AEROFLO User's Manual

TTC Technologies, Inc.

P.O. Box 1527

Stony Brook, New York 11790

May 2, 2006; July 1, 2007; October 3, 2007;   February 5, 2008

Preface

What is in This Manual

The Standard AEROFLO User's Guide contains the following:

- Basic overview of the AEROFLO Multi-Disciplinary Software
- Software installation guide
- Introduction to the user interface (GUI)
- Detailed procedures for performing general CFD simulations in AEROFLO
- Tutorial of simple CFD simulation
- Physical models used in AEROFLO
- Numerical models used in AEROFLO
- Parallel calculations in AEROFLO
- Keyword reference

What is in the Other Manuals

In addition to the AEROFLO User's Guide, there are other manuals available to help you use AEROFLO:

- AEROFLO CFD Sample Problems provide sample problems for AEROFLO CFD computation in AEROFLO
- AEROFLO Technique Reference provides a detailed technical reference for the physical and numerical models used in AEROFLO
- AEROFLO MHD Manual describes how to use AEROFLO to perform a magnetohydrodynamic simulation
- AEROFLO Aeroacoustics Manual describes how to use AEROFLO to simulate and analyze the acoustic generation by aerodynamic flows
- AEROFLO Combustion Manual contains the models and procedures used to simulate turbulent reacting flows
- AEROFLO Hypersonic Flow Manual describes how to simulate non-equilibrium hypersonic flows in AEROFLO
- AEROFLO Aeroelasticity Manual describes how to perform aeroelasticity calculations in AEROFLO
- AEROFLO Electromagnetics Manual describes how to solve an electromagnetic problem in AEROFLO

Technical Support

If you encounter difficulties while using AEROFLO, please first refer to the corresponding User's Manual. More resources about AEROFLO can be found on the AEROFLO website at http://www.ttctech.com/aero.asp. You can also contact TTC Technologies, Inc. to obtain further technical support.

Table of Contents

1. Introduction

AEROFLO is a high-order multi-disciplinary computational fluid dynamics (CFD) solver package developed by TTC Technologies. It is the first commercial high-order multi-disciplinary CFD simulation software that can solve flows with complex geometries at all speeds.

1.1 The Purpose of AEROFLO

With the rapid development of computer hardware techniques, many commercial CFD software packages have been developed that enable users to easily solve CFD problems for engineering applications. However, most of these commercial software packages are of low-order CFD simulation accuracy and short of the ability to handle complicated engineering problems. These requirements are becoming more and more important in the industries. People need a professional and easy-to-use CFD solution that can solve a large CFD problem, while providing high-order accurate results.

AEROFLO was developed to overcome these difficulties. AEROFLO takes the advantages of the recently-developed CFD fluid models (such as large-eddy simulation for turbulent flows) and high-order numerical schemes (such as WENO scheme) to enable high order CFD simulation to satisfy the increasing requirements for simulation accuracy in industrial applications. On the other hand, AEROFLO is a multi-disciplinary CFD package that consists of CFD solutions for many specific fluid problems, such as aeroacoustics, combustion, magnetohydrodynamics, etc. This enables AEROFLO to carry out CFD simulations for very complicated engineering projects that may contain various kinds of physics problems. AEROFLO can also carry out simulations that involve flows at different regimes, ranging from very low to high speeds. Finally, AEROFLO employs parallel, multi-block, and overset (chimera) techniques using blocks with generalized coordinates to ensure that very complicated, large, and realistic problems can be handled.

At this time, AEROFLO is focused on providing professional CFD solutions in the applications of aeronautics and astronautics. Therefore, AEROFLO only handles the simulation of gas flows.

1.2 AEROFLO Capabilities and Features

AEROFLO is a high-order multi-disciplinary computational fluid dynamics solver package. AEROFLO can perform the following kinds of simulations:

- CFD (Solution of Navier-Stokes Equations)
- Magnetohydrodynamics (MHD)
- Aeroacoustics
- Combustion
- Aeroelasticity
- Electromagnetics
- Hypersonic Flows

Figure 1.1 at the end of this section summarizes the program capabilities of AEROFLO.

AEROFLO contains various turbulent models to meet the different kinds of simulation requirements:

- Reynolds-Averaged Navier-Stokes model (RANS)
    - Spalart-Allmaras model
    - Abid's k-ε model
    - Launder-Sharma k-ε model
    - Menter's SST k-ω model
    - High Reynolds number k-ε model
- Large-Eddy Simulation model (LES)
    - Smagorinsky model
    - Dynamic model
- Direct Numerical Simulation model (DNS)
- Hybrid model

  - Detached Eddy Simulation (DES): based on Spalart-Allmaras model
  - Partially Resolved Numerical Simulation (PRNS): based on Abid's k-ε model

AEROFLO can handle flows at all speed regimes (subsonic, transonic, supersonic, and hypersonic). To perform these simulations, the package incorporates the following numerical solution schemes:

- Various high order spatial schemes
  - Páde type compact scheme (6th-order, for low-speed flow)
  - Weighted essentially non-oscillatory (WENO) scheme (5th-order, for high-speed flow)
- Standard low-order spatial schemes
  - MUSCL (Monotone Upstream-centered Schemes for Conservation Laws) scheme (2nd-order)
  - First-order Roe scheme
  - Second-order central schemes
- Several time integration schemes
  - High-order Runge-Kutta procedures (4th-order)
  - Implicit Beam-Warming schemes (2nd-order)
  - Beam-Warming with preconditioning formulation (very low-speed flow)

AEROFLO also uses the parallel computation technique to calculate very complicated, large, and realistic problems:

- Multi-block technique
- Overset (chimera) technique
- Automatic domain decomposition
- Automatic detection of interior sub-domain interface

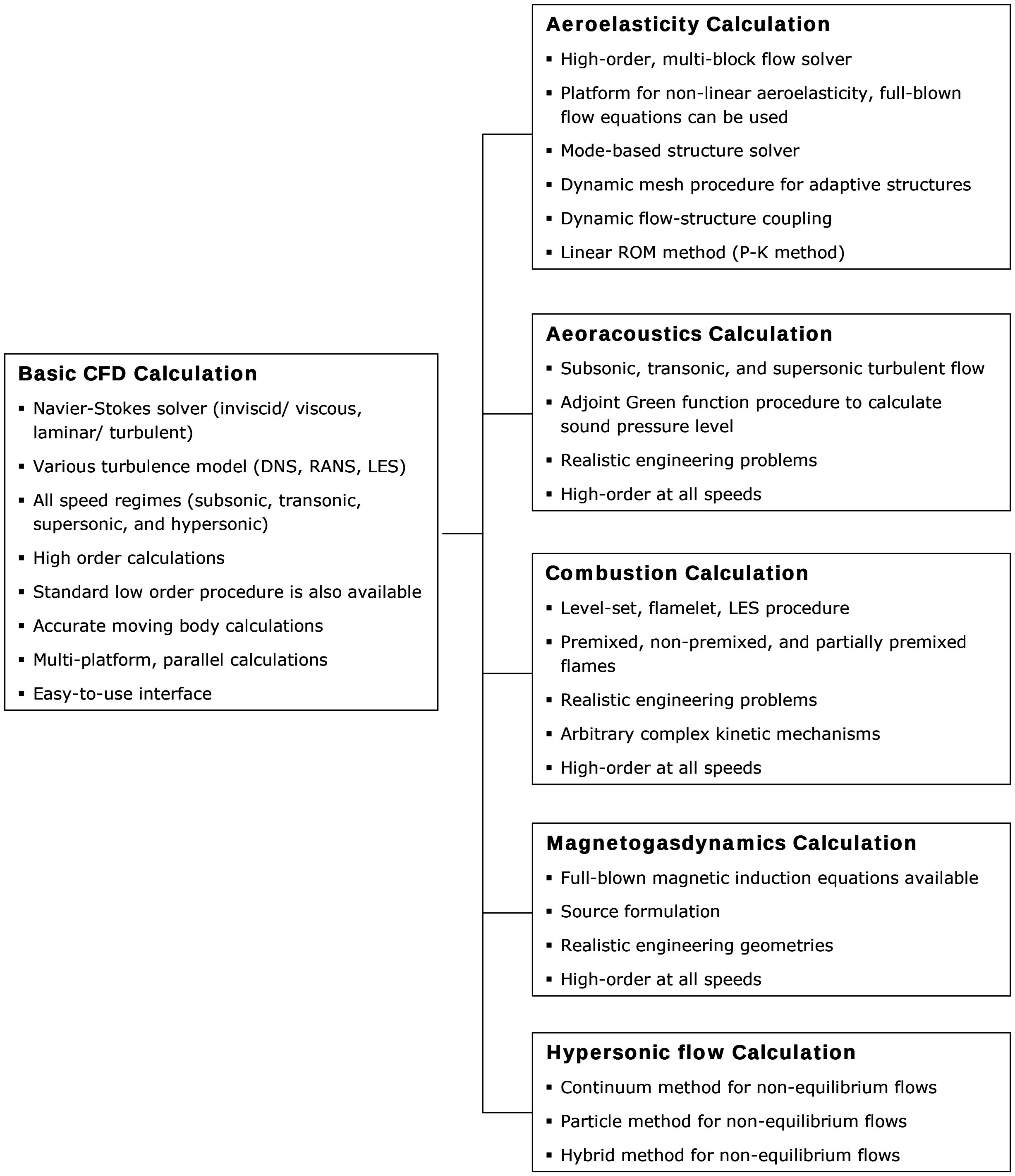

Basic CFD Calculation
▪ Navier-Stokes solver (inviscid/ viscous, laminar/ turbulent)
▪ Various turbulence model (DNS, RANS, LES)
▪ All speed regimes (subsonic, transonic, supersonic, and hypersonic)
▪ High order calculations
▪ Standard low order procedure is also available
▪ Accurate moving body calculations
▪ Multi-platform, parallel calculations
▪ Easy-to-use interface
Aeroelasticity Calculation
▪ High-order, multi-block flow solver
▪ Platform for non-linear aeroelasticity, full-blown flow equations can be used
▪ Mode-based structure solver
▪ Dynamic mesh procedure for adaptive structures
▪ Dynamic flow-structure coupling
▪ Linear ROM method (P-K method)
Aeoracoustics Calculation
▪ Subsonic, transonic, and supersonic turbulent flow
▪ Adjoint Green function procedure to calculate sound pressure level
▪ Realistic engineering problems
▪ High-order at all speeds
Combustion Calculation
▪ Level-set, flamelet, LES procedure
▪ Premixed, non-premixed, and partially premixed flames
▪ Realistic engineering problems
▪ Arbitrary complex kinetic mechanisms
▪ High-order at all speeds
Magnetogasdynamics Calculation
▪ Full-blown magnetic induction equations available
▪ Source formulation
▪ Realistic engineering geometries
▪ High-order at all speeds
Hypersonic flow Calculation
▪ Continuum method for non-equilibrium flows
▪ Particle method for non-equilibrium flows
▪ Hybrid method for non-equilibrium flows

Figure 1.1. Capabilities of AEROFLO and its Main Features.

1.3 Program Structure

Your AEROFLO package includes the following programs:

- AEROFLO GUI
- Sequential AEROFLO solver
- Parallel AEROFLO solver
- Manuals and sample problem files

AEROFLO does not include grid-generation tools. However, AEROFLO supports most common grid formats and therefore supports the grids generated by most grid-generation software. Computational mesh formats supported in AEROFLO include:

- PLOT3D format
- CGNS format (CGNS stands for CFD General Notation Scheme and is an widely accepted industry standard for CFD data)
- AEROFLO native format

AEROFLO also does not include post-processing and visualization tools. However, the output results of AEROFLO are well-organized and can be processed by most popular post-processing and visualization software, such as TECPLOT. AEROFLO supports the following output formats:

- TECPLOT format
- PLOT3D format
- CGNS format

AEROFLO also supports the native formats of TTC Technologies' other products, including INSTED and iSCRIPT.

1.4 The Structure of this Manual

This manual is divided into several parts:

- Chapter 2 describes the installation procedures for AEROFLO
- Chapter 3 outlines the general procedures for performing an AEROFLO calculation
- Chapter 4 describes the AEROFLO GUI environment
- Chapter 5 explains how to use the AEROFLO GUI to solve a simple problem
- Chapter 6 describes the input keywords in AEROFLO project files
- Chapter 7 describes the procedure to run and restart a CFD problem in AEROFLO
- Chapters 8 and 9 introduce the format of input mesh files and output result files
- Chapters 10 and 11 explain how to specify the initial and boundary conditions in AEROFLO
- Chapters 12 and 13 discuss the turbulence models and numerical schemes used in AEROFLO
- Chapter 14 explains how to implement a parallel calculation
- Chapter 15 discusses the overset technique used in AEROFLO
- Chapter 16 contains a complete reference of AEROFLO keywords

2. Installation

This chapter details the AEROFLO installation procedures for different platforms (Windows, Linux).

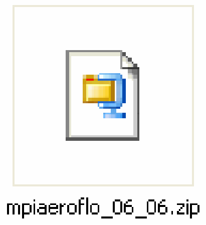
2.1 Windows

System Requirements:

The installation of the AEROFLO in Windows requires the following:

- Windows 2000/XP/2003
- Intel(R) Pentium(R) or equivalent processor
- 128MB of RAM

Installation File:



The installation file **mpiaeroflo_xx_xx.zip** can be found in the product CD or can be downloaded from the TTC Technologies website (http://www.ttctech.com). “**xx_xx**” in the filename represents the version information. For example, **mpiaeroflo_06_06.zip** represents the June 2006 version of AEROFLO.

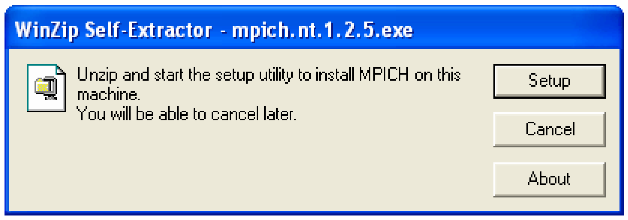
Besides the AEROFLO installation file, you may also need the MPICH for the parallel calculation. MPICH, developed by Argonne National Laboratory, is a free, portable implementation of MPI, a standard for message-passing protocol for distributed-memory applications used in parallel computing. MPICH can be downloaded from http://www.mcs.anl.gov/mpi/impich/.

Another MPI implementation package, MPIPRO, is also supported by AEROFLO. However, MPICH is assumed to be used in this manual.

<u>Installation Steps</u>:

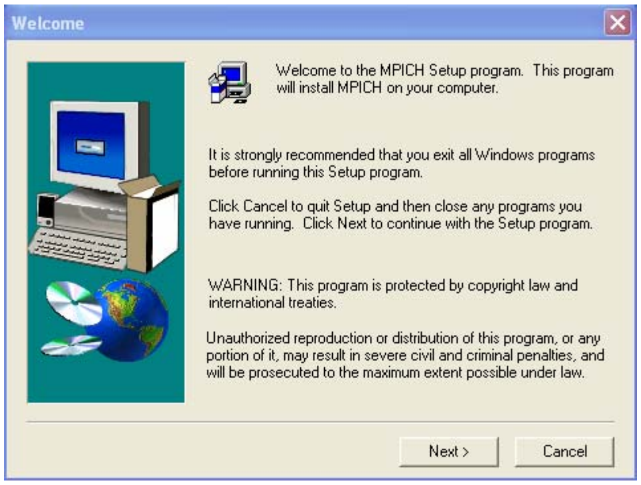
(1) To install AEROFLO in Windows, you must first install MPICH. To install MPICH, you can use the following steps (If your system already has MPICH installed, skip Step (1)):

- Double-click the installation file.


WinZip Self-Extractor - mpich.nt.1.2.5.exe

Unzip and start the setup utility to install MPICH on this machine.
You will be able to cancel later.

- Click "Setup."



Welcome to the MPICH Setup program. This program will install MPICH on your computer.

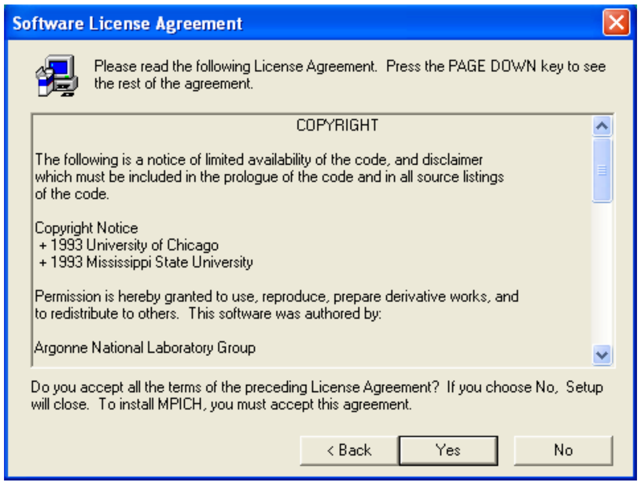
It is strongly recommended that you exit all Windows programs before running this Setup program.

Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

- Click "Next."



Please read the following License Agreement. Press the PAGE DOWN key to see the rest of the agreement.

COPYRIGHT

The following is a notice of limited availability of the code, and disclaimer which must be included in the prologue of the code and in all source listings of the code.

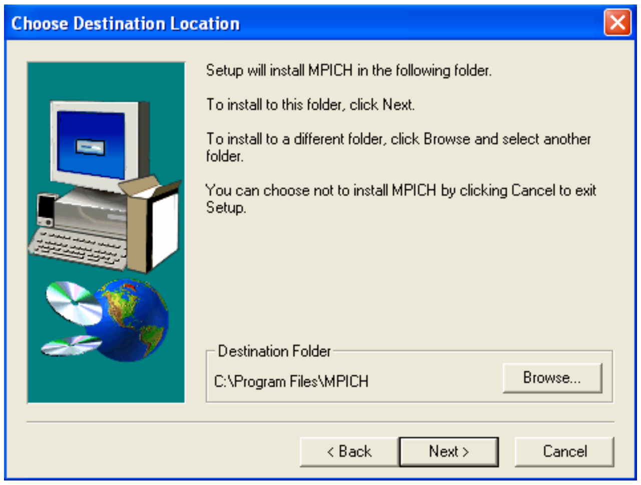
Copyright Notice
+ 1993 University of Chicago
+ 1993 Mississippi State University

Permission is hereby granted to use, reproduce, prepare derivative works, and to redistribute to others. This software was authored by:

Argonne National Laboratory Group

Do you accept all the terms of the preceding License Agreement? If you choose No, Setup will close. To install MPICH, you must accept this agreement.

- Select where to install MPICH and click "Next."



Choose Destination Location

Setup will install MPICH in the following folder.

To install to this folder, click Next.

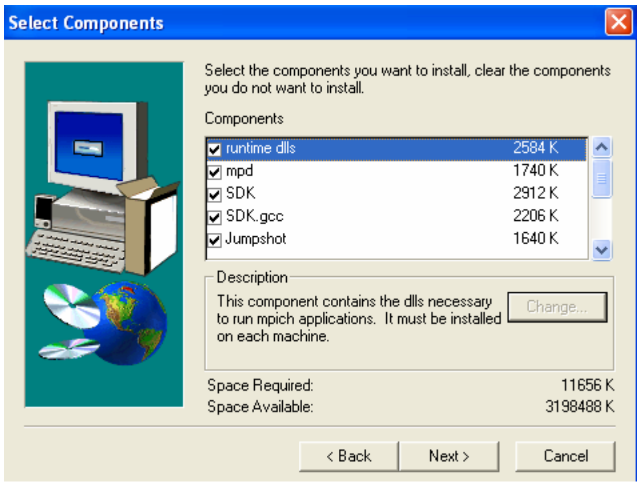
To install to a different folder, click Browse and select another folder.

You can choose not to install MPICH by clicking Cancel to exit Setup.

Destination Folder

C:\Program Files\MPICH

- Click "Next."



Select Components

Select the components you want to install, clear the components you do not want to install.

Components

✓ runtime dlls	2584 K
✓ mpd	1740 K
✓ SDK	2912 K
✓ SDK.gcc	2206 K
✓ Jumpshot	1640 K

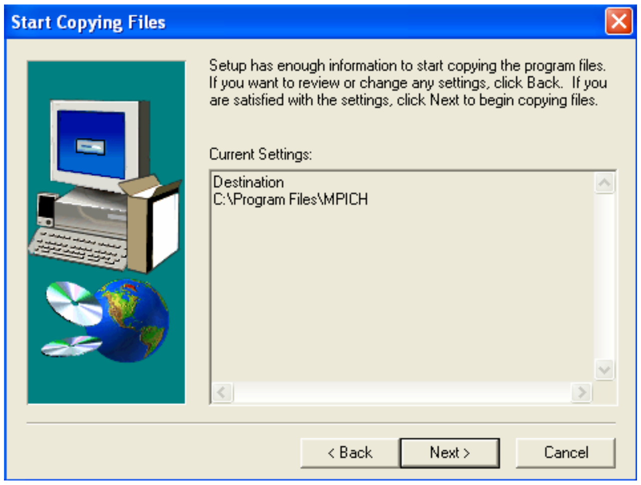
Description

This component contains the dlls necessary to run mpich applications. It must be installed on each machine.

Space Required: 11656 K

Space Available: 3198488 K

- Click "Next."

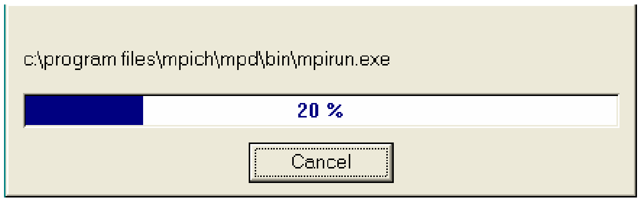

Start Copying Files

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

Destination
C:\Program Files\MPICH

- Click "Next."


c:\program files\mpich\mpd\bin\mpirun.exe

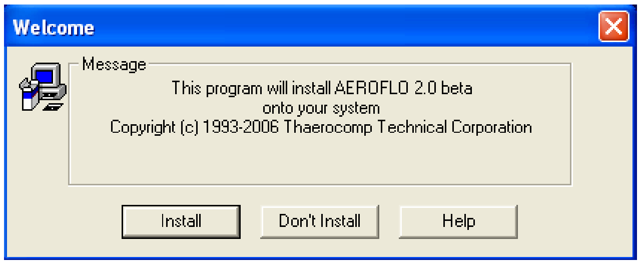
20 %

More information about the installation of MPICH can be found on the MPICH website.

(2) After installing MPICH, you may now install AEROFLO:

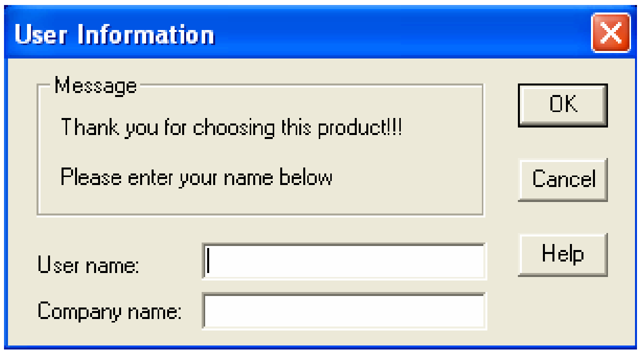
- Unzip the installation file to any temporary folder. You may need unzipping software, such as WINZIP or WINRAR, for this step.

- In the folder where you unzipped the installation file, go to the */run* subfolder and double-click "*setup.exe*." The installation interface will be shown.



This program will install AEROFLO 2.0 beta
onto your system
Copyright (c) 1993-2006 Thaerocomp Technical Corporation

- Click "Install."



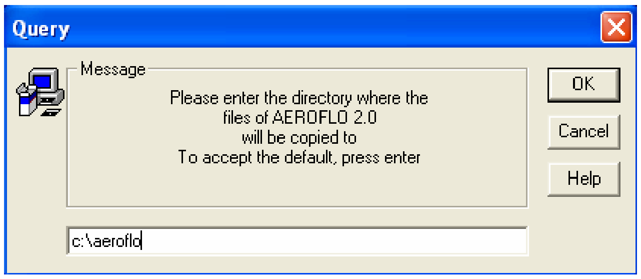
Thank you for choosing this product!!!

Please enter your name below

User name:

Company name:

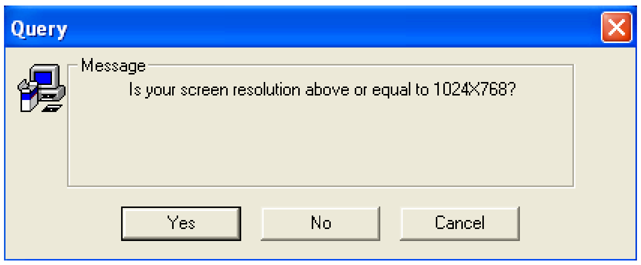
- Enter the "User name" and "Company name," then click "OK."



Please enter the directory where the
files of AEROFLO 2.0
will be copied to
To accept the default, press enter

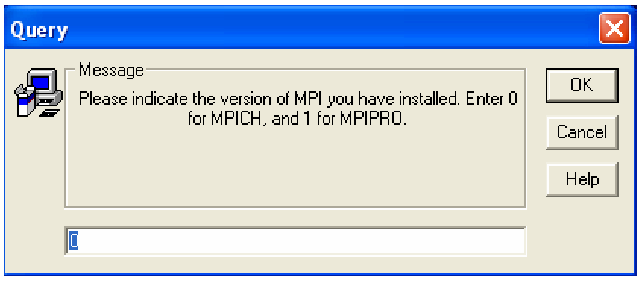
c:\aeroflo

- Select the installation path or leave it as the default value (*C:\aeroflo*). Click "OK."



Is your screen resolution above or equal to 1024X768?

- The AEROFLO GUI requires that the screen resolution be 1024x768 or better for optimum performance. Adjust the resolution if it is not, and then click “Yes.”

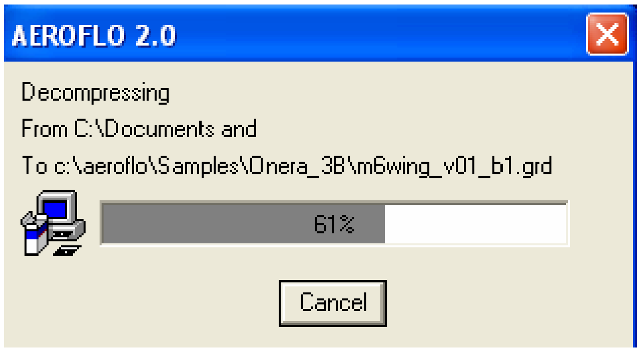


Message

Please indicate the version of MPI you have installed. Enter 0 for MPICH, and 1 for MPIPRO.

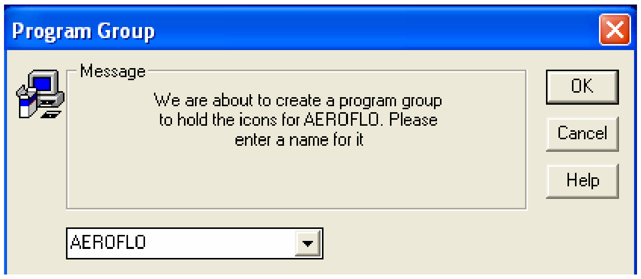
0

- Enter “0” if MPICH is installed or “1” if MPIPRO is installed. Leave the default value if MPICH is installed. Click “OK.”



Decompressing
From C:\Documents and
To c:\aeroflo\Samples\Onera_3B\m6wing_v01_b1.grd

- The installation process begins, which may take several minutes.

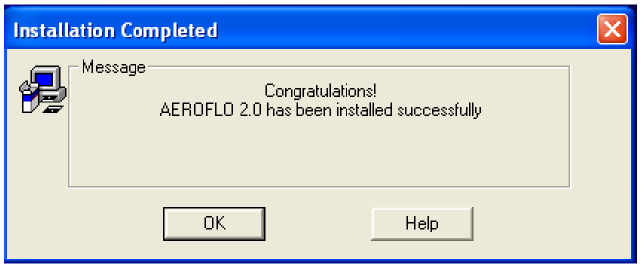


Message

We are about to create a program group to hold the icons for AEROFLO. Please enter a name for it

AEROFLO

- Create a program group or accept the default. Click “OK.”

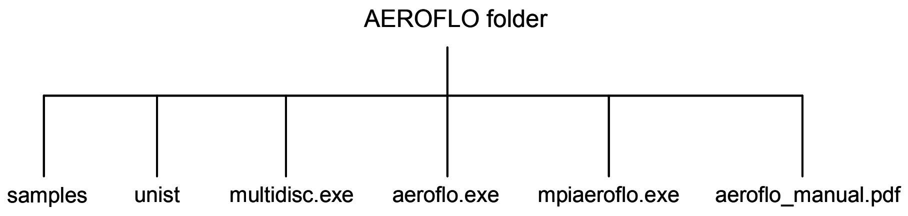


Congratulations!
AEROFLO 2.0 has been installed successfully

- Installation is successful. Click "OK."

<u>Installed Files</u>:

Once AEROFLO is installed, two folders (Samples, unist), three executable files (multidisc.exe, aeroflo.exe, mpiaeroflo.exe), and one PDF file (aeroflo_manual.pdf) will be installed in the installation directory (default is C:\aeroflo). These files or folders are:



samples –	contains the AEROFLO sample problems
unist –	contains the AEROFLO "uninstall" information
multidisc.exe –	AEROFLO GUI launch file
aeroflo.exe –	sequential AEROFLO solver
mpiaeroflo.exe –	parallel AEROFLO solver
aeroflo_manual.pdf –	AEROFLO User's Manual

2.2 Linux/Unix

AEROFLO installation on Linux is via the old-fashioned command line procedure. Please contact TTC for details.

3. Basic Procedures for CFD Analysis Using AEROFLO

In this chapter, the basic procedures for a CFD analysis in AEROFLO are introduced. Users are always advised to follow these procedures when using AEROFLO.

3.1 Detailed Procedures

Once you have determined the CFD problem you want to solve, you can follow the basic procedures shown below:

Step 1. Generate computational grids

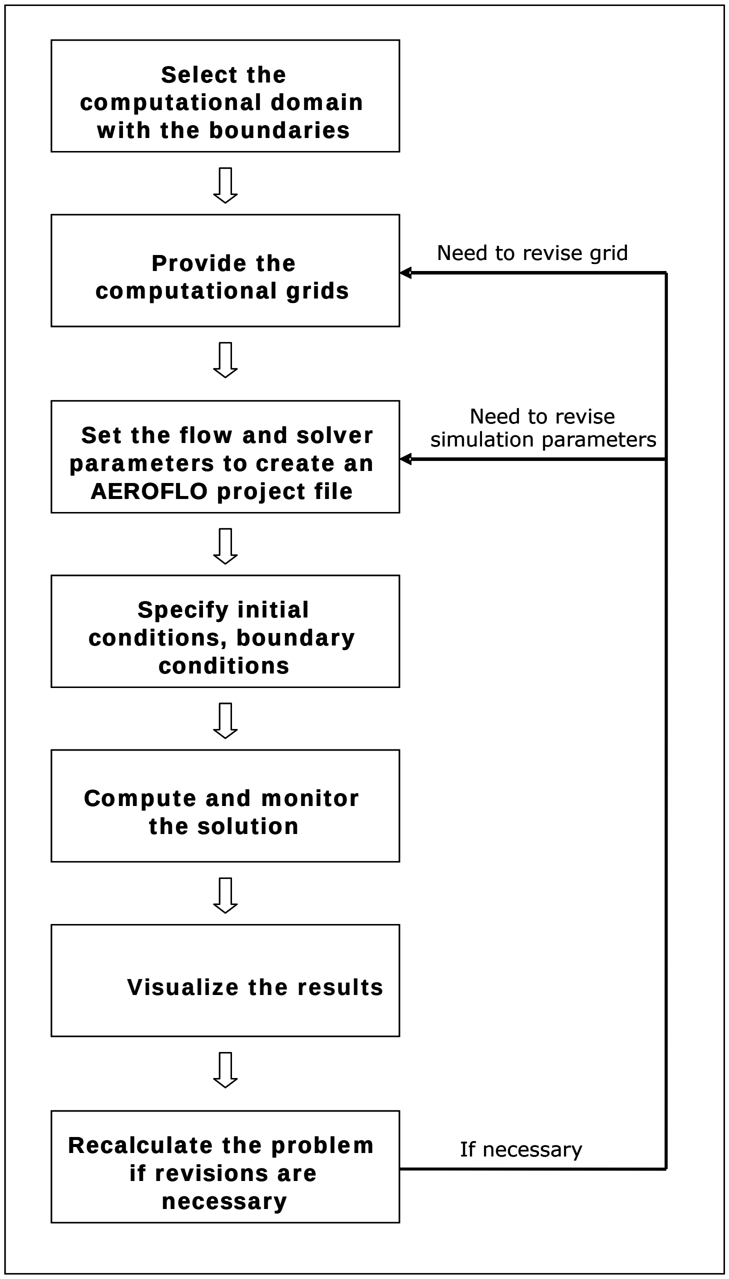
Step 2. Set up an AEROFLO project file

Step 3. Run the AEROFLO project

Step 4. Plot AEROFLO results

Step 5. Consider revisions to the grids and models, if necessary

Figure 3.1 is a flow chart of these procedural steps to solving a CFD problem in AEROFLO. Each procedure is also discussed in the following section of this chapter.



Select the computational domain with the boundaries
Provide the computational grids
Need to revise grid
Set the flow and solver parameters to create an AEROFLO project file
Need to revise simulation parameters
Specify initial conditions, boundary conditions
Compute and monitor the solution
Visualize the results
Recalculate the problem if revisions are necessary
If necessary

Figure 3.1. Basic Procedures for CFD Analysis in AEROFLO.

3.2 Creating Grids

AEROFLO currently only supports **structured grids**. The input grid formats may be PLOT3D, CGNS, or AEROFLO format. Note that AEROFLO does not provide the grid-generation tools. However, the user has flexibility in choosing grid-generation tools, since the grid format in AEROFLO is supported by most grid-generation and CAD/CAE software. The following are some commercial software that are supported by AEROFLO:

- GRIDGEN
- ICEM CFD
- VGRID

Detailed information about the grid format in AEROFLO will be discussed in Chapter 8.

3.3 Setting Up an AEROFLO Project

The input parameters for AEROFLO are stored in a single project file. To set up an AEROFLO project, the user must provide the following information:

- Simulation type (e.g., general CFD, MHD, combustion, aeroacoustics, etc.)
- Grid information
- Physical models (e.g., turbulence model, combustion model, etc.)
- Global flow conditions (e.g., Mach number, Reynolds number, etc.)
- Boundary conditions
- Initial conditions
- Numerical schemes (spatial differentiation and time integration schemes)
- Output controls

A project file can be created using AEROFLO's graphical user interface (GUI) or can be directly edited as a text file. Chapters 5 and 6 will introduce how to create project files using the AEROFLO GUI and how to directly edit them.

3.4 Running an AEROFLO Project

After the grid is created and the project file is set up, the project can immediately be run in the AEROFLO GUI or in a command line. The governing equations are solved iteratively. A number of iteration steps are required to obtain convergent results. The user can monitor the iteration residues to see if the calculation is converged. A detailed procedure of how to run and restart an AEROFLO calculation will be given in Chapter 7.

3.5 Plotting Results

AEROFLO outputs the simulation results at a frequency determined by the user. The format of the output file may be TECPLOT, PLOT3D, or CGNS. Note that AEROFLO does not provide post-processing and visualization tools. The user has flexibility in choosing any CFD visualization software to plot the results. The following are some commercial software that can process AEROFLO output files:

- TECPLOT
- FIELDVIEW

Chapter 9 will introduce the details of outputting files.

3.6 Revision and Recalculation

After you get the calculation solution, you may find that the solutions do not achieve your modeling goals. This may be because:

- The computational domain is too small or the grids are not fine enough
- The physical models you chose are not appropriate (e.g., turbulence models)
- The boundary and initial conditions are not correctly set
- The numerical schemes are not appropriate

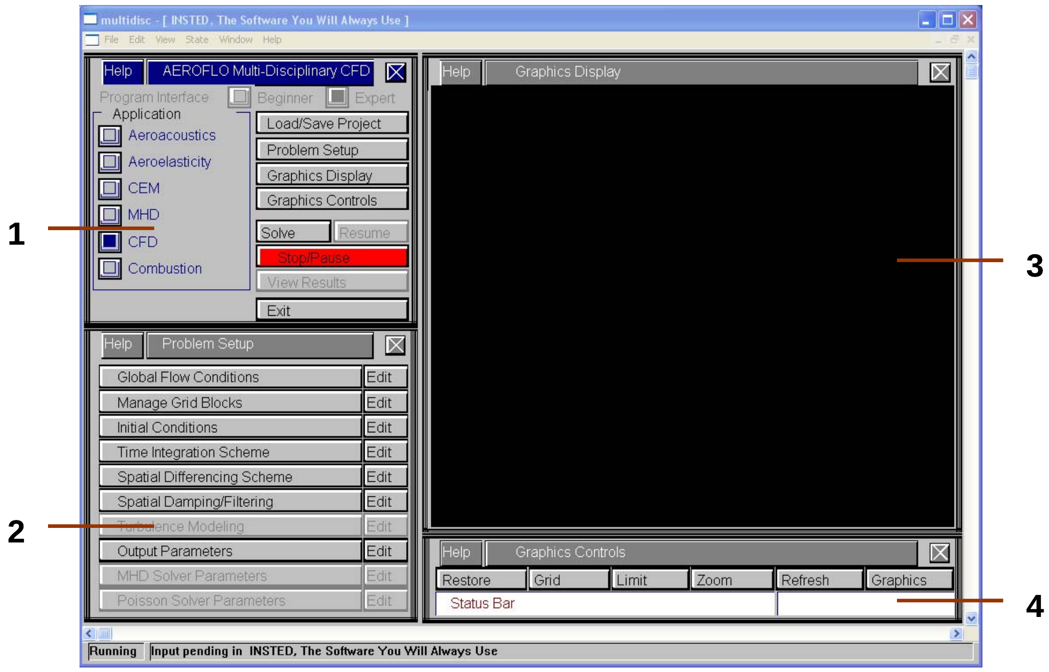
If any of these is true, you may need to change the grid file or revise the input parameters in the project file and recalculate the problem.

4. AEROFLO's Graphical User Interface (GUI)

The User Interface of AEROFLO provides an easy and visualizable way to set up the input parameters for an AEROFLO project. This chapter introduces the basic components of AEROFLO's Graphical User Interface (GUI). To start the AEROFLO GUI program, you can double-click the *multidisc.exe* in AEROFLO's installation directory or find the "*Programs > AEROFLO > AEROFLOGUI*" in the Start menu.

4.1 GUI Components

AEROFLO's User Interface is made up of four main components: Main Program Dialog Box, Program Setup Dialog Box, Graphics Display Dialog Box, and Graphics Controls Dialog Box. Users interact with these components to set up an AEROFLO project. Figure 4.1 is a complete screen shot of the AEROFLO GUI. The details of each component will be described in subsequent sections.



Running | Input pending in INSTED, The Software You Will Always Use

Figure 4.1. AEROFLO GUI Components:

1 – Main Program Dialog Box 2 – Program Setup Dialog Box

3 – Graphics Display Dialog Box 4 – Graphics Controls Dialog Box

4.2 Main Program Dialog Box

The Main Program Dialog Box contains the most basic controls for AEROFLO simulations, such as the selection of the simulation type, the project file manipulation, controls to the other dialog boxes, and the execution of the problem. Figure 4.2 is a screen shot of the Main Program Dialog Box.

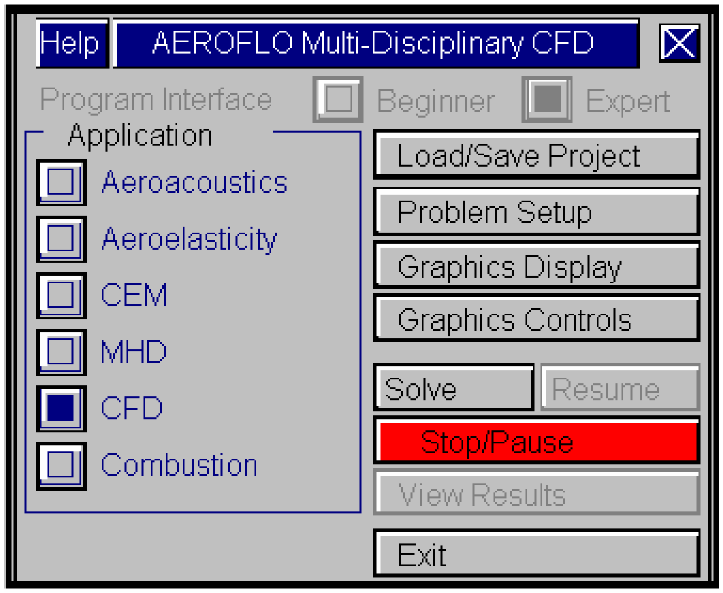


Figure 4.2. Main Program Dialog Box.

Program Interface

The program interface radio buttons allow the user to select the level of user interaction with the program. The two available options are **Beginner** and **Expert**. In **Beginner** mode, the program internally sets defaults for a large portion of the required inputs. This mode is recommended for a new user. For instance, with the compact scheme, you do not need to select a scheme, because the code will adopt a default high-order compact procedure. On the other hand, the **Expert** user would specify detailed parameters of the simulation.

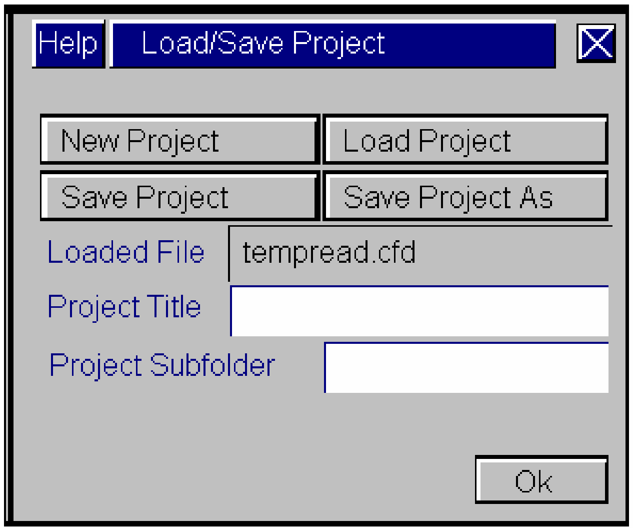
Application

The **Application** group contains the different types of simulations that can be performed with AEROFLO. Options include Aeroacoustics, Aeroelasticity, CEM, MHD, basic CFD or Navier-Stokes, and combustion calculations. <u>Depending on the license that you have purchased, some options may be</u>

unavailable. The selected application determines the required inputs, thus the dialog boxes that are presented to you for data input.

Load/Save Project button

This button opens the 'Load/Save Project' dialog box, where you can perform administrative functions, including loading an existing project, saving the project you are working on, and initializing all data fields to start a new session. Figure 4.3 is a screen shot of the Load/Save Project dialog box.



Loaded File tempread.cfd
Project Title
Project Subfolder

Figure 4.3. Load/Save Project Dialog Box.

In the Load/Save Project dialog box, the **New Project** button causes all input to be cleared and default values restored. The **Load Project** button opens up the file navigation dialog box to enable you load a previously saved project file. The **Save Project** button causes all project variables and parameters to be saved to the currently-opened project file. The **Save Project As** button opens up the file navigation dialog box to enable you save a project to a different file and folder. The **Loaded File** text box gives the name of the currently-open project file. The **Project Title** input box allows you to specify a project title, for administrative purposes. The **Project Subfolder** input box allows you to specify a project subfolder of the current folder where the grid and initial conditions file may be located. This is good for project organization.

<u>Problem Setup button</u>

This button opens the 'Problem Setup' dialog box, where you specify the inputs that are required for a simulation. The Problem Setup Dialog Box will be described in Section 4.3.

<u>Graphics Display button</u>

This button opens the 'Graphics Display' dialog box, which contains the visual display and output log. The Graphics Display Dialog Box will be described in Section 4.4.

<u>Graphics Control button</u>

This button opens the 'Graphics Control' dialog box, which contains buttons to control the Graphics Display dialog box. The Graphics Control Dialog Box will be described in Section 4.5.

<u>Solve button</u>

This button initiates the start of the simulation. This assumes that all required input data for the task have been specified. If this is not the case, the computation may not be successful and the missing input will be indicated via a (Warning) Message Box.

<u>Stop/Pause button</u>

This button stops a simulation that has been initiated with the **Solve** button. A simulation that is stopped or paused may be resumed with the **Resume** button.

<u>Resume button</u>

This button resumes a simulation that has already started and was suspended via the **Stop** button.

<u>View Results button</u>

This button opens up a dialog box where the output of a simulation can be viewed.

<u>Exit button</u>

Press this button to close the program and end the current session.

4.3 Program Setup Dialog Box

The Problem Setup dialog box opens up other dialog boxes from which input is provided. Figure 4.4 is a screen shot of the Problem Setup dialog box. For a particular project, you are only presented the dialog boxes that you will need for that project. For instance, the Turbulence Modeling dialog box will not be accessible if you asked for an inviscid simulation for all blocks in the project. Dialog boxes that have been opened and for which inputs have been provided are disabled ("grayed") to indicate that they have been processed. This helps your book-keeping. To return to a grayed dialog box, you simply click the Edit button on the right of the button that you wish you access.

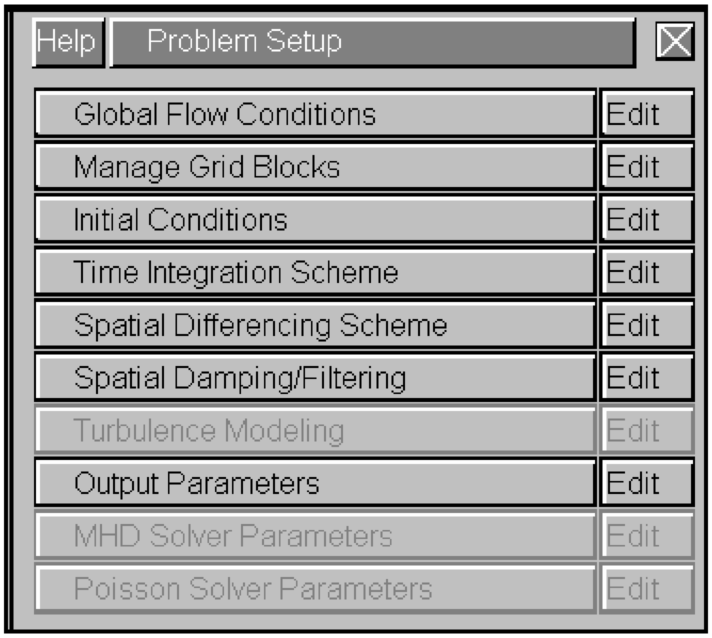
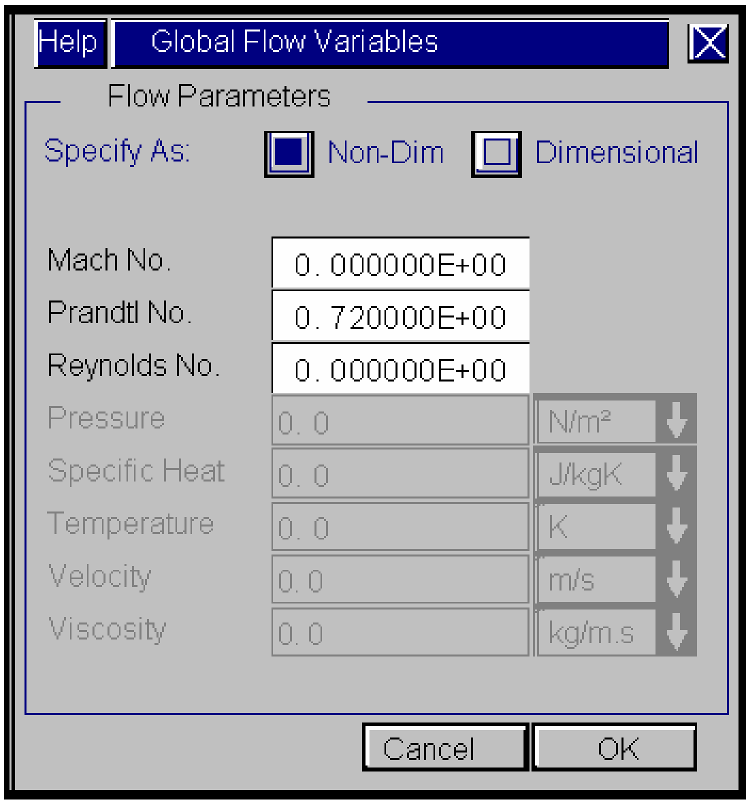


Figure 4.4. Problem Setup Dialog Box.

<u>Global Flow Conditions button</u>

This button opens up the ‘Global Flow Variables’ dialog box, which is used to specify the global flow parameters. Figure 4.5 is a screen shot of the Global Flow Variables dialog box. In AEROFLO, variables can be specified in either dimensional or non-dimensional form.

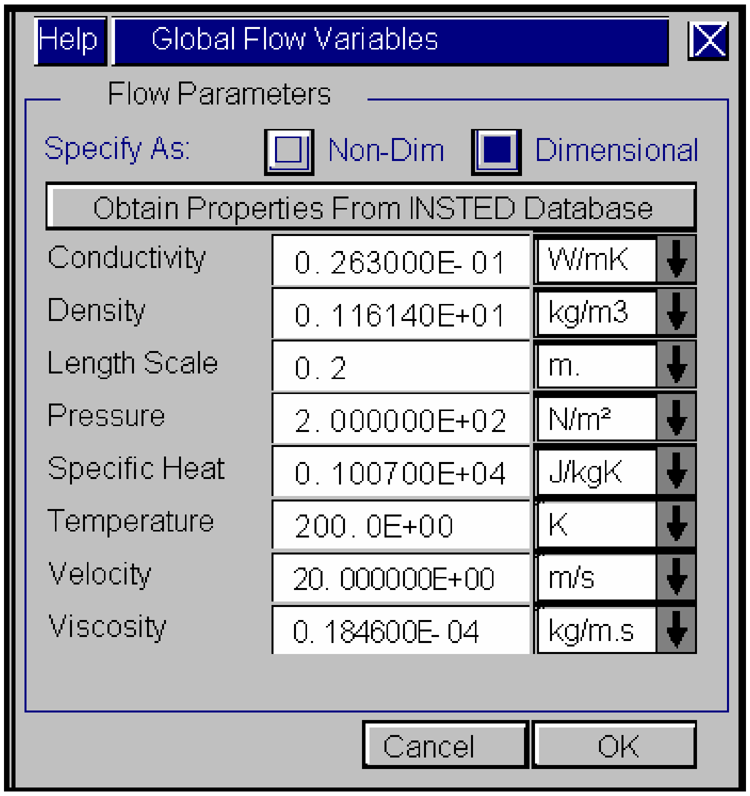


Specify As: [✓ Non-Dim] Dimensional

Mach No.	0. 000000E+00	
Prandtl No.	0. 720000E+00	
Reynolds No.	0. 000000E+00	
Pressure	0. 0	N/m²
Specific Heat	0. 0	J/kgK
Temperature	0. 0	K
Velocity	0. 0	m/s
Viscosity	0. 0	kg/m.s

Figure 4.5. Global Flow Variables Dialog Box.

Nondimensional global flow parameters include Mach number, Prandtl number, and Reynolds number. The dimensional global flow parameters include conductivity, density, length scale, pressure, specific heat, temperature, velocity, and viscosity. A screen shot of the inputs of dimensional flow parameters are shown in Figure 4.6. Note that the **Obtain Properties from INSTED Database** button opens up TTC's INSTED database to import the thermophysical properties of hundreds of liquids and gases over a range of temperatures.



Help | Global Flow Variables

Flow Parameters

Specify As: Non-Dim [✓ Dimensional]

Obtain Properties From INSTED Database

Conductivity	0. 263000E- 01	W/mK
Density	0. 116140E+01	kg/m3
Length Scale	0. 2	m.
Pressure	2. 000000E+02	N/m²
Specific Heat	0. 100700E+04	J/kgK
Temperature	200. 0E+00	K
Velocity	20. 000000E+00	m/s
Viscosity	0. 184600E- 04	kg/m.s

Cancel | OK

Figure 4.6. The Input of Dimensional Global Flow Parameters.

Manage Grid Blocks button

This button opens up the ‘Manage Grid Blocks’ dialog box, which is used to specify the structured grid blocks that are contained in a project and the boundary conditions for the blocks. Figure 4.7 is a screen shot of the Manage Grid Blocks dialog box. The **Add New Block** button opens up a new dialog box that allows you import a grid block to the project. The **Modify Selected Block** and **Delete Selected Block** buttons allow you to modify and delete a grid block that is selected in the **Selected Block** drop-down menu. The **Add Boundary Conditions** button opens up a new dialog box that allows you add new boundary conditions for a block at a selected block boundary that is selected in the **BC on Blocks** drop-down menu. The **Modify Selected BC** and **Delete Selected BC** buttons allow you to modify and delete a boundary condition that is selected in the **Selected BC** dialog box. The **Display Block** button allows you to turn the display of the selected dialog box on or off. The **Overset Setup** button opens the Overset Setup dialog box and allows you to set the overset boundary condition.

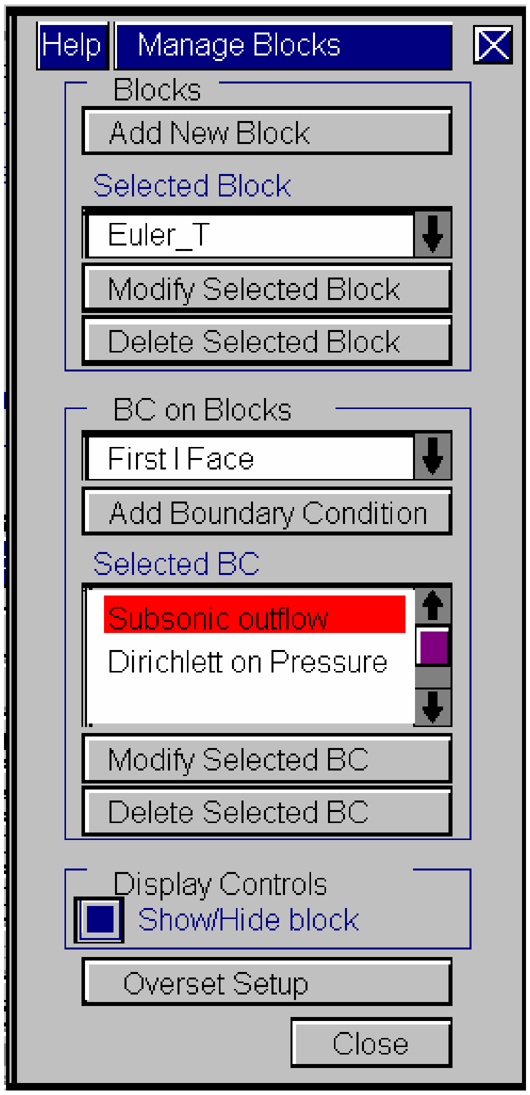
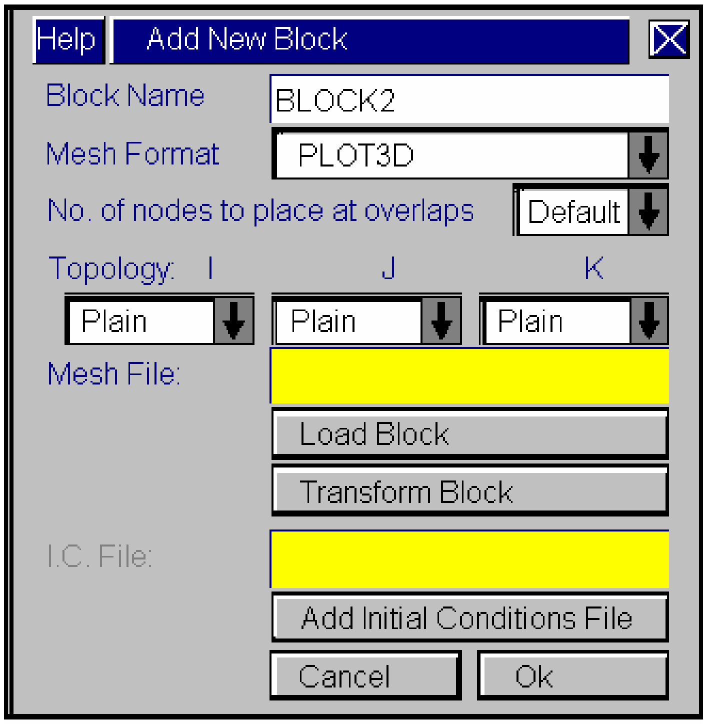


Figure 4.7. Manage Blocks Dialog Box.

Figure 4.8 is a screen shot of the Add/Modify Block dialog box. This dialog box opens when you click the **Add New Block** or **Modify Selected Block** buttons. You can define or modify the block name in the **Block Name** input box. The **Mesh Format** drop-down menu allows the format of the mesh to be selected (AEROFLO supports the following formats: PLOT3D, CGNS, and AEROFLO formats). The **No. of nodes to place at overlaps** drop-down menu controls how many fringe nodes are used at block boundaries for a multi-block calculation. The **Topology** drop-down menu options allow you to indicate the topology of your grid block in each of the grid directions. Options include Plane, O-grid, periodic, and 2D grids. The **Load Block** button opens up a file navigation dialog box to enable you to load a grid block file. The **Mesh File** text box shows the name of the mesh file that is currently loaded. The **Transform**

Block button opens up the Transformation dialog box, which enables you to specify an initial transformation of the block. Transformation operations include scaling, translation, and rotation. AEROFLO allows the project to import the initial conditions from an initial-condition file for each of the block. The **Add Initial Conditions File** button opens up a file navigation dialog box to enable you to load an initial condition file. The **I.C. File** text box shows the name of the initial condition file that is currently loaded.



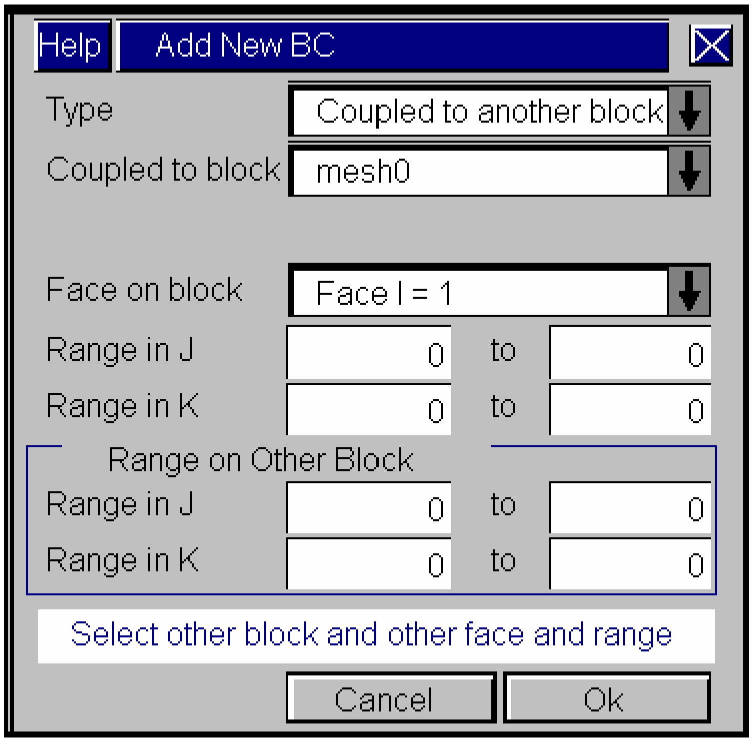
Block Name: BLOCK2
Mesh Format: PLOT3D
No. of nodes to place at overlaps: Default
Topology: I: Plain, J: Plain, K: Plain
Mesh File:
I.C. File:

Figure 4.8. Add/Modify Block Dialog Box.

Figure 4.9 is a screen shot of the Add New BC/Modify BC dialog box. This dialog box opens when you click the **Add Boundary Condition** or **Modify Selected BC** buttons. This dialog box is used to add or modify selected boundary conditions of a grid block. The first selection in this dialog box is the boundary conditions type. All other inputs in this dialog box depend on the type of boundary condition selected. The **Type** drop-down menu is used to select the type of boundary condition. AEROFLO supports the following types:

- Block coupling
- C-Grid
- Dirichlett
- Flux, specified value
- Neumann
- Overset boundary
- Periodicity coupling
- Slip wall
- Solid wall
- Specified equation
- Subsonic freestream
- Subsonic inflow
- Subsonic outflow
- Symmetry

Detailed information about these boundary conditions will be given in Chapter 11.



Type: Coupled to another block
Coupled to block: mesh0
Face on block: Face I = 1
Range in J: 0 to 0
Range in K: 0 to 0

Range on Other Block
Range in J: 0 to 0
Range in K: 0 to 0

Select other block and other face and range

Figure 4.9. Add/Modify BC Dialog Box.

Initial Conditions button

This button opens the Initial Condition dialog box, which is used to specify the global initial condition. This initial condition is applied to all the nodes of the indicated block or blocks. Figure 4.10 is a screen shot of the Initial Condition dialog box. The **Block** drop-down menu is used to select the block(s) to which an initial condition is being applied. The **Add New Initial Conditions** and **Modify Selected Initial Conditions** buttons open up the **New Initial Condition/Modify Initial Condition Dialog Box** that allows you to specify or modify a global initial condition on a variable. The **Delete Selected Initial Conditions** button deletes a selected initial condition.

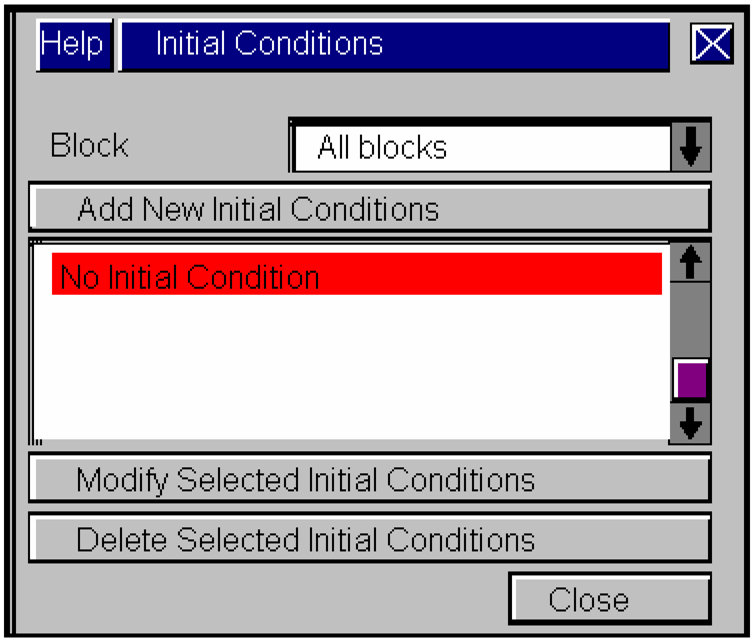
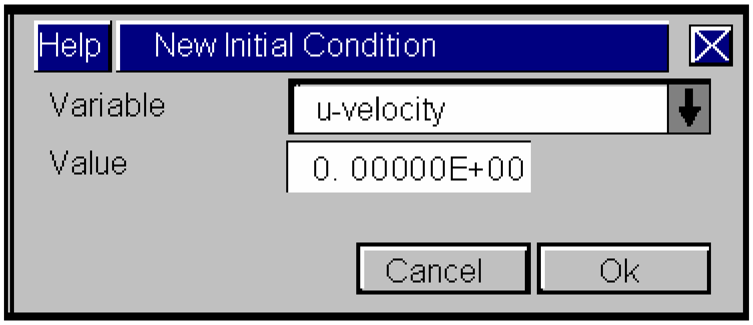

Help
Initial Conditions
Block
All blocks
Add New Initial Conditions
No Initial Condition
Modify Selected Initial Conditions
Delete Selected Initial Conditions
Close

Figure 4.10. Initial Conditions Dialog Box.

Figure 4.11 is a screen shot of the New Initial Condition/Modify Initial Condition dialog box. The **Variable** drop-down menu is used to select the variable to which an initial condition applies. The **Value** input box is used to specify the initial condition value.



Variable u-velocity

Value 0. 00000E+00

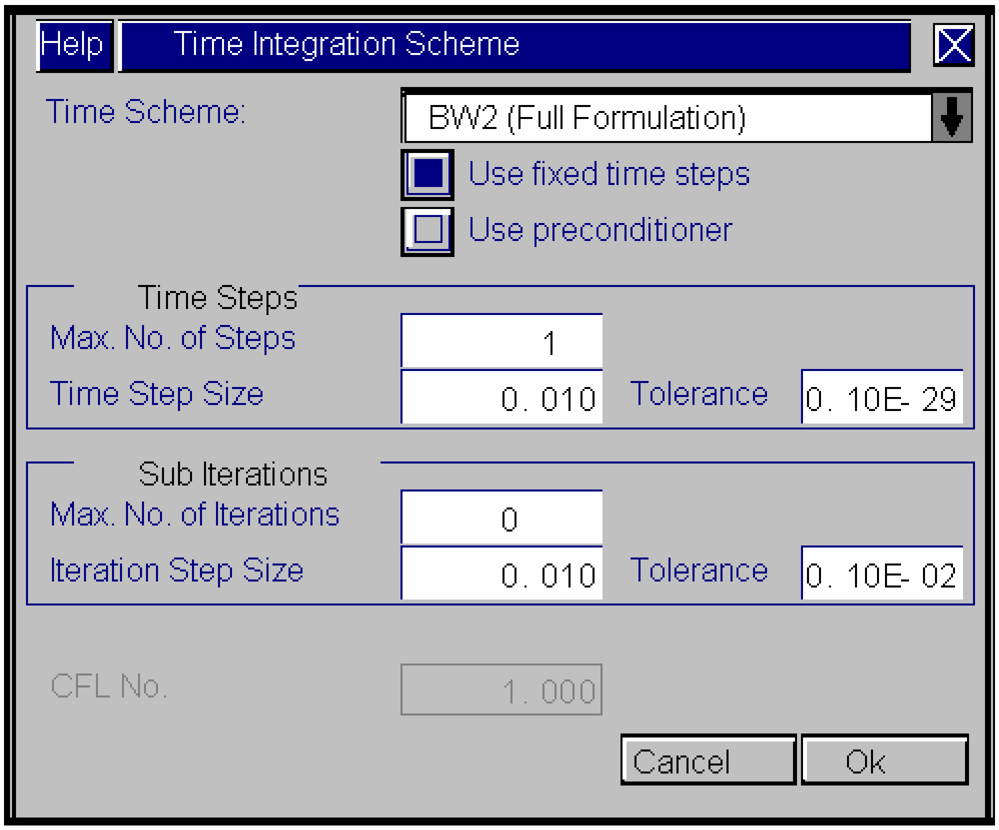
Figure 4.11. New Initial Condition/Modify Initial Condition Dialog Box.

Time Integration Scheme button

This button opens up a Time Integration Scheme dialog box, which allows you to specify the time integration scheme. Figure 4.12 is a screen shot of this dialog box. The **Time Scheme** drop-down menu allows the user to specify the time integration scheme. The currently-supported options in AEROFLO include:

- Beam Warming scheme
- Beam Warming scheme with approximate diagonalization scheme
- Runge-Kutta scheme

The **Use fixed time steps** button causes fixed time steps to be used for time advancement. When it is not selected, variable time steps are used. The value of the time steps is calculated based on the flow conditions and the CFL number. The **Use preconditioner** button causes preconditioner formulation to be used with the time advancement scheme. This allows stable calculations of low-speed flow ($Ma << 0.1$) with larger time steps than when this option is not selected. The **Max. No. of Time Steps** input box is used to specify the maximum number of time steps for the simulation. The **Time Step Size** input box is used to specify the time step size. When variable time steps are used, this input is used only for the initial time step size. The **Tolerance** input box is used to specify the value of the residual at which the simulations are considered to have converged. Sub-iterations may be used for the Beam Warming scheme. This allows bigger time step sizes to be used. The **Max. No. of Sub Iterations** input box is used to specify the number of sub-iterations to use per time step. The **Sub-Iteration Step Size** input box is used to specify the sub-iteration step size. The **Sub-Iteration Tolerance** input box is used to specify the value of the residual at which the calculations are considered to have converged for the current time step. The **CFL No.** input box is used to specify the CFL number to use with the variable time step option for calculating the time step.


Help | Time Integration Scheme

Time Scheme: BW2 (Full Formulation)

[✓] Use fixed time steps

[ ] Use preconditioner

Time Steps

Max. No. of Steps: 1

Time Step Size: 0. 010 Tolerance: 0. 10E- 29

Sub Iterations

Max. No. of Iterations: 0

Iteration Step Size: 0. 010 Tolerance: 0. 10E- 02

CFL No. 1. 000

Cancel | Ok

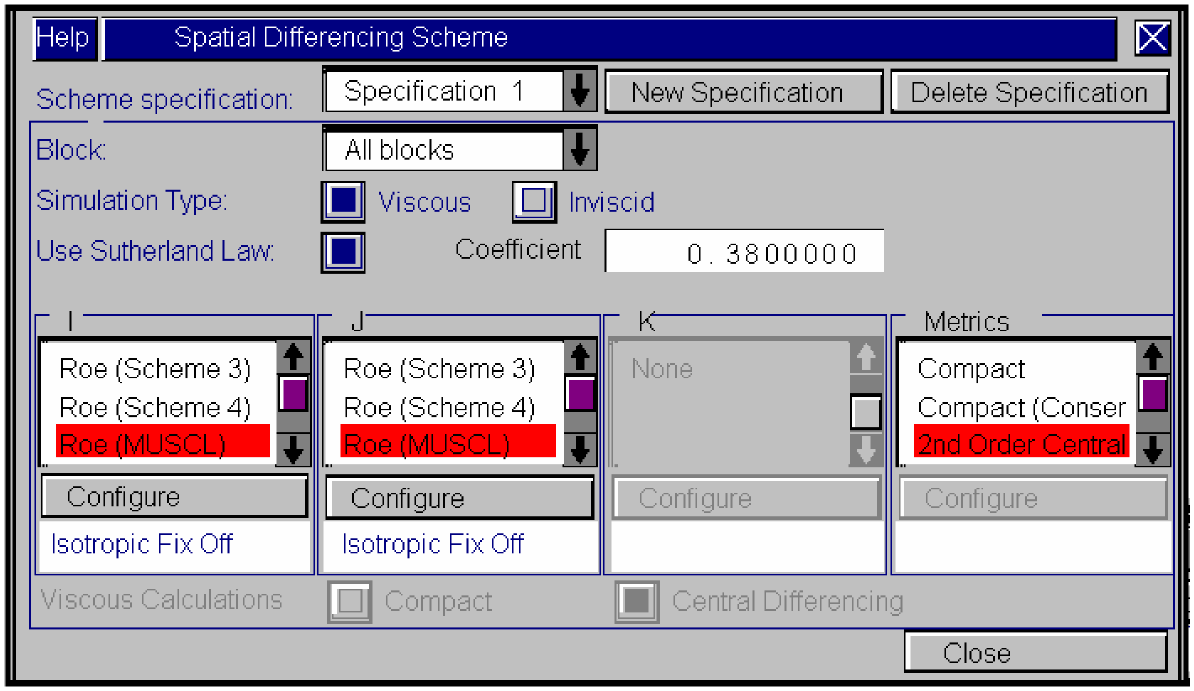
Figure 4.12. Time Integration Scheme Dialog Box.

Spatial Differencing Scheme button

This button opens a Spatial Differencing Scheme dialog box, which is used to specify the spatial differencing scheme. Figure 4.13 is a screen shot of this dialog box. The **Scheme Specification** dialog box provides the environment for specifying the spatial differencing scheme. AEROFLO allows the specification of various spatial differencing schemes in different blocks and in different directions. The spatial scheme options include:

- Compact scheme
- Roe schemes
- MUSCL scheme
- WENO schemes

Once a spatial scheme has been selected, parameters can be added via the subsequent dialog boxes that are activated. The **New Specification** button creates a new spatial scheme specification for the CFD project. You should access this feature if you intend to perform simulations with different spatial schemes in different blocks. The **Delete Specification** button deletes the current spatial scheme specification. The **Block** drop-down menu allows you to select the block to which the current specification applies. The **Simulation Type** radio button allows you to specify viscous or inviscid calculations for the block in the current specification. The **Sutherland Law** button is used to apply the Sutherland law for viscosity. Provide the value of the Sutherland law coefficient in the accompanying input box. The **Metrics** dialog box selects the scheme for calculating the metrics of the coordinate transformation. The **Configure** button opens a dialog box for the specification of the parameters of the selected spatial scheme. The **Viscous Calculations** button allows you to specify the numerical procedure used in computing the viscous terms.


Help Spatial Differencing Scheme

Scheme specification: Specification 1 | New Specification | Delete Specification

Block: All blocks

Simulation Type: [✓ Viscous] Inviscid

Use Sutherland Law: ✓ Coefficient 0.3800000

I: Roe (Scheme 3), Roe (Scheme 4), [✓ Roe (MUSCL)] — Configure — Isotropic Fix Off

J: Roe (Scheme 3), Roe (Scheme 4), [✓ Roe (MUSCL)] — Configure — Isotropic Fix Off

K: None — Configure

Metrics: Compact, Compact (Conser, [✓ 2nd Order Central] — Configure

Viscous Calculations: Compact [✓ Central Differencing]

Close

Figure 4.13. Spatial Differencing Scheme Dialog Box.

Spatial Damping/Filtering button

This button opens a Spatial Damping/Filtering dialog box, which is used to specify damping and filtering schemes. Figure 4.14 is a screen shot of the dialog box. The **Damping/Filtering** buttons are used to select the damping and filtering options. The information required in the remainder of the dialog box depends on the option selected here. If the **Filtering** option is selected, as shown in Figure 4.14, the following information is required:

- **Filtering On/Off:** Filtering can be turned on and off selectively in any of the *i, j*, or *k* coordinate directions.
- **Frequency:** Specify the number of filtering operations to be performed per time step in this input box.
- **Configure:** This button opens up dialog boxes for the specification of the details of the filtering procedure.

If the **Damping** option is selected, as shown in Figure 4.15, the following information is required:

- **Boundary Layer Damping:** This specifies the number of nodes from the boundary below which damping is not applied. The default value is 1, which implies that damping is applied at all points.
- **Damping Coefficients:** These are the coefficients for the fourth- and second-order terms of the damping function, respectively.
- **Implicit Damping Coefficients:** These are the coefficients for the fourth- and second-order terms of the implicit damping function, respectively.
- **Implicit Damping Modifiers:** The first term is a relaxation parameter for the implicit damping function, while the second adds a small source term to the damping function.

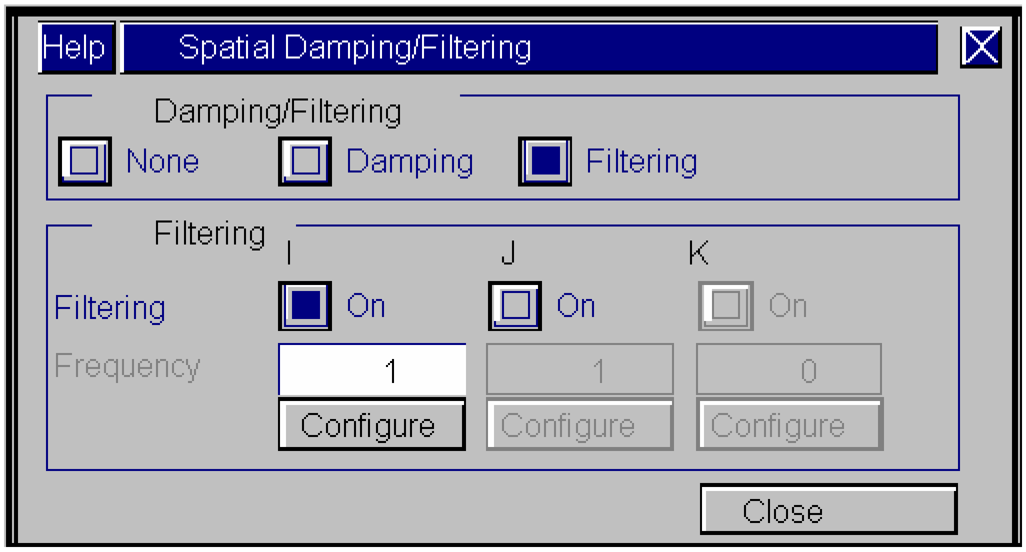
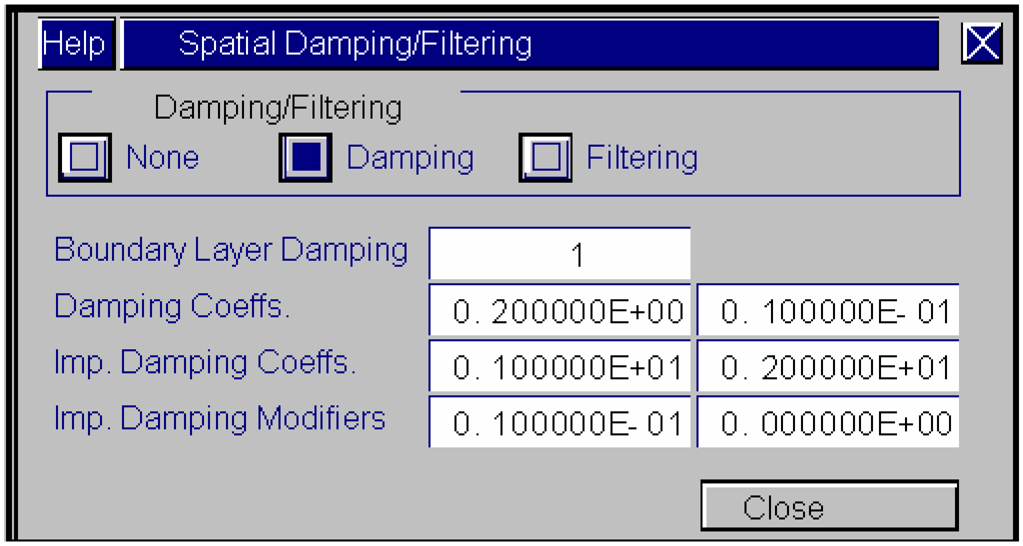


Figure 4.14. Spatial Damping/Filter Dialog Box with Filtering On.


Help Spatial Damping/Filtering

Damping/Filtering

None [✓ Damping] Filtering

Boundary Layer Damping	1	
Damping Coeffs.	0.200000E+00	0.100000E-01
Imp. Damping Coeffs.	0.100000E+01	0.200000E+01
Imp. Damping Modifiers	0.100000E-01	0.000000E+00

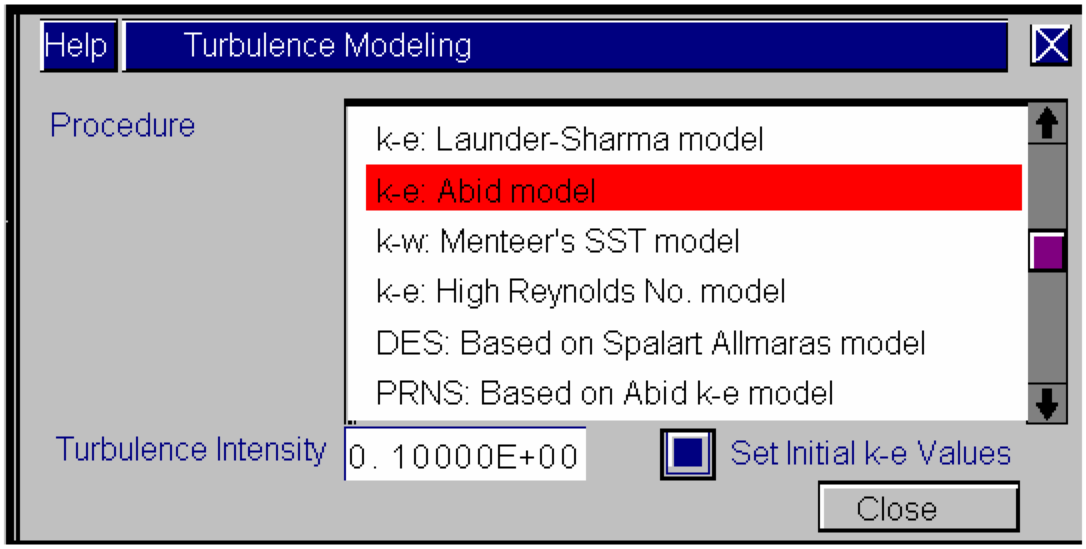
Figure 4.15. Spatial Damping/Filter Dialog Box with Damping On.

Turbulence Modeling button

This button opens the Turbulence Modeling dialog box, which is used to specify the turbulence model. Figure 4.16 is a screen shot of this dialog box. The **Procedure** dialog box lists the turbulence models supported by AEROFLO. These options are:

- LES: Smagorinsky with compact differencing of the LES terms
- LES: Smagorinsky with 2nd-order differencing of the LES terms
- LES: Smagorinsky with 1st-order differencing of the LES terms
- LES: Dynamic model
- Spalart-Allmaras one-equation model
- k-e: Launder-Sharma model
- k-e: Abid model
- k-w: Menter's SST model
- k-e: High Reynolds No. model
- DES: Based on Spalart-Allmaras model
- PRNS: Based on Abid k-e model
- PRNS: Based on High Re k-e model

The **Turbulence Intensity** input box sets the value of the inlet turbulence intensity for the k-ε models. The **Set Initial k-ε Values** button generates initial conditions for k-ε values using the inlet turbulence intensity.



Procedure

k-e: Launder-Sharma model
k-e: Abid model
k-w: Menteer's SST model
k-e: High Reynolds No. model
DES: Based on Spalart Allmaras model
PRNS: Based on Abid k-e model

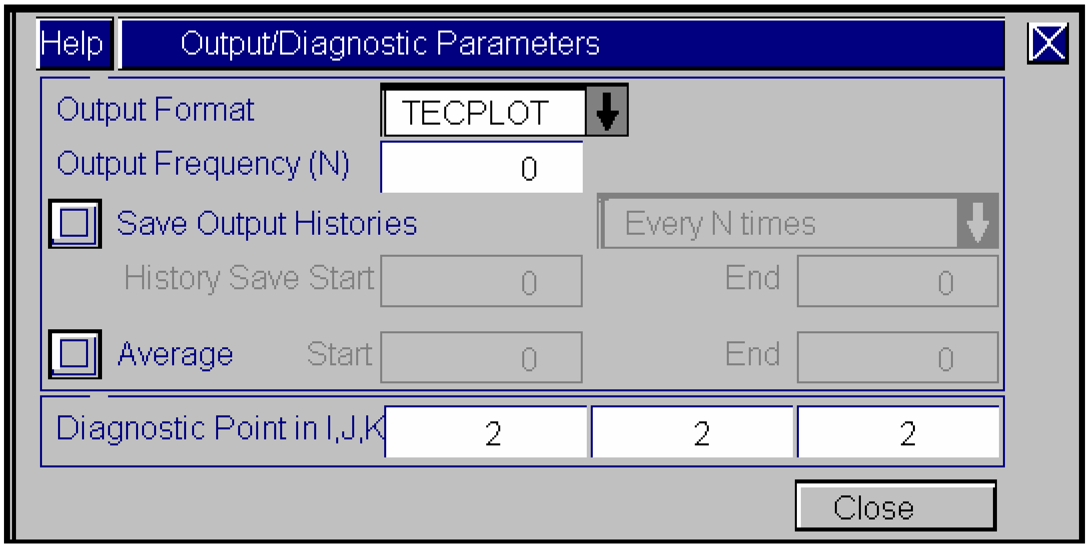
Turbulence Intensity 0.10000E+00

Set Initial k-e Values

Figure 4.16. Turbulence Modeling Dialog Box.

Output/Diagnostic Parameters button

This button opens an Output/Diagnostic Parameters dialog box, which is used to specify the output parameters. Figure 4.17 is a screen shot of this dialog box. The **Output Format** drop-down menu selects the format of the output files. The supported formats are CGNS, TECPLOT, and PLOT3D. The **Output Frequency** input box controls the number of time steps at which the output is written. The **Save Output Histories** button is used to indicate if history files are saved, and the associated input for the frequency of save. The **History Save Start/End** input boxes and the **Every xN times** drop-down menu control the output period and frequency of the history files. The **Average** buttons are used to generate average results at a point. The average is computed using the solutions generated between time steps **Start** and **End**. The **Diagnostic Point in I,J,K** input boxes select the point at which the diagnostic values of the solution are printed in the log screen or Display dialog box.


Help | Output/Diagnostic Parameters

Output Format: TECPLOT

Output Frequency (N): 0

☐ Save Output Histories — Every N times

History Save Start: 0 — End: 0

☐ Average — Start: 0 — End: 0

Diagnostic Point in I,J,K: 2 | 2 | 2

Close

Figure 4.17. Output/Diagnostic Parameters Dialog Box.

The **MHD Solver Parameters** and **Poisson Solver Parameters** are the options for the MHD calculation, which are not available in general CFD calculations.

4.4 Graphics Display Dialog Box

The Graphics Display dialog box serves two purposes: 1, to present the computational grid graphically to help you visually, while specifying boundary conditions, and 2, to present log information in a text mode as the simulation progresses. There are two modes, graphics mode and text mode, which correspond to those two purposes. Figure 4.18 is a screen shot of the graphics mode. In graphics mode, the computational grid geometries are shown and the selected boundary is highlighted. The graphics mode allows the user to visualize the selected surfaces and blocks, which aids in the application of boundary conditions, as well as block transformation.

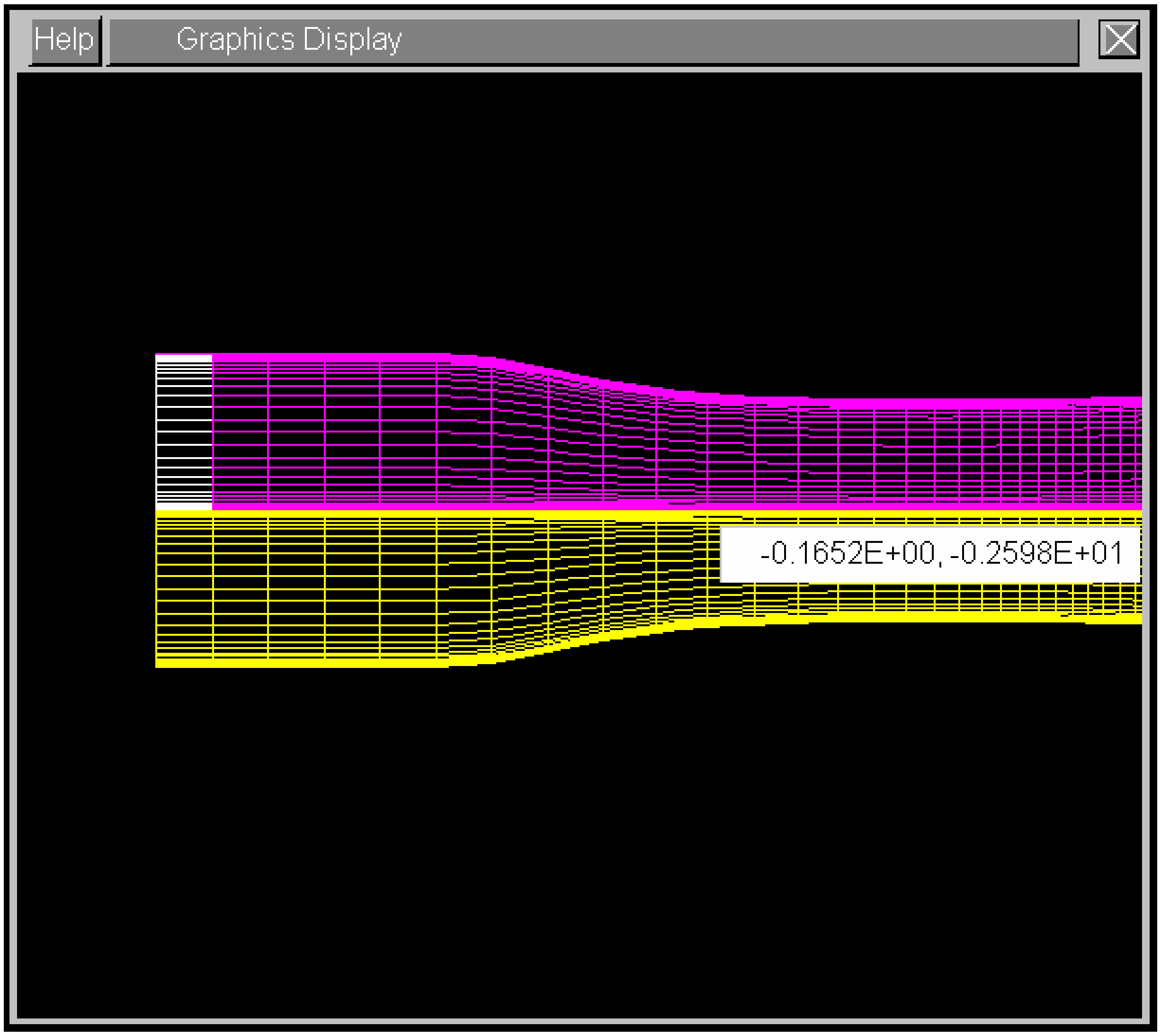
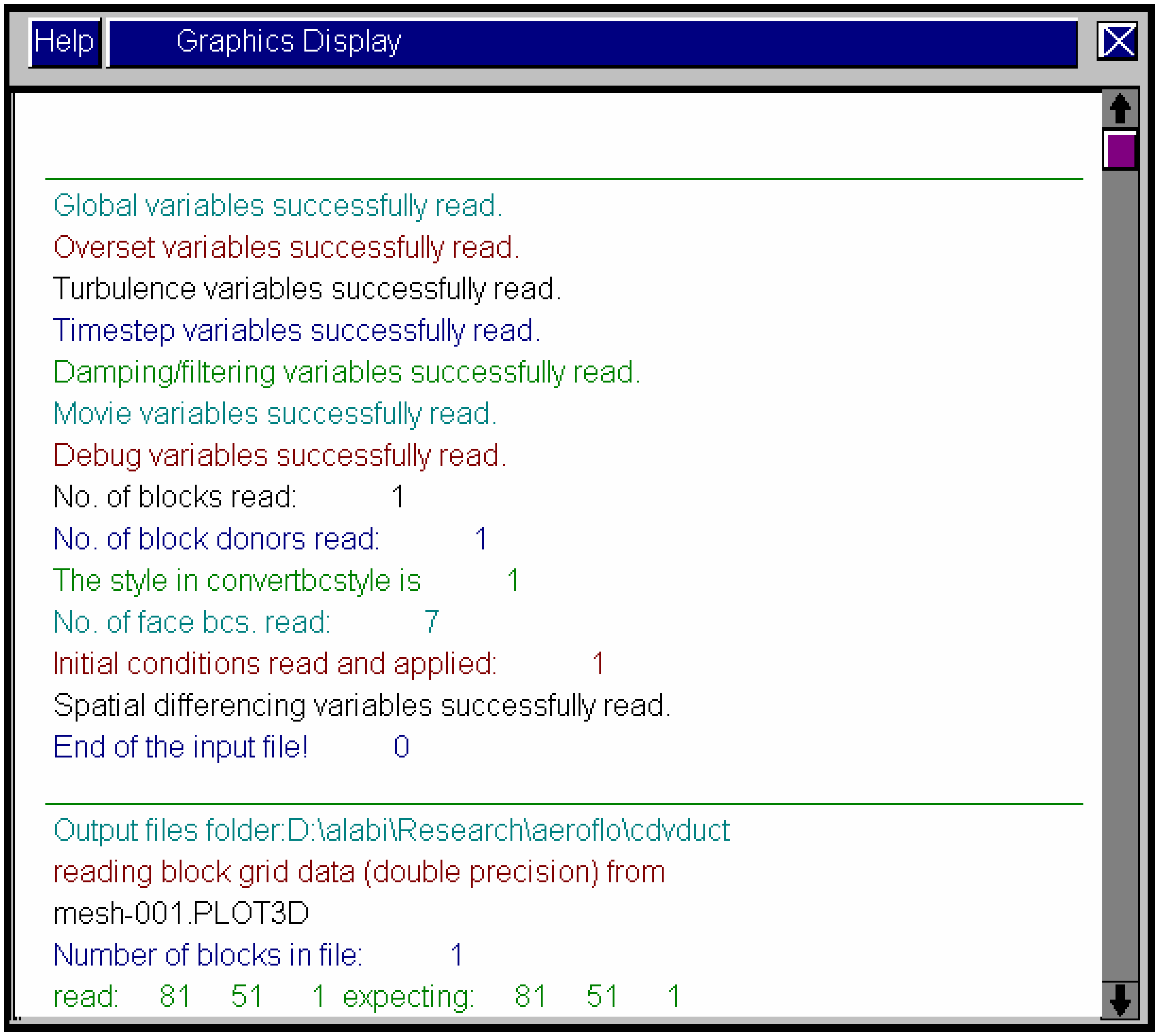


Figure 4.18. Graphics Display Dialog Box in Graphics Mode.

Figure 4.19 is a screen shot of the text mode. In text mode, the log information of the simulation will be outputted. The performance of the Graphics Display dialog box is controlled by the Graphics Control dialog box, which will be introduced in next section.



```
Global variables successfully read.
Overset variables successfully read.
Turbulence variables successfully read.
Timestep variables successfully read.
Damping/filtering variables successfully read.
Movie variables successfully read.
Debug variables successfully read.
No. of blocks read:           1
No. of block donors read:           1
The style in convertbcstyle is           1
No. of face bcs. read:           7
Initial conditions read and applied:           1
Spatial differencing variables successfully read.
End of the input file!           0

Output files folder:D:\alabi\Research\aeroflo\cdvduct
reading block grid data (double precision) from
mesh-001.PLOT3D
Number of blocks in file:           1
read:    81    51     1  expecting:    81    51     1
```

Figure 4.19. Graphics Display Dialog Box in Text Mode.

4.5 Graphics Controls Dialog Box

This dialog box controls the display in the Graphics Display dialog box. Figure 4.20 is a screen shot of the dialog box. The control functions are controlled by the following buttons:

- **Restore:** Returns the graphics display to its original state after a series of transformations (e.g., zoom).
- **Grid:** Toggles the display of a grid to help with graphic positioning, while using the mouse in the graphics area.
- **Limit:** Determines the physical limit of the display environment in (*x*,*y*,*z*). Note that the limits are also automatically set when blocks are loaded such that the blocks are contained optimally within the display area.

- **Zoom:** Performs a visual zoom of the display area on a user-selected part of the display. The user selects a zoom box via a mouse click of the top left and bottom right of the region to be zoomed into.
- **Refresh:** Redraws the graphics display area.
- **Graphics On/Off:** Toggles the graphics area between text and graphics modes. In graphics mode, the blocks are graphically displayed, while in text mode, messages and diagnostics are reported.

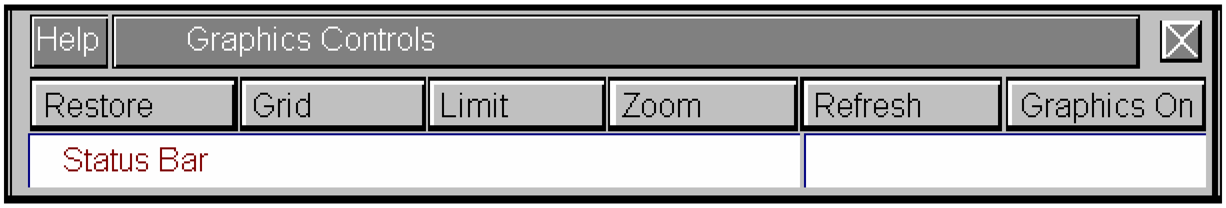


Figure 4.20. Graphics Controls Dialog Box.

4.6 Using the GUI Help

AEROFLO includes an integrated HTML-based online help system that provides easy access to the program documents. In each dialog box, there is a blue "Help" button on the top-left corner. By clicking the "Help" button, an HTML webpage that contains the help information for the dialog box will be automatically opened.

5. Tutorial: Using the AEROFLO GUI to Simulate Flow Through a Converging-Diverging Duct

In this chapter, the detailed procedures for simulating a flow through a converging-diverging duct are introduced. After following this tutorial, you will be familiar with the AEROFLO GUI and how to set up an AEROFLO project and run the project by using the AEROFLO GUI.

5.1 Problem Description

The selected sample problem is a flow through a converging-diverging duct. The physical domain has dimensions as shown in Figure 5.1, where h_{thr} = 0.14435 ft is used as the dimensional length scale. This provides a Mach number of 0.46 at the inlet and a Reynolds number of 732,676.8. The nondimensional inlet and outlet pressures are 3.375640 and 3.263250, respectively.

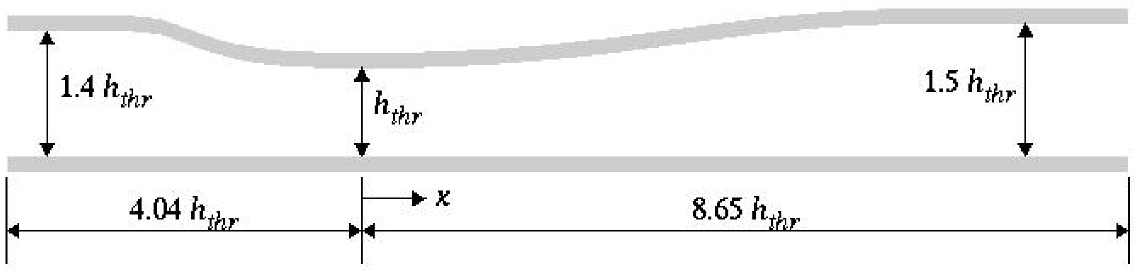


Figure 5.1. Physical Domain for Converging-Diverging Duct Calculation.

The boundary conditions for the problem are summarized in Table 5.1 below.

Boundary Conditions	
Inlet	$u=1; v=0;$ $\rho=1; P=3.37564$
Outlet	$\partial u/\partial x = 0$; $\partial v/\partial x = 0$; $\partial \rho/\partial x = 0$; $P=3.26325$
Lower Wall	solid wall
Upper Wall	solid wall
Initial Conditions	
$u=1; v=0; \rho=1; P=3.37564$	

Table 5.1. Boundary and initial conditions for flow through a converging-diverging duct.

The procedures for solving this problem are in accordance with the general procedures discussed in

Chapter 3. Each of these steps is described in the following sections.

5.2 Getting the Computational Grid

The computational grid file mesh-001.PLOT3D is available in the directory *\samples\cdvduct\* under AEROFLO's installation folder. It is a single-block grid with PLOT3D format. The grid size is 81x51x1. The geometry of the grid is shown in Figure 5.2.

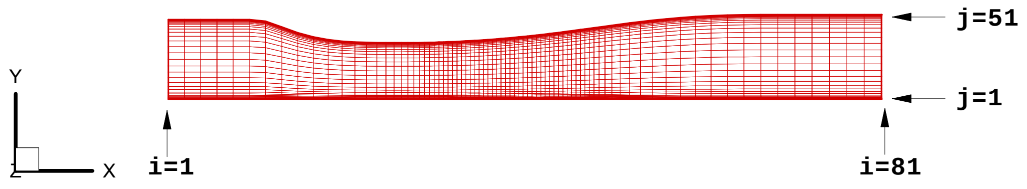


Figure 5.2. Computational Grid for Converging-Diverging Duct.

5.3 Setting Up the Project File Using the AEROFLO GUI

The project file is also available for this problem (cdvduct.afl in *\sample directory*). However, it is highly suggested that you follow the steps described in this section to prepare your own version of the project file. This will familiarize you with the AEROFLO GUI and teach you how to solve a CFD problem in AEROFLO. To create the project file, you need to follow the following steps:

(1) Launch AEROFLOGUI:

You can double-click the *multidisc.exe* in AEROFLO's installation directory or find the "*Programs > AEROFLO > AEROFLOGUI*" in the Start menu.

(2) Select Simulation Type:

Make sure the selection of the simulation type is CFD (this is the default selection).

(3) Create a New Project File:

a. Click the "Load/Save Project" button.

b. Click the "New Project" button.

c. In the "Project Title" input box, type "Converging/Diverging Duct." This is only for administrative purposes; you may change it to any other words.

(4) <u>Save the Project File</u>:

   a. Click the “Save Project As” button. A file navigation dialog box is opened.
   b. Change the directory to the folder where you want to save the project file. Here we assume the file is saved in folder *c:\aeroflo\test\*. If the folder does not exist, you must create it first.
   c. In the “File name” input box, type in the name of the project file. Here we assume the name of the file is “cdvduct.afl.” The file extension *.afl* a is suggested file extension for AEROFLO project files. You can also use other extensions that have the native plain text format (such as *.txt*).
   d. Click “Save” to save the file. A notice will automatically pop up, showing the message “Project Saved in: cdvduct.afl.” Click “OK” to continue.

(5) <u>Specify the Subfolder</u>:

   a. In the “Project Subfolder” input box, type in the name of the project subfolder. Here, we assume the name of the subfolder is “cdvduct.” The subfolder is the place where the project file can find the grid and initial files, and where AEROFLO outputs the results for the project. Properly setting the subfolder can make the project much more organized.
   b. Create the cdvduct folder in the directory *c:\aeroflo\test\* in your operating system. If the subfolder does not exist when you save the project, an error message will be shown.
   c. Click the “Save Project” button. A message is popped up to notice the saving of the file is successful. Click “OK” to continue.
   d. Click “OK” to close the Load/Save Project dialog box and return to the main graphical interface of AEROFLO.

(6) <u>Set Global Flow Conditions</u>:

   a. Click the “Problem Setup” button. This will open the Problem Setup dialog box if it is closed.
   b. Click the “Global Flow Conditions” button. The Global Flow Variables dialog box is opened.
   c. Make sure that the “Non-Dim” button is selected, as this particular problem is nondimensional.
   d. Input 0.46 and 732676.8 for the Mach No. input box and Reynolds No. input box, respectively. Leave the Prandtl No. as the default value (0.72), as it is not important for the current problem.
   e. Click “OK” to close the Global Flow Variables dialog box.
   f. You can find the “Global Flow Conditions” button is disabled ("grayed"). This indicates that it has been processed. This happens for every button in the Problem Setup dialog box. To return it to a grayed dialog box, you can simply click the Edit button on the right of the button that you wish you access.
   g. It is highly suggested that you save the project file after every step listed in this tutorial. To save the file, click the “Load/Save Project” button, and then click “Save Project.”

(7) Load the Grid File:

a. Copy the grid file (*\samples\cdvduct\mesh-001.PLOT3D* in AEROFLO installation folder) to your project subfolder (*c:\aeroflo\test\cdvduct\*).

b. When you return to the main graphic interface, click "Manage Grid Blocks" button. This opens up the Manage Blocks dialog box.

c. Click the "Add New Block" button to open Add New Block dialog box.

d. In the Block Name input box, type the name for the on-going grid block. Here we use "mesh0."

e. In the Mesh Format drop-down menu, select PLOT3D since the grid file is in PLOT3D format.

f. Click the "Load Block" button to open a file navigation dialog box. Change the directory to the project subfolder (*c:\aeroflo\test\cdvduct\*) to select the grid file. Click "Open" to load the grid file.

g. If the loading is successful, a message will pop up to show the mesh size. Click "OK" to close the Add New Block dialog box and finish the mesh loading.

(8) View the Grid:

Click the "Graphics On" button in the Graphics Control dialog box to change the Graphics Display dialog box to the graphic mode. The geometries of the computational grid will be shown.

(9) Set Boundary Conditions:

Boundary conditions must be specified at all of the grid boundaries. The following steps describe how to set the boundary conditions for each of the boundary faces. The boundary conditions are summarized in Table 5.1.

a. Inlet (I=1 face):

- Select "First I Face" in the BC on Blocks drop-down menu. The inlet boundary face is highlighted (white-colored) in the Graphics Display dialog box.
- To add the *u=1* boundary condition:
  - ➢ Click the "Add Boundary Condition" button to open the Add New BC dialog box.
  - ➢ In the Type drop-down menu, select "Dirichlett."
  - ➢ In the Variable drop-down menu, select "u-velocity."
  - ➢ In the Value input box, type 1.0.
  - ➢ Click "OK" to close the Add New BC dialog box.
- To add the v=*0* boundary condition:
  - ➢ Click the "Add Boundary Condition" button to open the Add New BC dialog box.
  - ➢ In the Type drop-down menu, select "Dirichlett."
  - ➢ In the Variable drop-down menu, select "v-velocity."
  - ➢ In the Value input box, type 0.0.

    - Click “OK” to close the Add New BC dialog box.
  - To add the ρ=*1* boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Dirichlett.”
    - In the Variable drop-down menu, select “Density.”
    - In the Value input box, type 1.0.
    - Click “OK” to close the Add New BC dialog box.
  - To add the *P=3.37564* boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Dirichlett.”
    - In the Variable drop-down menu, select “Pressure.”
    - In the Value input box, type 3.37564.
    - Click “OK” to close the Add New BC dialog box.

b. Outlet (I=81 face):
  - Select “Last I Face” in the BC on Blocks drop-down menu. By clicking “Refresh” in the Graphic Controls dialog box, the outlet boundary face is highlighted (white-colored) in the Graphics Display dialog box.
  - To add the $\partial u / \partial x = 0$ boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Neumann.”
    - In the Variable drop-down menu, select “u-velocity.”
    - Click “OK” to close the Add New BC dialog box.
  - To add the $\partial v / \partial x = 0$ boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Neumann.”
    - In the Variable drop-down menu, select “v-velocity.”
    - Click “OK” to close the Add New BC dialog box.
  - To add the $\partial \rho / \partial x = 0$ boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Neumann.”
    - In the Variable drop-down menu, select “Density.”
    - Click “OK” to close the Add New BC dialog box.
  - To add the *P=3.26325* boundary condition:
    - Click the “Add Boundary Condition” button to open the Add New BC dialog box.
    - In the Type drop-down menu, select “Dirichlett.”
    - In the Variable drop-down menu, select “Pressure.”
    - In the Value input box, type 3.26325.

        - ➢ Click “OK” to close the Add New BC dialog box.

c. Lower Wall (J=1 Face)

    - Select “First J Face” in the BC on Blocks drop-down menu. By clicking “Refresh” in the Graphic Controls dialog box, the outlet boundary face is highlighted (white colored) in the Graphics Display dialog box.
    - To add the *wall* boundary condition:
        - ➢ Click the “Add Boundary Condition” button to open the Add New BC dialog box.
        - ➢ In the Type drop-down menu, select “Solid wall.”
        - ➢ Click “OK” to close the Add New BC dialog box.

d. Upper Wall (J=51 Face)

    - Select “Last J Face” in the BC on Blocks drop-down menu. By clicking “Refresh” in the Graphic Controls dialog box, the outlet boundary face is highlighted (white colored) in the Graphics Display dialog box.
    - To add the *wall* boundary condition:
        - ➢ Click the “Add Boundary Condition” button to open the Add New BC dialog box.
        - ➢ In the Type drop-down menu, select “Solid wall.”
        - ➢ Click “OK” to close the Add New BC dialog box.

    Note: If you set the above boundary conditions in a wrong way and you wish to modify it, you can select the boundary condition that you wish to modify in the Selected BC list and click “Modify Selected BC” to modify it. You can also use “Delete Selected BC” to delete the wrong boundary condition and create a new boundary condition to replace it.

When you finish setting all boundary conditions, click “Close” to close the Manage Blocks dialog box and return to the main graphical interface.

(10) <u>Set Initial Conditions</u>:

There are four initial conditions need to be specified, which are listed in Table 5.1. Use the following steps to set each of the initial conditions.

a. Click the “Initial Conditions” button to open the Initial Conditions dialog box.

b. To specify the *u=1* initial condition:

    - Click the “Add New Initial Conditions” button to open the New Initial Condition dialog box.
    - In the Variable drop-down menu, select “u-velocity.”
    - In the Value input box, type 1.0.
    - Click “OK” to close the New Initial Condition dialog box.

c. To specify the *v=0* initial condition:

    - Click the “Add New Initial Conditions” button to open the New Initial Condition dialog box.
    - In the Variable drop-down menu, select “v-velocity.”
    - In the Value input box, type 0.
    - Click “OK” to close the New Initial Condition dialog box.

d. To specify the *ρ=1* initial condition:
   - Click the “Add New Initial Conditions” button to open the New Initial Condition dialog box.
   - In the Variable drop-down menu, select “Density.”
   - In the Value input box, type 1.0.
   - Click “OK” to close the New Initial Condition dialog box.

e. To specify the *P=3.37564* initial condition:
   - Click the “Add New Initial Conditions” button to open the New Initial Condition dialog box.
   - In the Variable drop-down menu, select “Pressure.”
   - In the Value input box, type 3.37564.
   - Click “OK” to close the New Initial Condition dialog box.

f. Click “Close” to close the Initial Conditions dialog box and return to the main graphical interface.

   Note: If you set the above initial conditions incorrectly and wish to modify them, you can select the initial condition that you wish to modify in the initial condition list and click “Modify Selected Initial Conditions” to modify it. You can also use “Delete Selected Initial Conditions” to delete the wrong initial condition and create a new initial condition to replace it.

(11) <u>Set Time Integration Scheme</u>:

For this problem, the Beam-Warming scheme is used for time integration. The time step size is set to 0.01 and the problem will be calculated for 10,000 steps. Use the following steps to set the time integration scheme:

a. Click the “Time Integration Scheme” button to open the Time Integration Scheme dialog box.
b. In the Time Scheme drop-down menu, select “BW2 (Full Formulation).”
c. Make sure that the “Use the fixed time steps” button is selected. Also check to make sure that the “Use preconditioner” button is not selected.
d. In the Max No. of Steps input box, type 10000.
e. In the Time Step Size input box, type 0.01.
f. Click “OK” to close the Time Integration Scheme dialog box.

(12) <u>Set Spatial Differencing Scheme</u>:

For this problem, the MUSCL scheme is used for the spatial scheme and the 2nd-order central difference scheme is used for the metrics differencing scheme. Use the following steps to set the spatial differencing scheme:

a. Click the “Spatial Differencing Scheme” button to open the Spatial Differencing Scheme dialog box.
b. In the Simulation Type buttons, select “Viscous,” since the flow in this problem is considered to be viscous.
c. Select the “Roe (MUSCL)” scheme for both I and J directions.

d. Select “Second Order Central” scheme for Metrics.

e. Click “Close” to close the Spatial Differencing Scheme dialog box.

(13) Set Spatial Damping/Filtering:

For this problem, there is no need to use spatial damping or filtering. Use the following steps to set the spatial damping/filtering scheme:

a. Click the “Spatial Damping/Filtering” button to open the Spatial Damping/Filtering dialog box.

b. Leave all the selections as default.

c. Click “Close” to close the Spatial Damping/Filtering dialog box.

(14) Set Turbulence Modeling:

For this problem, the Abid’s k-ε model is used for turbulence calculation. Use the following steps to set the turbulence model:

a. Click the “Turbulence Modeling” button to open the Turbulence Modeling dialog box.

b. In the Procedure list, select “k-e: Abid Model.”

c. Check to make sure the Turbulence Intensity input box is set to be 0.1 and the “Set Initial k-e Values” button is selected.

d. Click “Close” to close the Turbulence Modeling dialog box.

Note: If you find the “Turbulence Modeling” button is disabled, go back to the settings of the Spatial Difference Scheme to check if you forgot to select the “Viscous” button.

(15) Set Output Parameters:

In this project, we set the output files in TECPLOT format. The simulation results are set to be outputted every 500 steps. Use the following steps to set the output parameters:

a. Click the “Output Parameters” button to open the Output Parameters dialog box.

b. In the Output Format drop-down menu, select “TECPLOT.”

c. In the Output Frequency input box, type 500.

d. Click “Close” to close the Output Parameters dialog box.

(16) Save the Project File:

We have finished all of the settings for this project. You can go back to check if each of the settings are correct. You can modify all of them. After this, do not forget to save your project file.

5.4 Running the Project in the GUI

After the project file is saved, the problem can be calculated by clicking the “Solve” button. The Graphics Displays dialog box will show the log information of the simulation, including the iteration steps and norms. You can use the “Stop/Pause” button to stop (pause) the calculation. To resume a paused calculation, simply click the "Resume" button.

5.5 Simulation Results

The computation will take several hours. After the calculation is finished, you can check the simulation results. In this problem, we specify the TECPLOT format as the format of the output result. If the TECPLOT software is already installed in your system, you may view the result by clicking "View Results." If you prefer to use other visualization tools, you can launch the software outside of AEROFLO and find the result files in the project subfolder directory. Figure 5.3 is the simulation result of the Mach number contours for the flow field.

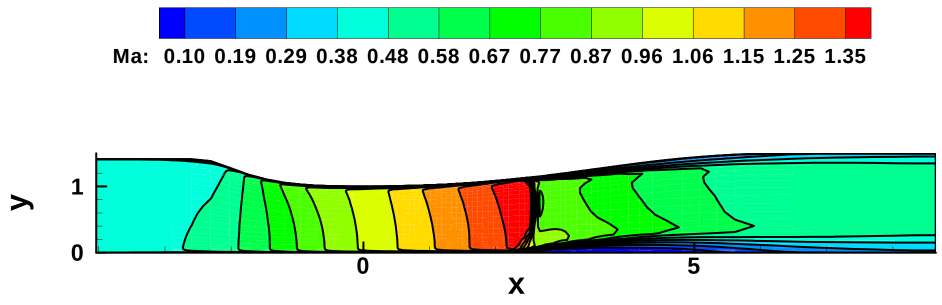


Figure 5.3. Mach Number Contour for Flow Through a Converging-Diverging Duct.

5.6 Remarks

By following the above tutorial, you have learned the basic procedures to solve a CFD problem in AEROFLO and should now be familiar with the AEROFLO GUI. Indeed, the AEROFLO GUI provides an environment by which the user can easily set the input parameters for AEROFLO. All of these parameters are saved in a project file. In the next chapter, we will use the project file created in this chapter to study the detailed structures of a project file. By understanding the meaning of each keyword in the project file, the user can modify or even create a project file directly without using the GUI program.

6. Introduction to AEROFLO Input Keywords

In this chapter, we will use the project file created in the last chapter to introduce the AEROFLO input keywords.

6.1 A First Look at the Project File

In the last chapter, we used the AEROFLO GUI to solve a problem of flow through a converging-diverging duct. By using the GUI environment, we set the problem input parameters, such as the computational grid information, boundary conditions, numerical schemes, and many other computation control parameters. In fact, the GUI will automatically save these input parameters in a project file with a special format that can be read and interpreted by the AEROFLO solver. It is this project file, together with the auxiliary grid files, initial files, and boundary condition files, that will be the inputs for AEROFLO. By understanding the structure and format of the project file, the user may directly create a new project file or modify an existing project file without using the GUI program. In this chapter, we will introduce the format of the project file.

The AEROFLO project file is an ASCII text file consisting of AEROFLO keywords or instructions and data. You can open and edit the project file by using any text edit software (NotePad, WordPad, WORD, etc.). Figure 6.1 is the content of the project file (cdvduct.afl) that we created for the flow trough converging-diverging duct problem in the last chapter. Additional comments and boxes have been added to help you to understand the structure of the file and the meaning of the primary keywords. In the next section, we will analyze this file in detail.

```
cdvduct .afl -------------------------------------------------------------------- Project File Name
```

```
$GLOBAL,
       TITLE  = 'Converging /Diverging  Duct',---------------------------------- Project Title
       FOLDER  = 'cdvduct ', --------------------------------------------- Project Subfolder Directory
       SIMTYPE  =             2 , ------------------------------------------ Simulation Type: 2 = CFD
       MACH  =     0.4600000      , ----------------------------------------------------- Mach Number
       REYNOLDS  =       732677.0       , ---------------------------------------------- Reynold Number
       PR  =    0.7200000         , -------------------------------------------------- Prandtl Number
       ORDER  =              2 ,
       BCSTYLE  = 2,
     $END
```

$GLOBAL

Global Project Data

```
$OVERSET,
     $END
```

$OVERSET

Overset Variables

```
$TURB,
       TURBTYPE  =              7 , ---------------------------------- Turbulence Model: 7 = Abid k-ε
       TUVALUE  =    0.1000000      ,------------------------------ Turbulence Intensity for k-ε model
       RESET  =             1 , -------------------------------------------- Set the initial k-ε values
     $END
```

$TURB

Turbulence Data

```
$SPATIAL,                                                                              Spatial Scheme:
       SCHEME  =              5 ,             5 ,          -999 , -----------          5 = MUSCL
       METRIC  =              0 , ---------------------------- Metric Scheme: 0 = 2nd order central
       VISCOUS  =              1 ,             1 ,             1 , ------ Viscous: 1 = true
       VISCHEME  =              0 ,
       VISCMON  =              1 ,
       CUTOFFI  =    5.0000001E-02  ,
       ISOTROPI  =              0 ,
       ICUT  =              0 ,             0 ,             0 ,
       CUTOFFJ  =    5.0000001E-02  ,
       ISOTROPJ  =              0 ,
       JCUT  =              0 ,             0 ,             0 ,
       SUTHERLN  =    0.3800000      ,
     $END
```

$SPATIAL

Spatial Differencing Data

```
$TIMESTEP,
       SCHEME  = 'BW2', ------------------------------------------ Time Scheme: 'BW2' = Beam-Warming
       MAXITER  =          10000  , ------------------------------ Maximum Number of Iteration Steps
       GLOBALDT  =     9.9999998E-03  , ------------------------------------ Iteration Time Step Size
       SUBDT  =    9.9999998E-03  ,
       GLOBTOL  =    1.0000000E-30  ,
       SUBTOL  =    1.0000000E-03  ,
     $END
```

$TIMESTEP

Time Differencing Data

```
$DAMPING,
       TYPE  =              3 ,
       ES2  =    0.2000000        ,
       ES4  =    9.9999998E-03   ,
       FES2  =     1.000000        ,
       FES4  =     2.000000        ,
       MAXRED  =              1 ,
       OMGAV  =    9.9999998E-03  ,
       SRCONST  =    0.0000000E+00  ,
       FILTER  =              0 ,             0 ,             0 ,
       NFILTER  =              1 ,             1 ,             0 ,
       ALPHAN  =    0.3000000       ,  0.3000000       ,  0.3000000       ,
       ALPHA1  =    0.0000000E+00  ,  0.0000000E+00  ,  0.0000000E+00  ,
       ALPHA2  =    0.3000000       ,  0.3000000       ,  0.3000000       ,
       ALPHA3  =    0.3000000       ,  0.3000000       ,  0.3000000       ,
       ALPHA4  =    0.3000000       ,  0.3000000       ,  0.3000000       ,
       ALPHA5  =    0.3000000       ,  0.3000000       ,  0.3000000       ,
       ORDERN  =             10 ,            10 ,            10 ,
       ORDER1  =              0 ,             0 ,             0 ,
       ORDER2  =              2 ,             2 ,             2 ,
       ORDER3  =              2 ,             2 ,             2 ,
       ORDER4  =              2 ,             2 ,             2 ,
       ORDER5  =              2 ,             2 ,             2 ,
     $END
```

$DAMPING

Damping or Filtering Data

Figure 6.1. The Project File cdvduct.afl for Flow Through Converging-Diverging Duct.

```
$OUTPUT,
       PRINTFRQ  =          500  , ------------------------------- Output Frequency
       FORMAT  = 'TECPLOT',    ------------------------------------ Output Format
     $END
```

$OUTPUT
Output Data

```
$DEBUG,
       POINT  =            2  ,            2  ,            2  ,
     $END
```

$DEBUG
Diagnostic Point

```
$BLOCK,
       NAME  = 'mesh0',----------------------------------------- Mesh Block Name
       MESHFILE  = 'mesh-001.PLOT3D', ---------------------------- Mesh File Name
       FILETYPE  = 'PLOT3DF', ------------------------------------- Mesh Format
       ICFILE  = 'NULL', --------------------------------- IC File Name: 'NULL' = N/A
       PERIODIC  =            0  ,            0  ,            3  , ------- Mesh Topology
       ISIZE  =           81  , ----------------------------- Grid Size in I Direction
       JSIZE  =           51  , ----------------------------- Grid Size in J Direction
       KSIZE  =            1  , ----------------------------- Grid Size in K Direction
       ORDER  =            2  ,
       ANGLE  =   0.0000000E+00  ,
       DISTANCE  =   0.0000000E+00  ,
       NOCROSS  =            0  ,
     $END
```

$BLOCK
Mesh Block Data

```
COMMENT:  BC            1  ------------------------------------------------ Comment
$FACEBC,
       BLOCK  = 'mesh0',  --------------------------------------------- Block Name
       SIDE  =            1  , ----------------- Position of Block Boundary: 1 = First I Face
       TYPE  =           10  , ------------------ Boundary Condition Type: 10 = Dirichlett
       VARIABLE  =            2  , ------------------- Preferred Variable: 2 = u-velocity
       VALUE  =    1.000000      , -------------------------------- Preferred Value
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
First I Face
u=1

```
COMMENT:  BC            2
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            3  , ------------------- Preferred Variable: 3 = v-velocity
       VALUE  =   0.0000000E+00  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
First I Face
v=0

```
COMMENT:  BC            3
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            1  , ---------------------- Preferred Variable: 1 = Density
       VALUE  =    1.000000      ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
First I Face
ρ=1

```
COMMENT:  BC            4
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            5  , -------------------  Preferred Variable: 5 = Pressure
       VALUE  =    3.375640      ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
COMMENT:  BC            5
```

$FACEBC
Boundary Condition
First I Face
P=3.37564

Figure 6.1. The Project File cdvduct.afl for Flow Through Converging-Diverging Duct (Cont.).

```
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            -1  ,  ------------------------ Position of Block Boundary: -1 = Last I Face
       TYPE  =            23  ,  --------------------------  Boundary Condition Type: 23 = Neumann
       VARIABLE  =            2  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
Last I Face
∂u/∂x=0

```
COMMENT:  BC              6
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            -1  ,
       TYPE  =            23  ,
       VARIABLE  =            3  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
Last I Face
∂v/∂x=0

```
COMMENT:  BC              7
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            -1  ,
       TYPE  =            23  ,
       VARIABLE  =            1  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
Last I Face
∂ρ/∂x=0

```
COMMENT:  BC              8
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            -1  ,
       TYPE  =            10  ,
       VARIABLE  =            5  ,
       VALUE  =     3.263250       ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
Last I Face
P=3.26325

```
COMMENT:  BC              9
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =             2  ,  --------------------------- Position of Block Boundary: 2 = First J Face
       TYPE  =             1  ,  ----------------------------  Boundary Condition Type: 1 = Solid Wall
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
First J Face
Solid Wall

```
COMMENT:  BC             10
$FACEBC,
       BLOCK  = 'mesh0',
       SIDE  =            -2  ,  ------------------------- Position of Block Boundary: -2 = Last J Face
       TYPE  =             1  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
```

$FACEBC
Boundary Condition
Last J Face
Solid Wall

```
$INITIAL,
       VARIABLE  =            2  ,  ------------------------------------------------ Preferred Variable
       VALUE  =     1.000000       ,  ------------------------------------------------  Preferred Value
     $END
```

$INITIAL
Initial Condition: *u=1*

```
$INITIAL,
       VARIABLE  =            2  ,
       VALUE  =    0.0000000E+00   ,
     $END
```

$INITIAL
Initial Condition: *v=0*

```
$INITIAL,
       VARIABLE  =            1  ,
       VALUE  =     1.000000       ,
```

$INITIAL
Initial Condition: *ρ=1*

Figure 6.1. The Project File cdvduct.afl for Flow Through Converging-Diverging Duct (Cont.).

```
      $END
 $INITIAL,
        VARIABLE  =            5 ,
        VALUE  =    3.375640      ,
      $END
```

$INITIAL

Initial Condition:
P=3.37564

Figure 6.1. The Project File cdvduct.afl for Flow Through Converging-Diverging Duct (Cont.).

6.2 Structure of the Project File

Let’s look at the project file now. The first line is the name of the project file. After that, the file consists of several blocks. In Figure 6.1, these blocks are highlighted and distinguished from each other by using green boxes. Each of these blocks starts with a keyword that has a $ in front of it, such as **$GLOBAL**, **$TURB**, **$SPATIAL**, **$TIMESTEP**, **$BLOCK**, **$FACEBC**, **$INITIAL**, etc. These keywords are called group keywords. The keyword **$END** is used to end a block. By using this block structure, the AEROFLO input parameters are categorized into several groups. For example, the **$GLOBAL** block specifies global project data, including simulation type, project folder, and flow parameters, such as Reynolds number and Mach number. Some group keywords are necessary for every problem (such as **$GLOBAL**, **$SPATIAL**, **$TIMESTEP**, and **$BLOCK**) and others are optional (such as **$TURB** and **$DAMPING**).

In each block, the input parameters are set by assigning a specified value or data to a special keyword. For example, the Mach number 0.42 is assigned to the keyword **MACH** by a statement “*MACH = 0.4200000.*” Each of the blocks contains several of these value-assignable keywords. These keywords are called sub-keywords. Under each of the group keywords, the block contains its own special group of sub-keywords. The meaning of some primary sub-keywords and the meaning of their values are explained in the comments in Figure 6.1.

In this project file, the first block is **$GLOBAL**, which sets the global project data, including the project title, project subfolder, simulation type, and flow parameters (Ma, Re, Pr). The next block is the **$OVERSET** block, which is used to set the overall boundary condition information. In this case, no overset boundary condition is used. Therefore, there is no sub-keyword is assigned inside this block. The **$TURB** block sets the turbulence modeling, the **$SPATIAL** block sets the spatial differencing scheme, and the **$TIMESTEP** block sets the time differencing scheme. The **$DAMPING** block contains the spatial scheme damping and filtering information, which is not used in this project. The **$OUTPUT** block sets the output format and frequency, while the **$DEBUG** block is the diagnostic point at which visual solution data can be printed out. One **$BLOCK** block is used to specify the single mesh block used in this problem. The data includes the format, grid sizes, and topology information for the mesh block. Ten **$FACEBC** blocks are used to specify ten boundary conditions and four **$INITIAL** blocks are used to specify four initial conditions. The meaning of each block is also noted in Figure 6.1.

6.3 AEROFLO Input Keywords

A detailed explanation of the primary keywords in this project file is given in Table 6.1. These keywords correspond to the parameters chosen by the user in the GUI in the last chapter. The other sub-keywords are automatically generated by the GUI and are set to default values. A complete reference of the AEROFLO keywords can be found in Chapter 16.

6.4 Editing the Project File Directly

After understanding the structure of the project file and the meaning of each of the AEROFLO keywords, the user can directly write a new project file or edit an existing project file. There are a few rules involved in creating an input file:

- All group keywords must have a "$" in front of them, e.g. $GLOBAL, $BLOCK.
- All keyword input groups must end with the "$END" limiter.
- All sub-keywords must be followed by an equal sign "=", followed by a value depending on the data type.
- All sub-keyword data entry lines must end with a comma ",".
- Sub-keyword values must be separated by a "," when more than one input value is required per sub-keyword, e.g., "POINT = 0.2, 0.4, 0.4," would be an input line for a coordinate point.
- All real data type input must have a ".". For example:
  MACH = 0.2,
  MACH = 2.0,
- String inputs should be within single string quotes of the type shown in the example below:
  MESHFILE = 'foil.in',

However, input into AEROFLO via an input file affords the following flexibility:

- Extra lines and spaces can be inserted anywhere.
- Comments and notes can be inserted anywhere. However, it is probably good practice to insert comments outside of keyword input blocks ($KEY - $END sections).
- Inputs that are not relevant to a particular problem do not need to be provided.
- Input may be provided in any order. For example:

```
$BLOCK
NAME = 'mesh0',
MESHFILE = 'mesh0.dat',
ISIZE = 245,
JSIZE = 50,
$END
```

would be read the same as

```
$BLOCK
NAME = 'mesh0',
ISIZE = 245,
JSIZE = 50,
MESHFILE = 'mesh0.dat',
$END
```

- Keyword input blocks may be ordered as desired. For instance, block inputs may be entered at the top of the files, while global inputs are at the bottom, but the global data would still be read first. This allows the user to make up input files easily by copying and modifying whole blocks from other input or sample files.

$GLOBAL	**Global project data**
TITLE	Specify a title for the project for information purposes
FOLDER	Subfolder directory: folder from which input files will be read
SIMTYPE	The type of simulation: 2 — general CFD
MACH	Mach number for simulation
REYNOLDS	Reynolds number for simulation
PR	Prandtl number for simulation
$TURB	**Turbulence modeling data**
TURBTYPE	Turbulence model: 7 — Abid k-ε
TUVALUE	Turbulence intensity for k-ε model
RESET	Set the initial k-ε values? (1 — yes ; 0 — no)
$SPATIAL	**Spatial differencing scheme data**
SCHEME	Spatial differencing scheme for all directions: 5 — MUSCL; -999 — 2D problem
METRIC	Metric differencing scheme: 0 — 2nd order central differencing
VISCOUS	Viscous specification in all three directions: 0 — inviscid; 1 — viscous
$TIMESTEP	**Time differencing scheme data**
SCHEME	Time differencing scheme: 'BW2' — Beam Warming
MAXITER	Maximum number of iterations
GLOBALDT	The time step size for iteration
$DAMPING	**Damping or filtering data**
$OUTPUT	**Output controlling data**
PRINTFQ	Frequency of printing results
FORMAT	Specifies the preferred output format: 'TECPLOT' — TECPLOT format
$DEBUG	**Debugging and monitoring data**
$BLOCK	**Mesh block data**
NAME	Name for the mesh block
MESHFILE	Location of the grid file for the block
FILETYPE	Format of the grid file: 'PLOT3D' — PLOT3D format
ICFILE	Location of the initial condition file for the block: 'NULL' — no file
PERIODIC	Periodic description of the block in all directions: 0 — no periodicity; 3 — 2D problem
ISIZE	Size of the grid in I direction
JSIZE	Size of the grid in J direction
KSIZE	Size of the grid in K direction
$FACEBC	**Pressure boundary condition data**
BLOCK	Name of the block to which the boundary condition refers
SIDE	Side of the block: 1 — first I face; -1 — last I face; 2 — first J face; -2 — last J face
TYPE	Boundary condition type: 1 — solid wall; 10 — Dirichlett; 23 — Neumann
VARIABLE	Variable which the boundary condition refers: *1 — ρ; 2 — u; 3 — v; 4 — w; 5 — P*
VALUE	Values assigned in the boundary condition
$INITIAL	**Initial boundary condition data**
VARIABLE	Variable which the initial condition refers: *1 — ρ; 2 — u; 3 — v; 4 — w; 5 — P*
VALUE	Values assigned to the variable in the initial condition

Table 6.1. The Primary Keywords Used in Project File cdvduct.afl.

7. Running AEROFLO

In this chapter, we will introduce how to run an AEROFLO project in command-line and how to restart a calculation from a previous one.

7.1 Running the Program in Command-Line

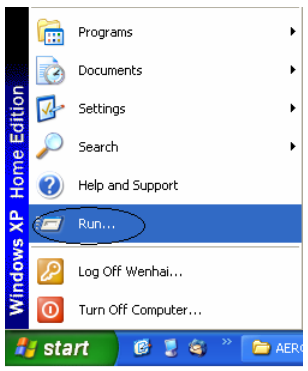
We have introduced how to run an AEROFLO problem in the GUI in Chapter 5. However, an AEROFLO project can also be run in command-line instead of in the GUI. In fact, the calculation speed is faster when running in command-line. However, before you start to run the program in command-line, make sure the following requirements are satisfied:

- The grid files must be put in the project sub-folder, or its path information must be included in the **$BLOCK/MESHFILE** keyword, so that AEROFLO can find the grid files.
- Other input files (initial condition files and boundary condition files) must be put in the project sub-folder.
- The AEROFLO executable command *aeroflo.exe* (or *mpiaeroflo.exe* for parallel calculation) should be in the directory of the project file. You can do this by adding the AEROFLO installation directory to the **PATH** system environment variable, or simply by copying the command *aeroflo.exe* (or *mpiaeroflo.exe*) into the directory of the project file.

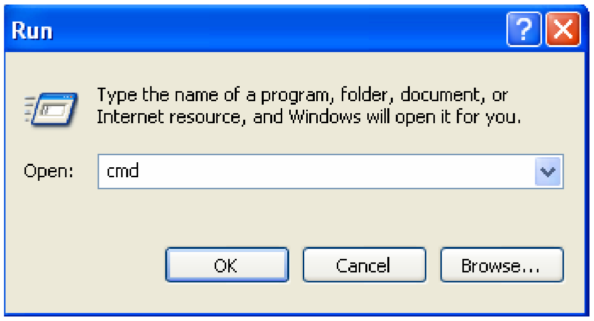
The following steps are used to run a program in command-line.

<u>WINDOWS</u>:

1. From the “Start” menu, select “Run…”

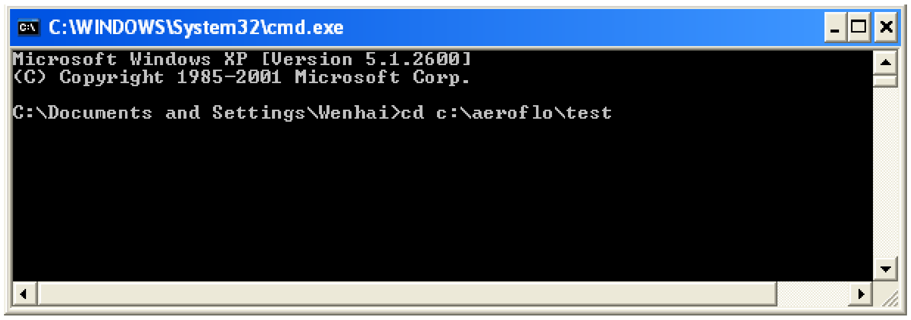


2. Type “cmd,” then click “OK” to prompt the command environment.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

```
cmd
```

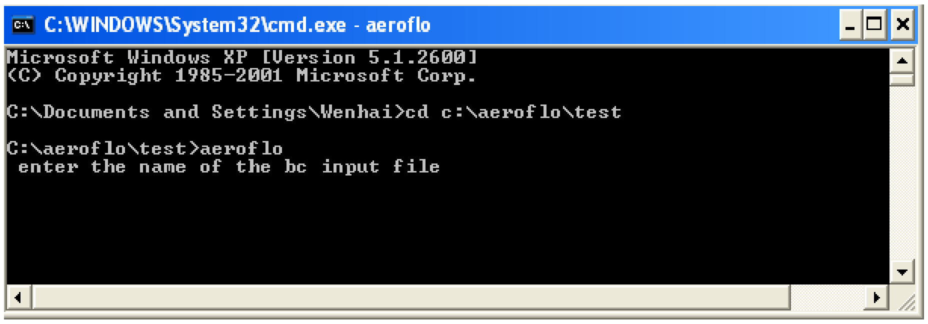
3. Change the directory to the folder where the project file resides. For the sample problem created in Chapter 5, type “cd c:\aeroflo\test,” where “c:\aeroflo\test” represents the directory of the project file cdvduct.afl.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Wenhai>cd c:\aeroflo\test
```

4. Run the code by typing “aeroflo,” and the program will request an input file.

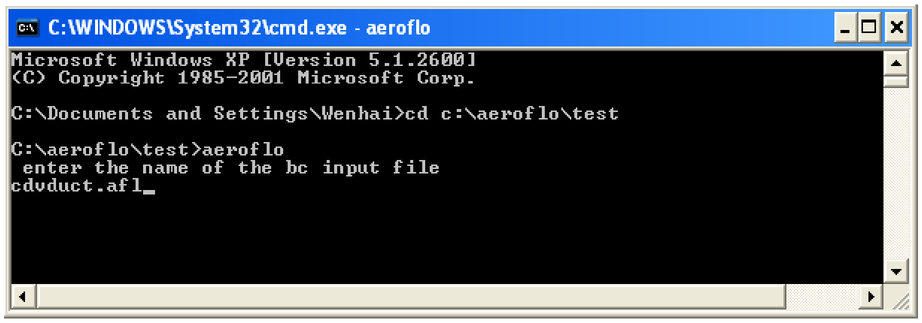


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Wenhai>cd c:\aeroflo\test

C:\aeroflo\test>aeroflo
 enter the name of the bc input file
```

5. Type in a project file name (“cdvduct.afl”).

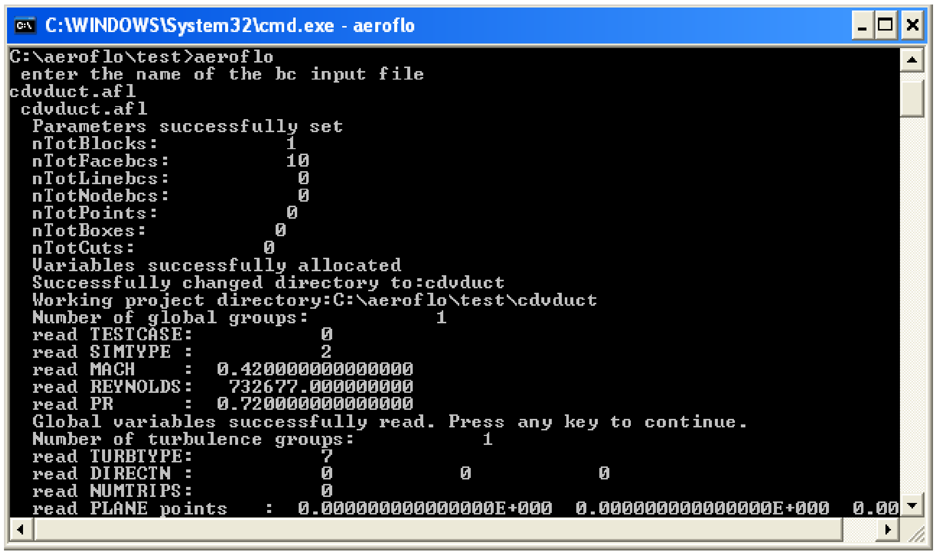


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Wenhai>cd c:\aeroflo\test

C:\aeroflo\test>aeroflo
 enter the name of the bc input file
cdvduct.afl_
```

6. The program reads the input files and begins to run the calculation. The log information of the calculation is output to the screen.



```
C:\aeroflo\test>aeroflo
 enter the name of the bc input file
cdvduct.afl
 cdvduct.afl
  Parameters successfully set
  nTotBlocks:           1
  nTotFacebcs:          10
  nTotLinebcs:           0
  nTotNodebcs:           0
  nTotPoints:           0
  nTotBoxes:           0
  nTotCuts:          0
  Variables successfully allocated
  Successfully changed directory to:cdvduct
  Working project directory:C:\aeroflo\test\cdvduct
  Number of global groups:            1
  read TESTCASE:           0
  read SIMTYPE :           2
  read MACH     :  0.420000000000000
  read REYNOLDS:   732677.000000000
  read PR       :  0.720000000000000
  Global variables successfully read. Press any key to continue.
  Number of turbulence groups:            1
  read TURBTYPE:           7
  read DIRECTN :           0           0           0
  read NUMTRIPS:           0
  read PLANE points    :  0.000000000000000E+000  0.000000000000000E+000  0.00
```

You may also use the input/output redirecting operator ("<" and ">" operators) to assign the project file to aeroflo.exe directly and to output the log information to an output file. For example, by typing "aeroflo.exe < cdvduct.afl" in Step 3, AEROFLO will run directly and the file name inputting step (step 4) will be omitted. Moreover, if you do not want to print the log information on the screen and want to save it in a text file, say, output.txt, you only need to type the execution command by "aeroflo.exe < cdvduct.afl > output.txt" in Step 3.

UNIX:

1. Open a UNIX shell.
2. Change the directory to the project folder.
3. Run the code by typing

   mpirun –np **n** *mpiaeroflo < project_file_name*

7.2 Restarting from a Previous Calculation

In Chapter 5, we introduced how to pause and resume an AEROFLO calculation in the GUI. The restart calculation can also be carried out in command-line mode. The restart calculation is possible because the AEROFLO output files contain the complete flow field information at the moment of outputting. Therefore, these files can be used to restart a terminated calculation. There are two kinds of files that can be used to restart the previous calculation: the CGNS output file *myres.cgns* or the AEROFLO output files "BLOCKNAME".tmp. To restart a calculation from these files, you simply need to modify your project file, then run the problem using the modified project file. The methods of using these two kinds of files to restart a calculation are described below.

- <u>Restarting from the CGNS output file</u>:
  1. Change the input meshfile for all the blocks to the single output CGNS file (myres.cgns).
  2. Change the filetype for each block to 'CGNS.'
- <u>Restarting from the AEROFLO output files</u>:
  1. Change the input meshfile for each block to the output AEROFLO file for each block.
  2. Change the filetype for each block to 'AEROFLO.'

7.3 Example of Modifying and Restarting a Calculation

If the user wants to restart the previous calculation, but also wishes to change the models, schemes, or other input parameters, the user can first do the corresponding change in the project file, and then follow the steps in Section 7.2 to restart the calculation. Here, we still use the converging-diverging duct flow problem as an example to see how to modify a project file and restart a previous calculation. Assume that you have finished the previous calculation (10,000 steps) and you wish to do the following changes to obtain better results:

- Change the turbulence model to Spalart-Allmaras
- Change the spatial scheme to WENO scheme
- Change the time step size to 0.005
- Change the output frequency to 200 steps
- Recalculate for 5000 steps

To implement the above changes, you can change the project file by using the GUI. Here, we do this by directly editing the project file. To do this, use the following steps:

1. Open the project file by using any text editor software.
2. Under the **$TURB** group keyword, change the value of the **TURBTYPE** keyword to 5 (represents Spalart-Allmaras model), and delete the **TUVALUE** and **RESET** keywords (these keywords are only meaningful for k-ε type turbulence model).
3. Under the **$SPATIAL** group keyword, change the values of the **SCHEME** keyword to *33, 33, -999* (33 represents the WENO scheme). Also change the value of the **METRIC** keyword to *13* (13 represents the WENO metric differencing scheme). Delete the **CUTOFFI**,

ISOTROPI, **ICUT**, **CUTOFFJ**, **ISOTROPJ**, and **JCUT** keywords, as they are only meaningful for Roe schemes.

4. Under the **$TIMESTEP** group keyword, change the value of the **MAXITER** keyword to *5000* and the value of the **GLOBALDT** keyword to *0.005*.
5. Under the **$OUTPUT** group keyword, change the value of the **PRINTFQ** to 200.
6. There are two ways to complete this step. Choose one of them:
   (a). Under the **$BLOCK** group keyword, change the value of the **MESHFILE** keyword to 'myres.cgns' and the value of the **FILETYPE** keyword to 'CGNS.'
   (b). Under the **$BLOCK** group keyword, change the value of the **MESHFILE** keyword to 'mesh0.tmp' and change the value of the **FILETYPE** keyword to 'AEROFLO.'
7. Save the file and check to make sure that either myres.cgns or mesh0.tmp is in the subfolder. Using the steps in Section 7.1 to run the project file in command-line.
8. The program will read the restart files (the modified project file and the restart block data myres.cgns or mesh0.tmp) to restart the calculation.

Figure 7.1 shows the modified project file. You can compare it with Figure 6.1 to see where the file is modified. The simulation result is shown in Figure 7.2.

```
cdvduct.afl

$GLOBAL,
       TITLE = 'Converging/Diverging Duct',
       FOLDER = 'cdvduct',
       SIMTYPE =            2 ,
       MACH =   0.4600000         ,
       REYNOLDS =    732677.0      ,
       PR =   0.7200000         ,
       ORDER =            2 ,
       BCSTYLE = 2,
     $END

$OVERSET,
     $END

$TURB,
       TURBTYPE =            5 ,
     $END
```

Change TURBTYPE to 5 (Spalart-Allmaras) and delete TUVALUE and RESET.

Change Turbulence Model

```
$SPATIAL,
       SCHEME =           33 ,          33 ,         -999 ,
       METRIC =           13 ,
       VISCOUS =           1 ,           1 ,            1 ,
       VISCHEME =           0 ,
       VISCMON =           1 ,
       SUTHERLN =   0.3800000         ,
     $END
```

Change SCHEME to 33 (WENO) for I, J directions and METRIC to 13 (WENO). Delete CUTOFFI, ISOTROPI, ICUT, CUTTOFFJ, ISOTROPJ, and JCUT.

Change Spatial Scheme

```
$TIMESTEP,
       SCHEME = 'BW2',
       MAXITER =         5000 ,
       GLOBALDT =   0.005 ,
       SUBDT =   9.9999998E-03  ,
       GLOBTOL =   1.0000000E-30  ,
       SUBTOL =   1.0000000E-03  ,
     $END
```

Change MAXITER to 5000 and GLOBALDT to 0.005

Change Time Steps

```
$DAMPING,
       TYPE =            3 ,
       ES2 =   0.2000000         ,
       ES4 =   9.9999998E-03  ,
       FES2 =    1.000000        ,
       FES4 =    2.000000        ,
       MAXRED =            1 ,
       OMGAV =   9.9999998E-03  ,
       SRCONST =   0.0000000E+00  ,
       FILTER =            0 ,            0 ,            0 ,
       NFILTER =            1 ,            1 ,            0 ,
       ALPHAN =   0.3000000      ,  0.3000000      ,  0.3000000      ,
       ALPHA1 =   0.0000000E+00  ,  0.0000000E+00  ,  0.0000000E+00  ,
       ALPHA2 =   0.3000000      ,  0.3000000      ,  0.3000000      ,
       ALPHA3 =   0.3000000      ,  0.3000000      ,  0.3000000      ,
       ALPHA4 =   0.3000000      ,  0.3000000      ,  0.3000000      ,
       ALPHA5 =   0.3000000      ,  0.3000000      ,  0.3000000      ,
       ORDERN =           10 ,           10 ,           10 ,
       ORDER1 =            0 ,            0 ,            0 ,
       ORDER2 =            2 ,            2 ,            2 ,
       ORDER3 =            2 ,            2 ,            2 ,
       ORDER4 =            2 ,            2 ,            2 ,
       ORDER5 =            2 ,            2 ,            2 ,
     $END

$OUTPUT,
       PRINTFRQ =          200 ,
       FORMAT = 'TECPLOT',
     $END
```

Change PRINTFRQ to 200

Change Output Frequency

```
$DEBUG,
       POINT =            2 ,            2 ,            2 ,
```

Figure 7.1. Revised Project File for Flow Through Converging-Diverging Duct.

```
     $END
 $BLOCK,
        NAME = 'mesh0',
        MESHFILE = 'myres.cgns',
        FILETYPE = 'CGNS',
        ICFILE = 'NULL',
        PERIODIC =            0 ,          0 ,           3 ,
        ISIZE =           81 ,
        JSIZE =           51 ,
        KSIZE =            1 ,
        ORDER =            2 ,
        ANGLE =   0.0000000E+00  ,
        DISTANCE =   0.0000000E+00  ,
        NOCROSS =            0 ,
     $END
```

or MESHFILE = 'mesh0.tmp'
FILETYPE = 'AEROFLO'

Change MESHFILE to 'myres.cgns' and FILETYPE to 'CGNS', or change MESHFILE to 'mesh0.tmp' and FILETYPE to 'AEROFLO'

Change Block Data to Restart Files

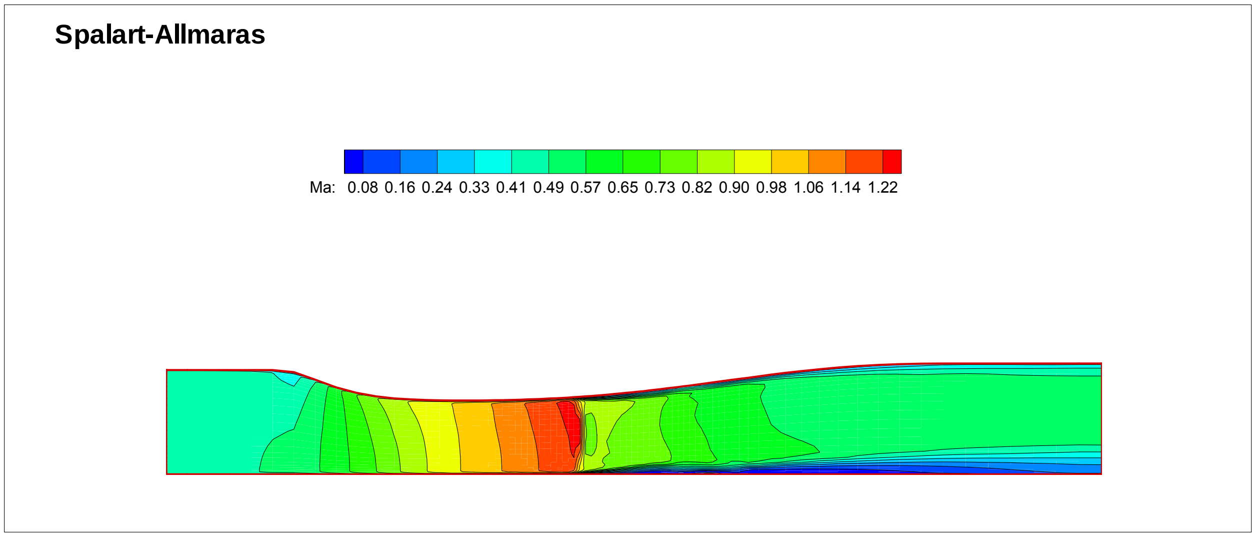
```
 COMMENT: BC            1
 $FACEBC,
        BLOCK = 'mesh0',
        SIDE =            1 ,
        TYPE =           10 ,
        VARIABLE =            2 ,
        VALUE =    1.000000       ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
     $END
 COMMENT: BC            2
 $FACEBC,
        BLOCK = 'mesh0',
        SIDE =            1 ,
        TYPE =           10 ,
        VARIABLE =            3 ,
        VALUE =   0.0000000E+00  ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
     $END
 COMMENT: BC            3
 $FACEBC,
        BLOCK = 'mesh0',
        SIDE =            1 ,
        TYPE =           10 ,
        VARIABLE =            1 ,
        VALUE =    1.000000       ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
     $END
 COMMENT: BC            4
 $FACEBC,
        BLOCK = 'mesh0',
        SIDE =            1 ,
        TYPE =           10 ,
        VARIABLE =            5 ,
        VALUE =    3.375640       ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
     $END
 COMMENT: BC            5
 $FACEBC,
        BLOCK = 'mesh0',
        SIDE =           -1 ,
        TYPE =           23 ,
        VARIABLE =            2 ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
```

Figure 7.1. Revised Project File for Flow Through Converging-Diverging Duct (Cont.).

```
     $END
COMMENT: BC            6
$FACEBC,
       BLOCK = 'mesh0',
       SIDE =          -1 ,
       TYPE =          23 ,
       VARIABLE =           3 ,
       DEPTH =            2 ,
       ISPLANE =            0 ,
       ISNORMAL =            1 ,
     $END
COMMENT: BC            7
$FACEBC,
       BLOCK = 'mesh0',
       SIDE =          -1 ,
       TYPE =          23 ,
       VARIABLE =           1 ,
       DEPTH =            2 ,
       ISPLANE =            0 ,
       ISNORMAL =            1 ,
     $END
COMMENT: BC            8
$FACEBC,
       BLOCK = 'mesh0',
       SIDE =          -1 ,
       TYPE =          10 ,
       VARIABLE =           5 ,
       VALUE =    3.263250      ,
       DEPTH =            2 ,
       ISPLANE =            0 ,
       ISNORMAL =            1 ,
     $END
COMMENT: BC            9
$FACEBC,
       BLOCK = 'mesh0',
       SIDE =           2 ,
       TYPE =           1 ,
       DEPTH =            2 ,
       ISPLANE =            0 ,
       ISNORMAL =            1 ,
     $END
COMMENT: BC           10
$FACEBC,
       BLOCK = 'mesh0',
       SIDE =          -2 ,
       TYPE =           1 ,
       DEPTH =            2 ,
       ISPLANE =            0 ,
       ISNORMAL =            1 ,
     $END

$INITIAL,
       VARIABLE =           2 ,
       VALUE =    1.000000      ,
     $END
$INITIAL,
       VARIABLE =           2 ,
       VALUE =   0.0000000E+00  ,
     $END
$INITIAL,
       VARIABLE =           1 ,
       VALUE =    1.000000      ,
     $END
$INITIAL,
       VARIABLE =           5 ,
       VALUE =    3.375640      ,
     $END
```

Figure 7.1. Revised Project File for Flow Through Converging-Diverging Duct (Cont.).



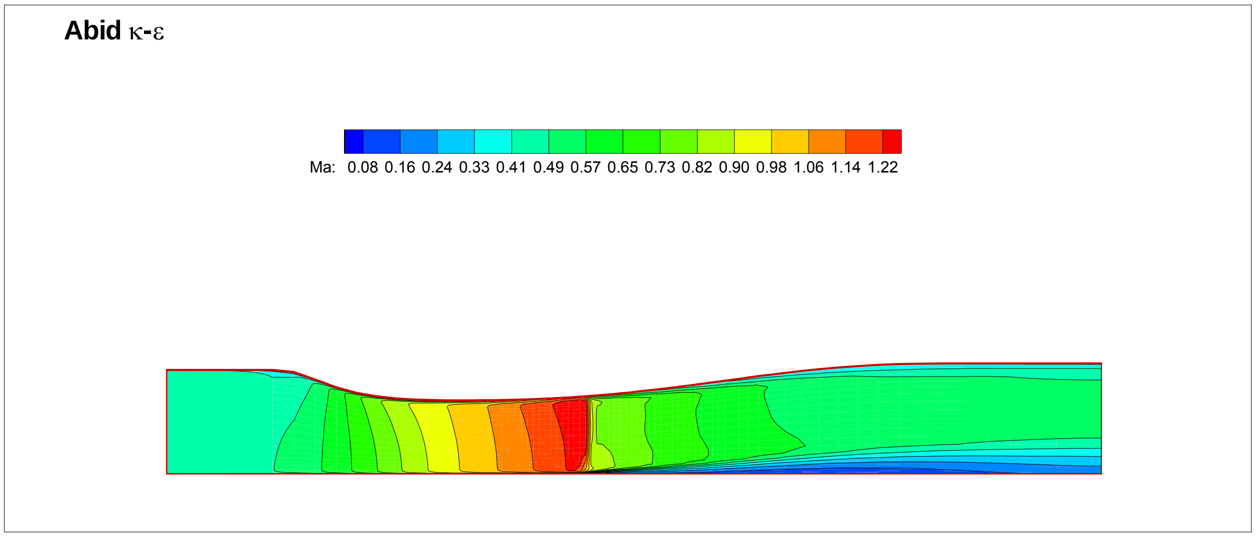


Figure 7.2. The Mach Number Contour for the Modified Converging-Diverging Duct Flow Compared with Original Results.

Discussion:

- To restart a calculation, the user sets the previous calculation results to be the initial condition of a new calculation.
- These previously-calculated results can be found in myres.cgns in the CGNS format, or in ‘meshname’.tmp in the AEROFLO format.
- These files can be loaded as mesh files for each of the blocks. Simply change the mesh file name and type in the project file.
- For multi-block problems, AEROFLO outputs only one myres.cgns file, which contains the information for all of the blocks. To restart from this file, simply let all of the blocks load the mesh files from this file.
- For multi-block problems, AEROFLO outputs the ‘meshname’.tmp files for every block. To restart from these files, let every block load its own *.tmp* file.

Note:

- If you look at the revised project file, you will find that the last part of the file is still set with the global initial conditions. In AEROFLO, if the initial condition data is loaded, it will automatically overwrite the global initial conditions. Therefore, the global initial condition settings in this project file do not take effect. However, you may also delete them if you want the project file to be cleaner.

8. Prepare AEROFLO Mesh File

In this chapter, we will introduce the requirement for the AEROFLO mesh file and the details of each kind of file format.

8.1 General Requirements

The grid must be provided by the user, and only structured grids are supported by AEROFLO. The user can generate grids by using grid-generation software (AEROFLO does not generate grids). High-quality grids will make your calculation faster and more accurate, while low-quality grids may cause the calculation to diverge or even cause unexpected errors in AEROFLO. High-quality grids always require that:

- The grid not be too loose for the problem.
- The grid be fine enough in the region where the flow quantities have a high gradient.
- The grid spacing vary smoothly.
- The grid stretching not be too severe.

AEROFLO solves CFD problems in a block-by-block fashion. In other words, a CFD problem will consist of several structured blocks containing grid points. The grids must be generated separately from AEROFLO. Each block that makes up part of a CFD problem must be described in the AEROFLO input file. The keyword for describing a block is the BLOCK command. The BLOCK command may be used several times in an input file – as many times as blocks that make up the project. Each block description includes a name for the block, a grid file (and path to the grid file), a file type specification, an initial conditions file, size of the grid, and periodic specification for the grid.

8.2 Grid Format

AEROFLO supports three kinds of grid formats: CGNS, PLOT3D, and a native AEROFLO format. Note that these formats support both grid files and solution files. In solution files, both the grid information and the solution variable quantities are written in a whole file. Therefore, these formats are also used for the output files and initial condition files in AEROFLO. Detailed information about each grid format is given below:

CGNS File Format:

The CGNS format was developed by the CFD General Notation Systems group, an organization of CFD software developers. The format consists of a description for recording grid and solution data, as well as boundary conditions data. As a result, a CGNS input file may contain both grid and initial conditions for a block. The CGNS format is supported by most grid-generation software. The layout of AEROFLO data in a CGNS file is shown in Figure 8.1.

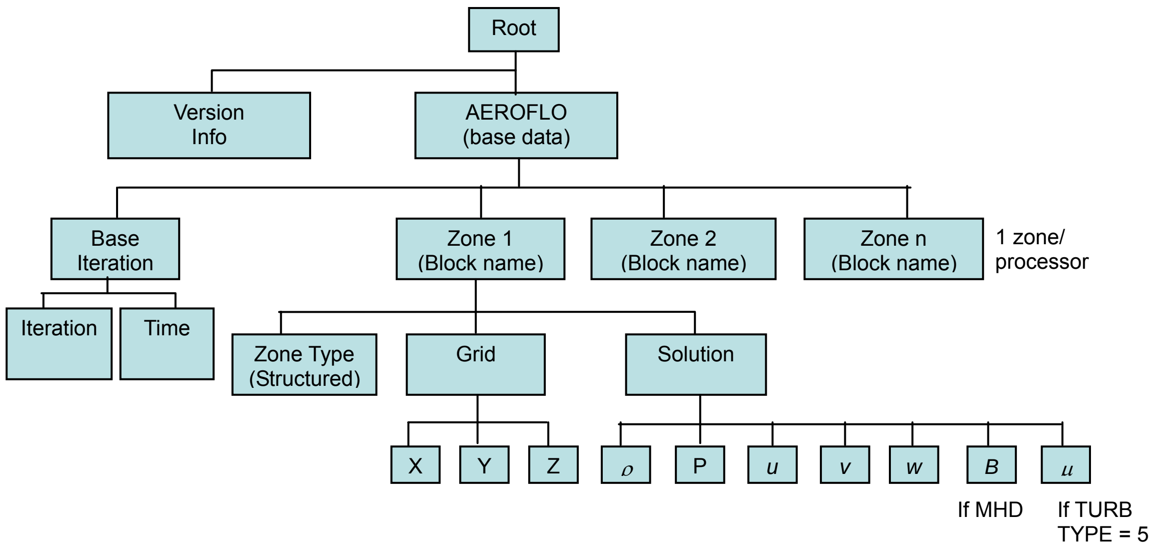


Figure 8.1. The Layout of AEROFLO Data in a CGNS File Format.

More information can be obtained at http://www.cgns.org/WhatIsCGNS.html. Software for viewing or converting CGNS files can be downloaded at http://sourceforge.net/projects/cgns.

PLOT3D File Format

The PLOT3D format is used to convey three-dimensional structured grid information. Exporting meshes in the PLOT3D format is supported by most grid-generation software. AEROFLO only supports PLOT3D files in an ASCII (formatted) and multi-block format. The data type in the PLOT3D file can be single or double precision. For PLOT3D grid files, the format is

```
nblock (number of zones in files)
(zone 1)   i_dim, j_dim, k_dim
(zone 2)   i_dim, j_dim, k_dim
...
(zone n)   i_dim, j_dim, k_dim
(zone 1)   all x values
           all y values
           all z values
(zone 2)   all x values
           all y values
```

```
                all z values
…
(zone n)        all x values
                all y values
                all z values
```

For PLOT3D solution files (Q files), the format is

```
nblock (number of zones in files)
(zone 1)   i_dim, j_dim, k_dim
(zone 2)   i_dim, j_dim, k_dim
...
(zone n)   i_dim, j_dim, k_dim
(zone 1)   mach (freestream Mach number)
           alpha (freestream angle of attack)
           re (freestream Reynolds number)
           time (time)
           all u values
           all v values
           all w values
           all P values
           all rho values
(zone 2)   mach (freestream Mach number)
           alpha (freestream angle of attack)
           re (freestream Reynolds number)
           time (time)
           all u values
           all v values
           all w values
           all P values
           all rho values
…
(zone n)   mach (freestream Mach number)
           alpha (freestream angle of attack)
           re (freestream Reynolds number)
           time (time)
           all u values
           all v values
           all w values
           all P values
           all rho values
```

AEROFLO File Format:

The AEROFLO data format is an unformatted binary file containing both the mesh and the solution variables. As a result, it can be used to provide the mesh data, as well as the initial conditions data. The format (including both grid and solution data) is shown below:

```
If (Turbulence flow with Sparlart-Allmaras model)
            IL,JL,KL,ITERATION,TIME,
            X
            Y
            Z
            U
            V
            W
            P
            RHO
            SP
If (Turbulence flow with κ-ε model)
            IL,JL,KL,ITERATION,TIME,
            X
            Y
            Z
            U
            V
            W
            P
            RHO
            k
            ε
Else
            IL,JL,KL,ITERATION,TIME
            X
            Y
            Z
            U
            V
            W
            P
            RHO
End if
```

For MHD simulation, the format is

```
      IL,JL,KL,ITERATION,TIME
      X
      Y
      Z
      U
      V
      W
      P
      RHO
      BFIELD
```

IL, JL, KL – integers representing the grid size in i, j, and k

ITERATION – current iteration step (0 for a new problem)
TIME – current time (0.0 for a new problem)
X, Y, Z – three-dimensional arrays exactly of size IL, JL, KL
U, V, W, P, RHO – three-dimensional arrays exactly of size IL, JL, KL for the u-velocity, v-velocity, w-velocity, pressure, and density variables.
SP, *k*, *ε* – three-dimensional arrays exactly of size IL, JL, KL for the Spalart-Allmaras variable, *k*, and *ε* for k-e turbulence models, respectively.
BFIELD – four-dimensional arrays exactly of size IL, JL, KL, 3 for the MHD B-field variables.

A FORTRAN 90 segment for writing a grid in the AEROFLO format for a CFD (Navier-Stokes) calculation is shown below:

```
  REAL, DIMENSION(IL,JL,KL) :: X,Y,Z,U,V,W,RHO,P

! Generate the mesh of size IL, JL, KL

  OPEN(2,FILE='test.in',FORM='UNFORMATTED', status='REPLACE')
!
  TINIT = 0.0
  WRITE(2) IL,JL,KL,0,TINIT
  WRITE(2) X
  WRITE(2) Y
  WRITE(2) Z
  WRITE(2) U
  WRITE(2) V
  WRITE(2) W
  WRITE(2) P
  WRITE(2) RHO
!
  CLOSE(2)
```

8.3 Right-Hand Rule

In AEROFLO, the right-hand rule must be obeyed by the grid coordinates *(x, y, z)* and the grid indices *(i, j, k)*. The rule can be explained as following:

Imagine pointing the right-hand index finger along the positive *x* (or *i*) axis.
Then curl the rest of the right-hand fingers toward the positive *y* (or *j*) axis.
The thumb, pointing straight out, is now in the positive *z* (or *k*) direction.

Figure 8.2 shows an example of the right-hand system.

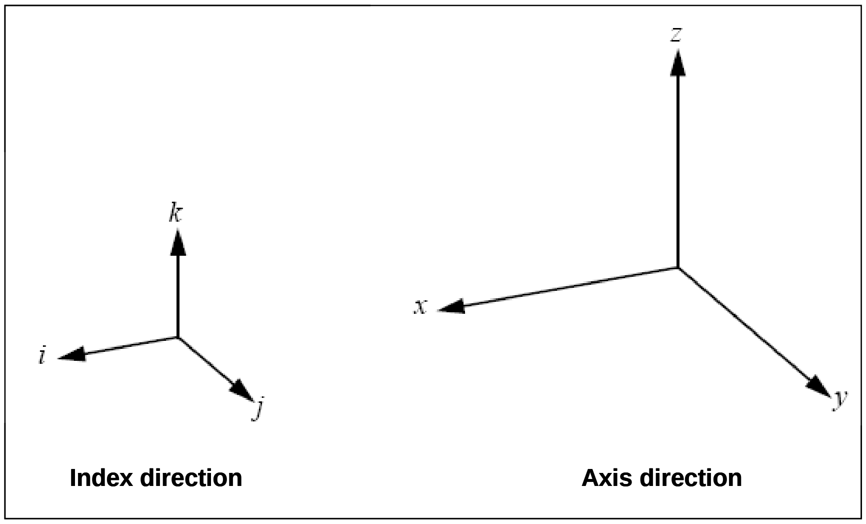


Figure 8.2. Right-Hand System.

In AEROFLO, the user can load 2D grids. However, all 2D grids will be converted to 3D grids inside AEROFLO by extruding the grids in the positive *z* direction. That means the *k* indices direction is fixed to positive *z* direction for a 2D grids. Therefore, the 2D grids still need to follow the right-hand rule. Figure 8.4 shows example 2D grids that either obey or violate the right-hand rule in AEROFLO.

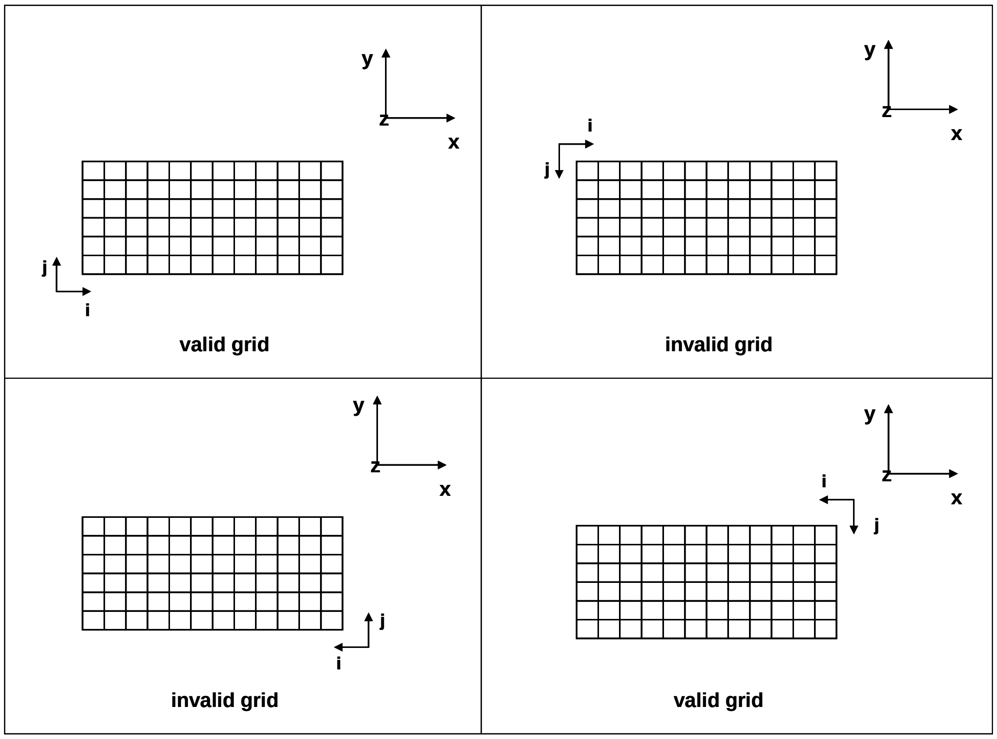


Figure 8.4. Examples of Valid and Invalid 2D Grids.

9. AEROFLO Output

In this chapter, detailed information about the AEROFLO output files is presented.

9.1 Result Files

The program generates output files that are controlled by the **$OUTPUT** group keyword. The output frequency and format are specified by **PRINTFRQ** and **FORMAT**. Three output formats are supported: CGNS, TECPLOT, and PLOT3D. Both CGNS and PLOT3D formats were introduced in the last chapter. The TECPLOT format is supported by TECPLOT, a CFD post-processing and results-plotting software developed by Amtec Engineering, Inc. The TECPLOT format output files generated by AEROFLO can be directly read by TECPLOT without any modifications.

AEROFLO generates the following overall flow field results files:

- A CGNS result output/restart file **myres.cgns**, no matter which output format is selected.
- An optional TECPLOT result output file **myres.dat**, if the output format is specified as TECPLOT.
- Two optional PLOT3D result output files **myres.PLOT3D** (grid file) and **myres.PLOT3Dq** (Q solution file), if the output format is specified as PLOT3D.
- A result output/restart file **"blockname".tmp** for each block with AEROFLO format.

In these files, the output variables include:

- x, y, z, P, Ω, ρ, u, v, and w values for general CFD problems
- additional μ, κ, ε (for κ-ε model), and ω (for κ- ω model) values for turbulence flows
- additional B values for MHD calculation,

where Ω is the vorticity, μ is the eddy viscosity, k is the turbulence kinetic energy, ε is the turbulence kinetic energy dissipation rate, ω is the turbulence vorticity, and B is the B-field for the MHD simulation.

There are other auxiliary result files that include additional information for the flow results:

- An optional TECPLOT result file **Cp.dat** containing the pressure *P* and pressure coefficient *Cp* data for the solid wall boundaries if the output format is specified as TECPLOT
- A **rnorm.dat** file that records the iteration norm for each time step. This can be used to check if the calculation is converged.

AEROFLO also allows the generation of the animation results, which include many intermittent results outputted during the calculation. It is controlled by the keywords **TECANIM** and **TRANGE** under the **$OUTPUT** group keyword. For example, the following commands in the project file will generate the intermittent result files every 200 steps when the time steps are between 0 and 30000:

```
$OUTPUT,
      PRINTFRQ = 100,
      FORMAT = 'TECPLOT',
      TECANIM = 1,
      TRANGE = 0, 30000, 2,
      $END
```

The intermittent result files include:

- The CGNS intermittent result files with names **res1-***.cgns**, where *** represents the block number between 000 and 999.
- The intermittent TECPLOT result files with names **res1-***.dat** if the output format is TECPLOT.
- The intermittent Cp data files with names **Cp1-***.dat** if the output format is TECPLOT.
- The intermittent PLOT3D Q result files with names **res1-***.3dq** if the output format is PLOT3D.

For multi-block calculations, there are additional files that are outputted for the overset calculation for diagnostic purpose:

- A TECPLOT format mesh file **resfe.dat**, which highlights the overset nodes
- A TECPLOT format mesh file **resorphans.dat**, which highlights the orphan nodes.
- A TECPLOT format result file **reselements.dat**, which is in a cell-based format. This will be useful when the user does not want to see the result in the hole/blanked region.

- A report file **igreport.dat**, which lists the status of overset nodes.
- **metr*** records the metrics for each block, where * represents the block number.
- **start.txt** and **end.txt**, which are created when the computation starts and ends, respectively.

There are also other binary files that contain the additional overset calculation information:

- **blnk*** records the overset information for the hole/blank regions
- **dono*** records the overset information for donors
- **rece*** records the overset information for receivers.

These files are useful for the restart calculation.

9.2 Log Output

At the beginning of the calculation, the program will read the input files and these input parameters will be outputted on the screen for diagnostic purposes. If there is an error detected in the project file input parameters, the calculation is terminated and the corresponding error information will be outputted. After the solver begins to run, the program will output the iteration time step, overall physical time, and iteration norms of the iterations and sub-iterations to let the user know the progress of the computation.

10. Specifying Initial Conditions

In AEROFLO, the user can specify initial conditions by an initial condition data file or through global initial conditions specified in the main input file. In addition, the user can simply accept the default initial conditions.

10.1 Default Initial Conditions

If no initial conditions are applied to AEROFLO calculations, AEROFLO imposes the following initial conditions on the primary variables:

$$\rho = 1.0$$
$$u = 0$$
$$v = 0$$
$$w = 0$$
$$P = \frac{1}{\gamma M^2}$$

10.2 Global Initial Conditions in the Project File

Initial conditions may be applied globally on primary variables using the **$INITIAL** group keyword anywhere in the input file. The **$INITIAL** keyword has two sub-keywords, **VARIABLE** and **VALUE**. The **VARIABLE** sub-keyword accepts integer values and is the same as those for the boundary conditions commands as reproduced below. The **VALUE** sub-keyword accepts real values. The primary variable specified by **VARIABLE** is initially set to the value provided in **VALUE** prior to the start of the calculations.

	Variable	
	Density	1
	u - velocity	2
	v - velocity	3
	w - velocity	4
	Pressure	5

The following commands present in the main input file, for example, will set the initial values.

```
$INITIAL,
        VARIABLE = 1,
        VALUE = 1.0,
        $END

$INITIAL,
        VARIABLE = 2,
        VALUE = 0.9961947,
        $END

$INITIAL,
        VARIABLE = 3,
        VALUE = 0.0871557,
        $END
```

This is equivalent to the following initial conditions:

$$\rho = 1.0$$
$$u = 0.9961947$$
$$v = 0.0871557$$
$$w = 0$$
$$P = \frac{1}{\gamma M^2}$$

As shown above, the values of *w*, and *P*, remain the same as the default AEROFLO initial values.

10.3 Initial Conditions Input File

An initial condition file may be supplied with each block and specified with the block description. The command to specify an initial condition file with a block description is **$BLOCK/ICFILE**. **ICFILE** will contain a character value specifying the path to an initial conditions data file for the block. The initial condition file must contain values for all grid points of the block (specified by the block size).

The initial condition file must be an unformatted binary file with data layout as shown below:

General CFD

```
IL,JL,KL,ITERATION,TIME,U,V,W,P,RHO
```

Turbulence Flow (Spalart-Allmaras)

```
IL,JL,KL,ITERATION,TIME,U,V,W,P,RHO,SP
```

MHD

```
IL,JL,KL,ITERATION,TIME,U,V,W,P,RHO,BFIELD
```

IL, JL, KL – integers representing the grid size in *i*, *j*, and *k*
ITERATION – current iteration step (0 for a new problem)
TIME – current time (0.0 for a new problem)
U, V, W, P, RHO – three-dimensional arrays exactly of size IL, JL, KL for the u-velocity, v-velocity, w-velocity, pressure, and density variables.
SP – three-dimensional arrays exactly of size IL, JL, KL for the Spalart-Allmaras variable
BFIELD – four-dimensional arrays exactly of size IL, JL, KL, 3 for the MHD B-field variables.

A FORTRAN 90 segment for writing a grid in the AEROFLO format for a CFD (Navier-Stokes) calculation is shown below:

```
  REAL, DIMENSION(IL,JL,KL) :: U,V,W,RHO,P

! Generate the mesh of size IL, JL, KL

  OPEN(2,FILE='test.in', FORM='UNFORMATTED', status='REPLACE')
!
  TINIT = 0.0
  WRITE(2) IL,JL,KL,0,TINIT, U,V,W,P,RHO
!
  CLOSE(2)
```

The commands to load this file for initial conditions in the main input file are

```
$BLOCK,
    NAME = ‘mesh0’
    MESHFILE = ‘grid.dat’,
    FILETYPE = ‘PLOT3D’,
    ICFILE = ‘test.in’,
    …
    $END
```

Note: **The specifications of an initial condition file supersede the global initial conditions specified in the main input file.**

10.4 Setting the Initial Conditions in Block Data Files

As we mentioned in Chapter 7, the restart file is a special block data file that contains the previous calculation results. This can be extended to the general cases that the user can write a block data file, which includes specified flow initial conditions for each of mesh node points. The command to specify this file with a block description is **$BLOCK/MESHFILE**. There is no need to use the **ICFILE** keyword in this case anymore.

A FORTRAN 90 segment to write a grid in the AEROFLO format for a CFD calculation is shown below:

```
REAL, DIMENSION(IL,JL,KL) :: X,Y.Z,U,V,W,RHO,P

! Generate the mesh of size IL, JL, KL

  OPEN(2,FILE='test.in', FORM='UNFORMATTED', status='REPLACE')
!
 TINIT = 0.0
  WRITE(2) IL,JL,KL,0,TINIT,X,Y,Z,U,V,W,P,RHO
!
  CLOSE(2)
```

Compared to the example given in the last section, the only difference is that the grid information *x, y, z* is also written to the file. The commands to load this file for initial conditions in the main input file are

```
$BLOCK,
        NAME = ‘mesh0’,
        MESHFILE = ‘test.in’,
        FILETYPE = ‘AEROFLO’,
        ICFILE = ‘NULL’,
        …
        $END
```

11. Specifying Boundary Conditions

AEROFLO solves CFD problems in a multi-block format. This means that the domain of the CFD problem may contain several structured blocks. Since the blocks are structured, each block will consist of six boundary faces. For the problem to be complete, boundary conditions must be specified on all faces of a block. The boundary conditions may range from simple boundary conditions, such as Dirichlett or Neumann, to compound boundary conditions, such as free-stream, inlet, or outflow boundary conditions. The main difference between simple and compound boundary conditions is that simple conditions act on a specific variable, while compound boundary conditions have a compound equation for all the variables at the boundary.

To apply boundary conditions on a face of a block, the **$FACEBC** group keyword is used. The **BLOCK** keyword indicates the block to which the boundary condition applies. The **SIDE** keyword indicates the face of the block to which the condition will be applied. The **TYPE** keyword specifies the type of boundary conditions. A list of the currently-supported types of boundary conditions is included below. The **VALUE** keyword specifies the value of the boundary condition. It is required for types such as Dirichlett or free stream. The **VARIABLE** keyword specifies the solution variable to apply the boundary condition on. It is required for simple boundary condition types.

The boundary condition commands may be invoked several times anywhere in the main input file.

11.1 Boundary Precedence Rules

The precedence of the boundary conditions is as follows:

23	Neumann
24	Specified flux
32	Subsonic free stream
31	Subsonic outflow
71	Slip wall
30	Subsonic inflow
1	No-ship wall
80	Equation
10	Dirichlett

41	Characteristic inflow boundary condition
42	Characteristic outflow boundary condition

Boundary conditions of the same type are simply applied in the order in which they are specified in the input file. For this reason, the user should exercise care in specifying the boundary conditions.

11.2 AEROFLO Boundary Condition Types

For all of the boundary condition equations below,

$$\phi^e = \{\text{Density, u - velocity, v - velocity, w - velocity, Pressure,...}\}$$

11.2.1 Simple Boundary Conditions

Dirichlett (Type = 10)

This applies a fixed value to the surface nodes indicated.

$$\phi_i^e = f$$

f is some real number.

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
VARIABLE
VALUE
```

Neumann (Type = 23), Flux (Type = 24)

$$\frac{\partial \phi}{\partial n} = f \big| f = 0 \text{ for type} = 23$$

f is some real number.

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
```

```
TYPE
VARIABLE
VALUE (for type = 24)
```

11.2.2 Compound Boundary Conditions

Solid Wall (Type = 1)

- Dirichlett ($f = 0.0$) for u, v, w – velocity)
- Neumann (for density, pressure)

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
```

Symmetry (Type = 11, 13, 14, 15), Slip (Type = 71)

Neumann (for density, pressure) $\frac{\partial \phi}{\partial n} = 0; \hat{n} \cdot \nabla \phi = 0$

Tangential velocity $\frac{\partial (\overline{q} \times \hat{n})}{\partial n} = 0$

Normal velocity $\overline{q} \bullet \hat{n} = 0$

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
```

Axisymmetry (Type = 60, 61, 62)

.

Inflow (Type = 30)

- Dirichlett $\hat{n} \bullet \overline{q} = f$ (for u, v, w – velocity)
- Neumann (for density, pressure)

Note that the inflow boundary condition assumes that the flow is normal to the surface. When this is not the case (e.g., the flow has an angle of attack relative to the inflow surface), use Dirichlett on each velocity variable.

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
VALUE
```

Outflow (Type = 31)

Neumann (for all variables) $\dfrac{\partial \phi}{\partial n} = 0; \hat{n} \cdot \nabla \phi = 0$

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
```

Free-Stream (Type = 32)

Neumann (for all variables) $\dfrac{\partial \phi}{\partial n} = 0; \hat{n} \cdot \nabla \phi = 0$

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
```

Characteristic Inflow/Outflow (Type – 41, 42)

The incoming and outgoing Riemann invariants are then

$$R_\infty = \mathbf{q}_\infty \cdot \mathbf{n} - \frac{2c_\infty}{\gamma - 1},$$

$$R_e = \mathbf{q}_e \cdot \mathbf{n} + \frac{2c_\infty}{\gamma - 1}.$$

From that, we have:

$$\mathbf{q} \cdot \mathbf{n} = \frac{1}{2}\left(R_e + R_\infty\right),$$

$$c = \frac{\gamma - 1}{4}\left(R_e - R_\infty\right).$$

At the outflow boundary,

$$\mathbf{q} = \mathbf{q_e} + (\mathbf{n} \cdot (\mathbf{q} - \mathbf{q_e}))\mathbf{n},$$

$$s = s_e.$$

At the inflow boundary,

$$\mathbf{q} = \mathbf{q}_\infty + (\mathbf{n} \cdot (\mathbf{q} - \mathbf{q}_\infty))\mathbf{n},$$

$$s = s_\infty.$$

The density and pressure can be calculated from *c* and *s*, which are:

$$s = \ln\left(\frac{p}{\rho^\gamma}\right),$$

$$c = \sqrt{\gamma \frac{p}{\rho}}.$$

Note that this boundary condition acts on existing values of *u, v, w, r*, and *P*. Consequently, it must be applied after either inflow or other Dirichlett conditions have set the appropriate values for those variables.

Specifying this boundary condition requires the following:

```
BLOCK
SIDE
TYPE
CNORM
```

<u>List of Boundary Condition Types</u>

BC No.	
1	Solid Wall
2	Coupling
3	Overset boundary condition
8	Coupling due to periodicity
9	C-Grid
10	Dirichlett
11	Symmetry
13	Symmetry in *i* (any plane)
14	Symmetry in *j* (any plane)
15	Symmetry in *k* (any plane)
23	Neumann (0 flux)
24	Specified flux

30	Inflow
31	Outflow
32	Free-stream
41	Characteristic Inflow B.C.
42	Characteristic Outflow B.C.
52	Initial value
59	Singular point surface (e.g. sphere grid)
60	*i*-axes (axisymmetric around I at specified face)
61	*j*-axes
62	*k*-axes
71	Slip wall
80	Equation
90	User Defined

Note: **The specifications of a boundary condition supersede the initial values at the boundary points, which are specified in the initial conditions.**

12. Turbulence Models

This chapter provides an elementary introduction to the turbulence models used in AEROFLO. Different turbulence models will be introduced and compared, with a few suggestions on the choice of an appropriate turbulence model.

12.1 Introduction of Turbulence Modeling

In turbulence, the flow fields exhibit stochastic behaviors. In this case, an instantaneous flow variable (or chemical species) can be decomposed into a mean value (time-averaged, ensemble-averaged) and a fluctuating value. The fluctuating quantities are associated with the small scales of the flow. If we want to directly resolve all the flow scales by solving the governing equations (Navier-Stokes), we would need a very fine mesh. This would cause the computation to be too expensive for realistic problems. Furthermore, the mean properties of the flow field are usually of interest in engineering applications. Therefore, the instantaneous governing equations are sometimes averaged to remove the small scales, so that the averaged equations can be solved form the mean quantities. This procedure, which is referred to as the Reynolds-averaged Navier-Stokes (RANS) equations, leads to governing equations that are less expensive to solve. However, because of the nonlinear nature of the Navier-Stokes equations, the averaged equations will always contain additional unknown terms that depend on the fluctuating components of the variables, leading to the so-called turbulence closure problem. Before closure models are introduced, we always have more unknown variables than equations. Thus, to close the averaged equations, we need to provide additional equations that can model the influence of the fluctuating quantities on the mean quantities. Many turbulence models have been developed, to different levels of fidelity and robustness.

12.2 Reynolds-Averaged Navier-Stokes (RANS)

The Reynolds-averaged Navier-Stokes (RANS) models include the following one-equation (Spalart-Allmaras) or two-equation models (k-ε, k-ω).

<u>Spalart-Allmaras Model</u>

In this model, a single-modeled transport equation is solved for the turbulent viscosity ν_t. In this case, it is not necessary to calculate another length scale using an algebraic equation. The Spalart-Allmaras model is computationally simpler than two-equation models.

Launder-Sharma k-ε model

This is a two-equation, low-Reynolds number k-ε model. The phrase "*low-Reynolds number*" refers to the fact that the local turbulence Reynolds number is small in the region close to wall due to the strong viscous effects; it does not mean that the global Reynolds number is small. In the low-Reynolds number model, the k-ε equations and the ν_t calculation are modified using certain damping functions to account for the viscous effects in the region close to the wall. With different damping functions and values of the model coefficients, there are many low-Reynolds number models. Developed by Launder and Sharma in 1974, the Launder-Sharma k-ε model is able to obtain good results for internal flows.

Abid's k-ε model

Developed by Abid in 1993, Abid's k-ε model is another low-Reynolds number model. In this model, a wall distance unit is used to construct the damping functions in order to obtain a reasonable near-wall distribution of the damping functions.

Menter's SST k-ω model

For boundary layer flows, the k-ω model has advantages, both in the treatment of the viscosity in the near-wall region and in its ability to account for the effects of pressure gradient. However, this model has difficulties in the treatment of the non-turbulent free-stream boundaries, which the k-ε model appears to handle quite well. In 1994, Menter developed a shear stress transport (SST) k-ω model to combine the k-ε and k-ω models, taking advantage, and avoiding the shortcomings, of both models. This is done by introducing a blending function which is zero close to the wall (a k-ω model) and unity far away from the wall (k-ε model).

High-Reynolds number k-ε model

As opposed to the low-Reynolds number models, the high-Reynolds number models do not focus on the viscous wall region. Instead, wall functions are introduced to avoid expensive near-wall computations. The main idea of the "wall function" approach is to apply boundary conditions some distance away from the wall, so that the turbulence-model equations do not need to be solved close to the wall.

The procedures for RANS simulation in AEROFLO are shown in Figure 12.1.

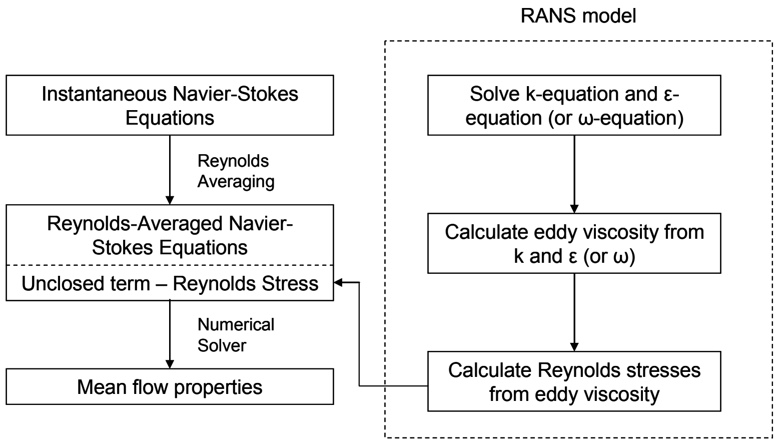

RANS model
Instantaneous Navier-Stokes Equations
Reynolds Averaging
Reynolds-Averaged Navier-Stokes Equations
Unclosed term – Reynolds Stress
Numerical Solver
Mean flow properties
Solve k-equation and ε-equation (or ω-equation)
Calculate eddy viscosity from k and ε (or ω)
Calculate Reynolds stresses from eddy viscosity

Figure 12.1. Procedures for RANS simulation in AEROFLO.

12.3 Large-Eddy Simulation (LES)

Kolmogorov's theory indicates us that the large eddies in turbulent flows are influenced by the flow geometries, and that the small-scale eddies have a universal character. Therefore, it is reasonable to resolve only the large-scale flow properties and model the small-scale eddies based on the larger ones. This method is called Large-Eddy Simulation (LES). In LES, a function is used to filter the Navier-Stokes equation. The LES procedure decomposes the primary variables into filtered components (large-scale value) and small-scale (subgrid-scale) components, with additional unclosed sub-grid scale (SGS) stress or flux terms. These SGS terms represent the influence of the small-scales on the larger scales. After the SGS terms have been introduced, the filtered equations can be solved to obtain the large-scale components of the flow.

The Smagorinsky Model

The Smagorinsky model is the simplest sub-grid scale model. The SGS stresses are calculated using an eddy-viscosity model. The approach for calculating the eddy-viscosity in LES is analogous to the mixing-length theory. However, a coefficient used in the eddy-viscosity model, C_s (Smagorinsky coefficient), is not universal to every problem. In AEROFLO, C_s is set to be 0.092, which is suitable for a wide range of flows.

The Dynamic Smagorinsky Model

In the Dynamic model, the Smagorinsky coefficient (C_s) is locally and dynamically calculated based on the information on the resolved scales. A grid filter and test filter, which have varying widths, are used, and the Smagorinsky coefficient is calculated based on the scale-invariance assumption that C_s is the same at the grid and test-filter levels (Germano's identity).

The procedures for LES simulation in AEROFLO are shown in Figure 12.2.

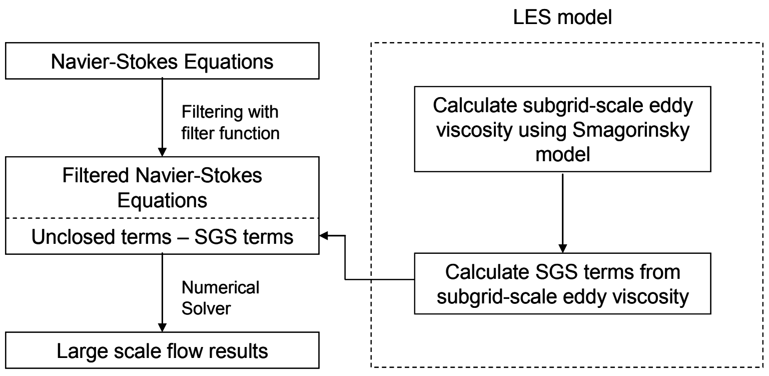

LES model
Navier-Stokes Equations
Filtering with filter function
Filtered Navier-Stokes Equations
Unclosed terms – SGS terms
Numerical Solver
Large scale flow results
Calculate subgrid-scale eddy viscosity using Smagorinsky model
Calculate SGS terms from subgrid-scale eddy viscosity

Figure 12.2. Procedures for LES simulation in AEROFLO.

12.4 Direct Numerical Simulation (DNS)

In the Direct Numerical Simulation (DNS) approach, the Navier-Stokes equations are solved without the introduction of any turbulence models. All of the scales of motion are resolved, from the smallest dissipative scale (Kolmogorov scale) up to the integral scale. DNS provides very accurate simulation results and can be used to perform numerical experiments to obtain turbulence information that may be difficult or impossible to obtain in the laboratory. To satisfy these resolution requirements, the computational mesh must be able to resolve the Kolmogorov scales, which are very tiny. This makes DNS very expensive. The computation cost of DNS sharply increases with the Reynolds number. Today, only very small Reynolds number flows can be simulated with DNS.

12.5 Hybrid Models

AEROFLO supports several hybrid RANS-LES methods, thus taking the advantages of both types of simulations: the low simulation cost of RANS and high simulation accuracy of LES.

Detached Eddy Simulation (Spalart-Allmaras):

The Detached Eddy Simulation (DES) was developed to overcome the near-wall problems occurring in LES calculations for the flow region close to the wall. This can be done by using RANS and LES for different flow regions. Regions near solid boundaries and regions where the turbulent length scale is less than the maximum grid dimension are solved using the RANS model, while regions where the turbulent length scale exceeds the grid dimension are solved using the LES model. The DES simulations in AEROFLO are based on the

Spalart-Allmaras model.

Partially Resolved Numerical Simulation (Abid κ-ε):

The Partially Resolved Numerical Simulation (PRNS) procedures are designed to provide a unified simulation strategy (from RANS to LES) for high-Reynolds number complex turbulent flows. The governing equations for the PRNS method are the temporally-filtered Navier-Stokes equations, in which the dependent variables can be construed as either the statistical mean (as in RANS), the partially resolved large-scale (as in LES), or the instantaneous (as in DNS) values of turbulence, while the effects of unresolved scales are modeled based on the size of the temporal filtering. In AEROFLO, unresolved stresses for near-wall flows are calculated using Abid's κ-ε model. This method is still under development.

12.6 Selecting Turbulence Models

AEROFLO incorporates the following turbulence models:

- Reynolds-Averaged Navier-Stokes model (RANS)
    - Spalart-Allmaras model
    - Abid's k-ε model
    - Launder-Sharma k-ε model
    - Menter's SST k-ω model
    - High Reynolds number k-ε model
- Large-Eddy simulation model (LES)
    - Smagorinsky model
    - Dynamic model
- Direct Numerical Simulation model (DNS)
- Hybrid model
    - Detached Eddy Simulation (DES): based on Spalart-Allmaras model
    - Partially Resolved Numerical Simulation (PRNS): based on Abid's k-ε model

RANS is the most popular turbulence simulation approach in the industry, due to its simplicity and relatively low computational cost. However, it is the least accurate. DNS, on the other hand, provides very accurate turbulence results since it can resolve all scales, but is only limited to very low-Reynolds number calculations because of its high computational cost. LES provides more accurate results than RANS, is also more computationally intensive, but costs less to simulate compared to DNS.

The selection of turbulence models is based on the flow characteristics, the simulation accuracy requirement, and the available computation resources. Also, different turbulence models have different requirements for the computational grids. For instance, high-accuracy turbulence models often times cannot use coarse grids.

The selection of turbulence models for RANS is also influenced by the flow boundary treatments. For example, if accurate near-wall flow structures are important, the grids close to the wall must be fine, and low-Reynolds number models are preferred. If the near-wall flow structures are not important and the grids close to the wall are coarse, it is then more appropriate to use high-Reynolds number turbulence models.

13. Numerical Schemes

This chapter is a basic introduction to the numerical differencing schemes used in AEROFLO for solving the flow and/or chemical species equations.

13.1 Introduction of Numerical Schemes

In AEROFLO, the governing differential equations are solved by finite difference in a generalized curvilinear coordinate system. Different spatial differencing and time-integration schemes are used in AEROFLO, which are summarized below.

13.2 Spatial Differencing Schemes

The parameters for the spatial schemes are set by using the **$SPATIAL** keyword. The following spatial schemes are supported by AEROFLO:

Inviscid Fluxes

The inviscid flux terms, which are the convective and pressure terms, have a hyperbolic character. This wave nature can be used to compute the flux with upwind methods.

- Roe Schemes: There are two representative upwind methods: the flux-differencing splitting method (Roe's scheme) and the flux-vector splitting method (van Leer's scheme). The Roe scheme is based on characteristic wave disturbance and, by design, can capture stationary discontinuities. It is less dissipative than the van Leer's scheme, and is therefore better for boundary layer flows. In the Roe scheme, a flux difference term is added to the central difference scheme to account for the upwind effects. To handle flow discontinuities (e.g., shock wave), a flux limiter is added to the extrapolation formulas to prevent a large gradient. There are different kinds of Roe schemes due to the different format of extrapolation formulas. AEROFLO includes the following Roe schemes:

  - First-order Roe
  - Fully-upwind second-order Roe
  - Second-order Roe scheme with Fromm reconstruction

- Third-order upwind-biased Roe scheme

- MUSCL Scheme: The MUSCL (Monotone Upstream-centered Schemes for Conservation Laws) scheme is a 2nd-order van Leer's scheme. In this scheme, no upwind information enters the reconstruction.

- WENO Scheme: The WENO (Weighted Essentially-Nonoscillatory) scheme is a high-order scheme designed for problems with piecewise smooth solutions containing discontinuities. The WENO scheme uses adaptive stencils to automatically achieve high-order accuracy and non-oscillatory properties near discontinuities. Reconstructions are done by weighting each of the possible high-order stencils (from fully upwind to fully downwind). In AEROFLO, the 5th- and 7th-order versions of the WENO scheme are implemented. The WENO scheme is used for transonic and supersonic flows. The 5th-order scheme is normally used in-house at TTC.

- Compact Scheme: The Padé Compact scheme is a high-order spatial approach for subsonic flows. It is a family of compact, high-order, central-type implicit schemes. The Compact scheme is inherently non-dissipative and more accurate compared to the central type explicit schemes, which make it particularly applicable to the simulation of waves with high frequency. However, the drawback of this scheme is that it may cause oscillations near the region of discontinuity. Therefore, the Compact scheme is only suitable for incompressible and subsonic flows. The 6th-order compact scheme is normally used in-house at TTC.

Viscous Fluxes:

The "viscous" flux terms, which are the viscous shear stress and heat flux terms, have an elliptic character. Those flux components can be computed with the central difference method. In AEROFLO, the viscous fluxes are discretized with explicit second-order central differences when the MUSCL scheme is used for the convective fluxes, and with high-order central differences when the Compact (sixth-order) or WENO (fifth-order) schemes are used for the convective fluxes.

For diagnostic and research purposes, the current version of AEROFLO enables the selection of the metric differencing scheme. In general, the selected metric and spatial schemes should be the same.

13.3 Time Integration Schemes

In AEROFLO, the following time integration schemes are used:

- Runge-Kutta Scheme: The classical fourth-order explicit Runge-Kutta (RK4) scheme is used in its low-storage form. Because of its relatively severe stability constraint (small CFL number is needed), the computational cost of RK4 is high.

- Beam-Warming Scheme: The implicit second-order Beam-Warming scheme is also supported in AEROFLO. The approximate factorization procedure is used to enhance scheme stability and make the computations go faster. Pulliam's simplified diagonalized method is used to reduce the computational costs. However, the approximate factorization and diagonalization procedures introduce errors at each time step which can degrade the temporal accuracy of the simulations. The sub-iteration method is used to remove these linearization and approximation errors. In general, the larger the time step and the grid sizes, the more the number of sub-iterations required at each time step in order to converge the sub-iteration operation.

Note that to obtain an accurate, oscillation-free, upwind-type numerical scheme, implicit or explicit artificial damping (dissipation) terms can be added to the central or forward difference schemes, which will suppress the oscillations. This is the damping (or filtering) method. In AEROFLO, the damping scheme developed by Pulliam is used for the Beam-Warming scheme. The damping parameters are set with the **$DAMPING** keyword.

The parameters for the time integration scheme are set by the **$TIMESTEP** keyword. Good time step sizes should be given by the user. If the time step size is too large, the calculation may be not able to converge and the solver may be terminated. A small time-step size leads to an expensive computational cost The selection of the time step size depends on both the flow conditions and the numerical schemes. A trial-and-error process is usually required in order to determine a good time step size.

13.4 Selecting Numerical Schemes

The selection of an appropriate numerical scheme depends on the flow conditions, accuracy requirements, and available computational resources. If a high-order calculation is preferred, WENO (5th-order) or Compact

(6th-order) spatial schemes should be used. The MUSCL scheme is appropriate for low-order (second-order) accuracy. If the flow speed is high, the WENO scheme should be used. The Compact scheme works well for subsonic flows. The Runge-Kutta method is 4th-order accurate, but is generally very expensive, and has poor stability performance. The Beam-Warming (2nd-order) method can provide fast convergence. Sub-iterations can be used to ensure time-accurate calculations. Good iteration time step sizes should be selected. If the time step size is too large, the sub-iteration calculations may not converge. If it is too small, the sub-iteration calculations may take too long to converge.

14. Parallel Computation in AEROFLO

This chapter introduces the implementation of a parallel calculation in AEROFLO.

14.1 Running in Parallel

Large problems can be solved in AEROFLO by using the parallel calculation technique. Parallel calculation in AEROFLO is done by using the domain decomposition. For multi-block problems, each block is sent to a parallel process. Aside from the use of multi-blocks, there are no other requirements for modifying grid and project files for parallel calculations. As long as MPICH (or MPIPRO) is installed on the user's parallel machines, the user can simply start parallel calculations for the multi-block problems. To run a parallel calculation, under the command-line environment, simply type:

```
mpirun –np n mpiaeroflo    < project_file_name
```

where **n** is the number of processes. Before you run this command, make sure your directory is where the project file is located, and that both the parallel AEROFLO solver *mpiaeroflo.exe* and the MPICH launch program *mpirun.exe* can be found by your operating system in this directory (If not, you can simply copy these two commands from their installation directories to your project directory, or add their directories to the **PATH** system environment variable).

Note:

- In AEROFLO, multi-block problems must be run in parallel (mpiaeroflo.exe).
- Parallel calculations do not have to be run on a parallel machine. For a single PC, if MPICH is installed, MPICH will automatically simulate the parallel environment, which allows you to run multi-block problems using more than one process.
- To restart a parallel calculation, the same number of processes is required.

14.2 An Example of a Parallel Calculation in AEROFLO

We will still use the flow through converging-diverging duct problem for a sample parallel calculation in AEROFLO. In this case, the grid we used in Chapter 5 is split into 2 blocks. The sizes of the both blocks are 41x51. The 2-block grids are shown in Figure 14.1.

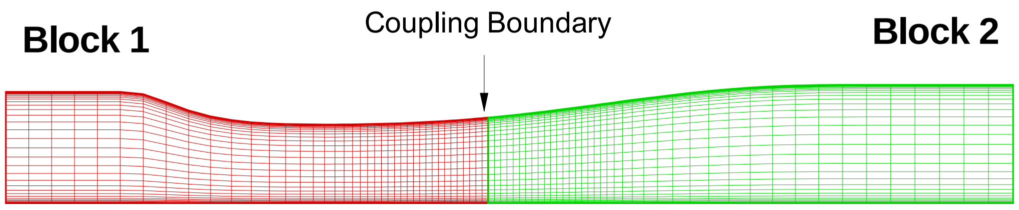


Figure 14.1. Computational Grid for 2-Block Converging-Diverging Duct Problem.

The project file is shown in Figure 14.2. Compared to the single-block version of the project file (Figure 6.1), only the followings parts have changed:

- There are two **$BLOCK** keywords that are used to load both of the two mesh files.
- The boundary conditions are needed to set each of the blocks. The inlet boundary conditions are assigned to Block 1, and the outlet boundary conditions are assigned to Block 2.
- The block interface boundary condition is set as the coupling boundary condition.

The simulation results are shown in Figure 14.3. Comparisons between the two-block and single-block results are also shown.

```
cdvductm.afl

$GLOBAL,
       TITLE = 'Converging/Diverging  Duct  (2 blocks)',
       FOLDER = 'cdvductm',
       SIMTYPE =            2 ,
       MACH =   0.4600000      ,
       REYNOLDS =    732676.8      ,
       PR =   0.7200000      ,
       ORDER =            2 ,
       BCSTYLE = 2,
     $END

$OVERSET,
     $END

$TURB,
       TURBTYPE =            7 ,
       TUVALUE =   0.1000000      ,
       RESET =            1 ,
     $END

$SPATIAL,
       SCHEME =            5 ,           5 ,        -999 ,
       METRIC =            0 ,
       VISCOUS =            1 ,           1 ,           1 ,
       VISCHEME =            0 ,
       VISCMON =            1 ,
       CUTOFFI =   0.0000001E-02  ,
       ISOTROPI =            0 ,
       ICUT =            0 ,           0 ,           0 ,
       CUTOFFJ =   0.0000001E-02  ,
       ISOTROPJ =            0 ,
       JCUT =            0 ,           0 ,           0 ,
       SUTHERLN =   0.3800000      ,
     $END

$TIMESTEP,
       SCHEME = 'BW2',
       MAXITER =        10000 ,
       GLOBALDT =   9.9999998E-03  ,
       SUBDT =   9.9999998E-03  ,
       GLOBTOL =   1.0000000E-30  ,
       SUBTOL =   1.0000000E-03  ,
     $END

$DAMPING,
       TYPE =            3 ,
       ES2 =   0.2000000      ,
       ES4 =   9.9999998E-03  ,
       FES2 =    1.000000      ,
       FES4 =    2.000000      ,
       MAXRED =            1 ,
       OMGAV =   9.9999997E-05  ,
       SRCONST =   0.0000000E+00  ,
       FILTER =            0 ,           0 ,           0 ,
       NFILTER =            0 ,           0 ,           0 ,
       ALPHAN =   0.4990000      ,  0.4990000      ,  0.4990000      ,
       ALPHA1 =   0.0000000E+00  ,  0.0000000E+00  ,  0.0000000E+00  ,
       ALPHA2 =   0.4990000      ,  0.4990000      ,  0.4990000      ,
       ALPHA3 =   0.4990000      ,  0.4990000      ,  0.4990000      ,
       ALPHA4 =   0.4990000      ,  0.4990000      ,  0.4990000      ,
       ALPHA5 =   0.4990000      ,  0.4990000      ,  0.4990000      ,
       ORDERN =           10 ,          10 ,          10 ,
       ORDER1 =            0 ,           0 ,           0 ,
       ORDER2 =            2 ,           2 ,           2 ,
       ORDER3 =            2 ,           2 ,           2 ,
       ORDER4 =            2 ,           2 ,           2 ,
       ORDER5 =            2 ,           2 ,           2 ,
     $END
```

Figure 14.2. The Project File cdvductm.afl for Flow Through Converging-Diverging Duct (two-Block).

```
$OUTPUT,
       PRINTFRQ  =           500  ,
       FORMAT  = 'TECPLOT',
     $END

$DEBUG,
       POINT  =            2  ,            2  ,            2  ,
     $END

$BLOCK,
       NAME  = 'BLOCK1',                                              Block 1 (41x51)
       MESHFILE  = 'mesh1.plot3d',
       FILETYPE  = 'PLOT3DF',
       ICFILE  = 'NULL',
       PERIODIC  =            0  ,            0  ,            3  ,
       ISIZE  =           41  ,
       JSIZE  =           51  ,
       KSIZE  =            1  ,
       ORDER  =            2  ,
       ANGLE  =   0.0000000E+00  ,
       DISTANCE  =   0.0000000E+00  ,
       NOCROSS  =            0  ,
     $END                                                                      Two blocks
$BLOCK,                                                                        are loaded
       NAME  = 'BLOCK2',
       MESHFILE  = 'mesh2.plot3d',                                    Block 2 (41x51)
       FILETYPE  = 'PLOT3DF',
       ICFILE  = 'NULL',
       PERIODIC  =            0  ,            0  ,            3  ,
       ISIZE  =           41  ,
       JSIZE  =           51  ,
       KSIZE  =            1  ,
       ORDER  =            2  ,
       ANGLE  =   0.0000000E+00  ,
       DISTANCE  =   0.0000000E+00  ,
       NOCROSS  =            0  ,
     $END

COMMENT:  BC            1
$FACEBC,
       BLOCK  = 'BLOCK1',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            2  ,
       VALUE  =    1.000000      ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
COMMENT:  BC            2
$FACEBC,
       BLOCK  = 'BLOCK1',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            3  ,
       VALUE  =   0.0000000E+00  ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
       ISNORMAL  =            1  ,
     $END
COMMENT:  BC            3
$FACEBC,
       BLOCK  = 'BLOCK1',
       SIDE  =            1  ,
       TYPE  =           10  ,
       VARIABLE  =            1  ,
       VALUE  =    1.000000      ,
       DEPTH  =            2  ,
       ISPLANE  =            0  ,
```

Figure 14.2. The Project File cdvductm.afl for Flow Through Converging-Diverging Duct (two-Block) (Cont.).

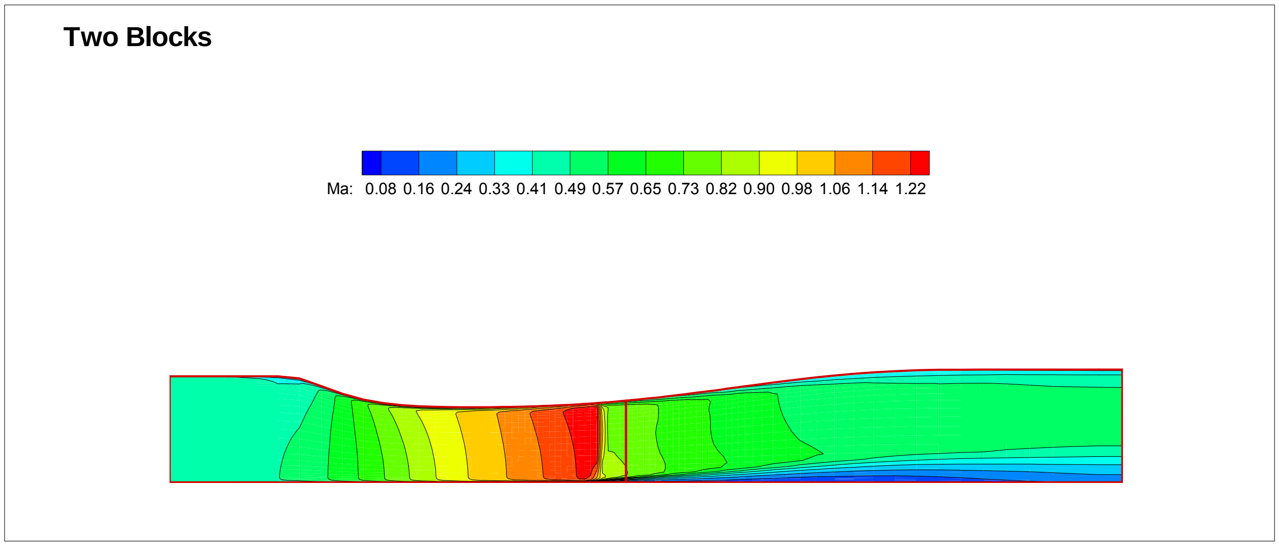
```
       ISNORMAL =             1 ,
     $END
COMMENT: BC             4
$FACEBC,
       BLOCK = 'BLOCK1',
       SIDE =             1 ,
       TYPE =            10 ,
       VARIABLE =             5 ,
       VALUE =    3.375640       ,
       DEPTH =             2 ,
       ISPLANE =             0 ,
       ISNORMAL =             1 ,
     $END
COMMENT: BC             5
$FACEBC,
       BLOCK = 'BLOCK1',
       SIDE =            -1 ,
       TYPE =             2 , ----------------------------------------------------- Type 2: Coupling boundary condition      Block 1 is
       DEPTH =             2 ,                                                                                         coupled with
       ISPLANE =             0 ,                                                                                       block 2 in
       ISNORMAL =             1 ,                                                                                      last I face
       OBLOCK = 'BLOCK2',  ------------------------------------- Specifies the block to which the current block is coupled
       OSIDE =             1 ,
     $END
COMMENT: BC             6
$FACEBC,
       BLOCK = 'BLOCK1',
       SIDE =             2 ,
       TYPE =             1 ,
       DEPTH =             2 ,
       ISPLANE =             0 ,
       ISNORMAL =             1 ,
     $END
COMMENT: BC             7
$FACEBC,
       BLOCK = 'BLOCK1',
       SIDE =            -2 ,
       TYPE =             1 ,
       DEPTH =             2 ,
       ISPLANE =             0 ,
       ISNORMAL =             1 ,
     $END
COMMENT: BC             8
$FACEBC,
       BLOCK = 'BLOCK2',
       SIDE =             1 ,
       TYPE =             2 ,                                                                                      Block 2 is
       DEPTH =             3 ,                                                                                     coupled with
       ISPLANE =             0 ,                                                                                   block 1 in
       ISNORMAL =             1 ,                                                                                  first I face
       OBLOCK = 'BLOCK1',
       OSIDE =            -1 ,
     $END
COMMENT: BC             9
$FACEBC,
       BLOCK = 'BLOCK2',
       SIDE =            -1 ,
       TYPE =            23 ,
       VARIABLE =             2 ,
       DEPTH =             2 ,
       ISPLANE =             0 ,
       ISNORMAL =             1 ,
     $END
COMMENT: BC            10
$FACEBC,
       BLOCK = 'BLOCK2',
       SIDE =            -1 ,
       TYPE =            23 ,
       VARIABLE =             3 ,
       DEPTH =             2 ,
```

Figure 14.2. The Project File cdvductm.afl for Flow Through Converging-Diverging Duct (two-Block) (Cont.).

```
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
      $END
  COMMENT: BC          11
  $FACEBC,
        BLOCK = 'BLOCK2',
        SIDE =          -1 ,
        TYPE =          23 ,
        VARIABLE =            1 ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
      $END
  COMMENT: BC          12
  $FACEBC,
        BLOCK = 'BLOCK2',
        SIDE =          -1 ,
        TYPE =          10 ,
        VARIABLE =            5 ,
        VALUE =    3.263250      ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
      $END
  COMMENT: BC          13
  $FACEBC,
        BLOCK = 'BLOCK2',
        SIDE =           2 ,
        TYPE =           1 ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
      $END
  COMMENT: BC          14
  $FACEBC,
        BLOCK = 'BLOCK2',
        SIDE =          -2 ,
        TYPE =           1 ,
        DEPTH =            2 ,
        ISPLANE =            0 ,
        ISNORMAL =            1 ,
      $END

  $INITIAL,
        VARIABLE =            2 ,
        VALUE =    1.000000      ,
      $END
  $INITIAL,
        VARIABLE =            3 ,
        VALUE =   0.0000000E+00  ,
      $END
  $INITIAL,
        VARIABLE =            1 ,
        VALUE =    1.000000      ,
      $END
  $INITIAL,
        VARIABLE =            5 ,
        VALUE =    3.375640      ,
      $END
```

Figure 14.2. The Project File cdvductm.afl for Flow Through Converging-Diverging Duct (two-Block) (Cont.).



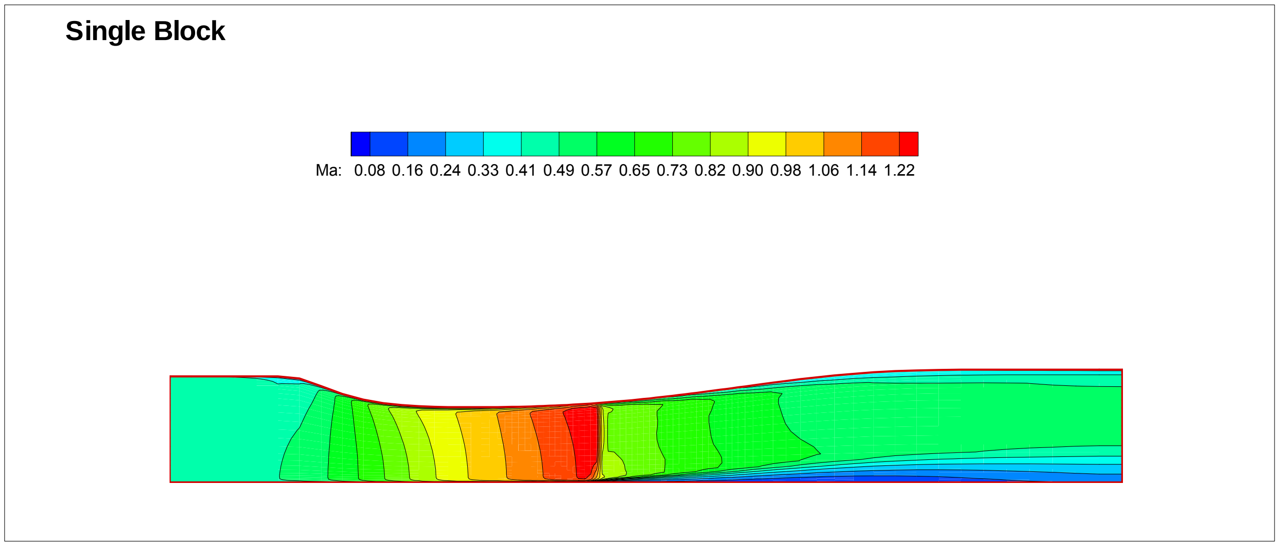


Figure 14.3. Mach Number Contour for the Two-Block Simulation of Converging-Diverging Duct Flow, Compared to the Single Block Results.

14.3 Automatic Domain Decomposition

AEROFLO can decompose your domain to match the available number of processors, independent of the original number of blocks. For example, you could ask the code to solve a 4-block problem on 10 processes, or conversely, 10 blocks on 4 processes. In these cases, the code will internally perform decomposition/coarsening of the original block structure. Note that input and output are in terms of the original block structure.

To use this capability, you only need to specify the number of processes in the syntax for submitting the job for execution. For the example in Section 14.1, the 2-block converging-diverging duct flow problem, the command to execute AEROFLO on 2 processes will be

mpirun –np 2 mpiaeroflo < cdvductm.afl

However, to run the 2 blocks on 4 processes, you will use

mpirun –np 4 mpiaeroflo < cdvductm.afl

Note that the minimum number of grid points in any direction that required for splitting is 20. Below this, AEROFLO does not allow you to split the original blocks.

14.4 Automatic Determination of Interior Block Interface

AEROFLO also supports a tool that can automatically determines the connection information between the blocks. You do not need to input the interior boundary conditions for multi-block problems. AEROFLO can automatically detect which blocks are adjacent and implement the coupling between them. You only need to provide the conditions on the physical boundaries of the problem. To use this tool, you need to insert the keyword:

$Global/SURMATCH = 1

in the setup file.

15. Performing Overset Calculations

AEROFLO can complete multi-block calculations that involve coincident node overlap or non-coincident node (chimera or overset) overlap between blocks. AEROFLO multi-block calculations are also parallel calculations, and the number of nodes used must be equal to the number of blocks. However, the blocks may be of different sizes.

To perform multi-block calculations, the user simply has to indicate the overset boundaries, like other boundary conditions, with the **$FACEBC** keyword. The boundary condition type for the overset boundary is type 3. All other boundary condition variables that define the boundary are indicated as usual. For instance, on an *I = 1* face (first *I* face), if only part of the face is overset within another block, then the *j* and *k* node limits may be indicated with the **JBCS/JBCE** and **KBCS/KBCE** keywords of the $**FACEBC** command.

AEROFLO also includes additional commands that help to speed up the calculations at an overset boundary. These commands are invoked using sub-keywords of the **$OVERSET** keyword and are explained below.

15.1 Basic Multi-Block Calculations with Coincident Node Overlap

AEROFLO can easily perform calculations involving basic multi-blocks with the blocks connected such that there is a continuity of mesh across the interblock region. (Note that these blocks may have simply been the result of a block subdivision from one huge grid). To perform calculations on blocks with coincident node overlap, the user only needs to indicate the overlap boundaries or faces for each block. No information on topology is necessary. AEROFLO will perform a search of all blocks to determine the appropriate blocks to provide values for each overlapping boundary.
To speed up the calculations, the user may indicate possible donors using the **$BLOCK/DONORS** keyword.

In addition, the user may indicate that the type of overset is a coincident node using the keyword **$OVERSET/TYPE = 1**. In such a situation, AEROFLO simply supplies the value from the matching donor node, rather than interpolating for values for the overset node from the donor block.

15.1.1 Coupling Grid Blocks

When two grid blocks touch but do not overlap, AEROFLO can create an overlap between the two blocks.

This sub-section discusses the automatic generation of an overlap between two blocks that have compatible (coincident) nodes at the boundaries at which they touch.

To specify coincident node overlap or coupling of two blocks, use the **TYPE = 2** boundary condition with the **$FACEBC** command. Consider the simple example in Figure 15.1(a):

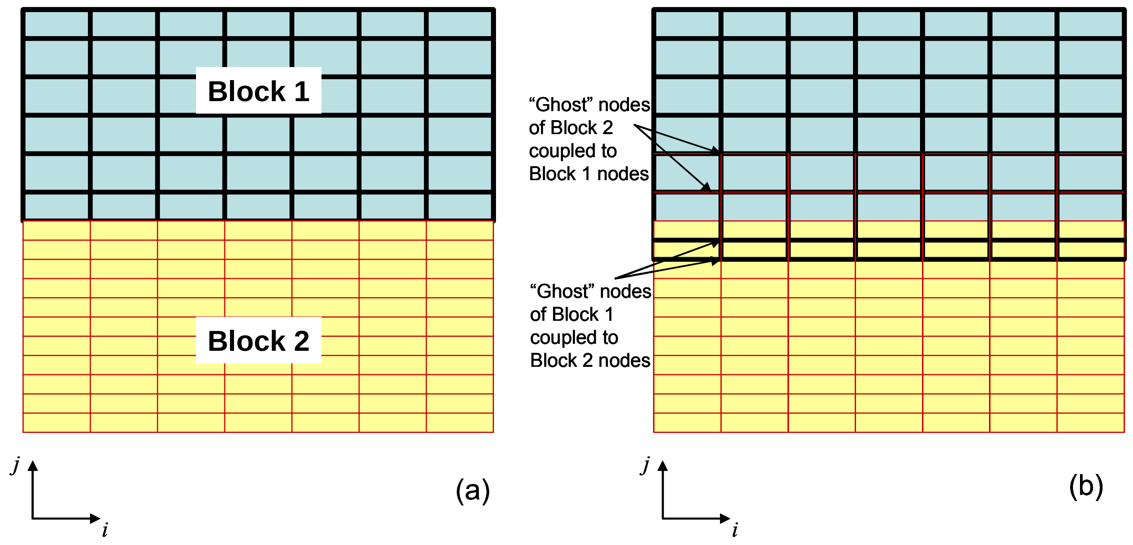

Block 1
Block 2
"Ghost" nodes of Block 2 coupled to Block 1 nodes
"Ghost" nodes of Block 1 coupled to Block 2 nodes
j
i
(a)
(b)

Figure 15.1. Simple 2D Example of Automatic Coincident Node Overlapping of Two Blocks, (a) Original blocks supplied by user, (b) Blocks showing the "ghost" nodes following the automatic coupling.

Assuming both blocks have the same *ijk* orientation as indicated in Figure 15.1, the command for coupling both blocks would be as follows:

```
$FACEBC
 BLOCK = 'BLOCK1',
 SIDE = 5,
 TYPE = 2,
 OBLOCK = 'BLOCK2',
 OSIDE = 6,
 $END

$FACEBC
 BLOCK = 'BLOCK2',
 SIDE = 6,
 TYPE = 2,
 OBLOCK = 'BLOCK1',
 OSIDE = 5,
 $END
```

The result of the above command is shown in Figure 15.1(b). Note that the coupling process is transparent to the user, as coupled or "ghost" nodes are not included in output files. To view the details of the coupling or grid assembly, plot the resfe.dat file, which is generated each time a grid assembly is done in the pre-processing stages of a calculation.

15.1.2 Partial Coupling Grid Blocks

The example provided in this section considers the partial coupling of grid blocks.

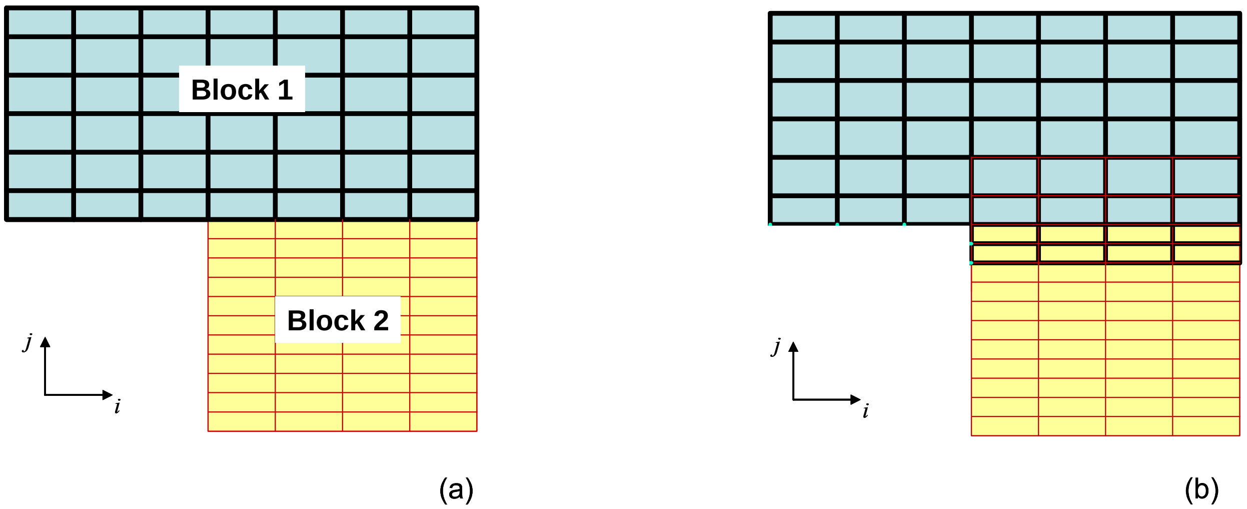


Figure 15.2. Simple 2D Example of Automatic Partial Coincident Node Overlapping of Two Blocks, (a) Original blocks supplied by user, (b) Blocks showing the “ghost” nodes following the automatic coupling.

Assuming both blocks have the same *ijk* orientation and number of grids are as indicated in Figure 15.2, the command for coupling both blocks would be as follows:

```
$FACEBC
 BLOCK = ‘BLOCK1’,
 SIDE = 5,
 TYPE = 2,
 IBCS = 4,
 OBLOCK = ‘BLOCK2’,
 OSIDE = 6,
 IOBCS = 1,
 $END

$FACEBC
 BLOCK = ‘BLOCK2’,
 SIDE = 6,
 TYPE = 2,
 IBCS = 1,
 OBLOCK = ‘BLOCK1’,
 OSIDE = 5,
 IOBCS = 4,
 $END
```

The result of the above command is shown in Figure 15.2 (b). Note that the end points of the coupling match are not indicated. As with all boundary conditions using the **$FACEBC** command, when not specified, the start point defaults to the start of the grid block (**including any ghost nodes**), and the end point defaults to the end of the grid block (**including any ghost nodes**). The commands below yield the same results as those presented in Figure 15.2(a):

```
$FACEBC
 BLOCK = 'BLOCK1',
 SIDE = 5,
 TYPE = 2,
 IBCS = 4,
 IBCE = 8,
 OBLOCK = 'BLOCK2',
 OSIDE = 6,
 IOBCS = 1,
 IOBCE = 5,
 $END

$FACEBC
 BLOCK = 'BLOCK2',
 SIDE = 6,
 TYPE = 2,
 IBCS = 1,
 IBCE = 5,
 OBLOCK = 'BLOCK1',
 OSIDE = 5,
 IOBCS = 4,
 IOBCE = 8,
 $END
```

However, the second set of commands is preferred to the former, as it helps to avoid incorrect specification of boundary conditions with its loose specifications. This is illustrated with an example in the next section.

15.1.3 When to Indicate or Omit Coupling Limits

Consider the five block example in Figure 15.3:

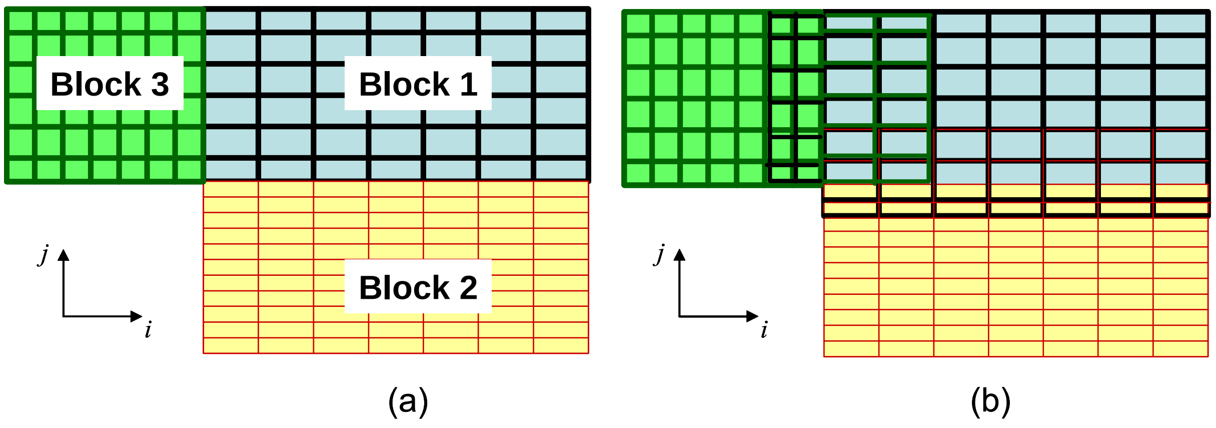


Figure15.3. 2D Example of Automatic Partial Coincident Node Overlapping of Three Blocks, (a) Original blocks supplied by the user, (b) Blocks showing the “ghost” nodes following the automatic coupling.

```
$FACEBC
 BLOCK = 'BLOCK1',
 SIDE = 5,
 TYPE = 1,
 $END
```

In Figure 15.3(a), specifying a loose wall boundary condition (as shown above) on the *j* face of Block 3 will result in the application of wall conditions on all of the nodes at the *j = 1* face, including any ghost nodes. This includes the original 8 nodes of Block 3, as well as the additional 2 ghost nodes coupled with Block 1. This is due to the precedence of the wall boundary condition over the overset boundary condition. The correct boundary condition is shown below:

```
$FACEBC
 BLOCK = 'BLOCK1',
 SIDE = 5,
 TYPE = 1,
 IBCS = 1,
 IBCE = 8,
 $END
```

Similarly, specifying a loose match between Blocks 1 and 2 results in an incompatibility and the blocks are extruded rather than coupled (which is usually not a major problem, as high-order interpolation is used to exchange results between the grids).

15.2 Overset Multi-Block Calculations with Non-Coincident Node Overlap

AEROFLO also performs overset calculations involving multi-blocks with non-coincident node overlap across the interblock regions. To perform these calculations, the user does not need to provide any special input. However, to speed up the calculations, the user may indicate possible donors using the **$BLOCK/DONORS** keyword.

15.2.1 Extruding Incompatible Grids

Abutting (touching) grids can be extruded in two ways:

(a) When a coupling (**TYPE =2**) boundary specification is used to connect both grids but the grids do not have coincident nodes at the interface, and
(b) By using **TYPE = 85** of the **$FACEBC** command.

An example of grids extruded due to an incompatibility between abutting grids is shown in Figure 15.4. Note that the grids are extruded at a normal and with similar grid size as the preceding grid cell.

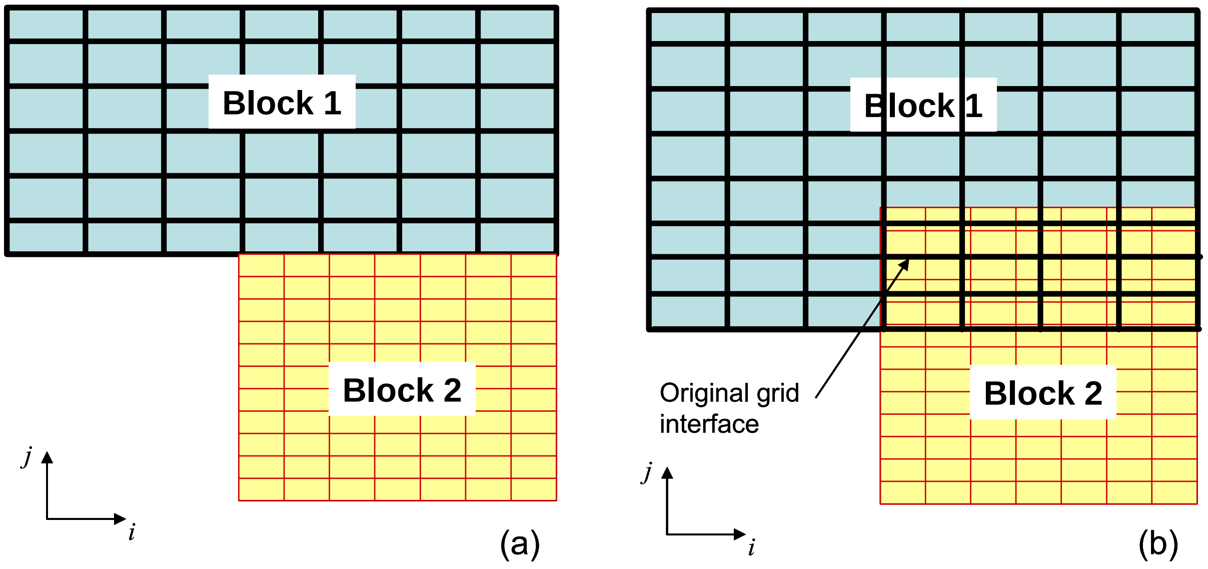


Figure 15.4. 2D Example of Overset Blocks from Abutting Grids with Incompatible Nodes at the Abutting interface, (a) Original blocks supplied by the user, (b) Blocks showing the "ghost" nodes following the automatic overlap.

15.3 Mixed Block Calculations

To perform calculations involving coincident node overlap across some blocks and non-coincident node overlap across other blocks, the user may set up the problems in the same way as those for non-coincident node overlap. However, using the keyword **$OVERSET/COINNODE = 1**, AEROFLO will first search for an exact node donor for an overset node before searching for an interpolating donor.

15.4 Specifying Blank Regions

Three methods have been provided specifying blank regions in AEROFLO.

1) Direct specification using grid points:

   Blank regions are specified this way using the **$FACEBL** keyword and **TYPE=95**. The region to be blanked is further qualified by the IBCS, IBCE, JBCS, JBCE, KBCS, and KBCE keywords. For example, the specification below causes the grid points, *i* = 1-5, *j* = 10, 25, *k* = 1-End to be blanked.

```
$FACEBL
    TYPE    = 95,
    IBCE    = 5,
    JBCS    = 10,
    JBCE    = 25,
$END
```

2) Direct specification using a geometric box:

   The geometric box is specified with the box keyword. The box is further defined by the **POINT** keywords.

```
$POINT
    COORDS = 0.0, 0.0, 0.0,
$END

$POINT
    COORDS = 0.0, 1.0, 0.0,
$END

$BOX
    POINTS = 1, 2, 3, 4, 5, 6, 7, 8,
    BLOCKS = 'BLOCK1',
$END
```

   The example above causes sections of BLOCK with name 'BLOCK1' that lie within the box specified the corner points above to be blanked.

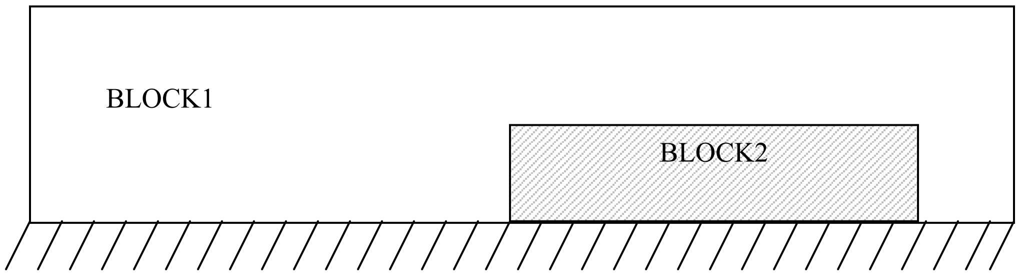
3) Automatic cuts based on block overlaps:

   These types of cuts may be specified using the **$MESHCUT** keyword. The procedure equates the specification of a primary block or cutter. One or more blocks, whose grids will be blanked in the regions in which they overlap the primary block, are then specified.

To further qualify these types of cuts, the order and periodicity of the cut may be specified. The order refers to how many overset nodes will be placed at the fringe of the cut, as well as how many nodes will be located at the offset between both blocks. Periodicity takes on the following values:

PERIODICITY (*i*, *j*, or *k*)	
0	Places an overlap at the boundary of the cut on both ends.
1, 2, 3	Extends the cut to the boundary of both ends and leaves no overlap. In particular, for 30 problems, the cut is intended to extend through the periodic direction.
4	The cut extends to *i*, *j*, or *k* = 1 without any offset or overlap.
5	The cut extends to *i*, *j*, or *k* = End without any offset or overlap.

In the example below, which represents a 2D, two-block setup with injection of the wall boundary of BLOCK2, the grid BLOCK1 blanks all regions of BLOCK2, overlapping it all the way to the wall region.



```
$MESHCUT
    CUTTER       = 'BLOCK2',
    BLOCKS       = 'BLOCK1',
    PERIODICITY = 0, 4, 3,
$END
```

16. AEROFLO Input Keywords

In this chapter, the keywords, expected values, and default values of input AEROFLO parameters are described. A summary of group keywords in AEROFLO is shown below:

Keywords of global scope include:

▪	$GLOBAL	– global project data including simulation type, project folder, project files, and flow parameters such as Reynolds number, Mach number, etc.
▪	$TURB	– specifies turbulence data. When not provided, no turbulence model is activated for the problem.
▪	$MHD	– specifies MHD data. Required for MHD simulation types SIMTYPE = 5 or 6.
▪	$SPATIAL	– spatial differencing data.
▪	$TIMESTEP	– time differencing data.
▪	$DAMPING	– damping or filtering data. When not provided, no damping or filtering is applied.
▪	$POISSON	– Poisson solver data. Required for MHD simulation types SIMTYPE = 5 or 6.
▪	$CASESPEC	– case specific data for certain internally-constructed problems.
▪	$OUTPUT	– output and animation generation data.
▪	$DEBUG	– specification of points at which to print out visual solution data.

Keywords of block scope include:

▪	$BLOCK	– specifies a mesh block. This keyword provides data such as the dimensions of the mesh, location of the grid file and initial condition file, as well as a unique name for the block. This input group may appear several times. Each occurrence specifies a unique mesh block. All projects require the specification of at least one block.
▪	$FACEBC	– specifies a surface boundary condition.
▪	$LINEBC	– specifies a line boundary condition.
▪	$NODEBC	– specifies a boundary condition on nodes.
▪	$INITIAL	– applies initial conditions to blocks.

Keywords used to overset specifications:

▪ $POINT	–	used to define a geometric point in the project.
▪ $BOX	–	used to define a box in space for blanking of regions.
▪ $MESHCUT	–	used to define a region of one or more blocks that will be blanked by another block.
▪ $OVERSET	–	specification of overset variables. Not mandatory.

16.1 $GLOBAL Keywords

The sub-keywords in the **$GLOBAL** keyword block specify the global project data, including simulation type, project folder, project files, and flow parameters, such as Reynolds number, Mach number, etc. For most projects, this block may be mandatory because the flow parameters are required.

TITLE

This input specifies a title for the simulation for informational purposes.

SIMTYPE

The type of simulation:

1-4	DNS/LES
5-6	MHD

MACH

Mach number for the simulation.

REYNOLDS

Reynolds number for the simulation.

ORDER

Specifies how many nodes at a boundary will be applied to a boundary condition.

FOLDER

Folder from which input files (grid and initial condition file) will be read and in which output files will be generated. **When not specified, the AEROFLO folder will be used as the project folder.** For organizational purposes, the user should create separate folders for each project as subfolders of the main AEROFLO folder.

The included sample problems are organized in this manner.

STORAGE

Specifies if all processors for a multi-block calculation have a shored storage. If so, the storage takes a value of 1, which is its default value. Otherwise, its value is 0.

16.2 $TURB Keywords

The sub-keywords in the **$TURB** keyword block specify turbulence data. This keyword block is not mandatory. When not provided, no turbulence model is activated for the problem.

TURBTYPE

Turbulence model. Available options include:

0 – No turbulence model is used (Laminar, Turbulent DNS or MILES)
1 – Smagorinsky with compact differencing of the turbulence equations
2 – Smagorinsky with 2nd-order central differencing of the turbulence equations
3 – Smagorinsky with 1st-order differencing of the turbulence equations
4 – Dynamic SGS. This choice requires the specification of homogenous directions
5 – Spalart-Allmaras. This choice also requires the specification of trip points along the solid walls of the model.
6 – Launder-Sharma κ-ε model
7 – Abid's κ-ε model
8 – Menter's SST κ-ω model
9 – High Reynolds Number κ-ε model
10 – DES (based on Spalart-Allmaras)
11 – PRNS (based on Abad's κ-ε)
12 – PRNS (based on high-Reynolds κ-ε)

DIRECTN

Homogenous directions. For instance, (1, 0, 0,) means homogenous in the *I*-direction only. This input is required for **TURBTYPE = 4**.

NUMTRIPS

Number of trip points on solid surfaces. This input is required for **TURBTYPE = 5**.

```
NUMTRIPS = 2,
```

The above specification means that two trip points will be employed at the solid wall surfaces of the mesh

(See the sample problem **cylinder.txt)**. The keyword is scheduled to be returned. If not provided, the trip points are automatically coded to be at the start of the boundary layer.

POINT

The trip points are specified as *(x, y, z)* for as many trip points as **NUMTRIPS**. Each point is separately specified on its own line.

PLANE

Description of the symmetry plane *(x, y, z)* via three points specified as *x, y, z*. For O-grids, points on opposite sides of this plane are not affected by trip points on the other side. For grids in which all the points lie on the same side of the solid wall, the plane should be defined below the wall so that all points lie on the same side as the trip points.

TUVALUE

The turbulence intensity for κ-ε and κ-ω model.

RESET

Determine if the turbulence intensity set in **TUVALUE** is used to initial the turbulence calculation.

16.3 $SPATIAL Keywords

The sub-keywords in the **$SPATIAL** keyword block determine the spatial difference data. This keyword block is required.

SCHEME

Spatial differencing scheme. Specified for all three directions on the same line, e.g.,

```
SCHEME = 5, 5, 5,
```

specifies MUSCL scheme in all three directions.

0	Original second-order central scheme
1	First-order Roe
2	Fully-upwind second-order Roe (using MUSCL with κ = -1)
3	Second-order Roe scheme with Fromm reconstruction (κ = 0)
4	Third-order upwind-biased Roe scheme (κ = 1/3)
5	Second-order subset of the Roe/MUSCL scheme (κ = 1)
16	Compact scheme
33	WENO scheme
-999	No solution

METRIC

Metric differencing scheme.

0	2nd-Order central differencing
1	Compact scheme
2	Same scheme used for inviscid and viscous fluxes
12	Compact scheme
13	WENO

VISCOUS

Viscous specification in all three directions. Specified on the same line, e.g.,

```
VISCOUS = 1, 1, 1,
```

implies that viscous calculations will be assumed in all three directions

VISCHEME

Scheme used to compute the viscous terms.

0	Central order formulation for computing the viscous terms
1	Compact difference formulation for computing the viscous terms

VISCMON

Determines if the cross derivative terms will be computed as well.

SUTHERLN

Sutherland's law parameter. Choosing a negative value results in an inviscid calculation. Choosing a value greater than 9999 means that the viscosity is considered constant and Sutherland's law does not applied.

COMPACTI

This input specifies the compact scheme for differencing in the *I*-direction. Used if SCHEME = 15 in the *I*-direction, this character variable consists of five fields designating the scheme to be employed at points 1, 2, interior, N-1, and N, respectively.

For instance, C4, CC4, C6, AC4, C4 requests the fourth-order compact scheme at point 1, the decoupled fourth-order compact scheme at point 2, compact sixth-order in the interior, the symmetric compact fourth-order scheme at point N-1, and the fourth-order compact scheme at point N. Note: For efficiency, if the interior scheme is explicit, the tridiagonal system is not solved, resulting in faster solutions. However, it is not possible to combine explicit interior schemes with implicit boundary schemes. Please consult the appendix for tables and descriptions of the filtering schemes.

COMPACTJ

This input specifies the compact scheme for differencing in the *J*-direction. *See the notes for COMPACTI.*

COMPACTK

This input specifies the compact scheme for differencing in the *K*-direction. *See the notes for COMPACTI.*

COMPACTG

This input specifies the compact scheme for the grid differencing. *See the notes for COMPACTI.*

CUTOFFI

Controls the entropy cutoff for the Roe scheme in the *I*-direction. Used only if SCHEME is 1-5 in the *I*-direction.

CUTOFFJ

Controls the entropy cutoff for the Roe scheme in the *J*-direction. Used only if SCHEME is 1-5 in the *J*-direction.

CUTOFFK

Controls the entropy cutoff for the Roe scheme in the *K*-direction. Used only if SCHEME is 1-5 in the *K*-direction.

ISOTROPI

Controls the choice of an isotropic or anisotropic formula in *I*-direction. Used only if SCHEME is 1-5 in the I-direction.

ISOTROPJ

Controls the choice of an isotropic or anisotropic formula in *J*-direction. Used only if SCHEME is 1-5 in the J-direction.

ISOTROPK

Controls the choice of an isotropic or anisotropic formula in *K*-direction. Used only if SCHEME is 1-5 in the K-direction.

ICUT

Three variables that control the on/off switches for the linear *u, u+c*, and *u-c* eigenvalues respectively. Used only if **SCHEME** is 1-5 in the *I*-direction. For example,

```
ICUT = 0, 0, 0,
```

turns off all the switches for the eigenvalues.

JCUT

Switches for *J*-direction. *See notes for ICUT.*

<u>KCUT</u>

Switches for *K*-direction. *See notes for ICUT.*

<u>BLOCKS</u>

In certain calculations, it is desired to use different spatial schemes for different calculations. If BLOCK is specified, the current **SPATIAL** specification applies to the block. Otherwise, the specification applies to all blocks. If provided, this refers to the block to which the spatial scheme specified in the group applies.

16.4 $TIMESTEP Keywords

The sub-keywords in the $TIMESTEP keyword block specify the time differencing data. This keyword block is required.

MAXITER

This variable represents the maximum number of iterations for which to run the problem.

SCHEME

The time integration schemes supported in the current version of AEROFLO include:

BW2	Beam-Warming approximate factorization with diagnolization simplification
BW1	Beam-Warming approximate factorization
RK4	Fourth-Order Runge-Kutta (an explicit scheme)

DIAGONAL

This variable takes on values of 0 or 1, and determines if diagonal formulation will be used for the Beam-Warming computations. When a value of 1 is used, diagonal formulation is used and results in slightly faster time integration.

SUBON

Determines if sub-iterations will be performed. This option is only activated for the Beam-Warming schemes.

MAXSUB

Determines the maximum number of sub-iterations, if convergence is not reached first during the sub-iteration stages. The convergence criteria for sub-iterations are controlled by SUBTOL (see below).

GLOBALDT

The time step size for iterations. It is recommended that a value smaller than 1/2 * MIN(Δx, Δy, Δz) be used. (Δx, Δy, Δz) represent mesh spacing at cells in the mesh. For calculations using preconditioners, GLOBALDT can usually be set to a very large value.

SUBDT

Step size for sub-iterations. It is recommended that a value smaller than 1/2 * MIN(Δx, Δy, Δz) be used. (Δx, *Δy, Δz*) represent mesh spacing at cells in the mesh.

PRECON

Determines if preconditioners will be used for the time integration. Preconditioners are recommended for very low Mach number calculations (< 0.1). Otherwise, the required GLOBALDT or time step size required for the calculation will be small, making the calculations expensive.

IBETA

Controls whether the scheme computes the sub-iteration time step size or if the CFL is used to automatically compute the sub-iteration time step and drive sub-iterations to convergence. Takes on values of 0, or 1. If the value is 0, the value of SUBDT is used as the sub-iteration step size. Otherwise, a sub-iteration step size is computed using the CFL number provided.

CFL

The CFL number is used in conjunction with IBETA =1 to compute the sub-iteration step size.

CFLMHD

The CFLMHD number is used in conjunction with IBETA =1 to compute the sub-iteration step size for an MHD calculation.

NDTAU

NDTAU is the number of steps between the updating of the sub-iteration step size. Used in conjunction with IBETA = 1.

KTVDRK

Determines the version of Runge-Kutta employed. This input is only relevant if 'RK4' is the time integration scheme.

3	uses the third-order Runge (TVDRunge3)
4	uses the fourth-order Runge (TVDRunge4)

Otherwise, the standard formulation is used.

LREF

Used in conjunction with the preconditioner (PRECON = 1), LREF is a reference length scale.

SUBTOL

When sub-iteration is used (ISUBON = 1), sub-iteration is performed up to the maximum number of sub-iterations specified by MAXSUB or until sub-iteration convergence. The convergence tolerance is specified by SUBTOL and represents the ratio of the residual at first sub-iteration to the residual at convergence.

GLOBTOL

For a steady-state simulation, GLOBTOL represents the convergence tolerance. This tolerance represents the ratio of the residual at first iteration to the residual at convergence. Once convergence is reached, the iterations terminate, even if MAXITER has not been reached. For an unsteady-state simulation, set GLOBTOL to a very low value or leave it at its default value to ensure that the calculations do not terminate prematurely.

ACCEL

Set this variable to 1 to use the convergence accelerating scheme. This scheme increases the value of GLOBALDT as the solution error reduces.

16.5 $DAMPING Keywords

The sub-keywords in the **$DAMPING** keyword block specify the damping or filtering data. When not provided, no damping or filtering is applied.

TYPE

This variable determines the type of damping or filtering scheme that will be employed:

1	Not used
2	Damping
3	Filtering (every sub-iteration)
4	Filtering (every iteration or after all sub-iterations are completed)

ES4, ES2, FES4, FES2, OMGAV, SRCONST

These are implicit damping control parameters.

FILTER

This variable is used to control filtering in each of the three directions. This variable takes on a value of either 0 or 1. When the value is 1, filtering is done in that direction. Otherwise, filtering is not performed. For example,

```
FILTER = 1, 1, 0,
```

will result in filtering in the *i* and *j* directions only if the filtering option is chosen as the damping type (TYPE = 4, or TYPE = 3,).

NFILTER

This variable determines the number of times that filtering will be performed each time. For example,

```
NFILTER = 2, 2, 0,
```

will result in filtering twice in the *i* and *j* direction for each round of filtering. For instance, if damping TYPE = 3, filtering will be performed twice after every sub-iteration or twice after every iteration if damping TYPE = 4.

ORDERN, ORDER5, ORDER4, ORDER3, ORDER2, ORDER1

These variables control the order of accuracy of filters in the interior nodes, fifth, fourth, third, second node from the boundary, and boundary nodes, respectively. The values are specified for all three directions.

```
ORDERN = 10, 10, 0,
ORDER5 = 8, 8, 0,
ORDER4 = 6, 6, 0,
ORDER3 = 4, 4, 0,
ORDER2 = 2, 2, 0,
ORDER1 = 0, 0, 0,
```

The above specification will use a tenth-order filter at the interior nodes and eighth, sixth, fourth, order, and second order filters at the fifth, fourth, third, and second nodal points from the boundary, while no filtering will be performed at the boundary nodes. Please consult the appendix for notes and tables on acceptable values.

ALPHAN, ALPHA5, ALPHA4, ALPHA3, ALPHA2, ALPHA1

These variables control the filter coefficients in all three directions at the interior, fifth, fourth, third, second, and boundary nodes, respectively. Accepted values range from 0.4999 to -0.4999. A value of 0.4999 is the least dissipative.

```
ALPHAN = 0.499, 0.499, 0.0,
ALPHA5 = 0.45, 0.45, 0.0,
ALPHA4 = 0.45, 0.45, 0.0,
ALPHA3 = 0.49, 0.49, 0.0,
ALPHA2 = 0.499, 0.499, 0.0,
ALPHA1 = 0.0, 0.0, 0.0,
```

The above specification will use a filter coefficient of 0.499 at interior nodes of a simulation in the *i* and *j* directions and values of 0.45, 0.45, 0.49, 0.499, and 0.0 at the fifth, fourth, third, second nodal points from the boundary, and boundary nodes, respectively. Please consult the appendix for notes and tables on acceptable values.

16.6 $OUTPUT Keywords

The sub-keywords in the **$OUTPUT** keyword block specify the output and animation generation data.

PRINTFRQ

Frequency of printing results. A CGNS file “myres.cgns” and a TECPLOT file “myres.dat” are generated at each PRINTFRQ iteration.

FORMAT

Specifies the preferred output format. The default value is ‘TECPLOT.’ Note that the CGNS result/output/restart file myres.cgns is always generated, regardless of the value of the format.

CGNS

- myres.cgns result/output/restart file is generated
- Intermittent result files are generated with names res1-***.cgns (if TECANIM is 1, see TECANIM below)

TECPLOT

- myres.dat tecplot result file is generated
- Cp.dat tecplot result file is generated for all wall boundaries
- Intermittent tecplot result files are generated with names res1-***.dat (if TECANIM is 1, see TECANIM below)
- myres.cgns result/output/restart file is generated

PLOT3D

- myres.PLOT3D and myres.PLOT3DQ PLOT3D grid and Q result file are generated
- Intermittent PLOT3D Q result files are generated with names res1-***.3dq (if TECANIM is 1, see TECANIM below)
- myres.cgns result/output/restart file is generated

TECANIM

This variable determines if complete result files will be intermittently written and stored. Takes on a value of 0 or 1.

TRANGE

The range in time at which the intermittent result files will be written.

```
    TRANGE = 20000, 40000, 2,
```

The above specification will cause result files to be written between the 20,000 and 40,000th iterations at every 2*PRINTFRQ iteration. To print the initial conditions, the TRANGE start should be 0.

16.7 $DEBUG Keywords

The sub-keywords in the $DEBUG keyword block define the points at which to print out visual solution data. These are not mandatory.

POINT

A point in the block at which visual diagnostic results are printed at every sub-iteration. If not provided, diagnostic values are printed at the center of the block (IE/2, JE/2, KE/2).

```
POINT = 20,4,2,
```

The above specification will cause diagnostic results to be printed at node (20,4,2).

16.8 $BLOCK Keywords

The sub-keywords in the $BLOCK keyword block specify a mesh block. They provide data such as the dimension of the mesh, location of the grid file, and initial condition file, as well as a unique name for the block. This input group may appear several times. Each occurrence specifies a unique mesh block. All projects require at least one block be specified.

NAME

A unique name for identifying the block.

MESHFILE

Path or location of a grid file for the block.

FILETYPE

Format of the grid file is as follows:

CGNS	CGNS file format (specifies the grid and possibly the initial conditions)
AEROFLO	AEROFLO file format (specifies the grid as well as the initial conditions)
PLOT3D	PLOT3D file format (specifies the grid).

ICFILE

An initial condition file, to be used if the initial condition is not contained in the mesh file. If no initial condition file is specified, and the initial conditions are not read from the grid file, an initial condition of 0.0 will be assumed for the velocity field, $1/(\gamma Ma^2)$ for the pressure, and 1.0 for the density. See Chapter 10 for initial conditions specification.

ISIZE, JSIZE, KSIZE

Size of the grid. For instance,

```
ISIZE = 245,
JSIZE = 50,
KSIZE = 3,
```

which describes a 245 x 50 x 3 grid.

PERIODIC

Periodic description of the block. This keyword expects three integer variables representing the periodicity description in the *i*, *j*, and *k* directions. The keyword expects three values: 0, 1, and 2.

```
0 – No periodicity
1 – Periodic
2 – Periodic but no overlap.
3 – 2D. Convert to 3D in this direction.
```

AEROFLO solves periodicity with the expected overlap of at least five grid points in the periodic direction. If the mesh was generated such that the last node point is the same as the first node point, then there is no overlap and the user must specify a periodic value of 2 to ensure that AEROFLO appropriately overlaps the grid.

```
PERIODIC = 1, 0, 1,
```

The above specification, for example, describes a block that is periodic in *i*, and *k*. If a block is degenerate or 2D in one direction, the user should specify periodicity in this direction. This ensures that the same results are maintained at all nodes. Note that the current program does not permit 2D definition by using only one node, and at least three nodes must be specified in the degenerate direction. If the mesh is 2D, the **PERIODIC** indicator must be 3 in the 2D direction to ensure that AEROFLO properly generates 2 extra nodes in this direction. Also remember to set the **$SPATIAL/SCHEME** value to -999 in a degenerate direction.

ORDER

Number of nodes required at the boundary for overset formulation. If not specified, assumes the default global value of 2.

DONORS

List of other blocks that possibly overlap the block being specified. This list must use the block names to refer to the donor blocks. Ensure that all blocks on this list are also described with the **$BLOCK** commands.

ANGLE

Maximum angle at which a node lying close to a solid boundary may be projected.

DISTANCE

Maximum distance that a node lying close to a solid boundary may be projected.

16.9 $FACEBC Keywords

The sub-keywords in the **$FACEBC** keyword block specify a surface boundary condition. This input group may appear several times. Each occurrence specifies a unique boundary condition.

BLOCK

Name of the block to which the boundary condition refers.

BCSTYLE

There are two styles to label the side of the block. See **SIDE**.

SIDE

If **BCSTYTLE** = 1, side of the block to which the boundary condition refers:

```
1        k=KE
2        i=IE
3        k=1
4        i=1
5        j=1
6        j=JE
```

If **BCSTYTLE** = 2, side of the block to which the boundary condition refers:

```
 1       i=1
-1       i=IE
 2       j=1
-2       j=JE
 3       k=1
-3       k=KE
```

TYPE

The list of the available boundary condition types are listed in the table below.

BC No.	
1	Solid Wall
2	Coupling
3	Overset boundary condition
8	Coupling due to periodicity
9	C-Grid
10	Dirichlett
11	Symmetry
13	Symmetry in *i* (any plane)

14	Symmetry in *j* (any plane)
15	Symmetry in *k* (any plane)
23	Neumann (0 flux)
24	Specified flux
30	Inflow
31	Outflow
32	Free-stream
41	Characteristic Inflow B.C.
42	Characteristic Outflow B.C.
52	Initial value
59	Singular point surface (e.g., sphere grid)
60	*i*-axes (axisymmetric around *I* at specified face)
61	*j*-axes
62	*k*-axes
71	Slip wall
80	Equation
90	User Defined

VARIABLE

Variable to which the boundary condition refers:

	Variable	
	Density	1
	u - velocity	2
	v - velocity	3
	w - velocity	4
	Pressure	5
	Reserved for B_1	7
	Reserved for B_2	8
	Reserved for B_3	9
	k	11
	ε	12
	.	
	.	
	.	
	Scalar *n*	20 + *n*

The variable input is only required for the Dirichlett (TYPE = 10,), no stress (TYPE = 23,), and flux (TYPE = 24,) boundary condition types.

VALUE

This represents the value to which the boundary conditions will be set. The value input is only required for the Dirichlett (TYPE = 10,), flux (TYPE = 24,), and freestream (TYPE = 32,) boundary condition types.

DEPTH

Number of nodes at the boundary to which to apply the boundary condition. Defaults to $BLOCK/ORDER or $GLOBAL/ORDER. For instance, for a multi-block overset formulation, to maintain fourth-order accuracy at an overset boundary for a centered formulation, at least two nodes at the boundary must be specified as overset nodes.

ISPLANE

This variable determines if the surface normal direction will be computed at every node on the boundary. If ISPLANE = 1, the normal is calculated only once, as it is assumed that all of the nodes lie on a plane. This results in a faster scheme. Because boundary conditions are applied at every sub-iteration, it is important to designate plane surfaces to speed up the calculations. This variable takes on a value of 0 or 1. The default value is ISPLANE = 1.

ISNORMAL

This variable determines if boundary nodes are assumed to be normal at the boundary. If ISNORMAL = 1, the second nodal point from the boundary surface is assumed to form a normal with the adjacent boundary node. If the boundary nodes are not normal, interpolation is necessary to compute the values when the Neumann and flux conditions are applied at the boundary. This results in a faster scheme. Because boundary conditions are applied at every sub-iteration, it is important to designate normal surfaces to speed up the calculations. This variable takes on a value of 0 or 1. The default value is ISNORMAL = 1.

IBCS/IBCE

The start and end nodal points in i for a boundary lying in the i, j or k, i planes. The default values are 1 and IE, respectively.

JBCS/JBCE

The start and end nodal points in j for a boundary lying in the i, j or j, k planes. The default values are 1 and JE, respectively.

KBCS/KBCE

The start and end nodal points in *k* for a boundary lying in the *j*, *k* or *k*, *i* planes. The default values are 1 and KE, respectively.

CNORM

For characteristic inflow or outflow boundary conditions, CNORM is used to provide the coefficient for the unit normal in the direction of the inflow.

OBLOCK

For coupled conditions, this keyword specifies the block to which the current block is coupled.

OSIDE

For coupled conditions, this keyword specifies the side of the other block to which the current block is coupled.

VAR1

For boundary conditions specified in simple equation form (e.g., ρ = rho$^{\gamma}$), VAR1 refers to the second variable in the equation expression. The values of VAR1 are as follows:

-1	x		-5	q = $\sqrt{u_i^2}$		1	ρ
-2	y		-6	τ		2	u
-3	z		-7	nstep		3	v
-4	M_α					4	w
						5	p

IOBCS/IOBCE

The start and end nodal points in *i* for a boundary lying in the *i*, *j* or *k*, *i* planes for the block to which the current block is supplied. The default values are 1 and IE, respectively, for the block to which the current block is supplied

JOBCS/JOBCE

The start and end nodal points in *j* for a boundary lying in the *i*, *j* or *j*, *k* planes for the block to which the current block is supplied. The default values are 1 and JE, respectively, for the block to which the current block is supplied.

KOBCS/KOBCE

The start and end nodal points in *k* for a boundary lying in the *j*, *k* or *k*, *i* planes for the block to which the current block is supplied. The default values are 1 and KE, respectively, for the block to which the current block is supplied.

OPERATO1

The arithmetic operation in a boundary condition specified in equation form TYPE=80.

VAL1

The sub-keyword is used in conjunction with the sub-keywords VAR1 and OPERATO1 to apply simple equations of the boundary. The simple equation format of the boundary condition type=80 is:

```
VAR = VAL1          OPERATO1        VAR1
```

The format of the equations is:

OPERATO1	Equation
-	VAL * VAR1 – VAL1
+	VAL * VAR1 + VAL1
/	VAL * VAR1 / VAL1
*	VAL * VAR1 * VAL1
**	VAL * VAR1 ** VAL1
ABS	VAL * ABS(VAR1)
COS	VAL * COS(VAR1)
INT	VAL * INT(VAR1)
SIN	VAL * SIN(VAR1)

In the future, AEROFLO will be furnished with a function parser, enabling the specification of more complicated equations of the boundary.

16.10 $INITIAL Keywords

The sub-keywords in the $INITIAL keyword block specify an initial condition. This input group may appear several times. Each occurrence specifies a unique initial condition.

BLOCK

Name of the block to which the boundary condition refers. When the block is not specified, the initial condition is applied to all blocks.

VARIABLE

Variable to which the boundary condition refers:

	Variable	
	Density	1
	u – velocity	2
	v – velocity	3
	w – velocity	4
	Pressure	5
	Smagorinsky variable	10
	Scalar 1	11
	Scalar 2	12
	.	
	.	
	.	
	Scalar *n*	10 + *n*

VALUE

The value to which the initial condition will be set.

16.11 $OVERSET Keywords

This keyword block is used to specify overset variables.

TYPE

For calculations involving only non-coincident node overlap or a combination of coincident and non-coincident overlap, use TYPE = 0 which is the default. In other words, this keyword is not required for calculations involving only overset nodes or a combination of overset and coincident node overlap. For calculations involving only coincident node overlap, TYPE = 1 is more appropriate. This causes the program to pair the coincident node with exactly its donor node in the overlapping block. No interpolation is required in computing the values from the donor nodes, making the calculations faster. If TYPE = 0 is used for coincident-node calculation, the results are nearly the same but require more calculation operations.

RESTART

This keyword is used to generate or read overset information. To remove the overhead of determining donors for overset nodes at the start of a restart calculation, the overset information may be written to file by setting RESTART = 2. On subsequent restarts, RESTART = 1 will read the overset information from restart files rather than recompute donors for overset nodes. RESTART = 0 is the default and causes the donor information to be computed every time but not written.

COINNODE

In a mixed multi-block calculation involving both coincident and non-coincident node calculations, the computation of donors for overset nodes can still be sped up for coincident overlap nodes by setting COINNODE = 1. This causes AEROFLO to first search for an exact coincident node for an overset node. COINNODE = 0 is the default value.

PROJECT

This determines if projection will be used at the interface of solid walls for overset boundaries. PROJECT = 0 is the default value and infers that projection will not be performed.

NUMITER

While searching for donors for overset nodes, AEROFLO may encounter nodes that have no donors (i.e., orphan nodes). However, AEROFLO may widen the allowance for declaring a node an orphan by allowing extrapolation from a close cell. NUMITER controls this allowance. The higher the value of NUMITER, the higher the proximity of a node to a donor cell that is allowed to interpolate and provide values for it.

NUMITER is applied progressively such that the closest cells are always chosen. The default value of NUMITER is 2.

<u>USEADT</u>

This sub-keyword is used to indicate whether octree search/coded procedures should be used for the overset donor search process. This results in a faster search process. The value of this sub-keyword is 0 or 1. The default value is 1, which indicates that octree procedures should be used.

16.12 $MESHCUT Keywords

This keyword block is used to define a region of one or more blocks that will be blanked by another block.

<u>CUTTER</u>

Primary cutting block. Sections of this block will cut the other blocks specified in the BLOCKS list.

<u>BLOCKS</u>

List of blocks that will be cut when they overlap the block specified by the keyword CUTTER.

<u>ORDER</u>

Offset of nod at the edge of the cut.

<u>PERIODIC</u>

This sub-keyword specifies how the cut will be completed.

The values of PERIODIC are:

0	-	The cut extends all the way to the boundary of the cutting block with an offset for interpolation.
1, 2, 3	-	The cut extends to the boundaries of the cutting grid without any offset.
4	-	The cut extends to *i*, *j*, or *k* = 1 without any offset or overlap.
5	-	The cut extends to *i*, *j*, or *k* = End without any offset or overlap.

16.13 $POINT Keywords

This keyword block is used to define a geometric point in the project.

COORDS

Coordinates of the point as real numbers separated by commas, e.g.,

```
$POINT
        COORDS = 0.1, 0.0, 0.5,
$END
```

16.14 $BOX Keywords

This keyword block is used to define a box in space for the blanking of regions.

<u>POINTS</u>

List of points defining the eight corners of a 3D box in space.

<u>BLOCKS</u>

List of blocks whose sections may be blocked where they lie within the box defined by poles.

<u>Example</u>

```
$BOX
        POINTS = 1, 2, 3, 4, 5, 6, 7, 8,
        BLOCKS = ‘BLOCK1’, ‘BLOCK2’,
$END
```

FAQ

1. *My computation gives me the error message "**STOPPING IN SUBROUTINE CXMU DUE TO NEGATIVE TEMPERATURE,**" and then stops running.*

   ANS: This is because the time step size is too large. Try to use a smaller time step size.

2. *My computation gives me the error message "**SOLUTION INTEGRITY IS IMPAIRED,**" and then stops running.*

   ANS: This is also because the time step size is too large. Try to use a smaller time step size.

3. *My computation gives me the error message "**NEGATIVE JACOBIAN**" in the beginning of the calculation.*

   ANS: This is because the block grids violate the right-hand rule. Try to modify the grid file to solve the problem.

Summary of AEROFLO Keywords

AEROFLO input files expect three types of data:

- Character strings
- Character string arrays
- Integer
- Integer arrays
- Real
- Real arrays
- Logical (.T. or .F.)

Global Variables

$GLOBAL

Keyword	Data Type	Default Value	Range	Required
TITLE	Character			
SIMTYPE	Integer	2	1-7	
FOLDER	Character			
MACH	Real			
REYNOLDS	Real			
ORDER	Integer	2		
STORAGE	Integer	1	0,1	

$TURB

Keyword	Data Type	Default Value	Range	Required
TURBTYPE	Integer	0	0-5	
DIRECTN	Integer Array (3)	0		TURBTYPE=4
NUMTRIPS	Integer	0		TURBTYPE=5
POINT	Integer Array (3)			1 to NUMTRIPS
PLANE	Integer Array (3)			1-3

$MHD

Keyword	Data Type	Default Value	Range	Required
LORSIG	Character	NULL		
BRDOCOMP	Character	NULL		
ESOLVE	Character	NULL		
MHDPRES	Real			
MHDREYN	Real			
RINTERCT	Real			
HALLPARM	Real			
IONSPARM	Real			
JOULEHT	Real			

$SPATIAL

Keyword	Data Type	Default Value	Range	Required
SCHEME	Integer Array (3)			YES
METRIC	Integer			YES
VISCOUS	Integer Array (3)	0, 0, 0,		
VISCHEME	Integer	0		
VISCMON	Integer	0		
SUTHERLN	Real	10000		
COMPACTI	Character(20)			If SCHEME(1) is 15
COMPACTJ	Character(20)			If SCHEME(2) is 15
COMPACTK	Character(20)			If SCHEME(3) is 15
COMPACTG	Character(20)			If METRIC is 1, 2, or 12
CUTOFFI	Real			If SCHEME(1) is 1-5
CUTOFFJ	Real			If SCHEME(2) is 1-5
CUTOFFK	Real			If SCHEME(3) is 1-5
ISOTROPI	Integer			
ISOTROPJ	Integer			
ISOTROPK	Integer			
ICUT	Integer Array (3)			
JCUT	Integer Array (3)			
KCUT	Integer Array (3)			
PERIODIC	Integer Array (3)	0,0,0	0,1	
BLOCK	Character (40)		Block names	

$TIMESTEP

Keyword	Data Type	Default Value	Range	Required
SCHEME	CHARACTER(3)		BW1, BW2, RK4	YES
SUBON	Integer	0	0,1	
MAXITER	Integer			YES
MAXSUB	Integer	0		
PRINTFRQ	Integer	1		
KTVDRK	Integer	0	0-5	
IBETA	Integer	1		
CFL	Real	1.0		
GLOBALDT	Real			YES
SUBDT	Real			YES
NDTAU	Integer	99999		
CFLMHD	Real	1.0		
DIAGONAL	Integer	0	0,1	
PRECON	Integer	0	0,1	
LREF	Real	1.0E-08		
GLOBTOL	Real	1.0E-30		
SUBTOL	Real	1.0E-03		
ACCEL	Integer	0	0,1	

$DAMPING

Keyword	Data Type	Default Value	Range	Required
TYPE	Integer	0	0-5	
ES2	Real	0.2		
ES4	Real	0.01		
FES2	Real	1.0		
FES4	Real	2.0		
MAXRED	Integer	0		
OMGAV	Real	0.0		
SRCONST	Real	0.0		
FILTER	Integer Array (3)	0	0,1	
NFILTER	Integer Array (3)	0		
ALPHAN	Real Array (3)	0.0	0.0-0.4999	
ALPHA1	Real Array (3)	0.0	0.0-0.4999	
ALPHA2	Real Array (3)	0.0	0.0-0.4999	
ALPHA3	Real Array (3)	0.0	0.0-0.4999	
ALPHA4	Real Array (3)	0.0	0.0-0.4999	
ALPHA5	Real Array (3)	0.0	0.0-0.4999	
ORDERN	Integer Array (3)	10	0-10	
ORDER1	Integer Array (3)	0	0-10	
ORDER2	Integer Array (3)	2	0-10	
ORDER3	Integer Array (3)	4	0-10	
ORDER4	Integer Array (3)	6	0-10	
ORDER5	Integer Array (3)	8	0-10	

$CASESPEC

Keyword	Data Type	Default Value	Range	Required
WALLTEMP	Real	1.0		
RMUM	Real			If MHD
SIGMAREF	Real			If MHD
SIGTYPE	Character			If MHD
SIGCOMP	Character			If MHD
NGAUSS	Integer			If MHD
AMPGAUSS	Real Array ()			
IGAUSS	Integer Array ()			
EPSGAUSS	Real Array ()			
DELGAUSS	Real Array ()			
XCTR	Real Array ()			
YCTR	Real Array ()			
ZCTR	Real Array ()			If MHD
REFTEMP	Real			If MHD
RINDEX	Real			If MHD
RKVL	Real			CASE = (31-39)
OMEGA	Real			CASE = (61,67)
AVALUE	Real			
LEADEDGE	Integer			CASE = (11,701-800)
YSURFACE	Real			CASE = (81)
ZSURFACE	Real			CASE = (91-99,111)
MAXRED	Integer			CASE = (111)
NUMDIPS	Integer			CASE = (111,161-166,168-170)
XDIPLOCS	Real Array ()			1-NUMDIPS CASE = (111,161-166,168-170)
YDIPLOCS	Real Array ()			
ZDIPLOCS	Real Array ()			
IDIPCOL	Integer Array ()			1-NUMDIPS CASE = (111)
DIPNORM	Integer			CASE = (167)
RXDIP	Real Array ()			1-NUMDIPS CASE = (161-166,168-170)
RYDIP	Real Array ()			
RZDIP	Real Array ()			
RMAGDIPS	Real Array ()			
EUNIFX	Real Array ()			CASE = (131)
EUNIFY	Real Array ()			
EUNIFZ	Real Array ()			
BUNIFX	Real Array ()			
BUNIFY	Real Array ()			
BUNIFZ	Real Array ()			

IRAMP	Integer			CASE = (141)
ICOWL	Integer			
ISWST	Integer			
ISWND	Integer			
ICOMND	Integer			
STRIPSGN	Integer			
STRIPSAC	Integer			
UFIXD	Real			CASE = (151,155)
VFIXD	Real			
WFIXD	Real			
BXFIXD	Real			
BYFIXD	Real			
BZFIXD	Real			
TILTANG	Real			
NUMELEC	Integer			CASE = (153,154,167)
ELECST	Real Array ()			1-NUMELEC
ELECND	Real Array ()			CASE = (153,154,167)
ELECPOT	Real Array ()			1-NUMELEC CASE = (167)
SLOPE	Real			
INTERCPT	Real			
MACH2	Real			
P2P1	Real			
RHO2RHO1	Real			
FLDEFAN	Real			

$POISSON

Keyword	**Data Type**	**Default Value**	**Range**	**Required**
NUMITER	Integer	0		
FREQ	Integer	0		
PRINTFRQ	Integer	0		
RELAXER	Real	0.0		
ALPHALIM	Real Array (2)	0.0		
MVL	Integer	0		
IFILTER	Integer	0	0,1	
SFILTER	Integer Array (3)	0		
EFILTER	Integer Array (3)			YES
GAUSS	Integer	1		
SIGMIN	Real	0.0		
SIGFAC	Real	0.0		
SIGDEF	Real	0.0		
COMPACTB	Character			YES
ALPHAN	Real	0.0	0.0-0.499	
ALPHA1	Real	0.0	0.0-0.499	
ALPHA2	Real	0.0	0.0-0.499	
ALPHA3	Real	0.0	0.0-0.499	
ALPHA4	Real	0.0	0.0-0.499	
ALPHA5	Real	0.0	0.0-0.499	
ORDERN	Integer	10	1-10	

ORDER1	Integer	0	1-10	
ORDER2	Integer	2	1-10	
ORDER3	Integer	4	1-10	
ORDER4	Integer	6	1-10	
ORDER5	Integer	8	1-10	

$OUTPUT

Keyword	**Data Type**	**Default Value**	**Range**	**Required**
PRINTFRQ	Integer	100		
FORMAT	Character		'CGNS', 'TECPLOT', 'PLOT3D',	
MOVIEON	Integer	0	0,1	
MOVIEFRQ	Integer			
IRANGE	Integer Array (3)	1,IE,1		
JRANGE	Integer Array (3)	1,JE,1		
KRANGE	Integer Array (3)	1,KE,1		
FILENAME	Character	aeromv.		
GRIDNAME	Character	aeromv.grid		
TECANIM	Integer	0	0,1	
TRANGE	Integer Array (3)	0	0, MAXITER	
ENTROPY	Integer	0	0,1	
IPERPROC	Integer	0	0,1	

$DEBUG

Keyword	**Data Type**	**Default Value**	**Range**	**Required**
WRITONLY	Integer	0	0,1	
POINT	Integer Array (3)	IE/2,JE/2,KE/2	1-IE/JE/KE	
NFIELDS	Integer	0		
IFIELD	Integer		1-10	1 to NFIELDS

$INITIAL

Keyword	**Data Type**	**Default Value**	**Range**	**Required**
BLOCK	Character	All blocks	Block names	
VARIABLE	Integer		1-8	YES
VALUE	Real	0.0		

BLOCK VARIABLES

$BLOCK

Keyword	Data Type	Default Value	Range	Required
NAME	Character (40)			YES
MESHFILE	Character (80)			YES
FILETYPE	Character (40)	AEROFLO	CGNS, AEROFLO PLOT3D	
ICFILE	Character (80)	“”		
ISIZE	Integer			YES
JSIZE	Integer			YES
KSIZE	Integer			YES
PERIODIC	Integer array (3)	0, 0, 0	0,1,2	
Following input required only for multi-block formulation				
DONORS	Character array		Block names	
ORDER	Integer	$GLOBAL/ORDER		
ANGLE	Real	0.0		
DISTANCE	Real	0.0		

$FACEBC

Keyword	Data Type	Default Value	Range	Required
BLOCK	Character		Block names	YES
SIDE	Integer		1-6	YES
TYPE	Integer		0-100	YES
VALUE	Real	0.0		
VARIABLE	Integer		1-8	Depends on TYPE
DEPTH	Integer	$BLOCK/ORDER		
ISPLANE	Integer	1		
ISNORMAL	Integer	1		
IBCS	Integer	1		
IBCE	Integer	$BLOCK/ISIZE		
JBCS	Integer	1		
JBCE	Integer	$BLOCK/JSIZE		
KBCS	Integer	1		
KBCE	Integer	$BLOCK/KSIZE		
CNORM	Real Array	0.0, 0.0, 0.0		
OBLOCK	Character		Block names	
OSIDE	Integer		1-6	
VAR1	Integer		-7, 3	
VAL1	Integer			
OPERATO1	Character (40)		+, -, *, xy, ABS, INT, ATAN, SIN, COS, EXP	
IOBLS	Image			
IOBCE	Image			
JOBCS	Image			
JOBLE	Image			
KOBCS	Image			
KOBLE	Image			

$OVERSET

Keyword	Data Type	Default Value	Range	Required
TYPE	Integer	0	0,1	
RESTART	Integer Array (3)	0	0-2	
COINNODE	Integer	0	0,1	
PROJECT	Integer	0	0,1	
NUMITER	Integer	2		
HSEADT	Integer	1	0,1	

$POINT

Keyword	Data Type	Default Value	Range	Required
COORDS	Real array (3)			YES

$BOX

Keyword	Data Type	Default Value	Range	Required
POINTS	Integer Array (8)		Point list	YES
BLOCKS	Character array		Block name list	YES

$MESHCUT

Keyword	Data Type	Default Value	Range	Required
CUTTER	Character (40)		Block name list	YES
BLOCKS	Character array		Block name list	YES
ORDER		2		
PERIODIC	Integer Array	0, 0, 0	0-5	

AEROFLO Papers

1 Cai, X. D. & Ladeinde, F. 2005. Comparative Studies of Two POD Methods for Airfoil Design Optimization," In "Computational Fluid and Solid Mechanics 2005," Edit. K. J. Bathe, Science Publishers, Amsterdam, The Netherlands. pp. 1227-1230.

2 Cai, X. & Ladeinde, F. 2001. Performance of Sub-grid Flamelet Model in LES of Reacting, Turbulent Flows. In "DNS/LES: Progress and Challenges", Greyden Press, Columbus Ohio., Edit. C. Liu, L. Sakell, & T. Beutner, pp. 299-310.

3 Ladeinde, F., Cai, X., Ramons, R. A. & Schlinker, R.H. 2008. "On the Connection between Near-field and Far- field Solutions of High Subsonic Jet Noise," Paper AIAA-2008-0011, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, January 6-10, 2008.

4 Cai, X. and Ladeinde, F. 2008. "Performance of WENO Scheme on Generalized Curvilinear Coordinate Systems," Paper AIAA-2008-0036, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, January 6-10, 2008.

5 Ladeinde, F. 2008. "Further Development of a High-Order Prediction Tool for Combustion at All Speeds," Paper AIAA-2008-0510, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, January 6-10, 2008.

6 Cai, X., Ladeinde, F. and Alabi, K. 2007. Hybrid RANS/LES Calculations of High Speed Jet Noise, AIAA-2007-3870, 37th AIAA Fluid Dynamics Conference and Exhibit, Miami, Florida, June 25-28, 2007

7 Cai, X. and Ladeinde, F. 2007. A Hybrid LES/RANS Calculation of Subsonic and Supersonic, Hot Jet Noise, GT2007-28117, Turbo Expo 2007, Montreal, Canada, May 14-17, 2007.

8 Alabi, K., Ladeinde, F., 2007. High-Order Dynamic Overset Procedure Applied to Moving Body Calculations. AIAA Paper AIAA-2007-248, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2007.

9 Safta, C., Alabi, K., Ladeinde, F., Cai, X., 2007. A Combined Level- Set/Mixture Fraction/Progress- Variable Approach for Partially- Premixed Turbulent Reacting Flows. AIAA Paper AIAA-2007-1436, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2007.

10 Cai, X., Ladeinde, F., Alabi, K., 2007. Towards Predicting Supersonic, Hot Jet Noise. AIAA Paper AIAA-2007-826, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2007.

11 Ladeinde, F., Alabi, K., Safta, C., Cai, X., Johnson, F. 2006. The First High-Order CFD Simulation of Aircraft: Challenges and Opportunities. AIAA Paper AIAA-2006-1526, 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2006.

12 Safta, C., Alabi, K., Ladeinde, F., Cai, X., Kiel, B., Sekar, B. 2006. Comparative advantages of high-order schemes for subsonic, transonic, and supersonic flows. AIAA Paper AIAA-2006-299, 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2006.

13 Safta, C., Alabi, K., Ladeinde, F. 2006. Level-Set Flamelet/Large-Eddy Simulation of a Premixed Augmentor Flame Holder. AIAA Paper AIAA-2006-156, 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV,. Jan. 08-11, 2006.

14 Ladeinde, F., Safta, C., Sekar, B., Lynch, A. and Kiel, B. 2005. Validation of Advanced Large Eddy Simulation Methods for Augmentor Applications. 53rd Joint Army-Navy-NASA-Air Force (JANNAF) Meeting, Monterey, CA.

15 Cai, X. and Ladeinde, F. 2006. An Evaluation of the Partially-Resolved Numerical Simulation Procedure For Near-Wall Performance. AIAA 2006-0115, The 44th AIAA Aerospace Sciences Meeting and Exhibit, 9-12 January 2006, Reno, Nevada.

16 Safta, C., Alabi, K. and Ladeinde, F. 2005. Comparative advantages of high-order schemes for subsonic, transonic, and supersonic flows. AIAA-2006-0299, The 44th AIAA Aerospace Sciences Meeting and Exhibit, 9-12 January 2006, Reno, Nevada

17 Alabi, K., Safta, C. and Ladeinde, F. 2005. High-order hole-cut procedure for realistic geometries at all speeds. The 44th AIAA Aerospace Sciences Meeting and Exhibit, 9-12 January 2006, Reno, Nevada

18 Cai, X., Ladeinde, F. and Alabi, K. 2006. High-fidelity Aeroelastic Computations for Realistic Aerospace Systems. Paper Submitted to The 44th AIAA Aerospace Sciences Meeting and Exhibit, 9-12 January 2006, Reno, Nevada.

19 Cai, X. D. & Ladeinde, F. 2005. Comparative Studies of Two POD Methods for Airfoil Design Optimization," 3rd MIT Conference on CFD/CSM, Cambridge, Massachusetts.

20 Alabi, K. & Ladeinde, F. 2005. Treatment of Blank Nodes in a High-Order Overset Procedure. AIAA-2005-1269. 43rd AIAA Aerospace Sciences Meeting and Exhibit, 10-13 January 2005, Reno, Nevada

21 Cai, X.D. & Ladeinde, F. 2004. A Comparison of Two POD Methods for Airfoil Design Optimization, AIAA-2005-4912, 4th AIAA Theoretical Fluid Mechanics Meeting, 9-12 January 2006, Reno, NV.

22 Alabi, K. & Ladeinde, F. 2004. Parallel, High-Order Overset Grid Implementation for Supersonic Flows. AIAA Paper 2004-0437, 42nd AIAA Aerospace Sciences and Exhibit, Reno, NV, 5-8 January 2004.

23 Cai, X. & Ladeinde, F. 2004. High-Order Formulation for a Hybrid ROM method for linear aeroelasticity. AIAA Paper 2004-0891, 42nd AIAA Aerospace Sciences and Exhibit, Reno, NV, 5-8 January 2004.

24 Cai X. & Ladeinde, F. 2004. A Unified Computational Methodology for Rarefied and Continuum Flow Regimes. AIAA Paper 2004-1178, 42nd AIAA Aerospace Sciences and Exhibit, Reno, NV, 5-8 January 2004.

25 Ladeinde, F., Cai, X., Visbal, M.R., & Gaitonde, D. 2003. Parallel Implementation of Curvilinear High-Order Formulas. *Int. Journal of Computational Fluid Dynamics* Vol. 17 (6), pp. 467-485.

26 Ladeinde, F., Cai, X., Visbal, M.R., & Gaitonde, D. 2001. Turbulence Spectra Characteristics of High Order Schemes for Direct and Large Eddy Simulation. J. Applied Numerical Mathematics Vol. 36 (2001), pp. 447-474.

27 Ladeinde, F., Cai, X., Visbal, M.R., & Gaitonde, D. 2001. Parallel Computation of Complex Aeroacoustic Systems, AIAA 2001-1118, 39th Aerospace Sciences Meeting and Exhibit, Reno, NV.

28 Ladeinde, F., Cai, X., Sekar, B., & Kiel, B. 2001. Application of Combined LES and Flamelet Modeling to Methane, Propane, and Jet-A Combustion, AIAA 2001-0634, 39th Aerospace Sciences Meeting and Exhibit, Reno, NV

29 Ladeinde, F., Cai, X. and Sekar, B. 2000. Flamelet Studies of Reduced and Detailed Kinetic Mechanisms for Methane/Air Diffusion Flames. Paper 2000-GT-0144, Proceedings of IGTI, ASME TURBO EXPO 2000, May 8-11, 2000, Munich, Germany

30 Ladeinde, F., Cai, X., Visbal, M.R. & Gaitonde, D. 2000. Studies of DRP and Compact Schemes for Aeroacoustic Simulation. AIAA Paper 2000-2330, Fluids 2000, Denver, CO.

31 Ladeinde, F. 1998. Truly Automatic CFD Mesh Generation with Support for Reverse Engineering. AIAA Paper 99-0828.

Reference

1. Abid, R., “Evaluation of Two-Equation Turbulence Models for Predicting Transitional Flows,” *Int. J. Engng Sci.,* Vol. 32 (6), 1993, pp. 831-840.
2. Anderson, D. A., Tannehill, J. C., and Pletcher, R. H., *Computational Fluid Mechanics and Heat Transfer*, McGraw-Hill, New York, 1984.
3. Beam, R. M. and Warming, R.F., “An Implicit Factored Scheme for the Compressible Navier-Stokes Equations,” *AIAA Journal*, Vol. 16, No. 4, 1978, pp. 393-402.
4. Germano, M., Piomelli, U., Moin, P., and Cabot, W.H., “A Dynamic Subgrid-Scale Model for Compressible Turbulence and Scalar Transport,” *Phys. Fluids A,* 3(7), 1991, pp. 1760.
5. Launder, B. and Sharma, B., “Application of the Energy Dissipation Model of Turbulence to the Calculation of Flow near a Spinning Disc,” *Lett. Heat and Mass Transfer 1*, 1974, 131-138.
6. Lele, S.K., “Compact Finite Difference with Spectral-Like Resolution,” *Journal of Computational Physics,* Vol. 103, No. 1, 1992, pp. 16-42.
7. Menter, F.R., “Zonal Two-Equation k-omega Turbulence Models for Aerodynamic Flows,” AIAA 1993-2906.
8. Pulliam, H.T. and Chaussee, D. S., “A Diagonal Form of an Implicit Approximate-Factorization Algorithm,” *J. Comp. Phys.,* 39 (2), 1981, pp. 347.
9. Pope, S.B., *Turbulence Flows*, Cambridge University Press, 2000.
10. Roe, P.L., “Characteristic-Based Schemes for the Euler Equations,” *Annual Review of Fluid Mechanics,* Vol. 18, 1986, pp. 337-365.
11. Shih, T.-H. and Liu, N.-S., “Partially Resolved Numerical Simulation: from RANS towards LES for Engine Turbulent Flows,” AIAA 2004-0160.
12. Shu, C.-W., “Essentially Non-Oscillatory and Weighted Essentially Non-Oscillatory Schemes for Hyperbolic Conservation Laws,” ICASE Report No. 97-65, November 1997.
13. Smagorinsky, J., “General Circulation Experiments with the Primitive Equations: I. The Basic Equations,” *Mon. Weather Rev.,* 91, 1963, pp. 99-164.
14. Spalart, P.R. and Allmaras, S.R., “A One-Equation Turbulence Model for Aerodynamic Flows,” *La Rech. A’erospatiale*, V.1, 1994, pp. 5-21.
15. 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,” *In Advance in DNS/LES,* C. Liu and Z. Liu Eds, Greyden Press, Columbus, Ohio, 1997.
16. van, Leer, B., “Towards, the Ultimate Conservative Difference Scheme V. A Second-Order Sequential to Godunov’s Method,” *Journal of Computational Physics,* Vol. 32, 1979, pp. 101-136.
17. Wilcox, D.C., *Turbulence Modeling for CFD*, 2nd Edition, DCW Industries, 1998.