

Accumulated Time Step (`atstep`) and Current Time Step (`ctstep`) are available for both steady-state and transient runs, with Current Time Step being set to the same value as Accumulated Time Step.

Accumulated Iteration Number (`aitern`), Current Iteration Number (`citern`), Accumulated Coupling Step (`acplgstep`), and Current Stagger Iteration (`cstagger`) are not available in CFD-Post.

If multiple cases are loaded, the values obtained from evaluating these timestep-related variables relate to the last loaded case.

16.2.12.6. Expression Names

Your CEL expression name can be any name that does not conflict with the name of a CFX system variable, mathematical function, or an existing CEL expression. The `RULES` and `VARIABLES` files provide information on valid options, variables, and dependencies. Both files are located in *<CFX-ROOT>*`/etc/` and can be viewed in any text editor.

16.2.12.7. Scalar Expressions

A *scalar expression* is a real valued expression using predefined variables, user variables, and literal constants (for example, 1.0). Note that literal constants have to be of the same dimension. Scalar expressions can include the operators + - * / and ^ and several of the mathematical functions found in standard Fortran (for example, sin() and exp()).

An expression's value is a real value and has specified dimensions (except where it is dimensionless - but this is also a valid dimension setting).

For example, if *t* is time and *L* is a length then the result of *L*/*t* has the same dimensions as speed.

The + and - operators are only valid between expressions with the same dimensions and result in an expression of those dimensions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The * and / operators combine the dimensions of their operands in the usual fashion. *X*^*I*, where *I* is an integer, results in an expression whose dimensions are those of *X* to the power *I*. The trigonometric functions all work in terms of an angle in radians and a dimensionless ratio.

16.2.12.8. Expression Properties

There are three properties of expressions:

- An expression is a simple expression if the only operations are +, -, *, / and there are no functions used in the expression.
- An expression is a constant expression if all the numbers in the expression are explicit (that is, they do not depend on values from the solver).
- An expression is an integer expression if all the numbers in the expression are integers and the result of each function or operation is an integer.

For example, (3+5)/2 is a simple, constant, integer expression. However, 2*(1/2) is not a constant integer expression because the result of 1/2 is 0.5, not an integer. Also 3.*4 is not a constant integer expression because 3 is not an integer. Moreover, 2^3 is not a simple, constant, integer expression because ^ is not in the list (+, -, *, /).

Expressions are evaluated at run time and in single precision floating point arithmetic.

16.2.12.9. Available and Unavailable Variables

CFX System Variables and user-defined expressions will be available or unavailable depending on the simulation you are performing and the expressions you want to create. In some circumstances, System Variables are logically unavailable; for instance, time (`t`) is not available for steady-state simulations. In others, the availability of a System Variable is not allowed for physical model reasons. For example, density can be a function of pressure (`p`), temperature (`T`) and location (`x`, `y`, `z`), but no other system variables.

Information on how to find dependencies for all parameters is available in the `RULES` and `VARIABLES` files. Both files are located in `<CFXROOT>/etc/` and can be viewed in any text editor.

The expression definition can depend on any system variable. If, however, that expression depends on a system variable that is unavailable for a particular context, then that expression will also be unavailable.

16.3. Particle Variables Generated by the Solver

This section describes the following types of particle variables that you may have defined in CFX-Pre or that are available for viewing in CFD-Post and exporting to other files. Many variables are relevant only for specific physical models.

16.3.1. Particle Track Variables

16.3.2. Particle Field Variables

Some variables are defined only on the boundaries of the model. When using these variables in CFD-Post, there are a limited number of useful things that you can do with these. For details, see Boundary-Value-Only Variables in the *CFD-Post User's Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following information is given for particle variables described in this section:

- **Long Variable Name**: The name that you see in the user interface.
- **Short Variable Name**: The name that must be used in CEL expressions.
- **Units**: The default units for the variable. An empty entry [ ] indicates a dimensionless variable.

> **Note:**
>
> The entries in the Units columns are SI but could as easily be any other system of units.

- **Type (User Level, Boundary)**

  User Level: This number is useful when using the CFX Export facility. For details, see File Export Utility in the *CFX-Solver Manager User's Guide*. Note that the CFX-Solver may sometimes override the user-level setting depending on the physics of the problem. In these cases, the User Level may be different from that shown in the table below.

  Boundary (B): A **B** in this column indicates that the variable contains only non-zero values on the boundary of the model. See Boundary-Value-Only Variables in the *CFD-Post User's Guide* for more details.

This section does not cover the complete list of variables. For information on obtaining details on all variables, see RULES and VARIABLES Files in the *CFX-Solver Manager User's Guide*.

16.3.1. Particle Track Variables

Particle track variables are particle variables that are defined directly on each track. These variables are defined on the particle positions for which track information is written to the results file. Direct access to the particle track variables outside of CFD-Post is possible only if the raw track file is kept after a particle run.

You can do the following with particle track variables:

- Export them from particle tracks in CFD-Post
- Use them to color particle tracks in CFD-Post
- Use them in particle histograms
- Use them as input to Particle User Fortran

Particle track variables are not available for use in CEL expressions and general User Fortran. In addition, they cannot be monitored during a simulation.

The following table lists particle track variables that are available in general, including in CFD-Post:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Description	Availability
<Particle Type>.Mean Particle Diameter	mean particle diameter	[m]	Particle diameter	3 PR
<Particle Type>.<Particle Component>.Mass Fraction		[-]	Note: Only available for multi-component particles. Fraction of mass of a particular particle component	
<Particle Type>.Particle Number Rate	particle number rate	[s^-1]	Particle number rate	3 PR
<Particle Type>.Particle Time	pttime	[s]	Simulation time	2 PR
<Particle Type>.Particle Traveling Distance	ptdist	[m]	Distance along the particle track measured from the injection point	2 PR
<Particle Type>.Particle Traveling Time		[s]	Time measured from the time of injection of the particle. For steady-state simulations only, this time is identical to <Particle Type>.Particle Time.	2 PR
<Particle Type>.Particle Weber Number		[-]	Note: Only available if a secondary droplet breakup model is active. Particle Weber number along track $$We=\rho_f V_{slip}{}^2 \frac{d_p}{\sigma}$$ where ρ_f=fluid density V_{slip}=slip velocity d_p=particle diameter σ=surface tension coefficient	
<Particle Type>.Temperature	T	[K]	Particle temperature	1 PR
<Particle Type>.Total Particle Mass	ptmasst	[kg]	Particle total mass	2 PR

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Description	Availability
<Particle Type>.Velocity		[m s^-1]	Particle velocity	1 PR
<Particle Type>.Velocity u <Particle Type>.Velocity v <Particle Type>.Velocity w	u v w	[m s^-1]	Particle velocity components in X, Y, and Z direction	1 PR

For Particle User Fortran, these additional track variables, which are not available in CFD-Post, can be specified in the argument list for the user routine:

Long Variable Name	Short Variable Name	Units	Definition	Availability
Particle Eotvos Number	pteo	[ ]	$\mathrm{Eo}_p=\lvert g\rvert\cdot\lvert\rho_{\mathrm{fl}}-\rho_{\mathrm{p}}\rvert\cdot\frac{{d_{\mathrm{p}}}^2}{\sigma}$	2 PR
Particle Morton Number	ptmo	[ ]	$Mo_p=\lvert g\rvert\cdot\lvert\rho_{\mathrm{fl}}-\rho_{\mathrm{p}}\rvert\cdot\frac{{\mu_{\mathrm{fl}}}^4}{({\rho_{\mathrm{fl}}}^2\cdot\sigma^3)}$	2 PR
Particle Nusselt Number	ptnu	[ ]	$\mathrm{Nu}_{\mathrm{p}}=2+0.6\sqrt{\mathrm{Re}_{\mathrm{p}}}\cdot\left[\frac{\mu_{\mathrm{fl}}\cdot c_{p,\mathrm{fl}}}{\lambda_{\mathrm{fl}}}\right]^{\frac{1}{3}}$	2 PR
Particle Ohnesorge Number	pton	[ ]	$\mathrm{Oh}_{\mathrm{p}}=\frac{\mu_{\mathrm{p}}}{\sqrt{\rho_{\mathrm{p}}d_{\mathrm{p}}\sigma}}$	2 PR
Particle Reynolds Number	ptre	[ ]	$\mathrm{Re}_{\mathrm{p}}=d_{\mathrm{p}}V_{\mathrm{slip,p}}\cdot\frac{\rho_{\mathrm{fl}}}{\mu_{\mathrm{fl}}}$	2 PR
Particle Slip Velocity	ptslipvel	[m s^-1]	$V_{\mathrm{slip,p}}=\lvert U_{\mathrm{fl}}-U_{\mathrm{p}}\rvert$	2 PR
Particle Position	ptpos	[m]	Cartesian coordinates of current particle position	2 PR
Particle Impact Angle [a]	particle impact angle	[radian]		3 PR

[a] Note: The impact angle is measured from the wall.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The following particle variables are available only in particle histograms:

Long Variable Name	Short Variable Name	Units	Description	Availability
<Particle Type>.Particle Mass Flow Rate		[kg s^-1]	Particle mass flow rate = Particle Total Mass * Particle Number Rate	3 PR
<Particle Type>.Velocity Magnitude		[m s^-1]	Particle Speed	2 PR

16.3.2. Particle Field Variables

Particle field variables are particle variables that are defined at the vertices of the fluid calculation. In contrast to track variables, these variables can be used in the same way as "standard" Eulerian variables. This means that particle field variables are available for use in CEL expressions and User Fortran, they can be monitored during a simulation, and are available for general post-processing in CFD-Post. Additionally, particle field variables can be used in the same way as particle track variables as input to particle User Fortran and for coloring tracks. When used for coloring tracks, the field variables have to be interpolated onto the tracks, and so this operation will be slower than coloring with a track variable.

The following particle variables are available as field variables:

16.3.2.1. Particle Sources into the Coupled Fluid Phase

For fully-coupled particle simulations involving energy, momentum and mass transfer to the fluid phase, the following variables are written to the results file:

Long Variable Name	Short Variable Name	Units	Availability
Particle Energy Source	ptenysrc	[W m^-3]	2 A, C, M, P, R
Particle Energy Source Coefficient	ptenysrcc	[W m^-3 K^-1]	2 A, C, M, P, R
Particle Momentum Source	ptmomsrc	[kg m^-2 s^-2]	2 A, C, M, P, R
Particle Momentum Source Coefficient	ptmomsrcc	[kg m^-3 s^-1]	2 A, C, M, P, R
Total Particle Mass Source	ptmassrctot	[kg s^-1 m^-3]	2

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability
			A, C, M, P, R
Total Particle Mass Source Coefficient	ptmassrcctot	[kg s^-1 m^-3]	2 A, C, M, P, R
For multi-component mass transfer, the following Additional Variables are available [a]:			
Particle Mass Source	ptmassrc	[kg s^-1 m^-3]	2 A, C, M, P, R
Particle Mass Source Coefficient	ptmassrcc	[kg s^-1 m^-3]	2 A, C, M, P, R

[a] The variables for multi-component take the following form: <Particle Type>.<Particle Component>.<Variable Name>

Particle source terms are accumulated along the path of a particle through a control volume and stored at the corresponding vertex. A smoothing procedure can be applied to the particle source terms, which may help with convergence or grid independence. For details, see Particle Source Smoothing in the *CFX-Solver Modeling Guide*.

16.3.2.2. Particle Radiation Variables

Long Variable Name	Short Variable Name	Units	Availability
Particle Radiative Emission	ptremiss	[W m^-3]	2 A, C, M, P, R
Particle Absorption Coefficient	ptabscoef	[m^-1]	2 A, C, M, P, R

Particles can also interact with the radiation field and either emit or absorb radiation.

16.3.2.3. Particle Vertex Variables

By default, particle vertex variables are not written to the results file, except for the `Averaged Volume Fraction`. The other vertex variables can be written to the results file if they are selected from **Extra Output Variables List** in the **Output Control** section of CFX-Pre or if they are used in a monitor point, CEL expression or in (Particle) User Fortran.

The following particle variables are available:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability
Averaged Velocity	averaged vel	[m s^-1]	1 A, C, M, P, PR, R
Averaged Volume Fraction	vfpt	[ ]	1 A, C, M, P, PR, R
Averaged Temperature	averaged temperature	[K]	1 A, C, M, P, PR, R
Averaged Mass Fraction [a]	averaged mf	[ ]	1 A, C, M, P, PR, R
Averaged Particle Time	averaged pttime	[s]	2 A, C, M, P, PR, R
Averaged Mean Particle Diameter (D43)	averaged mean particle diameter	[m]	2 A, C, M, P, PR, R
Averaged Arithmetic Mean Particle Diameter (D10)	averaged arithmetic mean particle diameter	[m]	2 A, C, M, P, PR, R
Averaged Surface Mean Particle Diameter (D20)	averaged surface mean particle diameter	[m]	2 A, C, M, P, PR, R
Averaged Volume Mean Particle Diameter (D30)	averaged volume mean particle diameter	[m]	2 A, C, M, P, PR, R
Averaged Sauter Mean Particle Diameter (D32)	averaged sauter mean particle diameter	[m]	2 A, C, M, P, PR, R
Averaged Mass Mean Particle Diameter (D43)	averaged mass mean particle diameter	[m]	2 A, C, M, P, PR, R

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability
Averaged Particle Number Rate	averaged particle number rate	[s^-1]	2 A, C, M, P, PR, R
For simulations with the particle wall film model activated, the following additional vertex variables are available:			
Averaged Volume Fraction Wall	vfptw	[ ]	1 A, C, M, P, PR, R
Averaged Film Temperature	averaged film temperature	[K]	1 A, C, M, P, PR, R

[a] This variable takes the following form: <Particle Type>.<Particle Component>.<Variable Name>

The following are the formulae for particle vertex fields' size distributions:

Arithmetic Mean Diameter	$d_{10}=\frac{\sum n_i d_i}{\sum n_i}$
Surface Mean Diameter	$d_{20}=\sqrt{\frac{\sum n_i d_i^2}{\sum n_i}}$
Volume Mean Diameter	$d_{30}=\sqrt[3]{\frac{\sum n_i d_i^3}{\sum n_i}}$
Sauter Mean Diameter	$d_{32}=\frac{\sum n_i d_i^3}{\sum n_i d_i^2}$
Mass Mean Diameter	$d_{43}=\frac{\sum n_i d_i^4}{\sum n_i d_i^3}$

16.3.2.3.1. Variable Calculations

Particle vertex variables are calculated using the following averaging procedure:

$$\overline{\Phi}_P=\frac{\sum\left(\Delta t\, m_P \dot{N}_P \Phi_P\right)}{\sum\left(\Delta t\, m_P \dot{N}_P\right)} \tag{16.1}$$

With:

- Σ: Sum over all particles and time steps in a control volume
- Δt: Particle integration time step
- $\dot{N}_P$: Particle number rate
- m_P: Particle mass
- Φ: Particle quantity

Slightly different averaging procedures apply to particle temperature and particle mass fractions:

Averaged Particle Temperature

$$\bar{\Phi}_P = \frac{\sum\left(\Delta t\, m_P\, \dot{N}_P\, c_{P,P}\, T_P\right)}{\sum\left(\Delta t\, m_P\, \dot{N}_P\, c_{P,P}\right)} \tag{16.2}$$

With:

Averaged Mass Fraction

- $c_{P,P}$: Particle specific heat capacity
- T_P: Particle temperature

$$\bar{\Phi}_P = \frac{\sum\left(\Delta t\, m_{c,P}\, \dot{N}_P\right)}{\sum\left(\Delta t\, m_P\, \dot{N}_P\right)} \tag{16.3}$$

With:

- $m_{c,P}$: Mass of species c in the particle

Due to the discrete nature of particles, vertex variables may show an unsmooth spatial distribution, which may lead to robustness problems. To reduce possible problems a smoothing option is available. For details, see Vertex Variable Smoothing in the *CFX-Solver Modeling Guide*.

16.3.2.4. Particle Boundary Vertex Variables

Particle-boundary vertex variables are particle variables that are defined on the vertices of domain boundaries. They are normalized with the face area of the corresponding boundary control volume.

You can use these variables to color boundaries and to compute average or integrated values of the corresponding particle quantities.

You cannot use these variables in CEL expressions or User Fortran, and you cannot monitor them during a simulation.

Long Variable Name	Units	Availability
Available at inlet, outlet, openings and interfaces:		
Mass Flow Density	[kg m^-2 s^-1]	2

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Units	Availability
		B, R
Momentum Flow Density	[kg m^-1 s^-2]	2 **B**, R
Energy Flow Density	[kg s^-3]	2 **B**, R
Available at walls only:		
Wall Stress	[kg m^-1 s^-2]	2 **B**, R
Wall Energy Flow Density	[kg s^-3]	2 **B**, R
Wall Mass Flow Density	[kg m^-2 s^-1]	2 **B**, R
Erosion Rate Density	[kg m^-2 s^-1]	2 **B**, R
Available in transient runs:		
Time Integrated Mass Flow Density	[kg m^-2]	2 **B**, R
Time Integrated Momentum Flow Density	[kg m^-1 s^-1]	
Time Integrated Energy Flow Density	[kg s^-2]	2 **B**, R
Time Integrated Wall Energy Flow Density	[kg s^-2]	2 **B**, R
Time Integrated Wall Mass Flow Density	[kg m^-2]	2 **B**, R
Time Integrated Erosion Rate Density	[kg m^-2]	2 **B**, R

16.3.2.5. Particle RMS Variables

For some applications, it may be necessary to not only provide the mean values of particle quantities, but also their standard deviation in the form of particle RMS variables. Similar to particle vertex variables, these variables are also defined at the vertices of the fluid calculation. Particle RMS variables are available for use in CEL expressions and User Fortran; they can be monitored during a simulation, and are available for general postprocessing in CFD-Post. Additionally, particle RMS variables can be used in the same way as particle track variables as input to particle User Fortran and for coloring tracks.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

By default, particle RMS variables are not written to the results file; unless, they have been explicitly requested by you (selected from **Extra Output Variables List** in the **Output Control** section of CFX-Pre, usage in a CEL expression or in User Fortran) or if the stochastic particle collision model is used in a simulation.

The following particle variables are available as field variables, particularly useful for simulations that use the stochastic particle collision model:

Long Variable Name	Short Variable Name	Units	Availability
RMS Velocity	rms velocity	[m s^-1]	1 A, C, M, P, PR, R
RMS Temperature	rms temperature	[K]	1 A, C, M, P, PR, R
RMS Mean Particle Diameter	rms mean particle diameter	[m]	3 A, C, M, P, PR, R
RMS Particle Number Rate	rms particle number rate	[s^-1]	3 A, C, M, P, PR, R

16.3.2.5.1. Variable Calculations

Particle RMS variables are calculated using the following procedure:

$$\Phi = \overline{\Phi} + \Phi''$$
$$\Phi_{rms} = \sqrt{\overline{\Phi''^2}} = \sqrt{\overline{\left(\Phi - \overline{\Phi}\right)^2}} = \sqrt{\overline{\Phi^2} - \overline{\Phi}^2} \quad (16.4)$$

With:

- Φ: Instantaneous particle quantity
- $\overline{\Phi}$: Average particle quantity
- Φ'': Fluctuating particle quantity
- $\overline{\Phi^2}$: Average of square of particle quantity
- $\overline{\Phi}^2$: Square of average of particle quantity

A smoothing option, as available for particle vertex variables, is available for particle RMS variables. For details, see Vertex Variable Smoothing in the *CFX-Solver Modeling Guide*.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

16.4. Miscellaneous Variables

Note:

Variables with names shown in **bold text** are not output to CFD-Post. However, some of these variables can be output to CFD-Post by selecting them from **Extra Output Variables List** on the **Results** tab of the **Solver** > **Output Control** details view of CFX-Pre.

In the **Availability** column:

- A number represents the user level (1 indicates that the variable appears in default lists, 2 and 3 indicate that the variable appears in extended lists that you see when you click)
- A indicates the variable is available for mesh adaption
- C indicates the variable is available in CEL
- DT indicates the variable is available for data transfer to Ansys
- M indicates the variable is available for monitoring
- P indicates the variable is available for particle user routine argument lists
- PR indicates the variable is available for particle results
- R indicates the variable is available to be output to the results, transient results, and backup files
- RG indicates the variable is available for real gases
- TS indicates the variable is available for transient statistics

Long Variable Name	**Short Variable Name**	**Units**	**Availability**	**Definition**
Aspect Ratio	aspect ratio	[ ]	2 C, M, R, TS	
Autoignition	autoignition	[ ]	1 A, C, M, R, TS	
Boundary Scale	bnd scale	[ ]	3 C, M, R, TS	Similar to wall scale, this variable is used for controlling mesh stiffness near boundaries for moving mesh problems.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Burnt Absolute Temperature	burnt Tabs	[K]	2 A, C, M, R, TS	
Burnt Density	burnt density	[kg m^-3]	2 A, C, M, R, TS	
Clipped Pressure	pclip	[Pa]	1 M, R, TS	Negative absolute values clipped for cavitation
Conservative Size Fraction	sfc	[ ]	2 A, C, M, R, TS	
Courant Number	courant	[ ]	2 C, M, R, TS	
Cumulative Size Fraction	csf	[ ]	2 A, C, M, R, TS	
Current Density	jcur		1 C, M, R, TS	
Dynamic Diffusivity	diffdyn		2 C, M, P, R, TS	
Electric Field	elec		1 C, M, R, TS	
Electric Potential	epot		1 C, M, R, TS	
Electrical Conductivity	conelec		3 C, M, R, TS	
Electrical Permittivity	permelec		3 C, M, R, TS	
Electromagnetic Force Density	bfemag		3 R	
Equivalence Ratio	equivratio	[ ]	2 A, C, M, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
External Magnetic Induction	bmagext	[ ]	1 M, R, TS	External magnetic induction field specified by the user.
First Blending Function for BSL and SST model	sstbf1	[ ]	3 C, M, R, TS	
Second Blending Function for SST model	sstbf2	[ ]	3 C, M, R, TS	
Flame Surface Density	fsd	[m^-1]	1 A, C, M, R, TS	Combustion with flame surface density models.
Specific Flame Surface Density	spfsd		2 A, C, M, R, TS	Combustion with flame surface density models.
Frequency	freq		3 C	
Fuel Tracer	trfuel	[ ]	1 A, C, M, R, TS	Residual material model or exhaust gas recirculation (EGR)
Granular Temperature	grantemp	[m^2 s^-2]	1 A, C, M, R, TS	
Group I Index	groupi	[ ]	2 C	
Group J Index	groupj	[ ]	2 C	
Group I Diameter	diami		2 C	
Group J Diameter	diamj		2 C	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Group I Mass	massi		2 C	
Group J Mass	massj		2 C	
Group I Lower Mass	massi lower		2 C	
Group J Lower Mass	massj lower		2 C	
Group I Upper Mass	massi upper		2 C	
Group J Upper Mass	massj upper		2 C	
Ignition Delay Elapsed Fraction	ignfrc	[ ]	2 A, C, M, R, TS	
Ignition Delay Time	tigndelay	[s]	2 A, C, M, R, TS	
Particle Integration Timestep	particle integration timestep	[s]	3 P	
Isentropic Compressibility	compisoS	[m s^2 kg^-1]	2 C, M, R	$\left(\frac{1}{\rho}\right)\ \left(\frac{\partial \rho}{\partial p}\right)_s$
Isentropic Compression Efficiency	icompeff	[ ]	2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Isentropic Expansion Efficiency	iexpeff	[ ]	2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Isentropic Total Enthalpy	htotisen		2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Isentropic Static Enthalpy	enthisen		2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Isobaric Compressibility	compisoP	[K^-1]	2 C, M, R	$-\frac{1}{\rho}\frac{\partial \rho}{\partial T}\Big\vert_{p}$

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Isothermal Compressibility	compisoT	[m s^2 kg^-1]	2 C, M, R	$\frac{1}{\rho}\frac{\partial \rho}{\partial p}\Big\vert_T$
LES Dynamic Model Coefficient	dynmc	[ ]	1 A, C, M, P, R, TS	
Laminar Burning Velocity	velburnlam	[m s^-1]	2 A, C, R, TS	
Lighthill Stress	lighthill stress tensor		2 A, C, M, R, TS	
Magnetic Induction	bmag		1 C, M, R, TS	
Magnetic Field	hmag		2 C, M, R, TS	
Magnetic Vector Potential	bpot		1 C, M, R, TS	
Magnetic Permeability	permmag		3 C, M, R, TS	
External Magnetic Induction	bmagext		1 C, M, R, TS	
Mass Flux	mfflux		2 R	
Mesh Diffusivity	diffmesh	[m^2 s^-1]	2 C, M, R, TS	
Normal Area	normarea	[ ]	2 C	Normal area vectors.
Total Force Density	forcetden		3 DT	
Total Pressure in Rel Frame	ptotrel		2 A, C, M, P, R, TS	Based on relative frame total enthalpy.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Turbulent Burning Velocity	velburnturb	[m s^-1]	2 A, C, R, TS	
Mesh Velocity	meshvel		1 C, M, R, TS	
Mixture Fraction Scalar Dissipation Rate	mixsclds	[s^-1]	3 A, C, M, R, TS	
Molar Reaction Rate	reacrate		2 C, R, TS	
Nonclipped Absolute Pressure	pabsnc		3 A, C, M, R, TS	Nonclipped absolute pressure for cavitation source. This is written to the `.res` file for all cases that have cavitation.
Nonclipped Density	densitync	[kg m^-3]	2 C	Nonclipped density for cavitation source
Normal Vector	normal	[ ]	2 C	
Orthogonality Factor Minimum	orthfactmin	[ ]	2 C, M, R, TS	
Orthogonality Factor	orthfact	[ ]	2 C, M, R, TS	
Orthogonality Angle Minimum	orthanglemin		2 C, M, R, TS	
Orthogonality Angle	orthangle		2 C, M, R, TS	
Particle Laplace Number	ptla	[ ]	2 P	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Particle Turbulent Stokes Number	ptstt	[ ]	2 P	
Polytropic Compression Efficiency	pcompeff	[ ]	2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Polytropic Expansion Efficiency	pexpeff	[ ]	2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Polytropic Total Enthalpy	htotpoly		2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Polytropic Static Enthalpy	enthpoly		2 C, M, R, TS Available only when used in a built-in efficiency calculation; otherwise unavailable due to lack of association with an inlet/outlet.	
Reaction Progress	reacprog	[ ]	1 A, C, M, R, TS	For premixed or partially premixed combustion.
Weighted Reaction Progress	wreacprog	[ ]	2 A, C, M, R, TS	For premixed or partially premixed combustion.
Weighted Reaction Progress Source	wreacprogsrc		3 A, C, R, TS	For premixed or partially premixed combustion.
Residual Products Mass Fraction	mfresid	[ ]	1 A, C, M, R, TS	Residual material model or exhaust gas recirculation (EGR)
Residual Products Molar Fraction	molfresid	[ ]	2 A, C, M, R, TS	Residual material model or exhaust gas recirculation (EGR)
Restitution Coefficient	restitution coefficient	[ ]	3 C, M, R, TS	
Rotation Velocity	rotvel		2 C, R, TS	
Rotational Energy	rotenergy		2 C, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Shear Velocity	ustar		2 C	
Size Fraction	sf	[ ]	1 A, C, M, R, TS	
Solid Bulk Viscosity	solid bulk viscosity	[kg m^-1 s^-1]	3 C, M, R, TS	
Solid Pressure	solid pressure	[Pa]	3 A, C, M, R, TS	
Solid Pressure Gradient	solid pressure gradient	[ ]	3 C, M, R, TS	
Solid Shear Viscosity	solid shear viscosity	[kg m^-1 s^-1]	3 C, M, R, TS	
Spinodal Pressure	pspin	[Pa]	RG	Metastable condition boundary
Spinodal Temperature	tspin	[K]	RG	Metastable condition boundary
Static Entropy	entropy		3 A, C, M, P, R, TS	
Temperature Variance	Tvar		1 A, C, M, R, TS	
Time This Run	trun		2 C	
Total Boundary Displacement	bnddisptot		1 C, DT, M, R, TS	
Total Density	dentot	[kg m^-3]	2 A, C, M, R	Total Density is the density evaluated at the Total Temperature and Total Pressure.
Total Density in Stn Frame	dentotstn	[kg m^-3]	2	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
			A, C, M, R	
Total Density in Rel Frame	dentotrel	[kg m^-3]	2 A, C, M, R	
Total Force	forcet		3 DT	
Unburnt Absolute Temperature	unburnt Tabs	[K]	2 A, C, M, R, TS	
Unburnt Density	unburnt density	[kg m^-3]	2 A, C, M, R, TS	
Unburnt Thermal Conductivity	unburnt cond	[W m^-1 K^-1]	2 A, C, M, R, TS	
Unburnt Specific Heat Capacity at Constant Pressure	unburnt Cp	[J kg^-1 K^-1]	2 A, C, M, R, TS	
Volume Porosity	volpor	[ ]	2 C, M, R, TS	
Volume of Finite Volumes	volcvol		3 C, R, TS	
Vorticity	vorticity		2 A, C, M, R, TS	Note that Vorticity is the same as Velocity.Curl.
Vorticity in Stn Frame	vortstn		2 A, C, M, R, TS	
Wall External Heat Transfer Coefficient	htco		2 R, TS	
Wall Adjacent Temperature	tnw	[K]	2 C, DT, R, TS	
Wall Distance	wall distance	[m]	2 A, C, M, P, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Wall External Temperature	tnwo	[K]	2 DT, R, TS	User-specified external wall temperature for heat transfer coefficient boundary conditions.
Wall Film Thickness	film thickness	[m]	2 C, R	
Wall Heat Transfer Coefficient	htc		2 C, R, TS	
Wall Heat Flow	QwallFlow		3 C, DT, R, TS	
Wall Normal Velocity	nwallvel		2 C, R, TS	
Wall Scale	wall scale		3 R, M, TS	
Wavelength in Vacuum	wavelo		3 C	
Wavenumber in Vacuum	waveno		3 C	
Normalized Droplet Number	spdropn	[m^-3]	2 C, M, R, TS	
Droplet Number	spdrop		1 C, M, R, TS	
Dynamic Bulk Viscosity	dynamic bulk viscosity		1 A, C, M, R, TS	
Total MUSIG Volume Fraction	vft	[ ]	2 A, C, M, R, TS	
Smoothed Volume Fraction	vfs	[ ]	2 A, C, M, R, TS	

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Long Variable Name	Short Variable Name	Units	Availability	Definition
Temperature Superheating	Tsuperheat		3 C	Temperature above saturation
Temperature Subcooling	Tsubcool		3 C	Temperature below saturation

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 17: Power Syntax in Ansys CFX

Programming constructs can be used within CCL for advanced usage. Rather than invent a new language, CCL takes advantage of the full range of capabilities and resources from an existing programming language, Perl. Perl statements can be embedded in between lines of simple syntax, providing capabilities such as loops, logic, and much, much more with any CCL input file.

A line of Power Syntax is identified in a CCL file by an exclamation mark (!) in the first column of a line. In between Perl lines, simple syntax lines may refer to Perl variables and lists.

A wide range of additional functionality is made available to expert users with the use of Power Syntax including:

- Loops
- Logic and control structures
- Lists and arrays
- Subroutines with argument handling (useful for defining commonly re-used plots and procedures)
- Basic I/O processing
- System functions
- Many other procedures (Object programming, World Wide Web access, simple embedded graphical user interfaces).

Any of the above may be included in a CCL input file or CFD-Post session file.

Important:

You should be wary when entering certain expressions because Power Syntax uses Perl mathematical operators. For example, in CEL, 2^2 is represented as 2^2, but in Perl, it would be written 2**2. If you are unsure about the validity of an operator, you should check a Perl reference guide.

There are many good reference books on Perl. Two examples are *Learning Perl* (ISBN 1-56592-042-2) and *Programming Perl* (ISBN 1-56592-149-6) from the O'Reilly series.

This chapter describes:

17.1. Examples of Power Syntax
17.2. Predefined Power Syntax Subroutines

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.1. Examples of Power Syntax

The following are some examples in which the versatility of power syntax is demonstrated. They become steadily more complex in the later examples.

Some additional, more complex, examples of Power Syntax subroutines can be found by viewing the session files used for the **Macro Calculator**. These are located in `CFX/etc/`. You can execute these subroutines from the **Command Editor** dialog box the same as calling any other Power Syntax subroutine. The required argument format is:

```
!cpPolar(<"BoundaryList">, <"SliceNormalAxis">,
   <"SlicePosition">, <"PlotAxis">, <"InletLocation">,
   <"ReferencePressure">)
!compressorPerform(<"InletLocation">, <"OutletLocation">,
   <"BladeLocation">, <"MachineAxis">, <"RotationalSpeed">,
   <"TipRadius">, <"NumBlades">, <"FluidGamma">)
```

These subroutines are loaded when CFD-Post is launched, so you do not need to execute the session files before using the functions.

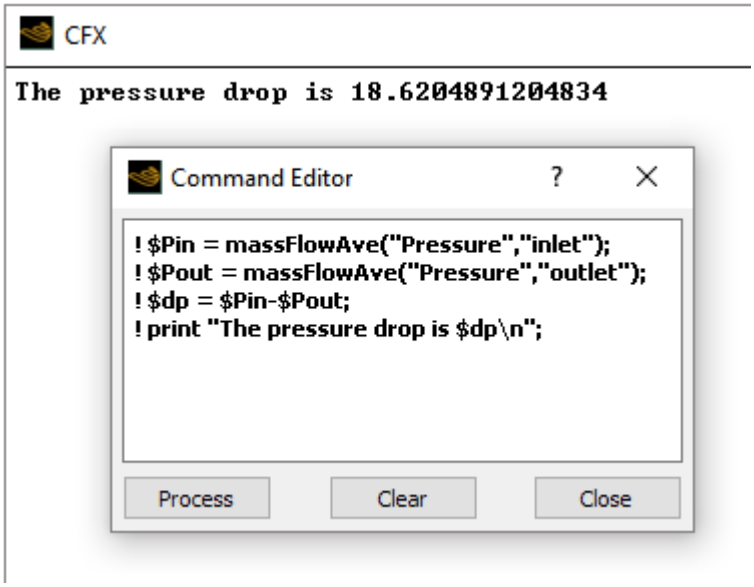
Additional information on these macro functions is available in Gas Compressor Performance Macro and Cp Polar Plot Macro.

All arguments passed to subroutines should be enclosed in quotations, for example `Plane 1` must be passed as `"Plane 1"` and `Eddy Viscosity` should be entered as `"Eddy Viscosity"`. Any legal CFX Command Language characters that are illegal in Perl need to be enclosed in quotation marks.

17.1.1. Example 1: Print the Value of the Pressure Drop Through a Pipe

```
! $Pin = massFlowAve("Pressure","inlet");
! $Pout = massFlowAve("Pressure","outlet");
! $dp = $Pin-$Pout;
! print "The pressure drop is $dp\n";
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.



Note:

Function-specific Perl subroutines do not allow phase-specific evaluations; that is, you can get only bulk results (such as mass flow for all phases). A workaround is to use "evaluate" subroutine, which evaluates any CEL expression.

For example instead of

```
! $val = massFlow("Inlet", "Water")   # Does NOT work
```

you need to use:

```
! ($val, $units) = evaluate( "Water.massFlow()\@Inlet");
```

17.1.2. Example 2: Using a for Loop

This example demonstrates using Power Syntax that wraps a `for` loop around some CCL Object definitions to repetitively change the visibility on the outer boundaries.

```
# Make the outer boundaries gradually transparent in
# the specified number of steps.
!$numsteps = 10;
!for ($i=0; $i < $numsteps; $i++) {
   ! $trans = ($i+1)/$numsteps;
   BOUNDARY:in
      Visibility = 1
      Transparency = $trans
   END
   BOUNDARY:out
      Visibility = 1
      Transparency = $trans
   END
   BOUNDARY:Default
      Visibility = 1
      Transparency = $trans
   END
!}
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

The first line of Power Syntax simply defines a scalar variable called `numsteps`. Scalar variables (that is, simple single-valued variables) begin with a `$` symbol in Perl. The next line defines a `for` loop that increments the variable `i` up to `numsteps`. Next, you determine the fraction you are along in the loop and assign it to the variable `trans`. The object definitions then use `trans` to set their transparency and then repeat. Note how Perl variables can be directly embedded into the object definitions. The final line of Power Syntax (`!}`) closes the `for` loop.

Note:

Function-specific Perl subroutines do not allow phase-specific evaluations; that is, you can get only bulk results (such as mass flow for all phases). A workaround is to use "evaluate" subroutine, which evaluates any CEL expression.

For example instead of

```
! $val = massFlow("Inlet", "Water")   # Does NOT work
```

you need to use:

```
! ($val, $units) = evaluate( "Water.massFlow()\@Inlet");
```

17.1.3. Example 3: Creating a Simple Subroutine

The following example defines a simple subroutine to make two planes at specified locations. The subroutine will be used in the next example.

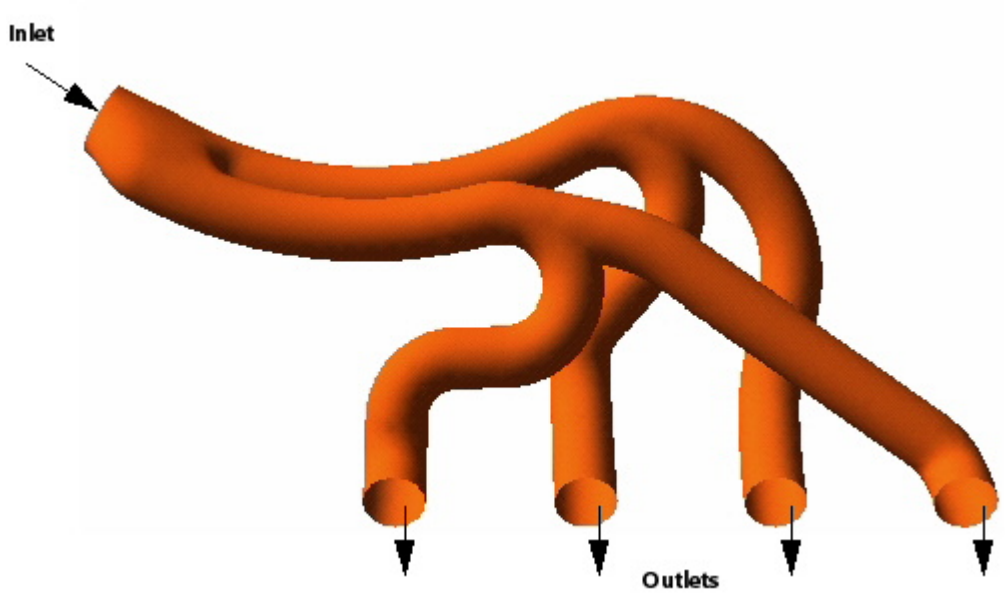
```
!sub makePlanes {
   PLANE:plane1
      Option = Point and Normal
      Point = 0.09,0,-0.03
      Normal = 1,0,0
      Draw Lines = On
      Line Color = 1,0,0
      Color Mode = Variable
      Color Variable = Pressure
      Range = Local
   END
   PLANE:plane2
      Option = Point and Normal
      Point = 0.08,-0.038,-0.0474
      Normal = 1,0,0
      Draw Faces = Off
      Draw Lines = On
      Line Color = 0,1,0
   END
!}
```

Although this subroutine is designed for use with the next example, you can execute it on its own by typing `!makePlanes();` in the **Command Editor** dialog box.

17.1.4. Example 4: Creating a Complex Quantitative Subroutine

This example is a complex quantitative subroutine that takes slices through the manifold geometry, as shown below, compares the mass flow through the two sides of the initial branch, and computes the pressure drop through to the four exit locations.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.




```
! sub manifoldCalcs{
# call the previously defined subroutine (Example 3) make the
# upstream and downstream cutting planes
! makePlanes();
#
# Bound the two planes so they each just cut one side of the branch.
PLANE:plane1
Plane Bound = Circular
Bound Radius = 0.025
END
PLANE:plane2
Plane Bound = Circular
Bound Radius = 0.025
END
# Calculate mass flow through each using the predefined
# 'evaluate' Power Syntax subroutine and output the results
! ($mass1, $mfunits) = evaluate( "massFlow()\@plane1" );
! ($mass2) = evaluate( "massFlow()\@plane2" );
! $sum = $mass1+$mass2;
! print "Mass flow through branch 1 = $mass1 [$mfunits]\n";
! print "Mass flow through branch 2 = $mass2 [$mfunits]\n";
! print "Total = $sum [$mfunits]\n";
# Now calculate pressure drops and mass flows through the exits
# calculate the average pressure at the inlet
!($Pin, $punits) = evaluate( "massFlowAve(Pressure)\@in1" );
# Set-up an array that holds the approximate X location of each
# of the 4 exits. We then loop over the array to move the outlet
# plane and re-do the pressure drop calculation at each exit.
! @Xlocs = (0.15,0.25,0.35,0.45);
! $sum = 0;
! for ($i=0;$i<4;$i++) {
PLANE:outlet
   Option = Point and Normal
   Normal = 0,-1,-1
   Point = $Xlocs[$i],-0.06,-0.2
   Plane Bound = Circular
   Bound Radius = 0.05
END
! ($Pout, $punits) =  evaluate( "massFlowAve(Pressure)\@outlet" );
! ($massFl) = evaluate( "massFlow()\@outlet" );
! $sum += $massFl;
! $Dp = $Pin-$Pout;
! $ii = $i+1;
! print "At outlet \#$ii: Dp=$Dp [$punits], Mass Flow=$massFl [$mfunits]\n";
! } # end loop
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
! print "Total Mass Flow = $sum  [$mfunits]\n";
!} # end subroutine
```

After processing these commands to define the subroutine, you can execute it, in the same way as any other subroutine, by typing `!manifoldCalcs();` in the **Command Editor** dialog box.

17.2. Predefined Power Syntax Subroutines

CFD-Post provides predefined subroutines that add Power Syntax functionality. You can view a list of these subroutines by entering `!showSubs();` in the **Command Editor** dialog box. The list is printed to the console window. The list shows all currently loaded subroutines, so it will include any custom subroutines that you have processed in the **Command Editor** dialog box.

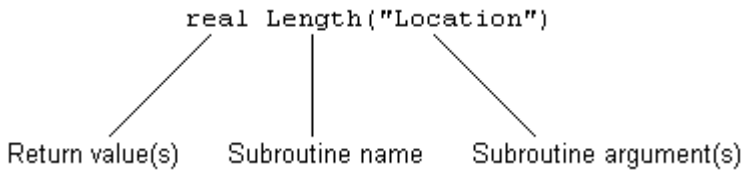
These subroutines provide access to the quantitative functionality of CFD-Post. Most of these routines provide results in a single return value. For example, if the Perl variable `$verbose = 1`, then the result is also printed to the screen. Information on the calculations performed by the subroutines is available. For details, see Function Selection.

The following sections describe these predefined subroutines:

- Power Syntax Subroutine Descriptions (p. 564)
- Power Syntax Usage (p. 565)
- Power Syntax Subroutines (p. 565)

17.2.1. Power Syntax Subroutine Descriptions

In the next section, each subroutine will appear in the following format:



Each of the subroutines contains an argument list (in brackets, separated by commas). If any argument contains more than one word (for example, `Plane 1`), it must be within quotes. You should enclose all arguments within quotes to avoid making possible syntax errors.

Each subroutine is preceded by its return value(s). For example:

```
real, string evaluate("Expression", "Locator")
```

will return two values, a real number and a string.

The return values will always be in the solution units of the CFX-Solver results file, even if you have changed the display units in the **Edit** menu. This means that if you have a plot of temperature in degrees C on `Plane 1`, the area averaged value of temperature on `Plane 1` returned by the `areaAve` command will still be in degrees K.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.2. Power Syntax Usage

All lines of power syntax must have an exclamation mark as the first character so that they are not treated as CCL statements. The statements must also end with a semicolon.

Assuming you have a plane named `Plane 1`, the following example returns the area of this plane:

```
! $areaVal = area("Plane 1");
! print "The area of Plane 1 is $areaVal \n";
```

Some subroutines return more than one value. To store return values for a subroutine that returns two variables (such as the `evaluate` function), you could use the following:

```
! ($value, $units) = evaluate('area()@Plane 1');
! print "The area of Plane 1 is $value in units of $units \n";
```

Note:

In this case, if single quotes are not used around the expression, `area()@Plane 1`, when calling the function, `evaluate()`, the @ symbol must be escaped (made literal) using the following power syntax instead:

```
! ($value, $units) = evaluate("area()\@Plane 1");
! print "The area of Plane 1 is $value in units of $units \n";
```

This is used to avoid Perl treating the @ symbol as a special character. See evaluate(Expression) (p. 567) for details.

17.2.3. Power Syntax Subroutines

17.2.3.1. area(Location, Axis)

```
real area("Location", "Axis")
```

Returns the area of a 2D locator. For details, see area (p. 479).

17.2.3.2. areaAve(Variable, Location, Axis)

```
real areaAve("Variable", "Location", "Axis")
```

Returns the area-weighted average of the variable at a 2D locator. For details, see areaAve (p. 480).

17.2.3.3. areaInt(Variable, Location, Axis)

```
real areaInt("Variable", "Location", "Axis")
```

Returns the result of the variable integrated over the 2D location. For details, see areaInt (p. 481).

17.2.3.4. ave(Variable, Location)

```
real ave("Variable", "Location")
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns the arithmetic average of the variable at a location. For details, see ave (p. 482).

17.2.3.5. calcTurboVariables()

```
void calcTurboVariables()
```

Calculates all 'extra' turbo variables. (Works only in turbo mode.)

17.2.3.6. calculate(function,...)

```
real calculate(function,...)
```

Evaluates the named function with the supplied argument list, and returns the float result. The function name is a required argument, which can be followed by a variable length list of arguments.

17.2.3.7. calculateUnits(function,...)

```
string,string calculateUnits(function,...)
```

Evaluates the named function with the supplied argument list, and returns the value and units.

17.2.3.8. collectTurboInfo()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.9. comfortFactors()

This is an internal subroutine that is used only to initialize report templates.

For details, see Comfort Factors Macro.

17.2.3.10. compressorPerform(Location, Location, Location, Var, Args)

This is a special macro; for details, see Gas Compressor Performance Macro. For example:

```
compressorPerform("Inlet", "Outlet", "Blade", "X", 600, 0.03, 10, 1.2)
```

17.2.3.11. compressorPerformTurbo()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.12. copyFile(FromPath, ToPath)

```
void copyFile("FromPath", "ToPath")
```

A utility function for copying files.

17.2.3.13. count(Location)

```
real count("Location")
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns the number of nodes on the location. For details, see count (p. 483).

17.2.3.14. countTrue(Expression, Location)

```
real countTrue("Expression", "Location")
```

Returns the number of mesh nodes on the specified region that evaluate to "true", where true means greater than or equal to 0.5. `"Expression"` should contain one of the logical operators =, >, <, <=, or >=. The countTrue function is valid for 1D, 2D, and 3D locations. For details, see countTrue (p. 484).

17.2.3.15. cpPolar(Location, Var, Arg, Var, Location, Arg)

This is a special macro; for details, see Cp Polar Plot Macro. For example:

```
cpPolar("Plane 1", "Y", 0.3, "X", "Inlet", 10000)
```

17.2.3.16. evaluate(Expression)

```
real,string evaluate("Expression")
```

Returns the value of the expression and the units. Only one expression can be evaluated each time the subroutine is executed. The main advantage of using `evaluate` is that it takes any CEL expression. This means that you do not have to learn any other quantitative power syntax routines described in this section. Also, evaluate will return the result units in addition to the value.

An example is:

```
evaluate("areaAve(Velocity v)\@Location 1")
```

In this case, another subroutine is evaluated. The `evaluate` command takes an any expression as the argument, or more precisely, any expression that resolves to a quantity. This means that you cannot use:

```
"2*Pressure"
```

but you can use:

```
"2*minVal(Pressure)\@locator 1"
```

or

```
"100 [m]"
```

This is simply an alternative way of typing:

```
! $myVal = 2 * minVal("Pressure", "Location");
```

The reason that the @ is escaped calling `evaluate()` is to avoid Perl treating it as a special character.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.3.17. evaluateInPreferred(Expression)

```
real,string evaluateInPreferred("Expression")
```

Returns the value of the expression in your *preferred units*. Preferred units are the units of the data that CFD-Post uses when information is displayed to you and are the default units when you enter information (as contrasted with units of the data that are stored in results files). Use the **Edit** > **Options** > **Common** > **Units** dialog box to set your preferred units.

17.2.3.18. exprExists(Expression)

```
bool exprExists("Expression")
```

Returns true if an expression with this name exists; false otherwise.

17.2.3.19. fanNoiseDefault()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.20. fanNoise()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.21. force(Location, Axis)

```
real force("Location", "Axis")
```

Returns the `force` on a 2D locator. For details, see force (p. 484).

17.2.3.22. forceNorm(Location, Axis)

```
real forceNorm("Location", "Axis")
```

Returns the per unit width force on a line in the direction of the specified axis. It is available only for a polyline created by intersecting a locator on a boundary. For details, see forceNorm (p. 486).

17.2.3.23. getBladeForceExpr()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.24. getBladeTorqueExpr()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.25. getCCLState()

This is an internal debugging call.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.3.26. getChildrenByCategory(Object Path, Category)

```
string getChildrenByCategory("Object Path", "Category")
```

Returns the children of an object that belong to the specified category in a comma-separated list. Each object type (for example, a PLANE) can have multiple categories associated with it such as "geometry", "surface", and so on). Categories are specified in <*CFXROOT*>`/etc/CFXPreRules.ccl` and <*CFXROOT*>`/etc/CFXPostRules.ccl`.

For example, to get a comma-separated list of all surfaces in a state at the top level (that is, not sub-objects of other objects):

```
! $surfaces = getChildrenByCategory("/", "surface" );
```

Use '`split ","`' to convert the string into an array of strings.

17.2.3.27. getChildren(Object Path, Child Type)

```
string getChildren("Object Path", "Child Type")
```

Returns the children of an object in a comma-separated list. If `Child Type` is not an empty string, this subroutine return only children of the specified type.

17.2.3.28. getExprOnLocators()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.29. getExprString(Expression)

```
string getExprString("Expression")
```

Returns the value and the units of the expression in the form "value units". For example: "100 m".

17.2.3.30. getExprVal(Expression)

```
real getExprVal("Expression")
```

Returns only the "value" portion of the expression (units are not included).

17.2.3.31. getObjectName(Object Path)

```
string getObjectName("Object Path")
```

Extracts the name of an object from its full path. For example:

```
!string = getObjectName("/USER SURFACE:User Surface 1")
```

returns "User Surface 1". This is the form needed for evaluating a CEL expression.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.3.32. getParameterInfo(Object Path, Parameter Name, Info Type)

```
string getParameterInfo("Object Path", "Parameter Name", "Info Type")
```

Returns the requested information for a parameter of an object. `Object Path` returns the name or path of an object; "/" or an empty string specifies the root.`Parameter Name` returns the name of the parameter. `Info Type` returns the type of data requested; this can be one of "type", "value", "default value", or "allowed values". For example:

```
! $info = getParameterInfo("/USER DEFINED/POINT:Point 1", "Symbol Size", "default value");
! print "getParameterInfo returned=$info\n";
```

prints:

```
getParameterInfo returned=2.5
```

17.2.3.33. getParameters(Object Path)

```
string getParameters("Object Path")
```

Returns the parameters of an object in a comma-separated list. Use '`split ","`' to convert the string into an array of strings.

17.2.3.34. getTempDirectory()

```
string getTempDirectory()
```

Returns the temporary directory path.

17.2.3.35. getType(Object Path)

```
string getType("Object Path")
```

Returns the object type.

17.2.3.36. getValue(Object Path, Parameter Name)

```
string getValue("Object Path", "Parameter Name")
```

Takes a CCL object and parameter name and returns the value of the parameter.

Returns the value stored in `Parameter Name`.

17.2.3.36.1. Example

1. Create a text object called **Text 1**.
2. In the **Text String** box, enter `Here is a text string`.
3. Click **Apply** to create the text object.
4. In the **Command Editor** dialog box, enter the following:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

```
!string = getValue( "/TEXT:Text 1/TEXT ITEM: Text Item 1", "Text String");
! print $string;
```

5. Click **Process**, and the string will be printed to your terminal window.

The same procedure can be carried out for any object.

17.2.3.37. getViewArea()

```
string,string getViewArea()
```

Calculates the area of the scene projected in the view direction. Returns the area and the units in an array of strings.

17.2.3.38. isCategory(Object Path, Category)

```
bool isCategory("Object Path", "Category")
```

A return of 1 indicates that the object matches the passed category; 0 otherwise. Categories are specified in <*CFXROOT*>`/etc/CFXPreRules.ccl` and <*CFXROOT*>`/etc/CFXPostRules.ccl`.

For example, the following prints "Plane 1 is a surface":

```
! if( isCategory( "Plane 1", "surface" )) {
!   print "Plane 1 is a surface\n";
! }
```

17.2.3.39. Length(Location)

```
real Length("Location")
```

Returns the length of a line locator. For details, see length (p. 487).

Note:

While using this function in Power Syntax the leading character is capitalized to avoid confusion with the Perl internal command "length."

17.2.3.40. lengthAve(Variable, Location)

```
real lengthAve("Variable", "Location")
```

Returns the length-based average of the variable on the line locator. For details, see lengthAve (p. 488).

17.2.3.41. lengthInt(Variable, Location)

```
real lengthInt("Variable", "Location")
```

Returns the length-based integral of the variable on the line locator. For details, see lengthInt (p. 488).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

17.2.3.42. liquidTurbPerformTurbo()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.43. liquidTurbPerform()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.44. massFlow(Location)

```
real massFlow("Location")
```

Returns the mass flow through the 2D locator. For details, see massFlow (p. 490).

17.2.3.45. massFlowAve(Variable, Location)

```
real massFlowAve("Variable","Location")
```

Returns the average value of the variable, weighted by mass flow, through the 2D locator. For details, see massFlowAve (p. 491).

17.2.3.46. massFlowAveAbs(Variable, Location)

```
real massFlowAveAbs("Variable","Location")
```

Returns the average value of the variable, weighted by absolute mass flow, through the 2D locator. For details, see massFlowAveAbs (p. 492).

17.2.3.47. massFlowInt(Variable, Location)

```
real massFlowInt("Variable", "Location")
```

Returns the integral of the variable, weighted by mass flow, over the 2D locator. For details, see massFlowInt (p. 494).

17.2.3.48. maxVal(Variable, Location)

```
real maxVal("Variable", "Location")
```

Returns the maximum value of the variable at the location. For details, see maxVal (p. 494).

17.2.3.49. minVal(Variable, Location)

```
real minVal("Variable", "Location")
```

Returns the minimum value of the variable at the location. For details, see minVal (p. 495).

17.2.3.50. objectExists(Object Path)

```
bool objectExists("Object Path")
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

A return of 1 indicates that the object exists; 0 otherwise.

17.2.3.51. probe(Variable, Location)

```
real probe("Variable", "Location")
```

Important:

This calculation should only be performed for point locators described by single points. Incorrect solutions will be produced for multiple point locators.

Returns the value of the variable at the point locator. For details, see probe (p. 495).

17.2.3.52. pumpPerform()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.53. pumpPerformTurbo()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.54. range(Variable, Location)

```
real,real range("Variable", "Location")
```

Returns the minimum and maximum values of the variable at the location.

17.2.3.55. reportError(String)

```
void reportError("String")
```

Pops up an error dialog box.

17.2.3.56. reportWarning(String)

```
void reportWarning("String")
```

Pops up a warning dialog box.

17.2.3.57. showPkgs()

```
void showPkgs()
```

Prints to the console a list of packages available that may contain other variables or subroutines in Power Syntax.

17.2.3.58. showSubs(packageName)

```
void showSubs("packageName")
```

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Prints to the console a list of the subroutines available in the specified package. If no package is specified, CFD-Post is used by default.

17.2.3.59. showVars(packageName)

```
void showVars("packageName")
```

Prints to the console a list of the Power Syntax variables and their current value defined in the specified package. If no package is specified, CFD-Post is used by default.

17.2.3.60. spawnAsyncProcess(command, arguments)

```
bool spawnAsyncProcess("command", "arguments")
```

Spawns a forked process. For example:

```
! spawnAsyncProcess("dir", "c:/");
```

Displays the contents of c:\ in the console window.

17.2.3.61. sum(Variable, Location)

```
real sum("Variable", "Location")
```

Returns the sum of the variable values at each point on the locator. For details, see sum (p. 497).

17.2.3.62. torque(Location, Axis)

```
real torque("Location", "Axis")
```

Returns the computed value of torque at the 2D locator about the specified axis. For details, see torque (p. 498).

17.2.3.63. turbinePerform()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.64. turbinePerformTurbo()

This is an internal subroutine that is used only to initialize report templates.

17.2.3.65. verboseOn()

```
bool verboseOn()
```

Returns 1 or 0 depending if the Perl variable `$verbose` is set to 1.

17.2.3.66. volume(Location)

```
real volume("Location")
```


Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Returns the volume of a 3D locator. For details, see volume (p. 498).

17.2.3.67. volumeAve(Variable, Location)

```
real volumeAve("Variable", "Location")
```

Returns the average value of a variable over the 3D locator. For details, see volumeAve (p. 499).

17.2.3.68. volumeInt(Variable, Location)

```
real volumeInt("Variable", "Location")
```

Returns the integral of a variable over the 3D locator. For details, see volumeInt (p. 499).

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Chapter 18: Bibliography

This bibliography contains entries referenced in the CFX documentation.

- References 1-20 (p. 577)
- References 21-40 (p. 580)
- References 41-60 (p. 583)
- References 61-80 (p. 586)
- References 81-100 (p. 589)
- References 101-120 (p. 592)
- References 121-140 (p. 595)
- References 141-160 (p. 598)
- References 161-180 (p. 601)
- References 181-200 (p. 604)
- References 201-220 (p. 607)
- References 221- (p. 610)

18.1. References 1-20

1

Hutchinson, B.R. and Raithby, G.D.,

"A Multigrid method Based on the Additive Correction Strategy", Numerical Heat Transfer, Vol. 9, pp. 511-537, 1986.

2

Rhie, C.M. and Chow, W.L.,

"A Numerical Study of the Turbulent Flow Past an Isolated Airfoil with Trailing Edge Separation",

AIAA Paper 82-0998, 1982

3

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Raw, M.J.,

"A Coupled Algebraic Multigrid Method for the 3D Navier-Stokes Equations",

10th GAMM Seminar, Kiel, 1994.

4

Launder, B.E., Reece, G.J. and Rodi, W.,

"Progress in the developments of a Reynolds-stress turbulence closure",

J. Fluid Mechanics, Vol. 68, pp.537-566, 1975.

5

Speziale, C.G., Sarkar, S. and Gatski, T.B.,

"Modelling the pressure-strain correlation of turbulence: an invariant dynamical systems approach",

J. Fluid Mechanics, Vol. 277, pp. 245-272, 1991.

6

Schiller, L. and Naumann, A.,

VDI Zeits, 77, p. 318, 1933.

7

Hughmark, G.A.,

AIChE J., 13 p. 1219, 1967.

8

Modest, M.,

"Radiative Heat Transfer", Second Edition

Academic Press, 2003.

9

Menter, F.R.,

"Two-equation eddy-viscosity turbulence models for engineering applications",

AIAA-Journal., 32(8), pp. 1598 - 1605, 1994.

10

Grotjans, H. and Menter, F.R.,

"Wall functions for general application CFD codes",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

In K.D.Papailiou et al., editor, ECCOMAS 98 Proceedings of the Fourth European Computational Fluid Dynamics Conference, pp. 1112-1117. John Wiley & Sons, 1998.

11

Wilcox, D.C.,

"Multiscale model for turbulent flows",

In AIAA 24th Aerospace Sciences Meeting. American Institute of Aeronautics and Astronautics, 1986.

12

Menter, F.R.,

"Multiscale model for turbulent flows",

In 24th Fluid Dynamics Conference. American Institute of Aeronautics and Astronautics, 1993.

13

Launder, B.E. and Spalding, D.B.,

"The numerical computation of turbulent flows",

Comp Meth Appl Mech Eng, 3:269-289, 1974.

14

White, F.M.,

"Viscous Fluid Flow", Second Edition,

McGraw-Hill, 1991.

15

Kader, B.A.,

"Temperature and concentration profiles in fully turbulent boundary layers",

International Journal of Heat and Mass Transfer, 24(9):1541-1544, 1981.

16

Huang, P.G., Bradshaw, P. and Coakley, T.J.,

"Skin friction and velocity profile family for compressible turbulent boundary layers",

American Institute of Aeronautics and Astronautics Journal, 31(9):1600-1604, 1993.

17

Bouillard, J.X, Lyczkowski, R.W.and Gidaspow, D.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Porosity Distribution in a Fluidised Bed with an Immersed Obstacle",

AIChE J., **35**, 908-922, 1989.

18

Gidaspow, D.,

"Multiphase Flow and Fluidisation", Academic Press, 1994.

19

Ishii, M. and Zuber, N.,

"Drag Coefficient and Relative Velocity in Bubbly, Droplet or Particulate Flows",

AIChE J., **25**, 843-855, 1979.

20

Lopez de Bertodano, M.,

"Turbulent Bubbly Flow in a Triangular Duct",

Ph.D. Thesis, Rensselaer Polytechnic Institute, Troy New York, 1991.

18.2. References 21-40

21

Lopez de Bertodano, M.,

"Two Fluid Model for Two-Phase Turbulent Jet",

Nucl. Eng. Des. **179**, 65-74, 1998.

22

Sato, Y. and Sekoguchi, K.,

"Liquid Velocity Distribution in Two-Phase Bubbly Flow",

Int. J. Multiphase Flow, **2**, p.79, 1975.

23

Siegel, R and J.R. Howell,

"Thermal Radiation Heat Transfer",

ISBN 0-89116-506-1.

24

Goldstein, M. and J.R. Howell,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Boundary Conditions for the Diffusion Solution of Coupled Conduction-Radiation Problems",

NASA Technical Note, NASA TN D-4618.

25

Raw, M.J.,

"Robustness of Coupled Algebraic Multigrid for the Navier-Stokes Equations",

AIAA 96-0297, 34th Aerospace and Sciences Meeting & Exhibit, January 15-18 1996, Reno, NV.

26

Kee, R. J., Rupley, F. M. and Miller, J. A.,

"Chemkin -II: A Fortran Chemical Kinetics Package for the Analysis of Gas-Phase Chemical Kinetics",

Sandia National Laboratories Report, SAND89-8009,(1991).

27

Brackbill, J.U, Kothe, D.B. and Zemach, C.,

"A Continuum Method for Modelling Surface Tension",

Journal of Computational Physics 100:335-354, 1992.

28

Barth, T.J., and Jesperson, D.C,

"The Design and Application of Upwind Schemes on Unstructured Meshes",

AIAA Paper 89-0366, 1989.

29

Bird, R.B., Stewart, W.E. and Lightfoot, E.N.,

"Transport Phenomena",

John Wiley & Sons, Inc., 1960.

30

Wilcox, D.C.,

"Turbulence Modelling for CFD",

DCW Industries, 2000, La Canada, CA 91011, p. 314.

31

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Launder, B.E., Tselepidakis, D. P., Younis, B. A.,

"A second-moment closure study of rotating channel flow",

J. Fluid Mech., Vol. 183, pp. 63-75, 1987.

32

Menter, F. R.,

"Eddy Viscosity Transport Equations and their Relation to the k–ε Model",

NASA Technical Memorandum 108854, November 1994.

33

Menter, F. R.,

"Eddy Viscosity Transport Equations and their Relation to the k–ε Model",

ASME J. Fluids Engineering, Vol. 119, pp. 876-884, 1997.

34

Smagorinsky, J.,

"General Circulation Experiments with the Primitive Equations",

Month. Weath. Rev. Vol. 93, pp. 99-165, 1963.

35

Clift, R., Grace, J.R., Weber, M.E.,

"Bubbles, Drops and Particles",

Academic Press, New York, U.S.A., 1978.

36

Liang, L., Michaelides, E. E.,

"The magnitude of Basset forces in unsteady multiphase flow computations",

Journal of Fluids Engineering, Vol. 114, pp. 417-419, 1992.

37

Peters, N.,

"Turbulent Combustion",

Cambridge monographs on mechanics, Cambridge University Press, 2000.

38

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Zimont, V.L., Polifke, W., Bettelini, M. and Weisenstein, W.,

"An efficient Computational Model for Premixed Turbulent Combustion at High Reynolds Numbers Based on a Turbulent Flame Speed Closure",

J. Engineering for Gas Turbines and Power (Transactions of the ASME), Vol. 120, pp. 526-532, 1998.

39

Hinze, J. O.,

"Turbulence",

McGraw-Hill, New York, U.S.A., 1975.

40

Zimont, V.L.,

"Gas Premixed Combustion at High Turbulence. Turbulent Flame Closure Combustion Model",

Proceedings of the Mediterranean Combustion Symposium, Instituto di Richerche sulla Combustione - CNR, Italy, pp. 1155-1165, 1999.

18.3. References 41-60

41

Zimont, V.L., Biagioli, F. and Syed, Khawar,

"Modelling turbulent premixed combustion in the intermediate steady propagation regime",

Progress in Computational Fluid Dynamics, Vol. 1, pp. 14-28, 2001.

42

Linan, A.,

"On the internal structure of laminar diffusion flames",

Technical note, Inst. nac. de tec. aeron., Esteban Terradas, Madrid, Spain, 1961.

43

Warnatz, J., Mass, U. and Dibble, R. W.,

"Combustion",

Springer, Verlag, 1996, pp.219-221.

44

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Magnussen, B. F.,

"The Eddy Dissipation Concept for Turbulent Combustion Modelling. Its Physical and Practical Implications",

Presented at the First Topic Oriented Technical Meeting, International Flame Research Foundation, IJmuiden, The Netherlands, Oct. 1989.

45

Tesner, P. A., Snegirova, T. D., and Knorre, V. G.,

"Kinetics of Dispersed Carbon Formation",

Combustion and Flame, Vol. 17, pp. 253-260, 1971.

46

Magnussen, B. F., and Hjertager, B. H.,

"On Mathematical Modeling of Turbulent Combustion with Special Emphasis on Soot Formation and Combustion",

Sixteenth Symp. (Int.) on Combustion, The Combustion Institute, p 719, 1976.

47

Vukalovich, M. P.,

"Thermodynamic Properties of Water and Steam",

Mashgis, Moscow, 6th ed., 1958.

48

Hottel, H.C. and Sarofim, A.F.,

"Radiative transfer",

McGraw-Hill, New York 1967.

49

Hadvig, S.,

"Gas emissivity and absorptivity",

J. Inst. Fuel, 43, pp. 129-135., 1970.

50

Leckner, B.,

"Spectral and total emissivity of water vapour and carbon dioxide",

Comb. Flame, **19**, pp. 33-48., 1972.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

51

Taylor, P.B. and Foster, P.J.,

"The total emissivities of luminous and non-luminous flames",

Int. J. Heat Mass Transfer, **17**, pp. 1591-1605., 1974.

52

Beer, J.M., Foster, P.J. and Siddall, R.G.,

"Calculation methods of radiative heat transfer",

HTFS Design Report No. 22, AEA Technology (Commercial)., 1971.

53

Prakash, C.,

"Two phase model for binary liquid-solid phase change",

Parts I and II, Numerical Heat Transfer, B 15, p. 171.

54

CFX Limited, Waterloo, Ontario, Canada,

CFX-TASCflow Theory Documentation,

Section 4.1.2, Version 2.12, 2002.

55

Menter, F. R. and Kuntz, M.,

"Development and Application of a Zonal DES Turbulence Model for CFX-5",

CFX-Validation Report, CFX-VAL17/0503.

56

Menter, F.R., Kuntz, M.,

"Adaptation of Eddy-Viscosity Turbulence Models to Unsteady Separated Flow Behind Vehicles",

Proc. Conf. The Aerodynamics of Heavy Vehicles: Trucks, Busses and Trains, Asilomar, Ca, 2002.

57

Spalart, P.R, Jou, W.-H., Strelets, M. and Allmaras, S.R.,

"Comments on the feasibility of LES for wings, and on a hybrid RANS/LES approach",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

1st AFOSR Int. Conf. On DNS/LES, Aug.4-8, 1997, Ruston, LA. In Advances in DNS/LES, C. Liu & Z. Liu Eds., Greyden Press, Colombus, OH.

58

Strelets, M.,

"Detached Eddy Simulation of Massively Separated Flows",

AIAA Paper 2001-0879, 39th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2001.

59

Ishii, M.,

"One-dimensional drift-flux model and constitutive equations for relative motion between phases in various two-phase flow regimes",

Argonne National Laboratory ANL-77-47, 1977.

60

Manninen, M. and Tavassalo, V.,

"On the Mixture Models for Multiphase Flow",

VTT Publications, 1996.

18.4. References 61-80

61

Luo, S.M., and Svendsen, H.,

"Theoretical Model for Drop and Bubble Breakup in Turbulent Dispersions",

AIChE Journal 42, 1225 -1233.

62

Prince, M. and Blanch, H.,

"Bubble Coalescence and Break-Up in Air-Sparged Bubble Columns",

AIChE Journal 36, 1485-1499.

63

Hutchings, I.M.,

"Mechanical and metallurgical aspects of the erosion of metals",

Proc. Conf. on Corrosion-Erosion of Coal Conversion System Materials, NACE (1979) 393.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

64

Dosanjh, S., and Humphrey, J.A.C.,

"The influence of turbulence C on erosion by a particle laden fluid jet, Wear",

V.102, 1985, pp. 309-330.

65

Aungier, R.H.,

"Centrifugal Compressors: A strategy for Aerodynamic Design and Analysis",

ASME Press, New York, 2000.

66

Westbrook, C.K., Dryer, F.L.,

"Simplified Reaction Mechanisms for the Oxidation of Hydrocarbon Fuels in Flames",

Combustion Science and Technology Vol. 27, pp. 31-43, 1981.

67

Faeth, G. M.,

"Mixing, transport and combustion in sprays",

Process Energy Combustion Science, Vol. 13, pp. 293-345, 1987.

68

Mijnbeek, G.,

"Bubble column, airlift reactors and other reactor designs", Operational Modes of Bioreactors, Chapter 4,

Butterworth and Heinemann, 1992.

69

Bello, R. A., Robinson, C. W., and Moo-Young, M.,

Canadian Journal of Chemical Engineering, Vol. 62, pp. 573. Chemical Institute of Canada and Canadian Society for Chemical Engineering, 1984.

70

García-Calvo, E. and Letón, P.,

"Prediction of gas hold-up and liquid velocity in airlift reactors using two-phase flow friction coefficients",

Journal of Chemical Technology & Biotechnology, Vol. 67, pp. 388-396,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Wiley Interscience, 1996.

71

Maneri, C. C. and Mendelson, H. D.,

American Institute of Chemical Engineers Journal, Vol. 14, p. 295. American Institute of Chemical Engineers, 1968.

72

Baker, J. L. L. and Chao, B. T.,

American Institute of Chemical Engineers Journal, Vol. 11, p. 268. American Institute of Chemical Engineers, 1965.

73

Hughmark, G. A.,

Industrial Engineering and Chemical Process Design and Development, Vol. 6, p. 218. 1967.

74

S. Lo, R. Bagatin and M. Masi.,

"The Development of a CFD Analysis and Design Tool for Air-lift Reactors",

Proceedings of the SAIChE 2000 Conference, Secunda, South Africa, 2000.

75

Ranz, W.E. and Marshall, W.R.,

Chem. Eng. Prog. 48(3), p. 141, 1952.

76

Bardina, J.E., Huang, P.G. and Coakley, T.J.,

"Turbulence Modeling Validation Testing and Development",

NASA Technical Memorandum 110446, 1997. (See also Bardina, J.E., Huang, P.G. and Coakley, T.J., "Turbulence Modeling Validation", AIAA Paper 97-2121.)

77

H. Schlichting.,

"Boundary Layer Theory",

McGraw-Hill, 1979.

78

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Badzioch, S., and Hawksley, P.G.W.,

"Kinetics of thermal decomposition of pulverised coal particles, Industrial Engineering Chemistry Process Design and Development, 9 p. 521", 1997.

79

S.J. Ubhayakar, D.B. Stickler, C.W. Von Rosenburg, and R.E. Gannon;

"Rapid devolatilization of pulverised coal in hot combustion gases",

16th Symposium (International) on Combustion, The Combustion Institute, p. 426, 1976.

80

Wall, T.F., Phelan, W.J., and Bartz, S.,

"The prediction of scaling of burnout in swirled pulverised coal flames",

InternationalFlame Research Foundation Report F388/a/3 IJmuiden, The Netherlands, 1976.

18.5. References 81-100

81

Sutherland, W.,

"The Viscosity of Gases and Molecular Force",

Phil. Mag. 5:507-531, 1893.

82

Hirschfelder, J.O., Taylor, and M.H., Bird, R.B.,

"Molecular Theory of Gases and Liquids",

Wiley, New York, 1954.

83

Chung, T.H., Lee, L.L., and Starling, K.E.,

"Applications of Kinetic Gas Theories and Multiparameter Correlation for Prediction of Dilute Gas Viscosity and Thermal Conductivity",

Ind. Eng. Chem. Fundam.

23:8, 1984.

84

Poling, B.E., Prausnitz, J.M., and O'Connell, J.P.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"The Properties of Gases and Liquids",

McGraw-Hill, New York, 2001.

85

Redlich, O., and Kwong, J.N.S.,

"On the Thermodynamics of Solutions. V. An Equation of State. Fugacities of Gaseous Solutions.",

Chem Rev 44:233, 1949.

86

Saffman, P. G.,

"The lift on a small sphere in a slow shear flow",

J. Fluid Mech., 22, p. 385, 1965.

87

Mei, R. and Klausner, J. F.,

"Shear lift force on spherical bubbles",

Int. J. Heat and Fluid Flow, 15, p. 62, 1994.

88

Antal, S. P., Lahey, R. T., and Flaherty, J. E.,

"Analysis of phase distribution in fully developed laminar bubbly two-phase flow",

Int. J. Multiphase Flow, 7, pp. 635-652, 1991.

89

Krepper, E., and Prasser, H-M.,

"Measurements and CFX Simulations of a bubbly flow in a vertical pipe",

in AMIFESF Workshop, Computing Methods in Two-Phase Flow, 2000.

90

Burns, A.D.B., Frank, Th., Hamill, I., and Shi, J-M.,

"The Favre Averaged Drag Model for Turbulent Dispersion in Eulerian Multi-Phase Flows",

5th International Conference on Multiphase Flow, ICMF-2004, Yokohama, Japan.

91

Moraga, J.F., Larreteguy, A.E., Drew, D.A., and Lahey, R.T.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Assessment of turbulent dispersion models for bubbly flows in the low Stokes number limit",

Int. J. Multiphase Flow, 29, p. 655, 2003.

92

CIBSE Guide A: Environmental Design

CIBSE, U.K., 1999.

93

ISO 7730-1984(E), Moderate thermal environments - Determination of the PMV and PPD indices and specification of the conditions for thermal comfort

1984.

94

Yamada, T. and R.D. Gunn,

J. Chem Eng. Data, 18, p. 234, 1973.

95

Pitzer, K.S., D.Z. Lippmann, R.F. Curl, C.M. Huggins, and D.E. Petersen,

J. Am. Chem. Soc., 77: 3433 (1955).

96

Aungier, R.H.,

"A Fast, Accurate Real Gas Equation of State for Fluid Dynamic Analysis Applications",

Journal of Fluids Engineering, Vol. 117, pp. 277-281, 1995.

97

H. Enwald, E. Peirano and A. E. Almstedt,

"Eulerian Two-Phase Flow Theory Applied to Fluidisation",

Int. J. Multiphase Flow, 22 Suppl., pp. 21-66,1996.

98

J. Ding and D. Gidaspow,

"A Bubbling Fluidisation Model using Theory of Granular Flow",

AIChEJ. 36, pp. 523-538, 1990.

99

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

C.K.K. Lun, S.B. Savage, D.J. Jeffery, and N. Chepurniy,

"Kinetic Theories for Granular Flow: Inelastic Particles in Couette Flow and Slightly Inelastic Particles in a General Flow Field",

J. Fluid Mech., 140, pp. 223-256, 1984.

100

C.K.K. Lun, and S.B. Savage,

"The Effects of an Impact Velocity Dependent Coefficient of Restitution on Stresses Developed by Sheared Granular Materials",

Acta Mechanica., 63, pp. 15-44, 1986.

18.6. References 101-120

101

Menter, F.R., Langtry, R.B., Likki, S.R., Suzen, Y.B., Huang, P.G., and Völker, S.,

"A Correlation based Transition Model using Local Variables Part 1- Model Formulation",

ASME-GT2004-53452, ASME TURBO EXPO 2004, Vienna, Austria.

102

Langtry, R.B., Menter, F.R., Likki, S.R., Suzen, Y.B., Huang, P.G., and Völker, S.,

"A Correlation based Transition Model using Local Variables Part 2 - Test Cases and Industrial Applications",

ASME-GT2004-53454, ASME TURBO EXPO 2004, Vienna, Austria.

103

Langtry, R.B., Menter, F.R.,

"Transition Modeling for General CFD Applications in Aeronautics",

AIAA paper 2005-522, 2005.

104

Mayle, R.E.,

"The Role of Laminar-Turbulent Transition in Gas Turbine Engines",

ASME Journal of Turbomachinery, Vol. 113, pp. 509-537, 1991.

105

R. Schmehl,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Advanced Modelling of Droplet Deformation and Breakup for CFD Analysis of Mixture Preparation",

ILASS-Europe 2002, 2002.

106

Miller A. and Gidaspow D,

AIChE Journal, Vol. 38, No. 11, p. 1811, 1992.

107

F.X. Tanner,

"Liquid Jet Atomization and Droplet Breakup Modeling of Non-Evaporating Diesel Fuel Sprays",

SAE Technical Paper Series, 970050, 1997.

108

B. Liu, D. Mather and R.D. Reitz,

"Effects of Drop Drag and Breakup on Fuel Sprays",

SAE Technical Paper 930072, 1993.

109

L.P. Hsiang and G.M. Faeth,

"Near-Limit Drop Deformation and Secondary Breakup",

International Journal of Multiphase Flow, Vol. 18, No. 5, pp. 635-652, 1992.

110

S. C. Kuensberg, S.-C., Kong and R. D. Reitz,

"Modelling the Effects of Injector Nozzle Geometry on Diesel Sprays",

SAE Paper 1999-01-0912, 1999.

111

H. Hiroyasu and T. Kadota,

"Fuel droplet size distribution in diesel combustion chamber",

SAE Technical Paper, 740715, 1974.

112

R. Schmehl, G. Maier and S. Wittig,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"CFD Analysis of Fuel Atomization, Secondary Droplet Breakup and Spray Dispersion in the Premix Duct of a LPP Combustor",

Proc. of 8th Int. Conf. on Liquid Atomization and Spray Systems, Pasadena, CA, USA, 2000.

113

Schlichting, H., and Gersten, K.,

"Grenzschicht-Theorie".

9. Auflage, Springer-Verlag Berlin, Heidelberg, New York, 1997

114

W.H. Nurick,

"Orifice Cavitation and Its Effect on Spray Mixing",

Journal of Fluids Engineering, Vol. 98, pp. 681-687, 1976.

115

R.D. Reitz and R. Diwakar,

"Structure of High-Pressure Fuel Sprays",

SAE Technical Paper, 870598, 1987.

116

P.J.O'Rourke and A.A. Amsden,

"The TAB Method for Numerical Calculation of Spray Droplet Breakup",

SAE Technical Paper 872089, 1987.

117

M. Pilch and C.A. Erdman,

"Use of Breakup Time Data and Velocity History Data to Predict the Maximum Size of Stable Fragments for Acceleration-Induced Breakup of a Liquid Drop",

Int. J. Multiphase Flow, Vol. 13, No. 6, pp. 741-757, 1987.

118

S.V. Patankar,

"Numerical Heat Transfer and Fluid Flow",

Hemisphere Publishing Corp., 1980.

119

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

S. Majumdar,

"Role of Underrelaxation in Momentum Interpolation for Calculation of Flow with Non-staggered Grids",

Numerical Heat Transfer 13:125-132.

120

C. Baumgarten, H. Lettmann and G.P. Merker,

"Modelling of Primary and Secondary Break-Up Processes in High Pressure Diesel Sprays",

Paper No. 7, CIMAC Congress, Kyoto 2004.

18.7. References 121-140

121

T. Iijima and T. Takeno,

"Effects of pressure and temperature on burning velocity",

Combust. Flame, Vol. 65, pp. 35-43, 1986.

122

B. Lewis and G. v. Elbe,

"Combustion, Flames and Explosions of Gases",

3rd Edition, Academic Press, London, 1987.

123

B. E. Milton and J.C. Keck,

"Laminar burning velocities in stoichiometric hydrogen and hydrogen-hydrocarbon gas mixtures",

Combust. Flame, Vol. 58, pp. 13-22, 1984.

124

M. Metghalchi and J.C. Keck,

"Burning Velocities of Mixtures of Air with Methanol, iso-octane and indolene at High Pressure and Temperature",

Combust. Flame, Vol. 48, pp. 191-210, 1982.

125

W. Wagner, and A. Kruse,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"The Industrial Standard IAPWS-IF97: Properties of Water and Steam",

Springer, Berlin, 1998.

126

Senecal, P.K., Schmidt, D.P., Nouar, I., Rutland, C.J., Reitz, R.D. and Corradin, M.L.,

"Modeling High-Speed Viscous Liquid Sheet Atomization",

International Journal of Multiphase Flow, 25, pp. 1073-1097, 1999.

127

Han, Z., Perrish, S., Farrell, P.V. and Reitz R.D.,

"Modeling Atomization Processes of Pressure-Swirl Hollow-Cone Fuel Sprays",

Atomization and Sprays, Vol. 7, pp. 663-684, Nov.-Dec. 1997.

128

Kato, M., Launder, B.E.,

"The modelling of turbulent flow around stationary and vibrating square cylinders",

Ninth Symposium on "Turbulent Shear Flows", Kyoto, Japan, August 16-18, 1993.

129

Menter, F. R.,

"Zonal two equation k–ω turbulence models for aerodynamic flows",

AIAA Paper 93-2906, 1993.

130

Menter F. R. and Egorov, Y.,

"Re-visiting the turbulent scale equation",

Proc. IUTAM Symposium; One hundred years of boundary layer research, Göttingen, 2004.

131

Menter, F.R. and Egorov, Y.,

"A Scale-Adaptive Simulation Model using Two-Equation Models",

AIAA paper 2005-1095, Reno/NV, 2005.

132

Menter, F. R, Kuntz, M., Bender R.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"A scale-adaptive simulation model for turbulent flow predictions",

AIAA Paper 2003-0767, 2003.

133

Menter. F. R., Kuntz, M. and Durand. L.,

"Adaptation of eddy viscosity turbulence models to unsteady separated flow behind vehicles",

The Aerodynamics Of Heavy Vehicles: Trucks, Buses And Trains, Monterey, Dec.2-6, 2002.

134

Rotta, J. C.,

"Turbulente Strömungen",

Teubner Verlag, Stuttgart, 1972.

135

Spalart P. R.,

"Young-Person's Guide to Detached-Eddy Simulation Grids",

NASA/CR-2001-211032, 2001.

136

Wilcox, D. C.,

"Turbulence Modelling for CFD",

DWC Industries, La Cañada, 1993.

137

Squires, K.,

"Detached eddy simulation: Current status and future perspectives",

Proc. DLES-5 Conference, München, 2004.

138

Jovic, S., Driver, D. M.,

"Backward-facing step measurement at low Reynolds number, Re_h=5000",

NASA TM 108807, 1994.

139

Roache, P. J.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Verification and Validation in Computational Science and Engineering",

Hermosa publishers, Albuquerque, New Mexico, 1998.

140

Casey, M. and Wintergerste W.,

"Best Practice Guidelines",

ERCOFTAC Special Interest Group on Quality and Trust in Industrial CFD, Report, 2000.

18.8. References 141-160

141

Ferziger, J. H. and Peric, M.,

"Computational methods for fluid dynamics",

Springer, Berlin.

142

Menter, F. R. and Esch T.,

"Elements of Industrial Heat Transfer Predictions",

16th Brazilian Congress of Mechanical Engineering (COBEM), Nov. 2001, Uberlandia, Brazil.

143

Menter, F.,

"CFD Best Practice Guidelines for CFD Code Validation for Reactor Safety Applications",

Evaluation of Computational Fluid Dynamic Methods for Reactor Safety Analysis (ECORA), European Commission, 5th EURATOM FRAMEWORK PROGRAMME, 1998-2002.

144

Menter F. R. and Egorov, Y.,

"Turbulence Models based on the Length-Scale Equation",

Fourth International Symposium on Turbulent Shear Flow Phenomena, Williamsburg, 2005 - Paper TSFP4-268, 2005.

145

Menter F. R. and Egorov, Y.,

"SAS Turbulence Modelling of Technical Flows",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

DLES 6 - 6th ERCOFTAC Workshop on Direct and Large Eddy Simulation September, Poitiers, 2005.

146

Spalart P. R.,

"Strategies for turbulence modelling and simulations",

Int. J. Heat Fluid Flow, 21, pp. 252-263, 2000.

147

A. D. Gosman and E. Ioannides,

"Aspects of computer simulation of liquid fuelled combustors",

AIAA Paper, No. 81-0323,1981.

148

F. Bakir, R. Rey, A.G. Gerber, T. Belamri and B. Hutchinson,

"Numerical and Experimental Investigations of the Cavitating Behavior of an Inducer",

Int J Rotating Machinery, Vol. 10, pp. 15-25, 2004.

149

Frank, Th.,

"Parallele Algorithmen für die numerische Simulation dreidimensionaler, disperser Mehrphasenströmungen und deren Anwendung in der Verfahrenstechnik",

Habilitationsschrift, Shaker Verlag Aachen, pp. 1-329, 2002.

150

Hussmann, B. et al.,

"A stochastic particle-particle collision model for dense gas-particle flows implemented in the Lagrangian solver of ANSYS CFX and its validation",

6th International Conference on Multiphase Flows, ICMF 2007, Leipzig, Germany, 2007.

151

Oesterlé, B., Petitjean, A.,

"Simulations of particle-to-particle interactions in gas-solid flows",

Int. J. Multiphase Flow, Vol. 19(1), pp. 199-211, 1993.

152

Sommerfeld, M.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Modellierung und numerische Berechnung von partikelbeladenen Strömungen mit Hilfe des Euler-Lagrange-Verfahrens",

Habilitationsschrift, Shaker Verlag Aachen, 1996.

153

Lavieville, J., Deutsch, E. and Simonin, O.,

"Large eddy simulations of interactions between colliding particles and a homogeneous isotropic turbulence field",

ASME FED Vol. 228, pp. 359-369, 1995.

154

Sommerfeld, M.,

"Validation of a stochastic Lagrangian modeling approach for inter-particle collision in homogeneous isotropic turbulence",

Int. J. Multiphase Flow, Vol. 27, pp. 1829 – 1858, 2001.

155

Huh, K.Y., Lee, E.,

"Diesel Spray Atomization Models Considering Nozzle Exit Turbulence Conditions",

Atomization and Sprays, Vol. 8, pp. 453-469, 1998.

156

Chryssakis, C.A., Assanis, D.N.,

"A Secondary atomization Model for Liquid Droplet Deformations and Breakup under High Weber Number Conditions",

ILASS Americas, 18th Annual Conference on Liquid Atomization and Spray Systems, Irvine, CA, 2005.

157

Peng, D.Y. and Robinson, D.B.,

"A New Two-Constant Equation of State",

Ind. Eng. Chem. Fundam., Vol. 15, No. 1, pp. 59 – 64, 1976.

158

Chung, T.H., M. Ajlan, L.L. Lee and K.E. Starling,

"Generalized Multiparameter Correlation for Nonpolar and Polar Fluid Transport Properties",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Ind. Eng. Chem. Res., Vol. 27, pp. 671 – 679, 1988.

159

Kurul, N. and Podowski, M. Z.,

"On the modeling of multidimensional effects in boiling channels",

ANS Proc. 27th National Heat Transfer Conference, Minneapolis, MN, July 28-31, 1991.

160

Kocamustafaogullari, G. and Ishii, M.,

"Interfacial area and nucleation site density in boiling systems",

Int. J. Heat Mass Transfer, 26 p. 1377, 1983.

18.9. References 161-180

161

Podowski, M. Z., Alajbegovic, A., Kurul, N., Drew, D.A. and Lahey, R. T.,

"Mechanistic modelling of CHF in forced-convection sub-cooled boiling",

Int. Conference on Convective Flow and Pool Boiling, Irsee, Germany, 1997a.

162

Podowski, R. M., Drew, D.A., Lahey, R. T. and Podowski, M. Z.,

"A mechanistic model of the ebullition cycle in forced-convection sub-cooled boiling",

NURETH-8, Kyoto, Japan, 1997b.

163

Egorov, Y. and Menter, F.,

"Experimental implementation of the RPI boiling model in CFX-5.6",

Technical Report ANSYS / TR-04-10., 2004.

164

Lemmert, M. and Chawla, J. M.,

"Influence of flow velocity on surface boiling heat transfer coefficient",

Heat Transfer and Boiling (Eds. E. Hahne and U. Grigull), Academic Press, 1977.

165

Tolubinski, V. I. and Kostanchuk, D. M.,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

"Vapour bubbles growth rate and heat transfer intensity at subcooled water boiling",

4th. International Heat Transfer Conference, Paris, France, 1970.

166

Cole, R.,

"A photographic study of pool boiling in the region of CHF",

AIChEJ, 6 pp. 533-542, 1960.

167

Mikic, B. B. and Rohsenow, W. M.,

"A new correlation of pool boiling data including the fact of heating surface characteristics",

ASME J. Heat Transfer, 91 pp. 245-250, 1969.

168

Del Valle, V. H. and Kenning, D. B. R.,

"Subcooled flow boiling at high heat flux",

Int. J. Heat Mass Transfer, 28 p. 1907, 1985.

169

Ceumern-Lindenstjerna, W. C.,

"Bubble Departure and Release Frequencies During Nucleate Pool Boiling of Water and Aqueous NaCl Solutions",

Heat Transfer in Boiling, Academic Press and Hemisphere, 1977.

170

Saffman, P. G.,

Corrigendum to: "The lift on a small sphere in a slow shear flow",

J. Fluid Mech., 31, p. 624, 1968

171

Legendre, D. and Magnaudet, J.,

"The lift force on a spherical bubble in a viscous linear shear flow",

J. Fluid Mech., 368, pp. 81–126, 1998.

172

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Tomiyama, A.,

"Struggle with computational bubble dynamics",

ICMF'98, 3rd Int. Conf. Multiphase Flow, Lyon, France, pp. 1-18, June 8-12, 1998.

173

Frank, Th., Shi, J. M. and Burns, A. D.,

"Validation of Eulerian Multiphase Flow Models for Nuclear Safety Applications",

3rd International Symposium on Two-Phase Flow Modelling and Experimentation, Pisa, Italy, 22-24, Sept. 2004.

174

Wellek, R. M., Agrawal, A. K. and Skelland, A. H. P.,

"Shapes of liquid drops moving in liquid media",

AIChE J, 12, pp. 854-862, 1966.

175

G. Elsässer,

"Experimentelle Untersuchung und numerische Modellierung der freien Kraftstoffstrahlausbreitung und Wandinteraktion unter motorischen Randbedingungen",

Dissertation, Logos Verlag, Berlin, 2001

176

C. Bai and A.D. Gosman,

"Prediction of spray wall impingement in reciprocating engines",

ILASS-Europe, July 1999

177

Frank, Th., Zwart, P. J., Krepper, E., Prasser, H. -M. and Lucas,

"Validation of CFD models for mono- and polydisperse air-water two-phase flows in pipes"

J. Nuclear Engineering & Design, Vol. 238, pp. 647–659, March 2008.

178

Lighthill, M. J.,

"On sound generated aerodynamically. I. General theory"

Proc. R. Soc. Series A, Vol. 211, p. 564, 1952.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

179

Lighthill, M. J.,

"On sound generated aerodynamically. II. Turbulence as a source of sound"

Proc. R. Soc. Series A, Vol. 222, 1954.

180

Ffowcs-Williams, J. E. and Hawkings, D. L.,

"Theory relating to the noise of rotating machinery"

J. Sound Vib., Vol. 10, pp. 10-21, 1969.

18.10. References 181-200

181

Wen, C. Y. and Yu, Y. H.,

"Mechanics of Fluidization"

Chem. Eng. Prog. Symp. Ser. 62, pp. 100-111, 1966.

182

Choi, C. R. and Huh, K. Y.,

"Development and validation of a coherent flamelet model for a spark-ignited turbulent premixed flame in a closed vessel,"

Combustion & Flame Vol. 114, No. 3-4, 336-348, 1998.

183

A. M. Douaud, P.Eyzat,

"Four-Octane-Number Method for Predicting the Anti-Knock Behavior of Fuels and Engines",

SAE Technical Paper 780080, SAE, 1978.

184

M. P. Halstead, L. J. Kirsch, C. P. Quinn,

"The Autoignition of Hydrocarbon Fuels at High Temperatures and Pressures – Fitting of a Mathematical Model",

Combustion and Flame, Vol. 30, pp. 45-60, 1977.

185

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

H. O. Hardenberg, F.W. Hase,

"An Empirical Formula for Computing the Pressure Rise Delay of a Fuel from Its Cetane Number and from the Relevant Parameters of Direct-Injection Diesel Engines",

SAE Technical Paper 790493, SAE, 1979.

186

Meneveau, C., and Poinsot, T.,

"Stretching and quenching of flamelets in premixed turbulent combustion",

Combustion and Flame, 86:311-332, 1991.

187

T. Poinsot, D. Veynante,

"Theoretical and Numerical Combustion",

Edwards, 2001.

188

Wallin, S. and Johansson A.,

"A complete explicit algebraic Reynolds stress model for incompressible and compressible flows",

Journal of Fluid Mechanics, 403, pp. 89-132, 2000.

189

Wallin, S., and Johansson A.,

"Modelling streamline curvature effects in explicit algebraic Reynolds stress turbulence models",

International journal of Heat and Fluid Flow, 23(5), pp. 721-730, 2002.

190

Hellsten, A.,

"New advanced k–ω turbulence model for high-lift aerodynamics",

AIAA Paper 2004-1120, Reno, Nevada, 2004.

191

Spalart, P.R., and Shur, M.

"On the sensitization of turbulence models to rotation and curvature",

Aerospace Sci. Tech., 1(5), pp. 297-302, 1997.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

192

Smirnov, P.E., and Menter, F.R.

"Sensitization of the SST turbulence model to rotation and curvature by applying the Spalart-Shur correction term",

ASME Paper GT 2008-50480, Berlin, Germany, 2008.

193

Coleman, H.W., Hodge, B.K., Taylor, R.P.,

"A Re-Evaluation of Schlichting's Surface Roughness Experiment",

Journal of Fluids Engineering, Vol. 106, 1984.

194

Lechner, R., and Menter, F.,

"Development of a rough wall boundary condition for ω-based turbulence models",

Technical Report ANSYS / TR-04-04, 2004.

195

Pimenta, M.M., Moffat, R.J. and Kays, W.M.,

" The Turbulent Boundary Layer: An Experimental Study of the Transport of Momentum and Heat with the Effect of Roughness",

Interim Report Stanford University, CA, 1975.

196

Launder, B.E.,

"Second-moment closure: present … and future".

Int. J. Heat and Fluid Flow, Vol. 10, No. 4, pp. 282-300, 1989.

197

Egorov, Y., and Menter, F.,

"Development and Application of SST-SAS Turbulence Model in the DESIDER Project",

Second Symposium on Hybrid RANS-LES Methods, Corfu, Greece, 2007.

198

Germano, M., Piomelli, U., Moin, P., Cabot, W.H.,

"A Dynamic Subgrid-Scale Eddy Viscosity Model",

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Phys. Fluids A 3 (7), pp. 1760-1765, 1991.

199

Lilly, D.K.,

"A Proposed Modification of the Germano Subgrid-Scale Closure Method",

Phys. Fluids A 4 (3), pp. 633-635, 1992.

200

Nicoud, F., Ducros, F.,

"Subgrid-Scale Stress Modelling Based on the Square of the Velocity Gradient Tensor",

Flow, Turbulence and Combustion, 62, pp. 183-200, 1999.

18.11. References 201-220

201

Abramzon, B. and Sirignano, W.A.,

"Droplet Vaporization Model for Spray Combustion Calculations"

Int. J. Heat Mass Transfer, 32, pp. 1605–1618, 1989.

202

Sazhin, Sergei S.,

"Advanced Models of Fuel Droplet Heating and Evaporation"

Progress in Energy and Combustion Science, 32, pp. 162–214, 2006.

203

Hughes, T.J.R.,

"The Finite Element Method"

Prentice-Hall, Englewood Cliffs, N.J., 1987.

204

Simo, J.C. and Wong, K.K.,

"Unconditionally Stable Algorithms for Rigid Body Dynamics that exactly Preserves Energy and Momentum"

Int. J. Num. Methods in Eng., 31, pp. 19-52, 1991.

205

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Hughes, Peter C.,

"Spacecraft Attitude Dynamic"

Dover, 2004.

206

Etkin, Bernard.,

"The Dynamics of Atmospheric Flight"

John Wiley & Sons, 1972.

207

Conaire, Marcus Ó, Curran, Henry J., Simmie, John M., Pitz, William J., Westbrook, Charles K.,

"A Comprehensive Modeling Study of Hydrogen Oxidation",

International Journal of Chemical Kinetics,

Vol. 36, Issue 11, pp. 603-622, 2004.

208

Frank, Th.

"Numerische Simulation der feststoffbeladenen Gasströmung im horizontalen Kanal unter Berücksichtigung von Wandrauhigkeiten"

Ph.D. Thesis, Techn. University Bergakademie Freiberg, Germany, 1992

209

Matsumoto, S., Saito, S., Maeda, S.

"Simulation of Gas-Solid Two-Phase Flow in Horizontal Pipe"

Journal of Chemical Engineering of Japan

Vol. 9, No. 1, pp. 23–28, 1976

210

Tsuji, Y., Oshima, T., Morikawa, Y.

"Numerical Simulation of Pneumatic Conveying in a Horizontal Pipe"

KONA Powder Science and Technology in Japan

No. 3, pp. 38–51, 1985

211

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Frank, Th.

"Parallele Algorithmen für die numerische Simulation dreidimensionaler, disperser Mehrphasenströmungen und deren Anwendung in der Verfahrenstechnik"

Doctorial Dissertation, Shaker Verlag, 2002

212

Sommerfeld, M.

"Numerical Simulation of the Particle Dispersion in Turbulent Flow: the Importance of Particle Lift Forces and Particle/Wall Collision Models"

ASME Symposium on Numerical Methods for Multiphase Flows, Toronto, Canada

pp. 1–8, 1990

213

B. P. Leonard,

"The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection,"

Comp. Methods Appl. Mech. Eng.,

88:17–74, 1991

214

Jasak, H.; Weller, H.G., Gosman, A.D.

"High resolution NVD differencing scheme for arbitrarily unstructured meshes,"

Int. J. Numer. Meth. Fluids, 1999,

pp. 413 – 449.

215

Erdos, J.A.

"Numerical Solution of Periodic Transonic Flow through a Fan Stage"

AIAA Journal, 1977.

pp. 1559-1568.

216

Gerolymos G.A.

"Analysis and Application of Chorochronic Periodicity in Turbomachinery Rotor/Stator Interaction Computations"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Journal of Propulsion and Power, 2002.

pp. 1139-1152.

217

Giles, M.

"Calculation of Unsteady Wake/Rotor Interaction"

J. Propulsion, 1988.

pp. 356-362.

218

He, L.

"An Euler Solution for Unsteady Flows Around Oscillating Blades"

Transactions of the ASME, 1990.

pp. 714-722.

219

Apsley, D.D. and Leschziner, M.A.

"A new low-reynolds-number nonlinear two-equation turbulence model for complex flows"

International Journal of Heat and Fluid Flow, 19, pp. 209-222, 1998.

220

Coffee, T. P.

"Comment on *Simplified Reaction Mechanisms for the Oxidation of Hydrocarbon Fuels in Flames by C. K. Westbrook and F. T. Dryer*"

Combustion Science and Technology, 43, pp. 333–339, 1985.

18.12. References 221-

221

Spalart, P. R., Deck, S., Shur, M. L., Squires, K. D., Strelets, M. Kh., Travin, A.

"A new version of detached-eddy simulation, resistant to ambiguous grid densities"

Theoretical and Computational Fluid Dynamics, Vol. 20, pp. 181-195, 2006.

222

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

C. Temperton

"Implementation of a Self-Sorting In-Place Prime Factor FFT Algorithm"

Journal of Computational Physics, 58, pp. 283–299, 1985.

223

Arnal, D., Habiballah, M., Coutols, E.

"Théorie de l'instabilité laminaire et critéres de transition en éncoulement bi et tridimensionne"

LaRecherche Aérospatiale, No. 2, pp. 125-143, 1984.

224

Menter F.R., Smirnov P. E., Liu, T., Avancha, R.

"A One-Equation Local Correlation-Based Transition Model"

Flow, Turbulence and Combustion, July 2015.

225

Viswanathan K.

"Aeroacoustics of hot jets"

Journal of Fluid Mechanics, 516, 39-82, 2004.

226

Gritskevich, M.S., Garbaruk, A.V., Schütze, J., and Menter, F.R.

"Development of DDES and IDDES Formulations for the k-ω Shear Stress Transport Model"

J. Flow Turbulence and Combustion, Vol. 88 Issue 3, pp. 431-449, 2012.

227

Zschaeck, G., Frank, T., Burns, A. D.

"CFD modelling and validation of wall condensation in the presence of non-condensable gases"

J. Nuclear Engineering & Design, Vol. 279, pp. 137–146, 2014.

228

Hall K. C., Thomas J. P., and Clark W. S.

"Computation of unsteady nonlinear flows in cascades using a harmonic balance technique"

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

AIAA Journal, 40 (5), 879-886, 2002.

229

Gopinath A. and Jameson A.

"Time Spectral Method for Periodic Unsteady Computations over Two and Three- Dimensional Bodies"

43rd AIAA Aerospace Sciences Meeting and Exhibit, AIAA Paper 2005-1220, Reno (USA), 2005.

230

B. Aupoix.

"Roughness Corrections for the k-ω Shear Stress Transport Model: Status and Proposals"

Journal of Fluids Engineering, 137, 2015.

231

auf dem Kampe, T., Völker, S., and Zehe, F.,

"A Model for Cylindrical Hole Film Cooling: Part I - A Correlation for Jet-Flow with Application to Film Cooling",

Proceedings of ASME Turbo Expo 2010 (GT2010-22787).

232

auf dem Kampe, T. and Völker, S.,

"A Model for Cylindrical Hole Film Cooling: Part II - Model Formulation, Implementation and Results",

Proceedings of ASME Turbo Expo 2010 (GT2010-22788).

233

Gyarmathy, G.,

"Grundlagen einer Theorie der Naβdampfturbine",

Dissertation, ETH Zürich, 1962.

234

Young, J. B.,

"The Spontaneous Condensation of Steam in Supersonic Nozzles",

PhysicoChemical Hydrodynamics, 3(1):57-82, 1982.

235

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Jörg Starzmann et al.,

"Results of the International Wet Steam Modelling Project",

Wet Steam Conference, Prague, September 12-14 2016.

236

Krepper, E. and Rzehak, R.,

"CFD for Subcooled Flow Boiling: Analysis of DEBORA Tests",

J. Computational Multiphase Flows, Vol. 6, No. 3, 2014.

237

Morel, C.,

"Turbulence modeling and first numerical simulation in turbulent two-phase flows",

CEA, Grenoble, France, Oct. 1997.

238

Marie, J. L., Moursali, E., Tran-Cong, S.,

"Similarity law and turbulence intensity profiles in a bubbly boundary layer at low void fractions",

International Journal of Multiphase Flow, Vol. 23, pp. 227-247, 1997.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information
of ANSYS, Inc. and its subsidiaries and affiliates.

Glossary

Symbols

<CFDPOSTROOT>	The directory in which CFD-Post is installed; for example: `C:\Program Files\ANSYS Inc\v251\CFD-Post\`
<CFXROOT>	The directory in which CFX is installed; for example: `C:\Program Files\ANSYS Inc\v251\CFX\`

A

absolute pressure	The summation of solver pressure, reference pressure, and hydrostatic pressure (if a buoyant flow) in the cavitation model. The absolute pressure is clipped to be no less than the vapor pressure of the fluid. It is used by the solver to calculate pressure-dependent properties (such as density for compressible flow).
absorption coefficient	A property of a medium that measures the amount of thermal radiation absorbed per unit length within the medium.
adaption criteria	The criteria that are used to determine where mesh adaption takes place.
adaption level	The degree that a mesh element has been refined during adaption. Each mesh element has an adaption level. Each time an element is split into smaller elements, the new elements have an adaption level that is one greater than the "parent" element. The maximum number of adaption levels is controlled to prevent over-refinement.
adaption step	One loop of the adapt-solve cycle in the mesh adaption process.
Additional Variable	A non-reacting, scalar component. Additional Variables are used to model the distribution of passive materials in the flow, such as smoke in air or dye in water. Additional Variables are typically specified as concentrations.
adiabatic	The description of any system in which heat is prevented from crossing the boundary of the system. You can set adiabatic boundary conditions for heat transfer simulations in Ansys CFX or in Fluent.
Advancing Front and Inflation (AFI)	The default meshing mode in CFX. The AFI mesher consists of a triangular surface/tetrahedral volume mesh generator that uses the advancing front method to discretize first the surface and then the volume into an unstructured (irregular) mesh. Inflation can be applied to selected surfaces to produce prismatic elements from the triangular surface mesh, which combine with the tetrahedra to form a hybrid mesh.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

all domains
: In immersed-solids cases in CFD-Post, "all domains" refers to all of the domains in the case *excluding* the immersed solid. This is done for backwards compatibility.

  Generally speaking, only the wireframe needs to keep track of both "all domains" *and* the immersed solid.

ASM (Algebraic Slip Model)
: A mathematical form in which geometry may be represented, known as parametric cubic.

aspect ratio
: Also known as normalized shape ratio. A measure of how close to a regular tetrahedron any tetrahedron is. The aspect ratio is 1 for a regular tetrahedron, but gets smaller the flatter the tetrahedron gets. Used for judging how good a mesh is.

B

backup file
: An intermediate CFX-Solver Results file that can be manually generated during the course of a solution from the CFX-Solver Manager interface by using the **Backup** action button. Backup files should be generated if you suspect your solution may be diverging and want to retain the intermediate solution from which you can do a restart.

batch mode
: A way to run some components of Ansys CFX without needing to open windows to control the process. When running in batch mode, a Viewer is not provided and you cannot enter commands at a command prompt. Commands are issued via a CFD-Post session file (`*.cse`), the name of which is specified when executing the command to start batch mode. The session file can be created using a text editor, or, more easily, by recording a session while running in line-interface or user-interface mode.

blend factor
: A setting that controls the degree of first/second order blending for the advection terms in discrete finite volume equations.

body
: A collection of surfaces that completely and unambiguously enclose a finite volume. Modelers that create so-called B-Rep models create "bodies." This term was coined to distinguish between the tri-parametric entities, known herein as solids, and the shell-like representations produced by most CAD systems.

boundary
: A surface or edge that limits the extent of a space. A boundary can be internal (the surface of a submerged porous material) or external (the surface of an airfoil).

boundary condition
: Physical conditions at the edges of a region of interest that you must specify in order to completely describe a simulation.

buoyant flow
: Flow that is driven wholly or partially by differences in fluid density. For fluids where density is not a function of temperature, pressure, or Additional Variables, the Boussinesq approximation is employed.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	If density is a function of one of these, then the Full Buoyancy model is employed.

C

CEL (CFX Expression Language)	A high level language used within CFX to develop expressions for use in your simulations. CEL can be used to apply user-defined fluid property dependencies, boundary conditions, and initial values. Expressions can be developed within CFX using the Expression Editor.
CFD (Computational Fluid Dynamics)	The science of predicting fluid flow, heat transfer, mass transfer (as in perspiration or dissolution), phase change (as in freezing or boiling), chemical reaction (as in combustion), mechanical movement (as in fan rotation), stress or deformation of related solid structures (such as a mast bending in the wind), and related phenomena by solving the mathematical equations that govern these processes using a numerical algorithm on a computer.
CFX-Solver Input file	A file that contains the specification for the whole simulation, including the geometry, surface mesh, boundary conditions, fluid properties, solver parameters and any initial values. It is created by CFX and used as input to CFX-Solver.
CHT (Conjugate Heat Transfer)	Heat transfer in a conducting solid.
clipping plane	A plane that is defined through the geometry of a model, in front of which no geometry is drawn. This enables you to see parts of the geometry that would normally be hidden.
command actions	Command actions are: • Statements in session files • Commands entered into the **Tools** > **Command Editor** dialog box • Commands entered in Line Interface mode. All such actions must be preceded with the > symbol. These commands force CFD-Post to undertake specific tasks, usually related to the input and output of data from the system. See also Power Syntax (p. 626).
component	A substance containing one or more materials in a fixed composition. The properties of a component are calculated from the mass fractions of the constituent materials and are based on the materials forming an ideal mixture.
compressible flow	Flow in which the fluid volume changes in response to pressure change. Compressible flow effects can be taken into consideration when the Mach number (M) approaches approximately 0.2.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

computational mesh
: A collection of points representing the flow field where the equations of fluid motion (and temperature, if relevant) are calculated.

control volume
: The volume surrounding each node, defined by segments of the faces of the elements associated with each node. The equations of fluid flow are solved over each control volume.

convergence
: A state of a solution that occurs when the change in residual values from one iteration to the next are below defined limits.

corrected boundary node values
: Node values obtained by taking the results produced by CFX-Solver (called "conservative values") and overwriting the results on the boundary nodes with the specified boundary conditions.

  The values of some variables on the boundary nodes (that is, on the edges of the geometry) are not precisely equal to the specified boundary conditions when CFX-Solver finishes its calculations. For instance, the value of velocity on a node on the wall will not be precisely zero, and the value of temperature on an inlet may not be precisely the specified inlet temperature. For visualization purposes, it can be more helpful if the nodes at the boundary do contain the specified boundary conditions and so "corrected boundary node values" are used. Corrected boundary node values are obtained by taking the results produced by CFX-Solver (called "conservative values") and overwriting the results on the boundary nodes with the specified boundary conditions. This will ensure the velocity is display as zero on no-slip walls and equal to the specified inlet velocity on the inlet, for example.

coupled solver
: A solver in which all of the hydrodynamic equations are solved simultaneously as a single system. The advantages of a coupled solver are that it is faster than a traditional solver and fewer iterations are required to obtain a converged solution. CFX-Solver is an example of a coupled solver.

curve
: A general vector valued function of a single parametric variable. In CFX, a line is also a curve. By default, curves are displayed in yellow in Ansys CFX.

D

default boundary condition
: The boundary condition that is applied to all surfaces that have no boundary condition explicitly set. Normally, this is set to the No Slip Adiabatic Wall boundary condition, although you can change the type of default boundary condition in CFX.

Detached Eddy Simulation (DES)
: A model that covers the boundary layer by a RANS model and switches to a LES model in detached regions.

Direct Numerical Simulation (DNS)
: A CFD simulation in which the Navier-Stokes equations are solved without any turbulence model.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

discretization

The equations of fluid flow cannot be solved directly. Discretization is the process by which the differential equations are converted into a system of algebraic equations, which relate the value of a variable in a control volume to the value in neighboring control volumes.

domain

Regions of fluid flow and/or heat transfer in CFX are called domains. Fluid domains define a region of fluid flow, while solid domains are regions occupied by conducting solids in which volumetric sources of energy can be specified. The domain requires three specifications:

- The region defining the flow or conducting solid. A domain is formed from one or more 3D primitives that constrain the region occupied by the fluid and/or conducting solids.
- The physical nature of the flow. This determines the modeling of specific features such as heat transfer or buoyancy.
- The properties of the materials in the region.

There can be many domains per model, with each domain defined by separate 3D primitives. Multidomain problems may be created from a single mesh if it contains multiple 3D primitives or is from multiple meshes.

dynamic viscosity

Dynamic viscosity, also called absolute viscosity, is a measure of the resistance of a fluid to shearing forces.

dynamical time

For advection dominated flows, this is an approximate timescale for the flow to move through the Domain. Setting the physical time step (p. 626) size to this value (or a fraction of it) can promote faster convergence.

E

eddy viscosity model

A turbulence model based on the assumption that Reynolds stresses are proportional to mean velocity gradients and that the Reynolds stress contribution can be described by the addition of a turbulent component of viscosity. An example of an eddy viscosity model is the k-ϵ model.

edge

The edge entity describes the topological relationships for a curve. Adjacent faces share at least one edge.

emissivity

A property of an object that describes how much radiation it emits as compared to that of a black body at the same temperature.

expansion factor

The rate of growth of volume elements away from curved surfaces and the rate of growth of surface elements away from curved boundaries. Expansion factor is also used to specify the rate of mesh coarsening from a mesh control.

expression editor

An interactive, form-driven facility within CFX for developing expressions.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

external flow — A flow field that is located outside of your geometry.

F

face — "Face" can have several meanings:

- A solid face is a surface that exists as part of a solid. It is also known as an implicit surface.
- An element face is one side of a mesh element.
- A boundary face is an element face that exists on the exterior boundary of the domain.
- Surfaces composed of edges that are connected to each other.

FLEXlm — The program that administers Ansys licensing.

flow boundaries — The surfaces bounding the flow field.

flow region — A volumetric space containing a fluid. Depending on the flow characteristics, you may have a single, uninterrupted flow region, or several flow regions, each exhibiting different characteristics.

flow symmetry — Flow where the conditions of the flow entering and leaving one half of a geometry are the same as the conditions of the flow entering and leaving the other half of the geometry.

fluid — A substance that tends to flow and assumes the shape of its domain, such as a gas in a duct or a liquid in a container.

free edges — Element edges belonging to only one element.

G

gas or liquid surface — A type of boundary that exhibits no friction and fluid cannot move through it. Also called a symmetry boundary.

general fluid — A fluid whose properties may be generally prescribed in Ansys CFX or Fluent. Density and specific heat capacity for general fluids may depend on pressure, temperature, and any Additional Variables.

global model tolerance — The minimum distance between two geometry entities below which CFX considers them to be coincident. The default setting of global model tolerance, defined in the template database, is normally .005 in whichever geometry units you are working.

geometric symmetry — The state of a geometry where each half is a mirror of the other.

group — A named collection of geometric and mesh entities that can be posted for display in viewports. The group's definition includes:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

- Group name
- Group status (current/not current)
- Group display attributes (modified under Display menu)
- A list of the geometric and mesh entities that are members of the group.

H

hexahedral element

A mesh element with the same topology as a hexahedron, with six faces and eight vertices.

home directory

The directory on all Linux systems and some Windows NT systems where each user stores all of their files, and where various set-up files are stored.

However, on some Windows NT systems, users do not have an equivalent to the Linux home directory. In this case, the Ansys CFX set-up file `cfx5rc` can be placed in `c:\winnt\profiles\<user>\Application Data\Ansys CFX\<release>`, where <*user*> is the user name on the machine. Other files can be put into a directory set by the variable **HOME**.

I

ideal gas

A fluid whose properties obey the ideal gas law. The density is automatically computed using this relationship and a specified molecular weight.

IGES (Initial Graphics Exchange Specification) file

An ANSI standard formatted file used to exchange data among most commercial CAD systems. IGES files can be imported into CFX.

implicit geometry

Geometry that exists as part of some other entity. For example, the edges of a surface are implicit curves.

import mesh

A meshing mode that enables import of volume meshes generated in one of a number of external CFD packages. The volume mesh can contain hexahedral, tetrahedral, prismatic, and pyramidal element types.

inactive region

A fluid or porous region where flow and (if relevant) temperatures are not being calculated, or a solid region where temperatures are not being calculated. By default, inactive regions are hidden from view in the graphics window.

incompressible flow

Flow in which the density is constant throughout the domain.

incremental adaption

The method of mesh adaption used by CFX where an existing mesh is modified to meet specified criteria. Incremental adaption is much

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	faster than remeshing; however, the mesh quality is limited by that of the initial mesh.
inertial resistance coefficients	Mathematical terms used to define porous media resistance.
initial guess	The values of dependent variables at the start of a steady-state simulation. These can set explicitly, read from an existing solution, or given default values.
initial values	The values of dependent variables at the initial time of a transient simulation. These can be either set explicitly, or read from an existing solution.
inlet boundary condition	A boundary condition (p. 616) for which the quantity of fluid flowing into the flow domain is specified, for example, by setting the fluid velocity or mass flow rate.
instancing	The process of copying an object and applying a positional transform to each of the copies. For example, a row of turbine blades can be visualized by applying instancing to a single blade.
interior boundary	A boundary that enables flow to enter and exit. These types of boundaries are useful to separate two distinct fluid regions from each other, or to separate a porous region from a fluid region, when you still want flow to occur between the two regions.
internal flow	Flow through the interior of your geometry, such as flow through a pipe.
interpolation	The process of transferring a solution from a CFX-Solver Results file containing one mesh onto a second file containing a different mesh.
isentropic	The description of a process where there is no heat transfer and entropy is held constant.
isosurface	A surface of constant value for a given variable. A three-dimensional surface that defines a single magnitude of a flow variable such as temperature, pressure, velocity, and so on.
Isovolume	A locator that consists of a collection of volume elements, all of which take a value of a variable greater than a user-specified value.

J

JPEG file	A common graphics file type that is supported by CFD-Post output options.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

K

k-epsilon turbulence model
: A turbulence model (p. 631) based on the concept that turbulence consists of small eddies that are continuously forming and dissipating. The k-epsilon turbulence model solves two additional transport equations: one for turbulence generation (k), and one for turbulence dissipation (epsilon).

kinematic diffusivity
: A function of the fluid medium that describes how rapidly an Additional Variable would move through the fluid in the absence of convection.

L

laminar flow
: Flow that is dominated by viscous forces in the fluid, and characterized by low Reynolds Number.

  A flow field is laminar when the velocity distributions at various points downstream of the fluid entrance are consistent with each other and the fluid particles move in a parallel fashion to each other. The velocity distributions are effectively layers of fluid moving at different velocities relative to each other.

Large Eddy Simulation Model (LES)
: The *Large Eddy Simulation model* decomposes flow variables into large and small scale parts. This model solves for large-scale fluctuating motions and uses "sub-grid" scale turbulence models for the small-scale motion.

legend
: A color key for any colored plot.

line interface mode
: A mode in which you type the commands that would otherwise be issued by the user interface. A viewer is provided that shows the geometry and the objects created on the command line. Line interface mode differs from entering commands in the Command Editor dialog box in that line interface action commands are not preceded by a > symbol. Aside from that difference, all commands that work for the Command Editor dialog box will also work in line interface mode, providing the correct syntax is used.

locator
: A place or object upon which a plot can be drawn. Examples are planes and points.

M

MAlt key (Meta key)
: The MAlt key (or Meta key) is used to keyboard select menu items with the use of mnemonics (the underscored letter in each menu label). By simultaneously pressing the MAlt key and a mnemonic is an alternative to using the mouse to click a menu title. The MAlt key is different for different brands of keyboards. Some examples of MAlt keys include the " " key for Sun Model Type 4 keyboards,

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	the "Compose Character" key for Tektronix keyboards, and the **Alt** key on most keyboards for most Windows-based systems.
mass fraction	The ratio of the mass of a fluid component to the total mass of the fluid. Values for mass fraction range from 0 to 1.
material	A substance with specified properties, such as density and viscosity.
meridional	A term used in Fluent documentation that is equivalent to the Ansys CFX term *constant streamwise location*.
mesh	A collection of points representing the flow field where the equations of fluid motion (and temperature, if relevant) are calculated.
mesh adaption	The process by which, once or more during a run, the mesh is selectively refined at various locations, depending on criteria that you can specify. As the solution is calculated, the mesh can automatically be refined in locations where solution variables are changed rapidly, in order to resolve the features of the flow in these regions. There are two general methods for performing mesh adaption. *Incremental adaption* takes an existing mesh and modifies it to meet the adaption criteria. The alternative is *remeshing*, in which the whole geometry is remeshed at every adaption step according to the adaption criteria. In CFX, incremental adaption is used because this is much faster; however, this imposes the limitation that the resulting mesh quality is limited by the quality of the initial mesh.
mesh control	A refinement of the surface and volume mesh in specific regions of the model. Mesh controls can take the form of a point, line, or triangle.
meshing mode	The method you use to create your mesh of nodes and elements required for analysis. There are two main meshing modes: • Advancing Front and Inflation (AFI) (p. 615) • import mesh (p. 621)
minimal results file	A file that contains only the results for selected variables, and no mesh. It can be created only for transient calculations. It is useful when you are only interested in particular variables and want to minimize the size of the results for the transient calculation.
multicomponent fluid	A fluid consisting of more than one component. The components are assumed to be mixed at the molecular level, though the proportions of each component may vary in space or time. The properties of a multicomponent fluid are dependent on the proportion of constituent components.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

N

Navier-Stokes equations
: The fundamental equations of fluid flow and heat transfer, solved by CFX-Solver. They are partial differential equations.

new model preferences
: Preferential settings for your model that define the meshing mode (p. 624), the geometry units, and the global model tolerance (p. 620).

node allocation parameter
: A parameter that is used in mesh adaption (p. 624) to determine how many nodes are added to the mesh in each adaption step (p. 615).

non-clipped absolute pressure
: The summation of solver pressure, reference pressure, and hydrostatic pressure (if a buoyant flow). This pressure, used by the solver to calculate cavitation sources, can be negative or positive.

non-Newtonian fluid
: A fluid that does not follow a simple linear relationship between shear stress and shear strain.

normal
: The direction perpendicular to the surface of a mesh element or geometry. The positive direction is determined by the cross-product of the local parametric directions in the surface.

O

open area
: The area in a porous region that is open to flow.

OpenGL
: A graphics display system that is used on a number of different types of computer operating systems.

outlet
: A boundary condition where the fluid is constrained to flow only out of the domain.

outline plot
: A plot showing the outline of the geometry. By setting the edge angle to 0, the surface mesh can be displayed over the whole geometry.

CFX-Solver Output file
: A text file produced by CFX-Solver, with a file name of the form `*.out`. This file details the history of a run. It is important to browse the CFX-Solver Output file when a run is finished in order to determine whether the run has converged and whether a restart is necessary.

P

parallel runs
: Separate solutions of sections (partitions) of your CFD model, run on more than one processor.

parametric equation
: Any set of equations that express the coordinates of the points of a curve as functions of one parameter, or express the coordinates of the points of a surface as functions of two parameters, or express

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	the coordinates of the points of a solid as functions of three parameters.
parametric solids	Six-sided solids parameterized in three normalized directions. Parametric solids are colored blue Ansys CFX.
parametric surfaces	Four sided surfaces parameterized in two normalized directions. Parametric surfaces are colored green Ansys CFX.
Particle-Particle Collision Model (LPTM-PPCM)	A model in Ansys CFX that takes inter-particle collisions and their effects on the particle and gas phase into consideration.
periodic pair boundary condition	A boundary condition where the values on the first surface specified are mapped to the second surface. The mapping can be done either by a translation or a rotation (if a rotating frame of reference is used).
physical time step	The time represented in each iteration of the solution.
pick list	The list processor interprets the contents of all selected data boxes. All selected data boxes in CFX expect character strings as input. The character strings may be supplied by the graphics system when you select an entity from a viewport, or you can type or paste in the string directly. The character strings are called "pick lists."
plot	Any means of viewing the results in CFD-Post. Types of plots include vectors, streamlines, and contour plots.
point	An ordered *n*-tuple, where *n* is the number of dimensions of the space in which the point resides.
point probes	Points placed at specific locations in a computational domain where data can be analyzed.
polyline	A locator that consists of user-defined points.
postprocessor	The component used to analyze and present the results of the simulation. For Ansys CFX, the postprocessor is CFD-Post.
Power Syntax	The CFX Command Language (CCL) is the internal communication and command language of CFD-Post. It is a simple language that can be used to create objects or perform actions in the postprocessor. Power Syntax enables you to embed Perl commands into CCL to achieve powerful quantitative postprocessing. Power Syntax programming uses the Perl programming language to enable loops, logic, and custom macros (subroutines). A Line of Power Syntax is identified in a `.ccl` file by an exclamation mark (!) in the first column of a line. In between Perl lines, simple syntax lines may refer to Perl variables and lists. For details, see Power Syntax in Ansys CFX (p. 559).
preprocessor	The component used to create the input for the solver. For Ansys CFX, the preprocessor is CFX-Pre.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

pressure

In the cavitation model, pressure is the same as solver pressure, but clipped such that the absolute pressure is non-negative. It is used for postprocessing only.

prism or prismatic element

A 3D mesh element shaped like a triangular prism (with six vertices). Sometimes known as a wedge element.

pyramid element

A 3D mesh element that has five vertices.

R

reference coordinate frame

The coordinate frame in which the principal directions of X or Y or Z are taken. X is taken in the local X of that frame, and so on. If the coordinate frame is a non-rectangular coordinate frame, then the principal axes 1, 2, and 3 will be used to define the X, Y, and Z directions, respectively. The default is CFX global system (Coord 0).

For domains, boundary conditions, and initial values, the reference coordinate frame is always treated as Cartesian, irrespective of coordinate frame type.

region

An area consisting of a fluid, a solid material, or a porous material.

residuals

The change in the value of certain variables from one iteration to the next.

The discretized Navier-Stokes equations (p. 625) are solved iteratively. The residual for each equation gives a measure of how far the latest solution is from the solution in the previous iteration. A solution is considered to be converged when the residuals are below a certain value.

CFX-Solver writes the residuals to the CFX-Solver Output file (p. 625) so that they can be reviewed. Fluent allows residuals to be plotted during the solution process.

CFX-Solver Results file

A file produced by CFX-Solver that contains the full definition of the simulation as well as the values of all variables throughout the flow domain and the history of the run including residuals (p. 627). A CFX-Solver Results file can be used as input to CFD-Post or as an input file to CFX-Solver, in order to perform a restart.

Reynolds averaged Navier-Stokes (RANS) equations

Time-averaged equations of fluid motion that are primarily used with turbulent flows.

Reynolds stress

The stress added to fluid flow due to the random fluctuations in fluid momentum in turbulent flows. When the Navier-Stokes equations (p. 625) are derived for time averaged turbulent flow to take into account the effect of these fluctuations in velocity, the resulting equations have six stress terms that do not appear in the laminar flow equations. These are known as Reynolds stresses.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

Reynolds stress turbulence model
: A model that solves transport equations for the individual Reynolds stress components. It is particularly appropriate where strong flow curvature, swirl, and separation are present. Reynolds stress models in general tend to be less numerically robust than eddy viscosity models such as the k-epsilon turbulence model (p. 623).

RNG k-epsilon turbulence model
: An alternative to the standard k-epsilon turbulence model (p. 623). It is based on renormalization group analysis of the Navier-Stokes equations. The transport equations for turbulence generation and dissipation are the same as those for the standard k-epsilon model, but the model constants differ, and the constant $C\epsilon 1$ is replaced by the function $C\epsilon 1RNG$.

Rotating Frame of Reference (RFR)
: A coordinate system that rotates. Ansys CFX and Fluent can solve for fluid flow in a geometry that is rotating around an axis at a fixed angular velocity.

run
: A process that requires the specification of the CFX-Solver input file (and an initial values file, if necessary), and produces a CFX-Solver Output file and a CFX-Solver Results file (if successful).

S

Sampling Plane
: A locator that is planar and consists of equally-spaced points.

scalar variable
: A variable that has only magnitude and not direction. Examples are temperature, pressure, speed (the magnitude of the velocity vector), and any component of a vector quantity.

Scale Adaptive Simulation (SAS) model
: A shear stress transport model used primarily for unsteady CFD simulations, where steady-state simulations are not of sufficient accuracy and do not properly describe the true nature of the physical phenomena. Cases that may benefit from using the SAS-SST model include:

  - Unsteady flow behind a car or in the strong mixing behind blades and baffles inside stirred chemical reactors
  - Unsteady cavitation inside a vortex core (fuel injection system) or a fluid-structure interaction (unsteady forces on bridges, wings, and so on).

  For these problems and others, the SAS-SST model provides a more accurate solution than URANS models, where steady-state simulations are not of sufficient accuracy and do not properly describe the true nature of the physical phenomena.

Second Moment Closure models
: Models that use seven transport equations for the independent Reynolds stresses and one length (or related) scale; other models use two equations for the two main turbulent scales.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

session file (CFX)

A file that contains the records of all the actions in each interactive CFX session. It has the extension `.ses`.

Shear Stress Transport (SST)

A $k-\omega$ based SST model that accounts for the transport of the turbulent shear stress and gives highly accurate predictions of the onset and the amount of flow separation under adverse pressure gradients.

singleton (CCL object)

A singleton object that consists of an object type at the start of a line, followed by a : (colon). Subsequent lines may define parameters and child objects associated with this object. The object definition is terminated by the string END on a line by itself. The singleton object for a session file is declared like this:

```
SESSION:
     Session Filename = <filename>.cse
END
```

The difference between a singleton object and a named object is that after the data has been processed, a singleton can appear just once as the child of a parent object. However, there may be several instances of a named object of the same type defined with different names.

slice plane

A locator that is planar, and that consists of all the points that intersect the plane and the mesh edges.

solid

A material that does not flow when a force or stress is applied to it.

The general class of vector valued functions of three parametric variables.

solid sub-domain

A region of the fluid domain that is occupied by a conducting solid. Ansys CFX can model heat transfer in such a solid; this is known as CHT (Conjugate Heat Transfer) (p. 617).

solver

The component that solves the CFD problem, producing the required results.

solver pressure

The pressure calculated by solving conservative equations; it can be negative or positive. In the CFX-Solver Output file, it is called Pressure.

spanwise coordinate

A term used in Fluent documentation that is equivalent to the Ansys CFX term *constant span*.

specific heat

The ratio of the amount of heat energy supplied to a substance to its corresponding change in temperature.

specific heat capacity

The amount of heat energy required to raise the temperature of a fixed mass of a fluid by 1 K at constant pressure.

speed of sound

The velocity at which small amplitude pressure waves propagate through a fluid.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

sphere volume	A locator that consists of a collection of volume elements that are contained in or intersect a user-defined sphere.
state files	Files produced by CFD-Post that contain CCL commands. They differ from session files in that only a snapshot of the current state is saved to a file. You can also write your own state files using any text editor.
STP (Standard Temperature and Pressure)	Defined as 0°C (273.15 K) and 1 atm (1.013x105 Pa).
steady-state simulation	A simulation that is carried out to determine the flow after it has settled to a steady state. Note that, even with time constant boundary conditions, some flows do not have a steady-state solution.
stream plot	A plot that shows the streamlines of a flow. Stream plots can be shown as lines, tubes, or ribbons.
streamline	The path that a small, neutrally-buoyant particle would take through the flow domain, assuming the displayed solution to be steady-state.
subdomains	Regions within a domain where special conditions can be applied. For example, a subdomain can be used to model a region of flow resistance or a heat source.
subsonic flow	The movement of a fluid at a speed less than the speed of sound.
surface plot	A plot that colors a surface according to the values of a variable. Additionally, you can choose to display contours.
symmetry-plane boundary condition	A boundary condition where all variables except velocity are mathematically symmetric and there can be no diffusion or flow across the boundary. Velocity parallel to the boundary is also symmetric and velocity normal to the boundary is zero.

T

template fluid	One of a list of standard fluids with predefined properties that you can use 'as is', or use as a template to create a fluid with your own properties.
thermal conductivity	The property of a fluid that characterizes its ability to transfer heat by conduction. A property of a substance that indicates its ability to transfer thermal energy between adjacent portions of the substance.
thermal expansivity	The property of a fluid that describes how a fluid expands as the result of an increase in temperature. Also known as the coefficient of thermal expansion, β.
theta	The angular coordinate measured about the axis of rotation following the right-hand rule. When looking along the positive direction of the axis of rotation, theta is increasing in the clockwise direction.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

	Note that the theta coordinate in CFD-Post does not increase over 360°, even for spiral geometries that wrap to more than 360°.
topology	The shape, node, edge, and face numbering of an element.
tracers	Particles that follow a flow pathline. Used in viewing CFD results in order to visualize the mechanics of the fluid flow.
transitions	Portions of a mesh that are the result of meshing geometry with two opposing edges that have different mesh seeds. This produces an irregular mesh.
turbulence intensity	The ratio of the root-mean-square of the velocity fluctuations to the mean flow velocity. A turbulence intensity of 1% or less is generally considered low and turbulence intensities greater than 10% are considered high. Ideally, you will have a good estimate of the turbulence intensity at the inlet boundary from external, measured data. For example, if you are simulating a wind-tunnel experiment, the turbulence intensity in the free stream is usually available from the tunnel characteristics. In modern low-turbulence wind tunnels, the free-stream turbulence intensity may be as low as 0.05%. For internal flows, the turbulence intensity at the inlets is totally dependent on the upstream history of the flow. If the flow upstream is under-developed and undisturbed, you can use a low turbulence intensity. If the flow is fully developed, the turbulence intensity may be as high as a few percent.
turbulence length scale	A physical quantity related to the size of the large eddies that contain the energy in turbulent flows. In fully-developed duct flows, the turbulence length scale is restricted by the size of the duct because the turbulent eddies cannot be larger than the duct. An approximate relationship can be made between the turbulence length scale and the physical size of the duct that, while not ideal, can be applied to most situations. If the turbulence derives its characteristic length from an obstacle in the flow, such as a perforated plate, it is more appropriate to base the turbulence length scale on the characteristic length of the obstacle rather than on the duct size.
turbulence model	A model that predicts turbulent flow (p. 632). The available turbulence models in Ansys CFX include: • k-epsilon turbulence model (p. 623) • RNG k-epsilon turbulence model (p. 628) • Reynolds stress turbulence model (p. 628) • zero equation turbulence model (p. 633)

Turbulence models enable a steady-state representation of (inherently unsteady) turbulent flow to be obtained.

turbulent — A flow field that is irregular and chaotic look. In turbulent flow, a fluid particle's velocity changes dramatically at any given point in the flow field, in time, direction, and magnitude, making computational analysis of the flow more challenging.

turbulent flow — Flow that is randomly unsteady over time. A characteristic of turbulent flow is chaotic fluctuations in the local velocity.

V

variable — A quantity such as temperature or velocity for which results have been calculated in a CFD calculation.

See also Additional Variable (p. 615).

vector plot — A plot that shows the direction of the flow at points in space, using arrows. Optionally, the size of the arrows may show the magnitude of the velocity of the flow at that point. The vectors may also be colored according to the value of any variable.

verification — A check of the model for validity and correctness.

viewer area — The area of Ansys CFX that contains the **3D Viewer**, **Operating Points Viewer**, **Table Viewer**, **Chart Viewer**, **Comment Viewer**, and **Report Viewer**, which you access from tabs at the bottom of the area.

viewport (CFX) — An assigned, named, graphics window definition, stored in the CFX database, that can be used to display selected portions of a model's geometry, finite elements, and analysis results. The viewport's definition includes:

- The viewport name
- The status of the viewport (posted or unposted; current or not current)
- Viewport display attributes
- A definition of the current view
- A current group
- A list of the posted groups for display
- A graphics environment accessed from Display, Preference, and Group menus that is common to all viewports.

There are the following types of CFX viewports:

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

current viewport

The viewport currently being displayed. The following actions can be performed only on the current viewport:

- Changing the view by using the **View** menu or mouse.
- Posting titles and annotations by using the **Display** menu.

posted viewport

A viewport that has been selected for display.

target viewport

A viewport selected for a viewport modify action. Any viewport (including the current viewport) can be selected as the target viewport.

viscosity

The ratio of the tangential frictional force per unit area to the velocity gradient perpendicular to the flow direction.

viscous resistance coefficients

A term to define porous media resistance.

Volume of Fluid (VOF) method

A technique for tracking a fluid-fluid interface as it changes its topology.

W

wall

A generic term describing a stationary boundary through which flow cannot pass.

workspace area

The area of CFX-Pre and CFD-Post that contains the **Outline**, **Variables**, **Expressions**, **Calculators**, and **Turbo** workspaces, which you access from the tabs at the top of the area. Each workspace has a tree view at the top and an editor at the bottom (which is often called the details view).

See also CFD-Post Graphical Interface.

Y

y+ (YPLUS)

A non-dimensional parameter used to determine a specific distance from a wall through the boundary layer to the center of the element at a wall boundary.

Z

zero equation turbulence model

A simple model that accounts for turbulence by using an algebraic equation to calculate turbulence viscosity. This model is useful for

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.

obtaining quick, robust solutions for use as initial fields for simulations using more sophisticated turbulence models.

Release 2025 R1 - © ANSYS, Inc. All rights reserved. - Contains proprietary and confidential information of ANSYS, Inc. and its subsidiaries and affiliates.