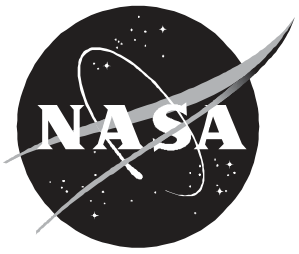
NASA/TM-1998-208444



CFL3D User's Manual (Version 5.0)

Sherrie L. Krist
BaNANePOS, Inc., Hampton, Virginia

Robert T. Biedron and Christopher L. Rumsey
Langley Research Center, Hampton, Virginia

June 1998

The NASA STI Program Office . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program Office plays a key part in helping NASA maintain this important role.

The NASA STI Program Office is operated by Langley Research Center, the lead center for NASA's scientific and technical information. The NASA STI Program Office provides access to the NASA STI Database, the largest collection of aeronautical and space science STI in the world. The Program Office is also NASA's institutional mechanism for disseminating the results of its research and development activities. These results are published by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part or peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services that help round out the STI Program Office's diverse offerings include creating custom thesauri, building customized databases, organizing and publishing research results . . . even providing videos.

For more information about the NASA STI Program Office, see the following:

- Access the NASA STI Program Home Page at ***http://www.sti.nasa.gov***

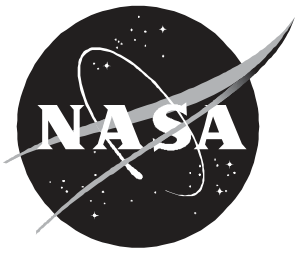
- Email your question via the Internet to help@sti.nasa.gov

- Fax your question to the NASA Access Help Desk at (301) 621-0134

- Phone the NASA Access Help Desk at (301) 621-0390

- Write to:
  NASA Access Help Desk
  NASA Center for AeroSpace Information
  7121 Standard Drive
  Hanover, MD 21076-1320

NASA/TM-1998-208444



CFL3D User's Manual (Version 5.0)

Sherrie L. Krist
BaNANePOS, Inc., Hampton, Virginia

Robert T. Biedron and Christopher L. Rumsey
Langley Research Center, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

June 1998

Available from the following:

NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
(301) 621-0390

National Technical Information Service (NTIS)
5285 Port Royal Road
Springfield, VA 22161-2171
(703) 487-4650

Table of Contents

Nomenclature

Symbol:	Definition:
A, **B**, **C**	flux Jacobians
a	speed of sound
b	part of viscous term in energy equation; also, compression parameter in minmod flux limiter
C	constant
C_μ	modeling variable in turbulence model equations
D	diffusion term in turbulence model equations
D_1, D_2	damping terms in Baldwin-Barth turbulence model
d	distance to nearest wall; also, directed distance to the wall
E	total energy
e	total energy per unit volume
F	function
f	damping function; also, frequency
F, **G**, **H**	inviscid fluxes
$\mathbf{F}_v$, $\mathbf{G}_v$, $\mathbf{H}_v$	viscous fluxes
G_1	modeling variable (similar to $-C_\mu$ term) in turbulence model equations
H	total enthalpy
I	restriction operator
I	identity matrix
i, j, k	grid indices
J	transformation Jacobian, $J = \partial(\xi, \eta, \zeta)/\partial(x, y, z)$
k_r	reduced frequency
k_g	growth rate

Symbol:	Definition:
k	kinetic energy in turbulence model equations
$\tilde{L}$	characteristic length
L_{ref}	length in grid corresponding to $\tilde{L}$
$\tilde{L}_R$	reference length used by code $\tilde{L}_R = \tilde{L}/L_{ref}$
l	reference length; also, length scale in Baldwin-Lomax turbulence model
M	Mach number, $M = \lvert\tilde{\mathbf{V}}\rvert/\tilde{a}$
$\mathbf{M}$	transformation matrix from conserved variables to primitive variables, $\partial\mathbf{Q}/\partial\mathbf{q}$
m	sub-iteration counter
N	spatially-factored implicit matrix term
n	direction normal to the wall
n	current iteration
P	production term in turbulence model equations
Pr	Prandtl number
p	static pressure
$\mathbf{Q}$	conserved variables
$\mathbf{q}$	primitive variables
$\dot{q}$	heat flux terms
$\mathbf{R}$	residual vector
R	turbulent Reynolds number term in Baldwin-Barth turbulence model; also, residual term
$Re_{\tilde{L}}$	Reynolds number, $Re_{\tilde{L}} = \tilde{\rho}_\infty\lvert\tilde{\mathbf{V}}\rvert_\infty\tilde{L}/\tilde{\mu}_\infty$
S	mean rate-of-strain tensor
S_P	production source term in turbulence model equations
S_D	destruction source term in turbulence model equations
s	entropy; also, parameter used in smooth flux limiter
$\mathbf{T}$	matrix of eigenvectors
t	time; also, parameter used in smooth flux limiter
U, V, W	contravariant velocities

Symbol:	Definition:
u, v, w	Cartesian velocities in x, y, z directions
u^+	law-of-the-wall variable, $u^+ = \tilde{u}\sqrt{\tilde{\rho}/\tilde{\tau}_w}$
V	corrections on coarser meshes, used to update finer mesh in the multigrid algorithm
V	velocity vector, (u, v, w)
$\|\mathbf{V}\|$	total velocity
W	mean vorticity tensor
X	represents either k, ω, or ε in general turbulence model equations
x, y, z	Cartesian coordinates
y^+	law-of-the-wall variable, $y^+ = \sqrt{\tilde{\rho}\tilde{\tau}_w}\tilde{y}/\tilde{\mu}$
α	angle of attack; also, used as constant in turbulence model equations
β	side-slip angle; also, used as constant in turbulence model equations
γ	ratio of specific heats, $\gamma = 1.4$; also, variable in turbulence model equations
Δ	incremental quantity; also, forward difference operator
δ	difference operator
ε	dissipation term in turbulence model equations; also, small constant used in flux limiters
κ	spatial differencing parameter; also, von Karman constant used in turbulence model equations
Λ	matrix of eigenvalues
λ	bulk viscosity coefficient
μ	molecular viscosity coefficient
ν	kinematic viscosity
$\hat{\nu}$	field equation variable in Spalart-Allmaras turbulence model
ξ, η, ζ	general curvilinear coordinates; also, η and ζ used as variables in EASM turbulence model equations
ρ	density
τ	shear stress tensor; also, relative truncation error
ϕ	parameter governing the temporal order of accuracy of the scheme
Ω	magnitude of vorticity

Symbol:	Definition:
ω	rotational velocity; also, variable in turbulence model equations, $\omega = \varepsilon/k$
∇	gradient operator $\nabla X = \partial X/\partial x_i$

Subscripts

Symbol:	Definition:
e	denotes estimated value
i, j, k	denote grid indices; also, summation convention where specified
inv	denotes inviscid part
k	denotes k turbulence model quantity
L	denotes left-hand state; also, denotes laminar quantity where specified
l	denotes summation convention
R	denotes right-hand state
T	denotes turbulent quantity
t	denotes total quantity; e.g. $p_t \Rightarrow$ total pressure ; also, denotes differentiation with respect to time
v	denotes viscous term
w	denotes wall condition
x, y, z	denote differentiation with respect to x, y, z ; x also denotes tensor notation where specified
ε	denotes ε turbulence model quantity
ω	denotes ω turbulence model quantity
∞	denotes reference conditions, typically free-stream conditions
+, –	denotes forward or backward difference operator
ξ, η, ζ	denotes reference to a particular coordinate direction

Superscripts

Symbol:	Definition:
b	denotes biased gradient
c	denotes correction term

Superscripts

Symbol:	Definition:
r	denotes residual smoothing term
^	denotes quantities in generalized coordinates
~	denotes dimensional value; also, denotes Roe-averaged variable where specified
′	denotes partial derivative with respect to **q** and intermediate values in the time advancement scheme
→	denotes a vector quantity
+, –	denotes forward or backward difference operator

Abbreviations

CFD	Computational Fluid Dynamics
CFL	Courant number
CFL3D	Computational Fluids Laboratory 3-Dimensional (flow solver)
CPU	Central Processing Unit
EASM	Explicit Algebraic Stress Model
FAST	Flow Analysis Software Toolkit[44]
LRR	Launder-Reese-Rodi
GRIDGEN	GRID GENeration[36]
MaGGiE	MultiGeometry Grid Embedder[11]
NACA	National Advisory Committee for Aeronautics
PLOT3D	PLOT 3-Dimensional[43]
RAE	Royal Aircraft Establishment
SSG	Speziale-Sarkar-Gatski
SST	Shear Stress Transport
TLNS3D	Thin-Layer Navier-Stokes 3-Dimensional (flow solver)[42]

CHAPTER 1 Introduction

Welcome to CFL3D (Version 5.0), a Reynolds-Averaged thin-layer Navier-Stokes flow solver for structured grids. The original version of CFL3D was developed in the early 1980's in the Computational Fluids Laboratory at NASA Langley Research Center; hence the name of the code, which is an acronym for the Computational Fluids Laboratory 3-Dimensional flow solver. As the number of people who have utilized CFL3D has increased over time, so have the demands on the code. Consequently, it is constantly "under construction" with numerous researchers having contributed to code upgrades. Currently, the primary developers/supporters of CFL3D are Dr. Christopher L. Rumsey and Dr. Robert T. Biedron of the Aerodynamics and Acoustics Branch at NASA Langley. An overview of the history of the code can be found in Rumsey, Biedron, and Thomas.[32]

CFL3D solves the time-dependent conservation law form of the Reynolds-averaged Navier-Stokes equations. The spatial discretization involves a semi-discrete finite-volume approach. Upwind-biasing is used for the convective and pressure terms, while central differencing is used for the shear stress and heat transfer terms. Time advancement is implicit with the ability to solve steady or unsteady flows. Multigrid and mesh sequencing are available for convergence acceleration. Numerous turbulence models are provided, including 0-equation, 1-equation, and 2-equation models. Multiple-block topologies are possible with the use of 1-1 blocking, patching, overlapping, and embedding. CFL3D does not contain any grid generation software. Grids must be supplied extraneously.

Version 5.0 of CFL3D has several additional utilities over earlier versions of the code. Most notably, Version 5.0 has the capability to employ sliding patched-zone interfaces, such as might be required to perform rotor-stator computations, for example. However, it is stressed here that CFL3D has been developed primarily as a tool for external aerodynamics analysis. Its use for internal turbomachinery applications has been only as a basic research code thus far; other Navier-Stokes codes specifically designed for turbomachinery applications, such as ADPAC[22], UncleTurbo[23], or ROTOR[29] may be better suited to the analysis of such flows.

The purpose of this user's manual is to describe the code and provide instruction for its usage. Chapter 2 explains the set up and running of the code. Various files available with CFL3D are described and directions for their usage are provided. Step by step instructions are listed for running CFL3D, as well as the pre-processing codes needed for grid-overlapping and patching.

Chapter 3 provides a line-by-line description of all the input parameters utilized in CFL3D. This will probably be the chapter referred to the most. It will aid users in understanding the sample problems as well as in setting up their own problems. While some parameters are strictly case dependent, some are fairly general and recommended values are usually indicated. Certain problems may, of course, require altering the recommended values of these parameters as well.

To avoid the use of feet, meters, pounds, grams, etc. in coding the equations, CFL3D solves the Navier-Stokes equations in "nondimensional" form. Each flow-field parameter is nondimensionalized by reference values. For example, all points on a wing may be nondimensionalized by the chord length of the wing. The nondimensionalizations used in CFL3D are described in Chapter 4.

Chapter 5 explains the file formats. The most important of these is the grid file format, since users will need to translate their grids into an appropriate CFL3D format. The restart file format may be needed for user-designed post-processing programs. This chapter also describes the various output files.

Besides providing general information about boundary condition usage in CFL3D, Chapter 6 discusses the various physical boundary conditions available. It also provides descriptions of the multiple block capabilities in CFL3D, mainly, 1-1 blocking, patching, grid overlapping, and grid embedding. Input examples for the various techniques are provided.

The advantages of using multigrid for convergence acceleration are explained in Chapter 7. The theory behind multigrid and its usage are discussed. The use of multigrid with embedded grids is described. Mesh sequencing is also defined and discussed. Input examples are provided there as well.

Chapter 8 discusses time accurate simulation of unsteady flows using sub-iteration schemes. Two types of sub-iterations are currently implemented in CFL3D. The effects of the different types of sub-iterations, as well as the strategy for pursuing time-accurate computations in general, are described.

Chapter 9 contains several test cases for the user to practice with when learning how to run CFL3D. Two-dimensional sample cases include a single block RAE airfoil case, a grid-overlapping NACA 0012 airfoil case, a patched grid NACA 0012 airfoil case, and a grid-overlapping multielement airfoil case. Also described are examples for a fixed flat plate, a vibrating plate, a multistream nozzle, and a rotor-stator. The three-dimensional test cases include an axisymmetric bump, a single block F-5 wing, a single block ONERA M6 wing, and a single block delta wing.

Finally, Chapter 10 is dedicated to troubleshooting. It is basically a compilation of suggestions for common problems encountered by users over the years. It is recommended that this chapter be read at the onset of any difficulties which occur while using CFL3D. Precious time may be saved by learning from the experiences (and, yes, the mistakes!) of

others. It is prudent at this point to suggest to the user that any problems encountered while using CFL3D be reported to the support personnel so that others may benefit from hitherto unknown experiences that may occur.

There are several appendices at the end of this manual which provide more detailed information about the code. Appendix A describes the governing equations. Appendix B explains the time advancement procedure. The spatial discretizations of the inviscid and viscous fluxes are described in Appendix C. The multigrid algorithm is described in Appendix D. The matrix needed to convert the equations from conservative variables to primitive variables is derived in Appendix E. A step-by-step explanation of how to convert the governing equations from Cartesian coordinates to generalized coordinates is provided in Appendix F. Appendix G shows how the forces and moments are calculated. Appendix H describes the turbulence models available in CFL3D. Appendix I illustrates the angle (α and β) definitions used in the code. Appendix J contains a list of all the subroutines in CFL3D and their purposes. Finally, Appendix K explains the differences between Versions 4.1 and 5.0 of CFL3D.

The last two sections of the manual are source listings. Beginning on page 337, there is a partial list of papers that have been published over the years as CFL3D has developed. The last section of the manual is a list of the references used for this document.

It is appropriate at this point to discuss some of the terminology used in the CFL3D *User's Manual*. Terms, such as grid, zone, mesh, and block, are often used interchangeably in CFD discussions and literature. For clarity in this manual, the following distinctions are defined. The *grid* refers to those sets of points that define the flow field and are generated by the user to be read in by CFL3D. A grid may be one entity or it may be composed of several component grids. These component grids may also be called *grid zones*. The grid zones may communicate with one another through 1-1 blocking, patching, overlapping, or embedding. The term *block* is used for bookkeeping purposes. All component grids are blocks and are assigned a block number. The grid or the set of component grids is often referred to as the *global* grid, encompassing the entire flow field. When the component grids are coarsened for such options as multigrid and mesh sequencing, the coarser levels are labelled as blocks and are also assigned block numbers. *Embedded* grids are finer than the component grid in which they reside, but do not encompass the entire zone. Instead, they are placed in regions known to have high gradients to further resolve those areas.

Another terminology definition that should be clarified is the phrase *free stream*. As used throughout this manual, the words *free stream* imply a reference state. CFL3D was developed primarily as an external flow solver, in which case the appropriate reference state *is* free stream. However, for purely internal flows, the concept of free stream has no meaning and the more general concept of *reference state* should be used.

Further distinctions between terms are required in discussing the methods of communication between blocks. When two blocks share a face or a portion of a face and the grid points correspond point to point, the boundary condition communication set up between

the two blocks is called *1-1 blocking*. The blocks in this case are often referred to as being C^0 *continuous*. Grid *patching*, on the other hand, refers to the boundary condition interpolations set up between blocks that share a common face or portion of a face, but which do not match point to point. For example, a fine grid face of one block could pass flow information to a coarse grid face of another block as long as the two blocks shared the face. Grid *overlapping* has neither the restriction of point to point connectivity or a common face between blocks. For example, a cylindrical "inner" grid could communicate with a cubic "outer" grid even though the grid topologies are completely different. The terms *overset grids* and *chimera grids* are synonymous with overlapped grids.

A distinction should also be made for the methods of time advancement. The term *time accurate* means that the flow is actually tracked in time. For example, if the flow is allowed to advance for ten seconds, the flow conditions at a certain point in the flow field might have ten different values, one at each second. For a *steady state* problem, after convergence is reached, the flow conditions will not change with time.

Here is one final tip for reading this manual. In discussions and examples, input variables are typed in **bold** print. Such items as variable names, array names, lines of actual code, input samples, etc. are printed in the `Courier` font. Note that these conventions do not necessarily convey in the World-Wide-Web version of this manual. The CFL3D internet address is: http://fmad-www.larc.nasa.gov/~rumsey/CFL3D/cfl3d.html .

Hopefully, this manual will answer most questions encountered when running CFL3D. If problems arise that are not addressed in the sample inputs or in Chapter 10, please contact

Dr. Christopher L. Rumsey
(757)864-2165
c.l.rumsey@larc.nasa.gov

or

Dr. Robert T. Biedron
(757)864-2156
r.t.biedron@larc.nasa.gov

NASA Langley Research Center
Mail Stop 128
Hampton, VA 23681-0001

Please also let these representatives know if any "bugs" or other useful insights are discovered during the usage of CFL3D.

Every effort has been made to insure that this document represents the correct usage and theoretical underpinnings of CFL3D. The intention has been to make this manual as complete and as self-consistent as possible. However, no guarantee can be given that this manual covers all possible aspects of the code and its usage or that this manual is without error.

CHAPTER 2 Getting Started

While the ultimate goal in utilizing CFL3D will be to solve problems pertinent to the user's work, it is highly recommended for new users to begin by running some of the test cases described in Chapter 9. Experience gained while running these will help the user when running a "real" problem. The following sections provide general instructions for compiling and running CFL3D.

2.1 Acquiring the Code and Example Files

The files needed to run CFL3D are located on Vonneumann and Eagle, two Cray supercomputers located at NASA Ames Research Center. The files are in

```
~rumsey/Cfl3dv5/cfl3dv5.ascii
```

The files are tarred, compressed, encrypted, and uuencoded. A keyword (obtained from the Aerodynamic and Acoustic Methods Branch at NASA Langley – see below) must be used to decode the file `cfl3dv5.ascii`. The steps for this procedure are as follows:

Step 1

```
uudecode cfl3dv5.ascii
```

Step 2

```
crypt keyword < cfl3dv5.crypt > cfl3dv5.tar.Z
```

Step 3

```
uncompress cfl3dv5.tar.Z
```

Step 4

```
tar xvf cfl3dv5.tar
```

This step should create a directory named `Cfl3dv5` with the appropriate CFL3D files in it.

Step 5

```
rm cfl3dv5.ascii cfl3dv5.tar cfl3dv5.crypt
```

If the test cases are not required, a more convenient option might be to obtain the file `cfl3dv5.codeonly.ascii`. This file is also tarred, compressed, encrypted, and uuencoded, but it is much smaller than `cfl3dv5.ascii`.

If access to Vonneumann or Eagle is not possible, the encrypted codes are also available on the Langley CFD anonymous ftp site. To obtain the code in this manner, take the following steps:

Step 1

```
ftp tabdemo.larc.nasa.gov
```

or

```
ftp 128.155.24.42
```

Step 2

When prompted for the username, type

```
anonymous
```

Step 3

Type the user's e-mail address as the password.

Step 4

```
cd incoming
```

Step 5

```
cd Rumsey
```

Step 6

```
get cfl3dv5.ascii
```

or

```
get cfl3dv5.codeonly.ascii
```

To obtain the current keyword needed to decode the source files, send mail or e-mail to Dr. Christopher L. Rumsey (see page 4) requesting the code, along with an explanation describing the planned usage of the code. Include a FAX number with the request. All users will be asked to fill out and sign a CFL3D Usage Agreement form. After the completed form is received, the keyword will be provided.

After obtaining CFL3D, it is required that the user remember not to distribute any part of the code to others outside of his or her own working group without prior permission

from NASA Langley. The CFL3D code is currently restricted to use within the United States only.

2.1.1 The Code and Supplementary Files

The items listed below are made available to the user when CFL3D is obtained. The names in **bold** face indicate a directory.

2dtestcases	isrcheq_wkstn.f	makeprecfl3d_hp_sngl
3dtestcases	lbcx.f	makeprecfl3d_rs6000
Advice	makecfl3d_cray	makeprecfl3d_sgi
Maggie	makecfl3d_decalpha	makeprecfl3d_sgi_R10000
Multitask_cray	makecfl3d_hp_dbl	makeprecfl3d_sgi_R8000
README	makecfl3d_hp_sngl	makeprecfl3d_sun
Ronnie	makecfl3d_rs6000	plot3d_hp_dbl.f
Tools	makecfl3d_sgi	precfl.h
af3f.f	makecfl3d_sgi_R10000	precfl3d.f
bc.f	makecfl3d_sgi_R8000	rhs.f
cbsem.f	makecfl3d_sun	second_hp_dbl.f
cvmgt_wkstn.f	makeprecfl3d_cray	second_rs6000.f
dynptch.f	makeprecfl3d_decalpha	second_wkstn.f
input.doc	makeprecfl3d_hp_dbl	turbs.f

The CFL3D code consists of one main driver subroutine package, `cbsem.f`, and six sets of library subroutines, `bc.f`, `rhs.f`, `af3f.f`, `lbcx.f`, `turbs.f`, and `dynptch.f`. `Cbsem.f` contains the main subroutine (`mgblk`) of the program as well as many of the routines needed for the input and output of information. The file `bc.f` contains the boundary condition subroutines, including physical boundary conditions, 1-1 blocking, and grid embedding. It also contains the necessary routines for processing supplemental overlapped-grid and static patched-grid interpolation information. `Rhs.f` contains the subroutines needed for solving the right-hand side of the governing equations, while `af3f.f` contains the subroutines needed for calculating the left-hand side of the governing equations. (See Appendix A.) `Lbcx.f` contains a variety of subroutines, including those needed for multigrid and mesh sequencing, those providing the metrics and other grid information, those involved with computing the forces, and several others. The file `turbs.f` contains the turbulence models subroutines. Finally, `dynptch.f` contains the subroutines needed for dynamic grid patching.

Some supplementary files are also needed for CFL3D. A “makefile” is used to compile and link subroutines and to create an executable for the code. A preprocessing program called `precfl3d.f` is used to determine the memory and array dimensions needed by CFL3D for a particular case. For `precfl3d.f`, use `makeprecfl3d_`*`machinename`*. Here, and in the discussions that follow, the terms in italics pertain to user-specific items. For

instance, *machinename* might be cray, sgi, or sun for the Cray supercomputer, SGI, and Sun workstations, respectively. Precfl.h is the parameter file required by this makefile. The array dimensions set in precfl.h should be large enough for nearly any problem and should rarely need modification; if a larger value of a particular parameter is required, precfl3d will stop and request the user to increase the relevant parameter.

A makefile called makecfl3d_*machinename* is also needed to compile and link the CFL3D subroutines and create the executable, cfl3d. Several files are needed to compile the code on workstations. These files contain Fortran source code for the corresponding Cray intrinsic functions. For instance, when utilizing makecfl3d_rs6000, make sure that second_rs6000.f is available. Likewise, when using makecfl3d_hp_dbl, make sure that plot3d_hp_dbl.f and second_hp_dbl.f are in the working directory. Note that for double precision on the HP, the user must actually *remove* the subroutines plot3d, plot3c, and plot3t from cbsem.f prior to compiling with makecfl3d_hp_dbl. For all other makecfl3d makefiles, cvmgt_wkstn.f, isrcheq_wkstn.f, and second_wkstn.f are utilized and therefore must be available. (Note that wkstn is short for "workstation".)

The file input.doc contains the information found in Chapter 3. It is a listing and description of the input parameters used in CFL3D.

The files in directory **2dtestcases** are:

0012_patch	**Flatplate**	**Rae10**
0012_xmera	**Multielem**	**Rotorstator**
README.2d	**Multistream**	**Vibrate**

The files in directory **3dtestcases** are:

README.3d	**Delta**	**Oneram6**
Axibump	**F5wing**	

Directory **2dtestcases** contains various two-dimensional test cases. A patching example involving the NACA 0012 airfoil is located in **0012_patch**. In **0012_xmera**, a case utilizing grid overlapping for the same airfoil is found. A three-element airfoil case involving grid overlapping is set up in **Multielem**. Directory **Rae10** contains a single block case for the RAE 2822 airfoil. **Flatplate** contains the files needed to run the flat plate test case. A multistream nozzle case is in **Multistream**. In **Vibrate** are the files for a vibrating plate case. A rotor-stator case is set up in **Rotor**.

The three-dimensional test cases are located in the directory **3dtestcases**. A case solving the flow over an axisymmetric bump is in **Axibump**. Other cases currently available are for a delta wing (in **Delta**), an F-5 wing (in **F5wing**), and an Onera M-6 wing (in **Oneram6**). The steps for running these test cases are described in Chapter 9.

The files in directory **Advice** are:

cfl3dadvice.give	cfl3d.references	v4_to_v5.inputdif
cfl3dadvice.grid	cfl3d.turb.references	

The first four files contain the same information given in Chapter 10 and in "CFL3D Papers" on page 337. The v4_to_v5.inputdif file describes the differences between the Version 4.1 and Version 5.0 input files. (This information is also given in Appendix K.)

The files in directory **Maggie** are:

README	maggie.doc	makemaggie_sgi
ismax_wkstn.f	maggie.f	makemaggie_sun
ismin_wkstn.f	makemaggie_cray	second_wkstn.f
mag1.h	makemaggie_hp_sngl	

The files in the **Maggie** directory are needed for cases involving grid overlapping. The chimera scheme code is called maggie.f. Maggie.doc explains the input parameters for maggie.f. Makefiles for various machines are available and require mag1.h, which contains the appropriate dimensions for the problem at hand. Make sure ismax_wkstn.f, ismin_wkstn.f, and second_wkstn.f are available before running MaGGiE on a workstation.

The files in **Multitask_cray** are:

README	makecfl3d_cray_multi	xlim_multi.f
fhat_multi.f	tinvr_multi.f	

These files should be used in conjunction with the standard CFL3D files when Cray multitasking is desired.

The files in directory **Ronnie** are:

README	makeronnie_sgi	ronnie.f
makeronnie_cray	makeronnie_sun	second_wkstn.f
makeronnie_decalpha	ron1.h	
makeronnie_hp_sngl	ronnie.doc	

When utilizing grid patching, the files in the **Ronnie** directory are needed. The code that sets up the patching interpolation stencils is called ronnie.f. Ronnie.doc explains the input parameters for ronnie.f. Makefiles for various machines are available and utilize ron1.h in the compilation. The user must set the parameters in ron1.h for each particular case. Also, make sure second_wkstn.f is available before running ronnie on a workstation.

The files in the **Tools** directory are:

```
README                  make1to1.f              reverseijk.f
everyother.f            make_p3dtotec3d_cray    v4tov5_input.f
everyotherp3d.f         p3dtotec3d.f            v4tov5_restart.f
historytec3d.f          p3dtotec3d_pre.c        v5inpdoubhalf.f
```

These programs are useful for converting files from one format to another. The code `everyother.f` takes an existing CFL3D-type grid and coarsens it by taking every other grid point. The `everyotherp3d.f` program performs the same task for PLOT3D-type grids. The file `historytec3d.f` reads in the residual history file and creates a corresponding Tecplot[4] file. It will also convert iterations to work units if desired. The `make1to1.f` program assists in the creation of the boundary condition and one-to-one blocking sections of the input file. The `p3dtotec3d.f` tool (along with `p3dtotec3d_pre.c` and `make_p3dtotec3d_cray`) creates a Tecplot file from PLOT3D grid and solution files. The `reverseijk.f` code reads a CFL3D-type grid file, swaps indices (as desired), and writes out a new CFL3D-type grid file. The `v4tov5_input.f` tool takes a Version 4.1 input file and converts it to Version 5.0 form. The `v4tov5_restart.f` tool converts a Version 4.1 restart file to Version 5.0 form. Finally, `v5inpdoubhalf.f` takes an existing input file and creates a new input file with either double or half the grid points of the original.

2.2 Obtaining a Grid File

The first step in solving a CFD problem is obtaining a grid. A pre-packaged tool such as GRIDGEN[36] could be used to create a grid, or perhaps a user-written program could be used for relatively simple configurations. How ever the grid is obtained, an essential step toward success with CFL3D will be having the grid file written in CFL3D or PLOT3D/TLNS3D format. See “Grid File” on page 65 for the appropriate formats for the file. Time can also be saved at this point by considering what options will be desirable later. For example, since the use of multigrid is highly recommended, choose grid dimensions wisely. See “A Word About Grid Dimensions” on page 127. If viscous cases will be run, make sure the grid spacing normal to the wall in the boundary layer is fine enough. At least 20 grid points are recommended in a *laminar* boundary layer. At least 3 points are recommended in the laminar sub-layer of a *turbulent* boundary layer (y^+ of first grid point off the wall should be $O(1)$). Basically, consider how the problem will be solved and choose dimensions, blocking strategies, grid-line stretching, etc., accordingly.

2.3 Creating the Input File

Chapter 3 lists and describes all the parameters in the input file. The easiest way to create an input file for a particular case is to copy the input file from the test case (see

Chapter 9) most resembling the case at hand. Then change those parameters that pertain to the current case. The other parameters should already be set to the recommended values. Besides the test case sample inputs, some input examples for 1-1 blocking and grid embedding are provided in Chapter 6 and several input examples for multigrid, mesh sequencing, and grid embedding are provided in Chapter 7. CFL3D provides additional help in setting up the input file with an extensive set of diagnostics which halt execution at detectable errors and provide the user with a message indicating the problem.

2.4 Utilizing Patched and/or Overlapped Grids

When using patched and/or overlapped grids, the files containing the appropriate interpolation coefficients must be obtained prior to running CFL3D. The preprocessors ronnie and MaGGiE[11] will generate the needed coefficients for patched grids and overlapped grids, respectively, and will output the interpolation information in the format needed by CFL3D. Information on using ronnie and MaGGiE can be found in the `ronnie.doc` and `maggie.doc` files, respectively. The preprocessor `precfl3d`, which sets all the array size parameters needed by CFL3D, will need the appropriate patch and/or overlap data files in order to properly size the arrays needed for these options. Therefore, be sure to run ronnie and/or MaGGiE *before* running `precfl3d`.

2.4.1 Running ronnie

When utilizing the grid patching option of CFL3D, the preprocessor ronnie must be run first. The basic steps for running this code are as follows.

Step 1

Prepare an input deck, typically called `ronnie.inp`, for the case. Also, modify the header file `ron1.h` as needed.

Step 2

Use the makefile to compile, link, and create the executable for the ronnie code (be sure `ron1.h` is in the current directory):

```
make -f makeronnie_machinename
```

Step 3

Run the ronnie code:

```
ronnie < ronnie.inp
```

If successful, ronnie will create a file containing patch interpolation data. The name of the patch file is specified by the user in the input file; it is typically called `patch.bin`.

2.4.2 Running MaGGiE

If grid overlapping is used in some portion of the grid configuration, then the preprocessor MaGGiE must be utilized. The basic steps for running this code are as follows.

Step 1

Prepare an input deck, typically called `maggie.inp`, for the case. Also, modify the header file `mag1.h` as needed.

Step 2

Use the makefile to compile, link, and create the executable for the MaGGiE code (be sure `mag1.h` is in the current directory):

```
make −f makemaggie_machinename
```

Step 3

Run the MaGGiE code:

```
maggie < maggie.inp
```

If successful, MaGGiE will create a file containing overlap interpolation data. The name of the overlap file is specified by the user in the input file; it is typically called `ovrlp.bin`.

2.5 Running CFL3D

The basic steps for running CFL3D are as follows. Remember, if using grid patching and/or grid overlapping, run ronnie and/or MaGGiE before proceeding with the following steps.

Step 1

Compile, link, and create the executable for the `precfl3d.f` code. Make sure the parameter file `precfl.h` is available.

```
make −f makeprecfl3d_machinename
```

The resulting executable will be called `precfl3d`.

Step 2

Run `precfl3d`.

```
precfl3d < inputfilename
```

The *`inputfilename`* here is the CFL3D input file name. Detailed diagnostic information is printed out in the file `precfl3d.out`. Most items in the input deck will be checked for

consistency. However, since precfl3d does not read in the grid file, it cannot detect grid problems and, most importantly, it cannot detect ordering errors in the 1-1 block interfaces. Check the output from CFL3D itself to verify that 1-1 block interfaces are set correctly. If precfl3d has not run successfully, then the most likely cause is an error in the input file; precfl3d echoes the input file as it goes along, so check the bottom of precfl3d.out to determine the approximate location of the input error. A successful execution of precfl3d will have one of the following lines at the end of the output file and print to the screen:

```
you *MUST* recompile cfl3d
```

or

```
you do not need to recompile cfl3d
```

When the code has run successfully, there will be no further need for this program until a new input file is developed.

Running precfl3d will either create or modify the parameter files, cfl1.h, cfl2.h, cfl3.h, cfl4.h, and cfl5.h, which are used when compiling CFL3D. Remember that these files *are* case dependent. Therefore, another case will generally require that precfl3d be run again and CFL3D recompiled.

Step 3

Compile, link, and create the executable for CFL3D.

```
make -f makecfl3d_machinename
```

The executable will be called cfl3d.

Step 4

Run cfl3d. This step can be performed interactively or a submittal file can be utilized to send the case to a particular queue of the machine being used.

```
cfl3d < inputfilename
```

Users have been known to submit cases for large numbers of iterations on the first run only to discover that an input parameter was set incorrectly. It is wise to begin running CFL3D with only a few iterations and then check the output. Look over the main output file. Make sure there are no warning messages and that, in general, it “looks right” for the particular case being run. Take the PLOT3D files to a Silicon Graphics (or compatible) workstation and check any block boundaries to see if the flow is passing from block to block as expected. *Then* submit the case for a more extensive computation.

CHAPTER 3

Input Parameters

Descriptions of each of the input line types are provided in this chapter where the section numbers denote the line type (LT) number referred to throughout the manual. An example input file is provided below for a flat plate case. Note that not all of the line types of information will be required for every application.

```
I/O FILES
grdflat5.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    turbulent flat plate (plate from j=17-65, prior to 17 is symmetry)
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
    0.2000    00.000       0.0    06.000     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
    -5.000         0       000   05.0000         0      10.0
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         0      1200         0         1         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0       001         0         0         7
    IDIM    JDIM    KDIM
      02      65      97
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         1         2         1        17         1        97
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         0         0         0
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         2         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1008         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1001         0         0         1        17         0
         1         2      2004         0         0        17        65         2
               TWTYPE        CQ
                  0.        0.
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
```

```
     MSEQ    MGFLAG    ICONSF       MTT      NGAM
        1         1         0         0        02
     ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
        0       0.3       0.3       0.3         0       0.3       0.3       0.3
     NCYC    MGLEVG     NEMGL     NITFO
     0800         03        00       000
     MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7      MIT8
       01        01        01        01        01         1         1         1
  1-1 BLOCKING DATA:
     NBLI
        0
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
 PATCH SURFACE DATA:
   NINTER
        0
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      0      0      0      0      0      0      0      0      0
IMOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

3.1 LT1 - Input/Output File Names

Input/Output file names, up to 60 characters each, starting in column 1. Specify the user file names in the following order (one name per line), using a "dummy" file name if a file is not actually needed for the current problem:

Input/Output File	File Type
grid (unit 1)	binary[1] input
PLOT3D grid (unit 3)	binary[2],[3] output
PLOT3D solution (unit 4)	binary[2],[3] output
primary case information (unit 11)	ascii output
residual and force coefficient history (unit 12)	ascii output
turbulence model residual history (*not* Baldwin-Lomax) (unit 13)	ascii output
Baldwin-Lomax (unit 14)	ascii output
secondary case information (unit 15)	ascii output

[1] Defaulted in the code to standard binary for the particular machine on which the code is executed. On Cray computers the default can be changed to IEEE Big unformatted by uncommenting "`call asnfile`" one line prior to the open statement for unit 1 in the main routine.

[2] Defaulted in the code to standard binary for the particular machine on which the code is executed. On Cray computers the default can be changed to IEEE Big unformatted by uncommenting the "`call asnfile`" lines one line prior to the open statements for both unit 3 and unit 4 in the main routine.

[3] Can be changed to formatted by changing the open statements for unit 3 and unit 4 to "`formatted`" and changing `ibin` from 1 to 0 in subroutine `qout`.

flow-field variables printout (unit 17) . . . . . ascii output

unsteady pressures from forced oscillations (**iunst** > 0) (unit 20) . . . . ascii output

interpolation coefficients for chimera grids (unit 21) . . . . . .binary input

interpolation coefficients for patched grids (unit 22) . . . . . binary input

restart (unit 2) . . . . . .binary input/output

3.2 LT2 - Case Title

title describing case

3.3 LT3 - Flow Conditions

xmach – free-stream[4] Mach number

alpha – angle of attack (See Appendix I.)

beta – side-slip angle (See Appendix I.)

reue – free-stream[4] Reynolds number per unit grid length (millions)

How to Set **reue**

Suppose it is desired to simulate a flow in which the Reynolds number based on some characteristic length (e.g. the chord) is Re. If L is the corresponding length in the grid (ignore the dimension, if any, of L), then set **reue** = Re/L. For example, if unit "1" in the grid corresponds to "1 inch" in the experiment, then give **reue** as the Reynolds number per inch. If unit "1" in the grid corresponds to "16.7845 furlongs", then give **reue** as the Reynolds number per 16.7845 furlongs. (Also remember that in the input file, **reue**/1×10^6 is the actual input value, so put a "20" for "20 million", etc.) (See "Reynolds Number Examples" on page 55.)

tinf – free-stream[4] temperature (degrees Rankine)

ialph – indicator for determining how angle of attack is measured in PLOT3D-type grids (**ngrid** < 0)

If	*Then*
ialph = 0	**alpha** is measured in the $x-z$ plane (with z "up"), i.e. **alpha** = 90° would give a free-stream velocity vector in the positive z direction.

[4] See the note on page 3 about the usage of the phrase *free stream*.

ialph > 0 — **alpha** is measured in the $x-y$ plane (with y "up"), i.e. **alpha** = 90° would give a free-stream velocity vector in the positive y direction.

Note

(1) For CFL3D-type grids, **alpha** is always measured in the $x-z$ plane, with z "up". (If it is desired to have y "up" with CFL3D-type grids, then set **ialph** > 0 and comment out the following line in `subroutine global`:

```
if (ngrid.gt.0) ialph=0
```

ihstry – determines which variables are to be tracked for a convergence history (i.e., which variables are output to file `cfl3d.res` (unit 12) and file `cfl3d.subit.res` (unit 23))

If	*Then*
ihstry = 0	standard convergence history: residual, C_l, C_d, C_y or C_z depending on whether z or y is "up", respectively, and C_{my} or C_{mz} (for z or y "up", respectively)
ihstry > 0	control surface history: residual and mass flow, pressure force, viscous force, thrust (momentum) force (forces are resultant forces, i.e. if f_x, f_y, and f_z are the force components in the x, y, and z directions, then the resultant force is $\sqrt{f_x^2+f_y^2+f_z^2}$; must have **ncs** > 0 and specify which control surfaces are to be tracked - only those surfaces with **inorm** 0 are included in the sum)

Note

(1) "LT3 - Flow Conditions" of CFL3D Versions 4.1 and earlier had additional parameters **isnd** and **c2spe** to govern wall temperature and heat transfer. These parameters are no longer used, but their functions are implemented in a more general fashion in boundary condition type 2004. (See notes under "LT14 - I0 Boundary Condition Specification" on page 32). The position formerly occupied by **isnd** is now occupied by **ialph**, so that the value of **ialph** for PLOT3D-type grids is no longer hard-wired in subroutine `rp3d` (as in earlier versions of the code).

3.4 LT4 - Reference Quantities

sref – reference area used to compute non-dimensional forces and moments; **sref** is generally taken to be the plan-form area (*not* the wetted area) of a wing, for example. It should be given in grid dimensions. (You must also account for the grid-distance between 2-D planes when **i2d**=1!)

cref – reference length used to compute non-dimensional moments

bref – reference span used to compute non-dimensional moments

xmc – moment center in x direction

ymc – moment center in y direction

zmc – moment center in z direction

3.5 LT5 - Time Step Parameters

dt – time step

If	*Then*
dt < 0	local time stepping, CFL number = **|dt|**; Use **ncyc** to control the number of cycles. Sub-iterations are not used.
dt > 0	constant time step equal to **dt**; Use **ntstep** to control the number of time steps and **ncyc** to control the number of sub-iterations. Two different sub-iteration strategies can be employed: t-TS and τ-TS. The t-TS method can have unacceptable time-step limitations depending on the case. The τ-TS method allows for much larger time steps, although larger time steps require more sub-iterations. Multigrid may be used to enhance convergence of sub-iterations and is recommended in general.

If	*Then*	
t-TS	**ntstep**	= number of time steps
	ita	= +1 or +2 (+2 recommended)
	mgflag	= 0 or 1 (1 recommended)
	mglevel	= number of multigrid levels
	ncyc	= number of sub-iterations + 1 (typically from 2 to 15) (**ncyc** = 1 will yield standard time-accurate method with *no* sub-iterations.)
τ-TS	**ntstep**	= number of time steps
	ita	= –1 or –2 (–2 recommended) (See "LT6 - Options and Specifications" on page 23)
	mgflag	= 0 or 1 (1 recommended)
	mglevel	= number of multigrid levels
	ncyc	= number of sub-iterations + 1 (typically from 5 to 25)

Notes

(1) The number of sub-iterations required depends partly on the size of the physical time step used. More sub-iterations are required for larger time steps. The use of multigrid will generally reduce the number of sub-iterations required, although the cost per sub-iteration will increase.

(2) It is *strongly* recommended that the sub-iteration convergence be monitored in the output file `cfl3d.subit_res` to determine if sufficient sub-iterations are being used for a particular case. The residual should drop and the force coefficients should tend toward constant values during each sub-iteration. However, from an efficiency standpoint, it is not generally desirable to fully converge each time step.

(3) The τ-TS scheme utilizes, in addition to the physical time step, a "pseudo" time step based on local time stepping. Thus, there is a CFL number associated with τ-TS (see **cfl_tau** on page 23).

(4) See "Time and Time Step" on page 56 for the nondimensionalization of the time step.

(5) To obtain second order accuracy in time ($|\mathbf{ita}| = 2$), sub-iterations (**ncyc** > 1) must be used.

irest – restart flag

If	*Then*
irest = 0	no restart
irest > 0	restart from previous solution, using restart file (unit 2)
irest < 0	restart from previous solution, but do not save previous history information

iflagts – time step ramping flag

If	*Then*
iflagts = 0	constant **dt**
iflagts > 0	**dt** ramped over **iflagts** steps to $\mathbf{dt} \times \mathbf{fmax}$

fmax – maximum increase in **dt**; $\mathbf{dt}_{\text{final}} = \mathbf{fmax} \times \mathbf{dt}_{\text{initial}}$

Notes

(1) The ramping of the time step/CFL number to $\mathbf{dt}_{\text{final}}$ is non-linear, occurring slowly at first and then increasing in rate.

(2) When the time step has successfully been ramped to $\mathbf{dt}_{\text{final}}$, remember to set the input value of **dt** to $\mathbf{dt}_{\text{final}}$ before restarting with the previous solution. Also, remember to change **iflagts** to zero and/or **fmax** to 1.0.

iunst – unsteady mesh flag

If	*Then*
iunst = 0	stationary grid
iunst = 1	dynamic grid (translation or rotation)

Notes
(1) If **dt** < 0, **iunst** is automatically set to 0.
(2) See "LT33 - Number of Translated Grids" on page 43 and beyond for specification of dynamic grid input.

cfl_tau – CFL number for τ-TS scheme; **cfl_tau** is always > 0 and it is not used for the t-TS scheme (**ita** > 0) or non-time-accurate runs (**dt** < 0). A value in the range from 5 to 10 is recommended; however, a smaller value may be required if the sub-iterations do not converge.

3.6 LT6 - Options and Specifications

ngrid – number of grids input = abs(**ngrid**)

If	*Then*
ngrid > 0	CFL3D grid format
ngrid < 0	PLOT3D (or TLNS3D) grid format

(See "Grid File" on page 65.)

nplot3d – number of flow-field data sets to be output in PLOT3D format
Notes
(1) If **nplot3d** < 0, then the PLOT3D files are automatically set to include all solid surfaces (no field points) for 3-d cases or all field points for 2-d cases.
(2) If **nplot3d** < 0, *do not* input any PLOT3D data lines in the PLOT3D input section below ("LT28 - PLOT3D Output Specifications"), that is, if **nplot3d** < 0, treat the PLOT3D input section as if **nplot3d** = 0.

nprint – number of data sets to be sent to an output file
Notes
(1) If **nprint** < 0, then the printout file is automatically set to include all solid surfaces (no field points) for 3-d cases or all field points for 2-d cases.
(2) If **nprint** < 0, *do not* input any print out data lines in the print out input section below ("LT30 - Print Out Specifications"), that is, if **nprint** < 0, treat the print out input section as if **nprint** = 0.

nwrest – number of iterations between updates of the binary restart file (Note that the binary restart file is also updated at the last iteration of the run regardless of the value of **nwrest**.) Must be > 0. Used to save an earlier restart file in case the full run does not finish. Generally, set **nwrest** > **ncyc** (or > **ntstep** for time-accurate) unless it is suspected that the run may not finish, or for a margin of safety. Restart file is overwritten each time it is updated.

ichk – checks for negative densities and/or pressures in subroutines `gfluxr`, `hfluxr`, `ffluxr`, and `conu` if set equal to 1.

Note

(1) Checking for negative densities and/or pressures requires additional CPU time, so utilize *only* to diagnose problem cases.

i2d – 2-d case flag

If	*Then*
i2d = 0	3-d case
i2d = 1	2-d case
i2d = –1	2-d case plus far-field point-vortex correction for **bctype** = 1003

Notes

(1) For **i2d** ≠ 0, *must* set **idim** = 2.

(2) For **i2d** ≠ 0, the $i = 1$ and $i = 2$ planes *must* each be planar.

(3) For **i2d** = -1, the grid *must* be the $x - z$ plane; the point vortex is applied at (**xmc**, **zmc**).

ntstep – number of cycles for time-accurate computations (**dt** > 0)

Notes

(1) For time-accurate computations, **ncyc** controls the number of sub-iterations.

(2) For steady-state computations (**dt** < 0), **ntstep** defaults to 1 and **ncyc** controls the number of cycles.

ita – order of time-accuracy/two-time scheme flag (used only for **dt** > 0)

If	*Then*
ita = +1	first order in time; physical time term only (t-TS method)
ita = +2	second order in time; physical time term only (t-TS method)
ita = –1	first order in time; physical *and* pseudo time terms (τ-TS method)
ita = –2	second order in time; physical *and* pseudo time terms (τ-TS method) (recommended)

Notes

(1) The approximate factorization scheme used to advance the solution in time introduces first order errors in time. Furthermore, if the diagonal version is utilized (**idiag** = 1), additional errors of order Δt are introduced. Sub-iterations can be used to drive these factorization/diagonalization errors to zero. Therefore, if a formally second-order (in time) solution is desired, sub-iterations *must* be used; the run will terminate if |**ita**| > 1 and **ncyc** = 1.

(2) The overhead for second order verses first order is relatively small when compared to the gain in accuracy and is therefore recommended.

(3) The inclusion of a pseudo time term increases (often dramatically) the maximum allowable time step one can take for a particular problem. However, sub-iterations (**ncyc** > 1) are therefore mandatory and multigrid is recommended. Also, note that larger time steps imply greater temporal error; so again, second order is recommended.

3.7 LT7 - Grid, Force, and Viscous Options

(Data for Line Type Seven should be repeated **ngrid** times.)

ncg – number of coarser grids to construct for multigrid and/or mesh sequencing (Set **ncg** = 0 for an embedded mesh.)

Note

(1) With the exception of embedded grids, all grids should have the same value of **ncg**.

iem – embedded mesh flag

If	*Then*
iem = 0	global grid
iem = **l**	level number (**l**) of this embedded grid above the global grid level

iadvance – flag to initiate residual/update calculations

If	*Then*
iadvance ≥ 0	evaluate the residual and update the solution in the current block
iadvance < 0	skip the residual/update calculations for the current block (rarely used)

iforce – flag to initiate the calculation of forces on block faces with solid surfaces; **iforce** is a 3-digit number of the form IJK:

Where	*Initiates*
I = 0	no force calculations on the i faces
I = 1	calculation of the force contribution from the i = 1 face
I = 2	calculation of the force contribution from the i = **idim** face
I = 3	calculation of the force contributions from both the i = 1 face *and* the i = **idim** face
J = 0	no force calculations on the j faces
J = 1	calculation of the force contribution from the j = 1 face
J = 2	calculation of the force contribution from the j = **jdim** face
J = 3	calculation of the force contributions from both the j = 1 face *and* the j = **jdim** face
K = 0	no force calculations on the k faces
K = 1	calculation of the force contribution from the k = 1 face

K = 2 calculation of the force contribution from the k = **kdim** face

K = 3 calculation of the force contributions from both the k = 1 face *and* the k = **kdim** face

Notes

(1) *Only* solid surfaces contribute to the force computations; i.e. only those boundaries with boundary condition types 1005 and 2004 can contribute to the force totals. Thus, wakes (with boundary condition type 0) are not computed in the force total, regardless of whether or not they are on a boundary that has been flagged with **iforce** > 0.

(2) The **iforce** parameter indicates the calculations of force contributions from entire faces; i.e. **iforce** ≠ 0 will cause *all* segments on that particular face to contribute to the force total (assuming those segments have solid-surface boundary conditions, as discussed previously). To eliminate from the force computation a segment that would otherwise contribute to the force total, set **segment** < 0 in "LT14 - I0 Boundary Condition Specification" on page 32 through "LT19 - KDIM Boundary Condition Specification" on page 35. This will eliminate that particular segment from the total. This feature is useful, for example, when the fuselage and sting are both on the k = 1 boundary of grid 1. The force contribution from the fuselage is desired, but not the contribution from the sting. In this case, let the k = 1 boundary be defined by two segments (the first for the fuselage and the second for the sting). Set **iforce** = 001 for grid 1 and then set **segment** = –2 for the second segment in "LT18 - K0 Boundary Condition Specification", e.g.:

```
K0: GRID SEGMENT BCTYPE   ISTA   IEND   JSTA   JEND  NDATA
       1       1   2004      1     97      0      0      2
       1      -2   2004     97    129      0      0      2
```

ivisc(m) – viscous/inviscid surface flag with $m = 1, 2, 3 \Rightarrow m = i, j, k$

If	Then
ivisc = 0	inviscid
ivisc = 1	laminar
ivisc = 2	turbulent – Baldwin-Lomax model
ivisc = 3	turbulent – Baldwin-Lomax with Degani-Schiff modification
ivisc = 4	turbulent – Baldwin-Barth model
ivisc = 5	turbulent – Spalart-Allmaras model
ivisc = 6	turbulent – Wilcox $k-\omega$ model
ivisc = 7	turbulent – Menter's $k-\omega$ SST model
ivisc = 8	turbulent – $k-\omega$ Explicit Algebraic Stress Model Gatski–Speziale (EASM Gatski–Speziale) in eddy-viscosity formulation
ivisc = 9	turbulent – $k-\varepsilon$ EASM Girimaji in eddy-viscosity formulation
ivisc = 10	turbulent – Abid $k-\varepsilon$ model

ivisc = 11 turbulent – $k-\varepsilon$ EASM Gatski–Speziale *Nonlinear*

ivisc = 12 turbulent – $k-\omega$ EASM Gatski–Speziale *Nonlinear*

ivisc = 13 turbulent – $k-\varepsilon$ EASM Girimaji *Nonlinear*

Notes

(1) If **ivisc**(m) < 0 on input, a wall function is employed. This option should only (officially) be used for attached flow, when the first grid point off the wall is not in the sub-layer ($y^+ > 10$ or more). The wall function with the Baldwin-Lomax model (**ivisc** = –2 or –3) can be particularly non-robust and, although it may sometimes work well, it is not recommended in general.

(2) The thin-layer viscous terms (laminar or turbulent) can be included in the j, k, or i directions, either separately or combined. Cross-derivative terms are not included. The exception is for turbulent flow using the Baldwin-Lomax model; terms can be included simultaneously in, at most, two directions, either $j-k$ or $i-k$, for any particular grid.

(3) Unlike previous versions of the code, the Baldwin-Lomax model can now be applied at any boundary or boundaries; i.e., the walls may now be at $j/k/i = 1$ and/or $j/k/i =$ **jdim/kdim/idim**.

(4) For multiple-zone applications, be aware that the Baldwin-Lomax model is *not* a field-equation model and it relies on distances to $j/k/i = 1$ and/or $j/k/i =$ **jdim/kdim/idim** walls *in a given zone*. If a particular zone does not contain walls but it is still desired that **ivisc** = 2 or 3 there, the model will default to using a length scale associated with a distance to the nearest $j/k/i = 1$ face *in that zone*. Another option is to set **ivisc** = 0 or 1 in the zones with no walls, but this may not be viable if the zone in question is very near to a wall(s) in other zones.

(5) For the reasons listed above, the field-equation modes 4 and up, which use the very general minimum distance function (which finds the minimum distance to the nearest wall, regardless of what zone it is in) are generally preferred, especially in multi-zone applications. However, the Baldwin-Lomax model (with the Degani-Schiff option) still seems to be the best model for vortical flows.

(6) It is preferable to let k be the primary viscous direction and i be the secondary viscous direction. See discussion in Section 5.1.1. If i is used as a primary viscous direction, it may be necessary to switch the order of inversions in subroutine `af3f` (e.g. from j,k,i to j,i,k). Keep in mind that, unlike i and k, altering the location of the j inversion in the approximate factorization sequence requires substantial recoding.

(7) The minimum distance function `smin` is computed from viscous walls *only*. If a wall is inviscid, then as far as `smin` is concerned, it is invisible. This is important to remember when viscous boundary conditions are turned on after running a case inviscidly for some time since `smin` may never have been computed!

Caution

The EASM models are currently very preliminary and should be considered as "researcher-oriented" as opposed to "production-oriented". Due to the sensitive nature of the variable coefficient `cmu`, the models are generally less robust than other models (particularly for 3-d cases and particularly for the nonlinear versions). It is recommended that EASM solutions be restarted from previously converged $k-\omega$ or $k-\varepsilon$ solutions for this reason. Even then, certain cases may *still* experience trouble! However, preliminary tests indicate that the Girimaji EASM $k-\varepsilon$ model (**ivisc** = 13) tends to be the most robust of the EASM models.

3.8 LT8 - Grid Dimensions

(Data for Line Type Eight should be repeated **ngrid** times.)

idim – number of grid points in the i direction (must be 2 for |**i2d**| =1)

jdim – number of grid points in the j direction

kdim – number of grid points in the k direction

3.9 LT9 - Laminar Region Specification

(Data for Line Type Nine should be repeated **ngrid** times.)

ilamlo – lower i grid point index defining the laminar region of the current block

ilamhi – upper i grid point index defining the laminar region of the current block

jlamlo – lower j grid point index defining the laminar region of the current block

jlamhi – upper j grid point index defining the laminar region of the current block

klamlo – lower k grid point index defining the laminar region of the current block

klamhi – upper k grid point index defining the laminar region of the current block

Notes

(1) These parameters are used for simulating transition only if **ivisc** > 1. Set **ilamlo, jlamlo, klamlo** = 0 for fully turbulent flow. Currently, only one transition region per grid is allowed.

(2) When **ilamlo**, etc. are used, they define a region inside which the production terms in the turbulence model equations (for **ivisc** > 3) are turned "off" (for the Baldwin-Lomax model, the eddy viscosity is merely zeroed out). The flow usually remains laminar inside that region as a result. In contrast, "fully turbulent" (**ilamlo**, etc. = 0) means that the production terms in the turbulence model equations are "on" everywhere. However, in practice, one might see behavior which indicates that the model is still "transitioning" on its own. In other words, one might see a low "laminar" C_f near the leading edge of an airfoil, transitioning fairly rapidly to a higher "turbulent" C_f, for example. Note that turning off the production term does *not* keep turbulence from convecting into a region, if turbulence exists upstream.

(3) Currently for multigrid, the turbulent viscosity is only computed on the finest level grid, then it is restricted to coarser levels. Therefore, the **ilamlo**, etc. values need not be "good" multigrid numbers when used.

3.10 LT10 - Embedded Mesh Specifications

(Data for Line Type Ten should be repeated **ngrid** times.)

inewg – restart flag for grid (not needed if **irest** = 0)

If	*Then*
inewg = 0	flow-field data is read from the restart file
inewg = 1	flow-field data is initialized by linear interpolation from coarser grid solutions

Notes

(1) The purpose of **inewg** is to allow embedded grids to be added *after* the solution process has begun; that is, if a solution has already been computed, but additional resolution in certain parts of the flow field is desired. In such cases, embedded grids may be appended to the grid file and the case may be restarted. Setting **inewg** = 1 initializes the solution on the new embedded level by interpolating from a solution on a coarser level, thereby providing a reasonable starting solution on the new embedded level (the solution for the new embedded grid is not contained in the restart file at this point).

(2) *Important:* **inewg** should be set to 1 *only for the first restart* after a new embedded level has been added. On subsequent restarts, **inewg** = 0 should be used, since the embedded solution has become part of the restart file and should not be re-initialized.

igridc – connection flag for embedded meshes

If	*Then*
igridc = 0	the current grid is a global mesh (**iem** = 0)

igridc > 0 **igridc** is the grid number to which the embedded mesh (**iem** > 0) connects

is, js, ks – starting indices in the connecting grid for placement of an embedded mesh (set **is** = 0, **js** = 0, **ks** = 0 for global meshes)

ie, je, ke – ending indices in the connecting grid for placement of an embedded mesh (set **ie** = 0, **je** = 0, **ke** = 0 for global meshes)

Note

(1) The embedded meshes must be a regular refinement in all directions of the grid to which it connects. The exception is in the i direction for which the embedded mesh *may* have the same grid spacing as the grid to which it connects if desired (this ensures that **idim** = 2 for all grid levels in 2-d cases, including embedded grid levels).

3.11 LT11 - Matrix Inversion and Flux Limiter

(Data for Line Type Eleven should be repeated **ngrid** times.)

idiag(m) – matrix inversion flag with $m = 1, 2, 3 \Rightarrow m = i, j, k$

If	*Then*
idiag = 0	5×5 block tridiagonal inversions
idiag = 1	scalar tridiagonal inversions

Notes

(1) The scalar tridiagonal inversions are recommended for steady-state computations. They may or may not be appropriate for time-accurate computations. For time-accurate computations with scalar tridiagonal inversions, it is suggested that sub-iterations be used to reduce the diagonalization errors.

(2) When **idiag** = 0, viscous terms are included on the left-hand side of the implicit time advancement scheme in the k direction *only.*

iflim(m) – flux limiter flag with $m = 1, 2, 3 \Rightarrow m = i, j, k$

If	*Then*
iflim = 0	unlimited
iflim = 1	smooth limiter
iflim = 2	min-mod limiter (recommended if limiter is needed and **rkap0** ≠ 1/3)
iflim = 3	smooth limiter tuned to $\kappa = 1/3$ with a cut-off to eliminate limiting in regions of small gradients (recommended if limiter is needed and **rkap0** = 1/3)

Notes

(1) Many purely subsonic flows do not require a limiter.

(2) Use of **iflim** = 3 will override the **rkap0** value input and force upwind-biased third order.

3.12 LT12 - Spatial Differencing

(Data for Line Type Twelve should be repeated **ngrid** times.)

ifds(m) – spatial differencing parameter for Euler fluxes with

$m = 1, 2, 3 \Rightarrow m = i, j, k$

If	*Then*
ifds = 0	flux-vector splitting
ifds = 1	flux-difference splitting (Roe's scheme) (recommended)

rkap0(m) – spatial differencing parameter for Euler fluxes with

$m = 1, 2, 3 \Rightarrow m = i, j, k$

If	*Then*
rkap0 = –1	fully upwind
rkap0 = 0	Frommes' scheme
rkap0 = 1	central
rkap0 = 1/3	upwind-biased third order (recommended)

3.13 LT13 - Boundary Condition Segments

(Data for Line Type Thirteen should be repeated **ngrid** times.)

boundary condition segment flags:

grid – current grid number (must be in order)

nbci0 – number of segments on $i = 0$ boundary

nbcidim – number of segments on $i =$ **idim** boundary

nbcj0 – number of segments on $j = 0$ boundary

nbcjdim – number of segments on $j =$ **jdim** boundary

nbck0 – number of segments on $k = 0$ boundary

nbckdim – number of segments on $k =$ **kdim** boundary

iovrlp – grid-overlapping flag

If	*Then*
iovrlp = 0	no overlapping occurs for the current grid
iovrlp = 1	the current grid receives data from an overlapped grid(s)

Note

(1) The minimum number of segments on any boundary is 1.

3.14 LT14 - I0 Boundary Condition Specification

(Data for Line Type Fourteen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbci0}(n)$ times.)

I0 Boundary:

grid – current grid number (must be in order)

segment – current segment (must be in order and the total must equal **nbci0**)

bctype – boundary condition type for the current segment

jsta – j starting grid point location for the current segment

jend – j ending grid point location for the current segment

ksta – k starting grid point location for the current segment

kend – k ending grid point location for the current segment

ndata – number of additional data values required for this boundary condition

Notes

(1) Setting **jsta** = **jend** = 0 and/or **ksta** = **kend** = 0 is a shorthand way of specifying the entire range of j and/or k for this segment.

The following notes pertain to "LT14 - I0 Boundary Condition Specification" on page 32 through "LT19 - KDIM Boundary Condition Specification" on page 35:

(2) If **ndata** > 0, then, following the current line, a header line must appear, followed by a single line with the **ndata** values for this boundary condition.

(3) When multigrid is being used, it is best if all *segment* lengths are also multigridable. This is not a requirement. However, if one-to-one connections are involved, it is possible that the use of non-multigridable segment lengths will result in geometric mismatches on coarser grid levels, causing a "mismatch" message to be written to unit 11 and, most likely, eventual disruption of the execution.

(4) If a particular segment should not be counted in the force total, set **segment** to be negative. (See *Note* (2) under **iforce** in Section 3.7.)

(5) Input values for **bctype** (i.e. boundary condition types currently supported) as follows:

bctype	boundary condition
1000	free stream
1001	general symmetry plane
1002	extrapolation
1003	inflow/outflow
1004	(no longer available, use 2004 instead)

1005	inviscid surface
1008	constant enthalpy and entropy inflow
1011	singular axis – half-plane symmetry
1012	singular – full plane
1013	singular axis – partial plane
2002	specified pressure ratio
2003	inflow with specified total conditions
2004	no-slip wall
2005	periodic in space
2006	set pressure to satisfy the radial equilibrium equation
2007	set all primitive variables
2102	pressure ratio specified as a sinusoidal function of time

Descriptions of the boundary conditions can be found in Chapter 6.

3.15 LT15 - IDIM Boundary Condition Specification

(Data for Line Type Fifteen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbcidim}(n)$ times.)

IDIM Boundary:

All entries in this line type are the same as in "LT14 - I0 Boundary Condition Specification" except:

segment – current segment (must be in order and the total must equal **nbcidim**)

3.16 LT16 - J0 Boundary Condition Specification

(Data for Line Type Sixteen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbcj0}(n)$ times.)

J0 Boundary:

grid – current grid number (must be in order)

segment – current segment (must be in order and the total must equal **nbcj0**)

bctype – boundary condition type for the current segment

ista – i starting grid point location for the current segment

iend – i ending grid point location for the current segment

ksta – k starting grid point location for the current segment

kend – k ending grid point location for the current segment

ndata – number of additional data values required for this boundary condition

Notes
(1) Setting **ista** = **iend** = 0 and/or **ksta** = **kend** = 0 is a shorthand way of specifying the entire range of i and/or k for this segment.
(2) See general notes in "LT14 - I0 Boundary Condition Specification".

3.17 LT17 - JDIM Boundary Condition Specification

(Data for Line Type Seventeen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbcjdim}(n)$ times.)

JDIM Boundary:

All entries in this line type are the same as in "LT16 - J0 Boundary Condition Specification" except:

segment – current segment (must be in order and the total must equal **nbcjdim**)

3.18 LT18 - K0 Boundary Condition Specification

(Data for Line Type Eighteen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbck0}(n)$ times.)

K0 Boundary:

grid – current grid number (must be in order)

segment – current segment (must be in order and the total must equal **nbck0**)

bctype – boundary condition type for the current segment

ista – i starting grid point location for the current segment

iend – i ending grid point location for the current segment

jsta – j starting grid point location for the current segment

jend – j ending grid point location for the current segment

ndata – number of additional data values required for this boundary condition

Notes
(1) Setting **ista** = **iend** = 0 and/or **jsta** = **jend** = 0 is a shorthand way of specifying the entire range of i and/or j for this segment.
(2) See general notes in "LT14 - I0 Boundary Condition Specification".

3.19 LT19 - KDIM Boundary Condition Specification

(Data for Line Type Nineteen should be repeated $\sum_{n=1}^{\mathbf{ngrid}} \mathbf{nbckdim}(n)$ times.)

KDIM Boundary:

All entries in this line type are the same as in "LT18 - K0 Boundary Condition Specification" except:

segment – current segment (must be in order and the total must equal **nbckdim**)

3.20 LT20 - Mesh Sequencing and Multigrid

mseq – mesh sequencing flag for global grids (maximum number of grid levels for mesh sequencing is 5)

If	*Then*
mseq = 1	single solution on the finest grid level
mseq = 2	solution on the second finest grid advanced **ncyc**(1) cycles, followed by **ncyc**(2) cycles on the finest grid. The initial solution on the finest grid is obtained by interpolation from the coarser grid solution. If **ncyc**(2) = 0, the computations are terminated on the second finest grid after **ncyc**(1) cycles and the restart file is written for the second finest grid.
mseq > 2	sequencing from coarsest to finest mesh as above

mgflag – multigrid flag

If	*Then*
mgflag = 0	no multigrid
mgflag = 1	multigrid on coarser global meshes
mgflag = 2	multigrid on coarser global meshes and on embedded meshes

iconsf – conservation flag

If	*Then*
iconsf = 0	nonconservative flux treatment for embedded grids
iconsf = 1	conservative flux treatment for embedded grids

mtt – flag for additional iterations on the "up" portion of the multigrid cycle

If	*Then*
mtt = 0	no additional iterations (recommended)
mtt > 0	**mtt** additional iterations

ngam – multigrid cycle flag

If	*Then*
ngam = 1	V-cycle
ngam = 2	W-cycle (not recommended for overlapped grids)

Note
(1) If **mglevg** = 2, **ngam** defaults to 1.

3.21 LT21 - Smoothing

issc – correction smoothing flag

If	*Then*
issc = 0	no correction smoothing (usually recommended, but see note 18 in Chapter 10)
issc = 1	correction smoothing

epsssc(*m*) – correction smoothing coefficient with $m = 1, 2, 3 \Rightarrow m = i, j, k$; typical values: 0.3, 0.3, 0.3

issr – residual smoothing flag

If	*Then*
issr = 0	no residual smoothing (usually recommended, but see note 18 in Chapter 10)
issr = 1	residual smoothing

epsssr(*m*) – residual smoothing coefficient with $m = 1, 2, 3 \Rightarrow m = i, j, k$; typical values: 0.3, 0.3, 0.3

3.22 LT22 - Iterations and Multigrid Levels

(Data for Line Type Twenty-Two should be repeated for each sequence 1 through **mseq** (from coarsest to finest).)

ncyc – number of cycles for steady-state computations (**dt** < 0); number of sub-iterations + 1 for time-accurate computations (**dt** > 0)

Notes
(1) Set **ncyc** ≥ 1, unless using mesh sequencing and terminating execution on a coarser grid level (see description of **mseq** = 2 for "LT20 - Mesh Sequencing and Multigrid" on page 35, as well as "Mesh Sequencing" on page 134).
(2) For time-accurate computations (**dt** > 0), when **ncyc** = 1, the code is iterating on each time step *once*, which means *no* sub-iterations are used. Using **ncyc** = 1 corresponds to the standard non-sub-iterative time-accurate scheme that was in all previous versions of CFL3D. Similarly, **ncyc** = 2 means that the code is iterating on each time step *twice*, which means sub-iterations are used, since it is *one extra* sub-iteration over the standard non-sub-iterative time-accurate scheme.
(3) For time-accurate computations (**dt** > 0) with multigrid (**mgflag** > 0), at least one sub-iteration must be used (**ncyc** ≥ 2).

mglevg – number of grids to use in multigrid cycling for the global meshes

If	*Then*
mglevg = 1	single grid level
mglevg = 2	two grid levels
mglevg = m	m grid levels

nemgl – number of embedded grid levels above the finest global grid

If	*Then*
nemgl = 0	no embedded grids
nemgl = 1	one embedded grid
nemgl = m	m embedded grids

Note

(1) Set **nemgl** = 0 for global grids coarser than the finest global grid.

nitfo1 – number of first order iterations (**nitfo1** = 0 recommended)

3.23 LT23 - Coarse Grid Iterations

(Data for Line Type Twenty-Three should be repeated for each sequence 1 through **mseq** (from coarsest to finest).)

mitL – iterations on level L for each level L from coarsest to finest (**mitL** = 1 recommended, in general, although for some cases 3 2 1 or 2 2 1 can yield better multigrid convergence than 1 1 1 when three levels are used, for example)

3.24 LT 24 - Number of 1-1 Interfaces

nbli – number of 1-1 grid-point-connecting block interfaces

3.25 LT25 - 1-1 Blocking Connections

(Data for Line Type Twenty-Five should be repeated **nbli** times.)

number – identifying number to aid the user with "bookkeeping" the 1-1 interfaces in the input file (not used by CFL3D)

grid – grid/block identifier indicating which grid in the common interface is being described in this line of input

ista – starting i limit for 1-1 block interface

jsta – starting j limit for 1-1 block interface

ksta – starting k limit for 1-1 block interface

iend – ending i limit for 1-1 block interface

jend – ending j limit for 1-1 block interface

kend – ending k limit for 1-1 block interface

isva1 – varying index on 1-1 block interface (**isva1** in Line Type Twenty-Five varies with **isva1** in Line Type Twenty-Six)

isva2 – varying index on 1-1 block interface (**isva2** in Line Type Twenty-Five varies with **isva2** in Line Type Twenty-Six)

Notes

(1) The code will check the input connection data by computing the geometric mismatch between both sides of the interface, based on the specified ranges. A true 1-1 interface will have zero (or machine zero) mismatch. Any mismatches larger than ε (where ε is the larger of 10^{-9} and 10 times machine zero) will cause a warning message to be printed out in the main output file (look for the words "`geometric mismatch`". *Always* check the main output file (unit 11) for these words, since the code will only print out the message; it will not stop. A good practice is to make one run with only one iteration, then check the output for "`geometric mismatch`". Small mismatches may be acceptable, but large mismatches [O(1)] most likely indicate that one of the segments on the block boundary has been specified incorrectly.

(2) Any block segments that have 1-1 connectivity must also be input in the boundary condition section ("LT13 - Boundary Condition Segments" on page 31 through "LT19 - KDIM Boundary Condition Specification" on page 35), setting **bctype** = 0.

3.26 LT26 - 1-1 Blocking Connections

(Data for Line Type Twenty-Six should be repeated **nbli** times.)

Line Type Twenty-Six has the same parameter types as Line Type Twenty-Five. It gives the grid/block 1-1 connection information on the opposite side of the interface.

3.27 LT27 - Number of Patched-Grid Interfaces

ninter – patched-grid flag (zonal interface along a common surface where points need not match)

If	*Then*
ninter = 0	no patched-grid data is needed (no patched interfaces)
ninter < 0	patched-grid data supplied in a separate file

Notes

(1) In Version 5.0 of CFL3D, patched-grid interpolation data for static patched interfaces (interfaces that do not change with time) must be obtained as a preprocessing step (as for chimera overlapped grids). The code ronnie is designed for this task. Dynamic patched interfaces (such as occur when blocks slide past one another) are computed internally to CFL3D. The input for dynamic patched interfaces is described in "LT41 - Number of Dynamic Patched-Grid Interfaces" on page 49 through "LT45 - Displacement and Rotation Amounts" on page 52.

(2) Any block segments that are patched must also be input in the boundary condition section ("LT13 - Boundary Condition Segments" on page 31 through "LT19 - KDIM Boundary Condition Specification" on page 35), setting **bctype** = 0.

3.28 LT28 - PLOT3D Output Specifications

(Data for Line Type Twenty-Eight should be repeated **nplot3d** times.)

grid – designated grid number for output

iptype – type of PLOT3D file output

If	*Then*
iptype = 0	grid point type – grid file and **Q** file output
iptype = 1	cell center type – grid file and **Q** file output
iptype = 2	cell center type – grid file and turbulence file output (**ivisc** > 1 only) (The current defaults for output are the production term $\tilde{\mathrm{P}} \cdot \tilde{L}_R/(\tilde{u}_\infty)^3$ and the Reynolds stress components $\overline{u'w'}/(\tilde{u}_\infty)^2$, $\overline{u'u'}/(\tilde{u}_\infty)^2$, and $\overline{w'w'}/(\tilde{u}_\infty)^2$ for 2-d; and the same plus $S\mathrm{k}/\varepsilon$ for 3-d. (See Ristorcelli[30] for a description of these and other available variables.) These may be modified to output other variables if the user desires. If no turbulence model is being used or if the turbulence model cannot obtain a specific parameter being asked for, a value of zero is output. The "hard wire" occurs in subroutine `plot3t`. Note that the u', v', and w' quantities are aligned with the x, y, and z axes of the grid, respectively.
iptype > 2	cell center type – grid file and PLOT3D function file output

Specific functions currently available:

iptype = 3	minimum distance to nearest viscous wall or directed distance (**ivisc** > 1 only)
iptype = 4	eddy viscosity (**ivisc** > 1 only)

istart – starting location in i direction

iend – ending location in i direction

iinc – increment factor in i direction

jstart – starting location in j direction

jend – ending location in j direction

jinc – increment factor in j direction

kstart – starting location in k direction

kend – ending location in k direction

kinc – increment factor in k direction

Notes

(1) Setting **istart** = **iend** = **iinc** = 0, **jstart** = **jend** = **jinc** = 0, and/or **kstart** = **kend** = **kinc** = 0 is a shorthand way of specifying the entire range of i, j and/or k, respectively.

(2) Grid files are written with a blank array.

(3) All files are in multi-block format (even if there is a single zone).

(4) Files are written as 2-d if **i2d** ≠ 0.

(5) Function files contain only one variable.

(6) Files are written out in single precision.

(7) On Cray computers, "uncommenting" the lines

```
      c     call asnfile(plt3dg, '-F f77 -N ieee', IER)
```

and

```
      c     call asnfile(plt3dq, '-F f77 -N ieee', IER)
```

in module `cbsem.f` will cause the files to be written in IEEE binary. The files may then be transferred directly to a Silicon Graphics workstation and read into FAST or PLOT3D using the unformatted option. If those lines are left as comments, then files are written in Cray native, and must first be "itransed" before being transferred to a Silicon Graphics workstation and read into FAST or PLOT3D using the default (binary) option.

(8) This output will appear in the PLOT3D files (units 3 and 4) named near the top of the input deck.

3.29 LT29 - Movie Option

movie – flag to append periodically to the PLOT3D output files (grid and solution) as the solution progresses. Only one grid file and one solution file are generated.

If *Then*

movie = 0 no output of intermediate solutions (if |**nplot3d**| > 0, then a single solution is written at the end of the run)

movie > 0 output of additional solutions every **movie** iterations (the contents of the augmented PLOT3D files are governed by the parameters under "LT28 - PLOT3D Output Specifications" on page 39)

movie < 0 output of the initial flow field at the beginning of the run and output of additional solutions every |**movie**| iterations

Caution

Use with care: PLOT3D files will get very large very quickly! If the complete flow field is output, this is really a viable option only in 2-d. In 3-d, it is generally a viable option only if subsets of the flow field are output (i.e. the surface data). Also, set **nprint** = 0 when **movie** > 0 or large print files will result as well.

3.30 LT30 - Print Out Specifications

(Data for Line Type Thirty should be repeated **nprint** times.)

grid – designated grid number for output

iptype – type of flow-field variables to print out

If	*Then*
iptype = 0	grid point type
iptype = 1	cell center type

istart – starting location in i direction

iend – ending location in i direction

iinc – increment factor in i direction

jstart – starting location in j direction

jend – ending location in j direction

jinc – increment factor in j direction

kstart – starting location in k direction

kend – ending location in k direction

kinc – increment factor in k direction

Notes

(1) Setting **istart** = **iend** = **iinc** = 0, **jstart** = **jend** = **jinc** = 0, and/or **kstart** = **kend** = **kinc** = 0 is a shorthand way of specifying the entire range of i, j and/or k.

(2) This output will appear in the printout file (unit 17) named near the top of the input deck.

Section 5.2.2 gives a detailed sample of the contents of the printout file.

3.31 LT31 - Number of Control Surfaces

ncs – the number of control surfaces; control surfaces are user-specified grid surfaces through which mass flow, thrust (momentum) force, pressure force, and viscous force may be monitored. Control surfaces *must* be specified if **ihstry** > 0. (See "LT3 - Flow Conditions" on page 19.)

3.32 LT32 - Control Surface Specifications

block – block number of the control surface

istart – starting location in i direction

iend – ending location in i direction

jstart – starting location in j direction

jend – ending location in j direction

kstart – starting location in k direction

kend – ending location in k direction

iwall – control surface type

If	*Then*
iwall = 0	for a flow surface
iwall = 1	for a solid wall

inorm – defines the direction of the control surface outward normal relative to the grid surface normal. This is only required in the calculation of the total forces for all control surfaces to obtain the correct signs.

If	*Then*
inorm = 1	the control volume outward normal is in the same direction as the grid surface normal
inorm = –1	the control volume outward normal is in the opposite direction to the grid surface normal
inorm = 0	do not add this surface into the summation of the total forces

Notes

(1) The starting and ending indices in one of the directions must be identical to define the surface. If they are not, no calculation is performed. In the remaining two directions, setting both the starting and ending indices to zero is a shorthand way of specifying the entire index range.
(2) If all the surfaces have **inorm** = 0, the totals are not calculated.
(3) This output will appear in the printout file named near the top of the input deck; the control surface data appears after the data generated by setting **nprint** > 0.

THE INPUT DECK TERMINATES HERE FOR STEADY-STATE CASES OR TIME-DEPENDENT CASES IN WHICH THE GRID IS STATIONARY (**iunst** = 0)

Line Types Thirty-Three through Forty-Five are required only if **iunst** > 0 (dynamic grid).

3.33 LT33 - Number of Translated Grids

ntrans – number of grids undergoing translation motion

Notes

(1) If **ntrans** = 0, only the headers for "LT34 - Translational Reference Length" on page 43 through "LT36 - Maximum Translational Displacements" on page 45 are required (i.e. no numerical values required).
(2) If the moment center also needs to be translated, set **ntrans** = "number of grids undergoing translation + 1". If the moment center is not translated, moments are computed relative to a fixed point in space.

3.34 LT34 - Translational Reference Length

(Data for Line Type Thirty-Four should be repeated **ntrans** times.)

lref – the "grid equivalent" of the dimensional reference length used to define either the reduced frequency needed for **itrans** = 2 or the decay rate needed for **itrans** = 3 (See "LT35 - Translational Information and Velocities" on page 44 below.) For **itrans** ≤ 1, any value may be input (it is reset to 1.0 internally).

Note

(1) For example, if the input value of reduced frequency for a sinusoidally plunging wing was defined based on the dimensional chord of the wing and, in the grid, the chord of the wing is 2.0, then set **lref** = 2.0.

3.35 LT35 - Translational Information and Velocities

(Data for Line Type Thirty-Five should be repeated **ntrans** times.)

grid – designated number of grid to be translated

Note

(1) If **grid** = 0, then the following input governs the motion of the moment center; on output, the value of 0 will be replaced by the letters "MC" as a reminder.

itrans – type of translation

If	*Then*
itrans = 0	no translation
itrans = 1	translation with constant speed
itrans = 2	sinusoidal variation of displacement
itrans = 3	smooth increase in displacement, asymptotically reaching a maximum displacement

rfreq – reduced frequency when **itrans** = 2; growth rate to maximum displacement when **itrans** = 3 (set **rfreq** = 0 for **itrans** = 0 or **itrans** = 1)

utrans – translation velocity in x direction when **itrans** = 1; maximum displacement in x direction when **itrans** > 1 (set **utrans** = 0 if there is no x displacement)

vtrans – translation velocity in y direction when **itrans** = 1; maximum displacement in y direction when **itrans** > 1 (set **vtrans** = 0 if there is no y displacement)

wtrans – translation velocity in z direction when **itrans** = 1; maximum displacement in z direction when **itrans** > 1 (set **wtrans** = 0 if there is no z displacement)

Notes

(1) Depending on the type of motion, the nondimensional input parameters listed above have different definitions and it is important to understand the distinction. These multiple definitions allow a range of grid motions with a minimal number of input parameters.

(2) The sign of **utrans/vtrans/wtrans** governs the direction of the motion: "+" for motion in the increasing coordinate direction, "–" for motion in the decreasing coordinate direction. For **itrans** = 2 (sinusoidal), the sign dictates the "initial" direction of motion.

Caution

If PLOT3D-type grids, where y is "up", are used, it is important to note that the grid is internally treated as if z is "up" and that **utrans**, **vtrans**, and **wtrans** refer to the internal velocities. Thus, if y is "up" in the input grid and translation in the "up" direction is desired, then **wtrans** must be used to set the speed. It is advised that a visualization package such as FAST or PLOT3D be used in the early stages of the computation to verify that the grid is moving in the desired direction.

3.36 LT36 - Maximum Translational Displacements

(Data for Line Type Thirty-Six should be repeated **ntrans** times.)

grid – designated number of grid to be translated

Note

(1) If **grid** = 0, then the following input governs the motion of the moment center; on output, the value of 0 will be replaced by the letters "MC" as a reminder.

dxmax – maximum (absolute) translational displacement in the x direction to be allowed for this grid, measured from the $t = 0$ position; set **dxmax** = 0 if no restriction is required

dymax – maximum (absolute) translational displacement in the y direction to be allowed for this grid, measured from the $t = 0$ position; set **dymax** = 0 if no restriction is required

dzmax – maximum (absolute) translational displacement in the z direction to be allowed for this grid, measured from the $t = 0$ position; set **dzmax** = 0 if no restriction is required

Note

(1) Setting **dxmax**, **dymax**, and/or **dzmax** > 0 will have the following effect: if the grid moves further than |**dxmax**|, |**dymax**|, and/or |**dzmax**| from its position at $t = 0$, the grid will be reset backward the appropriate amount, from where the specified grid motion will start again. The grid will be reset every time it reaches the specified limit. Though each block may have an independent limit, care should be taken to insure that any blocks that must be reset concurrently have the *same* values of **dxmax**, **dymax**, and/or **dzmax**. This resetting feature is used primarily in 2-d turbomachinery applications. For example, in 2-d rotor-stator interaction problems where only a few of an infinite number of rotor and stator blades are actually modeled in the grid system, resetting the grids periodically allows the simulation to be continued for an arbitrarily long period of time with only a limited number of zones. This resetting option is only applicable for constant translational speed (**itrans** = 1).

3.37 LT37 - Number of Rotational Grids

nrotat – number of grids undergoing rotational motion

Notes

(1) If **nrotat** = 0, only the headers for "LT38 - Rotational Reference Length" on page 46 through "LT40 - Maximum Rotational Displacements" on page 48 are required (i.e. no numerical values required).

(2) If the moment center also needs to be rotated, set **nrotat** = "number of grids undergoing rotation" + 1.

3.38 LT38 - Rotational Reference Length

(Data for Line Type Thirty-Eight should be repeated **nrotat** times.)

lref – the "grid equivalent" of the dimensional reference length used to define either the nondimensional rotation rate needed for **irotat** = 1, the reduced frequency needed for **irotat** = 2, or the growth rate needed for **irotat** = 3 (See "LT39 - Rotational Information and Velocities" below.) For **itrans** ≤ 1, any value may be input (it is reset to 1.0 internally).

Note

(1) For example, if the input value of reduced frequency for a sinusoidally pitching wing was defined based on the dimensional chord of the wing and, in the grid, the chord of the wing = 2.0, then set **lref** = 2.0.

3.39 LT39 - Rotational Information and Velocities

(Data for Line Type Thirty-Nine should be repeated **nrotat** times.)

grid – designated number of grid to be rotated

Note

(1) If **grid** = 0, then the following input governs the motion of the moment center; on output, the value of 0 will be replaced by the letters "MC" as a reminder.

irotat – type of rotation

If	*Then*
irotat = 0	no rotation
irotat = 1	rotation with constant angular speed
irotat = 2	sinusoidal variation of angular displacement
irotat = 3	smooth increase in displacement, asymptotically reaching a maximum angular displacement

rfreq – reduced frequency when **irotat** = 2; growth rate to maximum angular displacement when **irotat** = 3 (set **rfreq** = 0 for **irotat** = 0 or **irotat** = 1)

omegax – x component of rotational velocity when **irotat** = 1; maximum angular displacement about the x axis when **irotat** > 1; set **omegax** = 0 if there is no x rotation

omegay – y component of rotational velocity when **irotat** = 1; maximum angular displacement about the y axis when **irotat** > 1; set **omegay** = 0 if there is no y rotation

omegaz – z component of rotational velocity when **irotat** = 1; maximum angular displacement about the z axis when **irotat** > 1; set **omegaz** = 0 if there is no z rotation

xorig – x coordinate of origin of the rotation axis

yorig – y coordinate of origin of the rotation axis

zorig – z coordinate of origin of the rotation axis

Notes

(1) Depending on the type of motion, the nondimensional input parameters listed above have different definitions and it is important to understand the distinction. These multiple definitions allow a range of grid motions with a minimal number of input parameters. (See Section 4.5.2 in the Nondimensionalization chapter for details.)

(2) The sign of **omegax**, **omegay**, and **omegaz** governs the direction of the rotation: "+" for motion in the positive direction, "–" for motion in the negative direction where "positive" and "negative" are dictated by the right-hand rule in which the thumb points in the positive direction of the axis of rotation and the fingers curl in the positive rotational direction. (See "The Right-Hand Rule" on page 67.)

3.40 LT40 - Maximum Rotational Displacements

(Data for Line Type Forty should be repeated **nrotat** times.)

grid – designated number of grid to be rotated

Note

(1) If **grid** = 0, then the following input governs the motion of the moment center.

dthxmx – maximum (absolute) rotational displacement about the x axis to be allowed for this grid (set **dthxmx** = 0 if no restriction is required)

dthymx – maximum (absolute) rotational displacement about the y axis to be allowed for this grid (set **dthymx** = 0 if no restriction is required)

dthzmx – maximum (absolute) rotational displacement about the z axis to be allowed for this grid (set **dthzmx** = 0 if no restriction is required)

Notes

(1) Setting **dthxmx**, **dthymx**, and/or **dthzmx** > 0 will have the following effect: if the grid rotates more than |**dthxmx**|, |**dthymx**|, and/or |**dthzmx**| from it's position at $t = 0$, the grid will be reset backward the appropriate amount, from where the specified grid motion will start again. The grid will be reset every time it reaches the specified limit. Though each block may have an independent limit, care should be taken to insure that any blocks that must be reset concurrently have the *same* values of **dthxmx**, **dthymx**, and/or **dthzmx**. This resetting feature is used primarily in 3-d turbomachinery applications. For example, in 3-d rotor-stator interaction problems where only a few of the large number of rotor and stator blades are actually modeled in the grid system, resetting the grids periodically allows the simulation to be continued for an arbitrarily long period of time with only a limited number of zones. This resetting option is only applicable for constant rotational speed (**irotat** = 1).

(2) **dthxmx**, **dthymx**, and **dthzmx** are output in degrees.

3.41 LT41 - Number of Dynamic Patched-Grid Interfaces

ninter2 – dynamic patched-grid flag (zonal interface along a common interface, with grids in relative motion on either side of the interface)

If	*Then*
ninter2 = 0	no patched-grid data is needed (no patched interfaces)
ninter2 > 0	patched-grid data calculated at every time step

Notes

(1) This type of patched-grid data cannot be read in from a file since it must be recalculated for every time step. If there are interfaces for which static patch data is required, use the standard patch input method (precalculate with ronnie and set **ninter** < 0 in "LT27 - Number of Patched-Grid Interfaces" on page 38).

(2) Any block segments that are dynamically patched must also be input in the boundary condition section ("LT13 - Boundary Condition Segments" on page 31 through "LT19 - KDIM Boundary Condition Specification" on page 35), using **bctype** = 0.

(3) Further information on patching can be found in the `input.doc` file for the ronnie preprocessor. It is recommended that some experience be gained with static patching (via ronnie) before attempting dynamic patching with moving zones.

(4) While most of the input for static and dynamic patching is the same, dynamic patching requires a few extra pieces of information, described below.

3.42 LT42 - Dynamic Patched-Grid Specifications

(Data for Line Type Forty-Two should be repeated **ninter2** times.)

int – interpolation number

ifit – type of fit

If	*Then*
ifit = 1	bi-linear fit
ifit = 2	serendipity (degenerate) bi-quadratic fit
ifit = 3	quadratic fit in ξ ; linear fit in η
ifit = 4	linear fit in ξ ; quadratic fit in η

limit – maximum step size (number of cells) to use in the search routine (**limit** = 1 recommended)

itmax – maximum number of iterations allowed to find each "to" cell (**itmax** = maximum grid dimension recommended)

mcxie – flag to indicate whether the $\xi = 0$ boundaries on either side of the patch interface are to be rendered coincident (generally required if $\xi = 0$ corresponds to a viscous surface)

If	*Then*
mcxie = 0	do not render the $\xi = 0$ boundaries coincident
mcxie > 100	render the $\xi = 0$ boundaries coincident
mcxie = 1 to 100	The code will try to decide whether or not the boundaries should be rendered coincident. The larger the value, the greater the number differences between the two boundaries can be before deciding they should not be rendered coincident.

Note

(1) For multiple "from" blocks, in which multiple ξ, η coordinate systems are defined, the proper $\xi = 0$ boundary to consider is the one closest to the $\xi = 0$ boundary on the "to" side.

mceta – flag to indicate whether the $\eta = 0$ boundaries on either side of the patch interface are to be rendered coincident (generally required if $\eta = 0$ corresponds to a viscous surface)

ic0 –flag for C-0 continuous boundaries (*not applicable to dynamically patched interfaces; set* **ic0** = 0)

iorph – flag to allow points which cannot be located by the search routine to be tagged as "orphan" points and to be marked as being "interpolated" from block 0

If	*Then*
iorph = 0	do not allow orphan points
iorph = 1	allow orphan points (not recommended for dynamic patch interfaces at this time)

itoss – flag to allow faster patching for cases in which the interface lies along a constant (or very nearly so) coordinate surface

If	*Then*
itoss = 0	interface is non-planar
itoss = 1	interface is an x = constant plane
itoss = 2	interface is an y = constant plane
itoss = 3	interface is an z = constant plane

Notes

(1) While **itoss** = 0 can always be used, it will slow down the dynamic patching considerably. Thus, if the interface is on a constant coordinate surface, the appropriate value of **itoss** is strongly recommended.

(2) All of the above parameters except **itoss** are also used as input to ronnie.

3.43 LT43 - Dynamic Patching "To" Grid Specifications

(Data for Line Type Forty-Three should be repeated **ninter2** times.)

int – interpolation number

to – a three or four digit number indicating the block number and face of the "to" surface ("to" refers to the block being interpolated)
to = Nmn
where
N is the block number (N may be up to two digits)
m is the coordinate direction; 1 = i, 2 = j, 3 = k
n is the minimum or maximum face; 1 = minimum, 2 = maximum

xie1 – starting index in ξ for which interpolation coefficients will be found on the "to" side of the interface

xie2 – ending index in ξ for which interpolation coefficients will be found on the "to" side of the interface

eta1 – starting index in η for which interpolation coefficients will be found on the "to" side of the interface

eta2 – ending index in η for which interpolation coefficients will be found on the "to" side of the interface

nfb – number of block boundaries which make up the "from" side of the patch surface ("from" refers to the block(s) from which the interpolations are made)

Notes
(1) If the entire range is desired, a shortcut is to set **xie1** = **xie2** = **eta1** = **eta2** = 0.
(2) *Important*: For dynamic patching, additional "ghost" block boundaries are often required. The baseline value of **nfb** is set from the zonal setup at $t = 0$ (as in static patching). However, as soon as the zones start to move past one another, portions of some block faces may not have an opposing block to interpolate from. To provide the patching algorithm with a block to interpolate from, one or more of the "from" blocks that match to the "to" side at $t = 0$ are duplicated and then translated or rotated by an appropriate amount to provide coverage of the "to" side as the zones move past one another. The baseline value of **ntb** must then be increased accordingly.

3.44 LT44 - Dynamic Patching "From" Grid Specifications

(Data for Line Type Forty-Four should be repeated **nfb** times for *each* Line Type Forty-Three.)

from – a three or four digit number indicating the block number and face of the "from" surface ("from" refers to the block(s) from which the interpolations are made)
from = Nmn
where
N is the block number (N may be up to two digits)
m is the coordinate direction; 1 = i, 2 = j, 3 = k
n is the minimum or maximum face; 1 = minimum, 2 = maximum

xie1 – starting index in ξ for which interpolation coefficients will be found on "from" side of interface

xie2 – ending index in ξ for which interpolation coefficients will be found on "from" side of interface

eta1 – starting index in η for which interpolation coefficients will be found on "from" side of interface

eta2 – ending index in η for which interpolation coefficients will be found on "from" side of interface

factj – expansion factor in the ξ direction (not implemented at this time (set **factj** = 0))

factk – expansion factor in the η direction (not implemented at this time (set **factk** = 0))

Note
(1) If the entire range is desired, a shortcut is to input zeros for the beginning and ending indices.

3.45 LT45 - Displacement and Rotation Amounts

(Data for Line Type Forty-Five should be repeated **nfb** times for *each* Line Type Forty-Three.)

dx – amount (in grid units) to displace the "from" block in the x direction

dy – amount (in grid units) to displace the "from" block in the y direction

dz – amount (in grid units) to displace the "from" block in the z direction

dthetx – amount (in degrees) to rotate the "from" block in the x direction

dthety – amount (in degrees) to rotate the "from" block in the y direction

dthetz – amount (in degrees) to rotate the "from" block in the z direction

CHAPTER 4

Nondimensionalization

The goal of many CFD problems is to solve the flow over configurations that have been tested in wind tunnels or flight tests. In experimental tests, the model being studied is of known dimensions. Since there is no standard system of measurement for such testing, some models are measured in feet, some in meters, some in inches, etc. Similarly, wind speeds might be measured in feet/second or meters/second or some other way. Nondimensionalizing the flow-field parameters in a CFD code removes the necessity of converting from one system to another within the code.

Throughout this chapter, the term *free stream* is used. Please see the note on page 3 concerning its usage.

4.1 Dimensional Parameter Definitions

The following parameters are used to nondimensionalize the flow-field variables:

$\tilde{L}$ = characteristic length, feet (e.g. chord)

L_{ref} = corresponding length in the grid (nondimensional)

$\tilde{L}_R = \tilde{L}/L_{ref}$ = reference length used by code, feet (dimensional)

$\tilde{\rho}_\infty$ = free-stream density, slug/feet^3

$\tilde{a}_\infty$ = free-stream speed of sound, feet/second

$|\tilde{\mathbf{V}}|_\infty$ = free-stream velocity, feet/second

$\tilde{\mu}_\infty$ = free-stream molecular viscosity, slug/feet-second

$\tilde{t}$ = time, seconds

The ~ here and elsewhere indicates a dimensional quantity. Note that meters or inches, etc. could be substituted for feet in the above definitions, as long as it is done consistently.

4.2 General Flow-Field Variables

In CFL3D, the nondimensionalizations are performed as follows:

$$\rho = \frac{\tilde{\rho}}{\tilde{\rho}_\infty} \qquad \rho_\infty = 1$$

$$u = \frac{\tilde{u}}{\tilde{a}_\infty} \qquad u_\infty = M_\infty \cos\alpha\cos\beta$$

$$v = \frac{\tilde{v}}{\tilde{a}_\infty} \qquad v_\infty = -M_\infty \sin\beta \qquad \textbf{(4-1)}$$

$$w = \frac{\tilde{w}}{\tilde{a}_\infty} \qquad w_\infty = M_\infty \sin\alpha\cos\beta$$

$$p = \frac{\tilde{p}}{\tilde{\rho}_\infty(\tilde{a}_\infty)^2} \qquad p_\infty = \frac{1}{\gamma}$$

Also,

$$e = \frac{\tilde{e}}{\tilde{\rho}_\infty(\tilde{a}_\infty)^2} \qquad e_\infty = \frac{1}{\gamma(\gamma-1)} + \frac{(M_\infty)^2}{2}$$

$$a = \frac{\tilde{a}}{\tilde{a}_\infty} \qquad a_\infty = 1 \qquad \textbf{(4-2)}$$

$$T = \frac{\tilde{T}}{\tilde{T}_\infty} = \frac{\gamma p}{\rho} = a^2 \qquad T_\infty = 1$$

$$x = \frac{\tilde{x}}{\tilde{L}_R} \qquad y = \frac{\tilde{y}}{\tilde{L}_R}$$

$$z = \frac{\tilde{z}}{\tilde{L}_R} \qquad t = \frac{\tilde{t}\tilde{a}_\infty}{\tilde{L}_R} \qquad \textbf{(4-3)}$$

Also, Reynolds number based on the characteristic length is

$$Re_{\tilde{L}} = \frac{\tilde{\rho}_\infty |\tilde{\mathbf{V}}|_\infty \tilde{L}}{\tilde{\mu}_\infty} \tag{4-4}$$

Note that $Re_{\tilde{L}}$ is *not* input into the code, but rather the Reynolds number per unit grid length, in millions:

$$\mathbf{reue} = \frac{Re_{\tilde{L}}}{L_{ref}} \times 10^{-6} = \frac{\tilde{\rho}_\infty |\tilde{\mathbf{V}}|_\infty \tilde{L}_R}{\tilde{\mu}_\infty} \times 10^{-6} = Re_{\tilde{L}_R} \times 10^{-6} \tag{4-5}$$

The free-stream Mach number is

$$M_\infty = \frac{|\tilde{\mathbf{V}}|_\infty}{\tilde{a}_\infty} \tag{4-6}$$

where the free-stream velocity magnitude is

$$|\tilde{\mathbf{V}}|_\infty = \sqrt{(\tilde{u}_\infty)^2 + (\tilde{v}_\infty)^2 + (\tilde{w}_\infty)^2} \tag{4-7}$$

The molecular viscosity is defined from Sutherland's[45] law as follows:

$$\mu = \frac{\tilde{\mu}}{\tilde{\mu}_\infty} = \left(\frac{\tilde{T}}{\tilde{T}_\infty}\right)^{\frac{3}{2}} \left[\frac{\tilde{T}_\infty + \tilde{c}}{\tilde{T} + \tilde{c}}\right] \tag{4-8}$$

$$= (T)^{\frac{3}{2}} \left[\frac{1 + \frac{\tilde{c}}{\tilde{T}_\infty}}{T + \frac{\tilde{c}}{\tilde{T}_\infty}}\right]$$

where $\tilde{c} = 198.6°R = 110.4°K$ is Sutherland's constant.

4.3 Reynolds Number Examples

Experience has shown that some confusion arises when it comes to setting the parameter **reue** in CFL3D. Some examples may help.

Suppose the configuration to be analyzed has a body length of 48 inches and it is desired to run a simulation in which the Reynolds number based on the body length is 6×10^6. Thus, $\tilde{L} = 48$ inches and $Re_{\tilde{L}} = 6\times10^6$.

To carry out the simulation, a grid is required. Suppose three people each create a grid. In the first person's grid, the length of the body is 48, i.e. the configuration is full size in the grid coordinate system. Thus, L_{ref} in grid 1 is 48. Therefore, by Equation (4-5),

$$\mathbf{reue} = \frac{6\times10^{6}}{48}\times10^{-6} = 0.125 \tag{4-9}$$

The second person decides to normalize the grid such that the body has a length of 1. Thus, L_{ref} in grid 2 is 1 and by Equation (4-5),

$$\mathbf{reue} = \frac{6\times10^{6}}{1}\times10^{-6} = 6.0 \tag{4-10}$$

The third person was given the geometry definition in centimeters. Grid 3 is therefore generated with a body length of 121.9 (48 inches ≈ 121.9 centimeters), so $L_{ref} = 121.9$ and

$$\mathbf{reue} = \frac{6\times10^{6}}{121.9}\times10^{-6} = 0.04922 \tag{4-11}$$

4.4 Time and Time Step

Time is nondimensionalized as follows:

$$t = \frac{\tilde{t}\tilde{a}_{\infty}}{\tilde{L}_R} \tag{4-12}$$

If a particular dimensional time step (**dt** in "LT5 - Time Step Parameters" on page 21) is required, then this relation may be used to determine the corresponding input value of Δt.

For the time step,

$\tilde{\Delta}$t = desired time step, seconds

Therefore,

$$\Delta t = \frac{\tilde{\Delta}\tilde{t}\tilde{a}_{\infty}}{\tilde{L}_R} \tag{4-13}$$

4.5 Moving Mesh

These additional values may be needed for the nondimensionalization of the moving-mesh parameters:

$\tilde{f}$ = frequency, cycles/second

k_r = reduced frequency, $k_r = \tilde{f}\tilde{L}/\tilde{a}_\infty$ Note: This definition of reduced frequency differs from the "standard" definition $k_r = \tilde{f}\tilde{L}/|\tilde{\mathbf{V}}|_\infty$. Also, note that the quantities $\tilde{L}$ and L_{ref} used to nondimensionalize the moving mesh data need not be the *same* as those used to nondimensionalize x, y, z, and t. That is why L_{ref} is input explicitly in the moving mesh section of the input file.

$k_g = \tilde{L}/(2\pi\tilde{a}_\infty\tilde{t}_0)$ = rate of growth of displacement, where $\tilde{t}_0$ = the time to reach 63.2% of the maximum displacement, seconds

4.5.1 Translational Motion

These dimensional parameters are also needed for the translational nondimensionalizations:

$\tilde{u}_{trans}$ = x component of translational velocity, feet/second (needed for **itrans** = 1)

$\tilde{v}_{trans}$ = y component of translational velocity, feet/second (needed for **itrans** = 1)

$\tilde{w}_{trans}$ = z component of translational velocity, feet/second (needed for **itrans** = 1)

$\tilde{x}_{max}$ = maximum x displacement, feet (needed for **itrans** > 1)

$\tilde{y}_{max}$ = maximum y displacement, feet (needed for **itrans** > 1)

$\tilde{z}_{max}$ = maximum z displacement, feet (needed for **itrans** > 1)

Again note that meters or inches, etc. could be substituted for feet in the above definitions, as long as it is done consistently. Also, not all of the above dimensional parameters are relevant to a given problem. For example, in sinusoidal plunging (**itrans** = 2), k_g, $\tilde{u}_{trans}$, $\tilde{v}_{trans}$, and $\tilde{w}_{trans}$ are irrelevant.

4.5.1.1 Translation With Constant Velocity

For **itrans** = 1, input the following nondimensional parameters:

rfreq = 0.0

utrans = $\tilde{u}_{trans}/\tilde{a}_{\infty}$ (Mach number of x translation)

vtrans = $\tilde{v}_{trans}/\tilde{a}_{\infty}$

wtrans = $\tilde{w}_{trans}/\tilde{a}_{\infty}$

The nondimensional displacement in the code is then governed by

$$\begin{aligned} x &= u_{trans}t \\ y &= v_{trans}t \\ z &= w_{trans}t \end{aligned} \qquad \textbf{(4-14)}$$

4.5.1.2 Sinusoidal Plunging

For **itrans** = 2, input the following nondimensional parameters:

rfreq = k_r

utrans = $\tilde{x}_{max}/\tilde{L}_R$

vtrans = $\tilde{y}_{max}/\tilde{L}_R$

wtrans = $\tilde{z}_{max}/\tilde{L}_R$

Note that, $\tilde{x}_{max}/\tilde{L}_R$, for example, is simply the maximum x displacement in "grid units". The nondimensional displacement in the code is then governed by

$$x = u_{trans}\sin\left(2\pi\frac{k_r t}{L_{ref}}\right)$$
$$y = v_{trans}\sin\left(2\pi\frac{k_r t}{L_{ref}}\right) \quad \text{(4-15)}$$
$$z = w_{trans}\sin\left(2\pi\frac{k_r t}{L_{ref}}\right)$$

If the time step is to be chosen so that there are N steps per cycle, the required input value of **dt** (see "LT5 - Time Step Parameters" on page 21) may be determined from

$$\mathbf{dt} = \frac{L_{ref}}{k_r \cdot N} \quad \text{(4-16)}$$

The corresponding dimensional time step is

$$\tilde{\Delta t} = \frac{1}{N \cdot \tilde{f}} \quad \text{(4-17)}$$

4.5.1.3 Acceleration to Constant Displacement

For **itrans** = 3, input the following nondimensional parameters:

rfreq = k_g

utrans = $\tilde{x}_{max}/\tilde{L}_R$

vtrans = $\tilde{y}_{max}/\tilde{L}_R$

wtrans = $\tilde{z}_{max}/\tilde{L}_R$

Note that, $\tilde{x}_{max}/\tilde{L}_R$, for example, is simply the maximum x displacement in "grid units". The nondimensional displacement in the code is then governed by

$$x = u_{trans}\left[1.0 - \exp\left(-2\pi\frac{k_r t}{L_{ref}}\right)\right]$$
$$y = v_{trans}\left[1.0 - \exp\left(-2\pi\frac{k_r t}{L_{ref}}\right)\right] \quad \text{(4-18)}$$
$$z = w_{trans}\left[1.0 - \exp\left(-2\pi\frac{k_r t}{L_{ref}}\right)\right]$$

Thus, with **utrans** = x_{max}, x will attain 63.2% of it's maximum value, i.e.

$$\frac{x}{x_{max}} = 1 - \frac{1}{e} \quad \text{(4-19)}$$

at a nondimensional time of

$$t_0 = \frac{L_{ref}}{2\pi k_r} \quad \text{(4-20)}$$

rfreq for **itrans** = 3 may therefore be defined as

$$\mathbf{rfreq} = \frac{L_{ref}}{2\pi t_0} = \frac{\tilde{L}}{2\pi\tilde{a}_\infty\tilde{t}_0} \quad \text{(4-21)}$$

where $\tilde{t}_0$ is the time (seconds) to reach 63.2% of the maximum displacement. Increasing **rfreq** has the effect of decreasing the time to attain a specified percentage of the asymptotic maximum displacement.

4.5.2 Rotational Motion

These additional dimensional parameters are needed for the rotational nondimensionalizations:

$\tilde{\omega}_x$ = x component of rotational velocity, revolutions/second (needed for **irotat** = 1)

$\tilde{\omega}_y$ = y component of rotational velocity, revolutions/second (needed for **irotat** = 1)

$\tilde{\omega}_z$ = z component of rotational velocity, revolutions/second (needed for **irotat** = 1)

$\tilde{\theta}_{x,\,max}$ = maximum angular displacement about x axis, degrees (needed for **irotat** >1)

$\tilde{\theta}_{y,\,max}$ = maximum angular displacement about y axis, degrees (needed for **irotat** >1)

$\tilde{\theta}_{z,\,max}$ = maximum angular displacement about z axis, degrees (needed for **irotat** >1)

Again note that meters or inches, etc. could be substituted for feet in the above definitions, as long as it is done consistently. Not all of the above dimensional parameters are relevant to a given problem. For example, in sinusoidal pitching (**irotat** = 2) k_g, $\tilde{\omega}_x$, $\tilde{\omega}_y$, and $\tilde{\omega}_z$ are irrelevant.

Using these dimensional values, the nondimensionalizations are performed as follows:

4.5.2.1 Rotation With Constant Angular Velocity

For **irotat** = 1, input the following nondimensional parameters:

rfreq = 0.

omegax = $\tilde{\omega}_x \tilde{L} / \tilde{a}_\infty$

omegay = $\tilde{\omega}_y \tilde{L} / \tilde{a}_\infty$

omegaz = $\tilde{\omega}_z \tilde{L} / \tilde{a}_\infty$

For turbomachinery type applications, $\tilde{L}$ is typically chosen as the prop/blade diameter, so that $\tilde{\omega}_x$, $\tilde{\omega}_y$, $\tilde{\omega}_z$ = M_∞ / (advance ratio). The angular displacement (radians) in the code is then governed by

$$\begin{aligned} \theta_x &= 2\pi \frac{\omega_x t}{L_{ref}} \\ \theta_y &= 2\pi \frac{\omega_y t}{L_{ref}} \\ \theta_z &= 2\pi \frac{\omega_z t}{L_{ref}} \end{aligned} \qquad \text{(4-22)}$$

4.5.2.2 Sinusoidal Variation of Angular Displacement

For **irotat** = 2, input the following parameters:

rfreq $= k_r$

omegax = $\tilde{\theta}_{x,\,max}$, degrees

omegay = $\tilde{\theta}_{y,\,max}$, degrees

omegaz = $\tilde{\theta}_{z,\,max}$, degrees

The rotational displacement (radians) in the code is then governed by

$$\begin{aligned} \theta_x &= \omega_x \frac{\pi}{180} \sin\left(2\pi k_r \frac{t}{L_{ref}}\right) \\ \theta_y &= \omega_y \frac{\pi}{180} \sin\left(2\pi k_r \frac{t}{L_{ref}}\right) \\ \theta_z &= \omega_z \frac{\pi}{180} \sin\left(2\pi k_r \frac{t}{L_{ref}}\right) \end{aligned} \tag{4-23}$$

If the time step is to be chosen so that there are N steps per cycle, the required input value of **dt** (see “LT5 - Time Step Parameters” on page 21) may be determined from

$$\Delta t = \frac{L_{ref}}{k_r N} \tag{4-24}$$

The corresponding dimensional time step is:

$$\tilde{\Delta t} = \frac{1}{N \tilde{f}} \tag{4-25}$$

4.5.2.3 Acceleration to a Constant Rotational Displacement

For **irotat** = 3, input the following nondimensional parameters:

rfreq $= k_g$

omegax = $\tilde{\theta}_{x,\,max}$, degrees

omegay = $\tilde{\theta}_{y,\,max}$, degrees

omegaz = $\tilde{\theta}_{z,\,max}$, degrees

The rotational displacement (radians) in the code is then governed by

$$\theta_x = \frac{\pi\omega_x}{180}\left[1 - \exp\left(-2\pi k_r \frac{t}{L_{ref}}\right)\right]$$
$$\theta_y = \frac{\pi\omega_y}{180}\left[1 - \exp\left(-2\pi k_r \frac{t}{L_{ref}}\right)\right] \qquad \textbf{(4-26)}$$
$$\theta_z = \frac{\pi\omega_z}{180}\left[1 - \exp\left(-2\pi k_r \frac{t}{L_{ref}}\right)\right]$$

Thus, with $\omega_x = \tilde{\theta}_{x,\,max}$, θ_x will attain 63.2% of it's maximum value, i.e.

$$\theta_x / \theta_{x,\,max} = 1 - \frac{1}{e} \qquad \textbf{(4-27)}$$

at a nondimensional time of

$$t_0 = \frac{L_{ref}}{2\pi k_r} \qquad \textbf{(4-28)}$$

Increasing k_r has the effect of decreasing the time to attain a specified percentage of the asymptotic maximum displacement. **rfreq** for **irotat** = 3 may therefore be defined as

$$\textbf{rfreq} = \frac{L_{ref}}{2\pi t_0} = \frac{\tilde{L}}{2\pi \tilde{a}_\infty \tilde{t}_0} \qquad \textbf{(4-29)}$$

where $\tilde{t}_0$ is the time (seconds) to reach 63.2% of maximum displacement.

CHAPTER 5

File Formats

This chapter describes the input and output files listed in Section 3.1. Typical names are given below; however, these names can be changed by the user to pertain to certain cases.

```
I/O FILES
grid.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
```

5.1 Input Files

There are two input files that *must* be available to CFL3D whenever a case is submitted. These files are the grid file (unit 1) and the input file (unit 5) defining the case. The latter file is described in detail in Chapter 3. A restart file (unit 2) *must* be available for a restart case; however, for a start from "scratch", it is not needed. In addition, there are two files that *may* be needed at submission depending on the problem being solved. When overlapped grids are used, the interpolation and boundary data file (unit 21) must be available. When patching is utilized, the patched boundary data file (unit 22) must be available.

5.1.1 Grid File

grid.bin (unit 1)

The grid must be generated using grid-generation software (CFL3D does not generate grids). Take care when generating a grid. Grid spacings should vary smoothly; the grid should "look nice". (See the troubleshooting notes about grids in Chapter 10). Unless there is a good reason for not doing so, the grids, as well as all segments, should be multigridable (see Section 7.1.2 on page 127). Grid stretching should not be too severe if it can be avoided since it can degrade the accuracy and convergence of the code.

The grid file is in binary format. In the input file, the parameter **ngrid** is set as the total number of grids to be read in by CFL3D. The *sign* of **ngrid** indicates whether the grid will be read in CFL3D format or in PLOT3D format. (Note that TLNS3D[42] uses PLOT3D format.) CFL3D-type grids always have **alpha** measured in the $x-z$ plane, with z as the "up" direction (i.e. **alpha** = $90°$ would give a free-stream velocity vector in the positive z direction). PLOT3D-type grids may be utilized with either z being "up" (**alpha** measured in the $x-z$ plane) or y being "up" (**alpha** measured in the $x-y$ plane) depending on the value of **ialph** (see "LT3 - Flow Conditions" on page 19). See Appendix I for an illustrated definition of α. When PLOT3D-type grids are input, they are immediately converted internal to the code to CFL3D format. Thus the actual computations are *always* carried out internally as if z is "up".

If **ngrid** > 0, the following CFL3D grid format is used to read in the grid(s):

```
      do 10 n=1,ngrid
      read(1) jdim(n),kdim(n),idim(n)
      read(1) (((x(i,j,k,n),j=1,jdim(n)),k=1,kdim(n)),i=1,idim(n)),
     .        (((y(i,j,k,n),j=1,jdim(n)),k=1,kdim(n)),i=1,idim(n)),
     .        (((z(i,j,k,n),j=1,jdim(n)),k=1,kdim(n)),i=1,idim(n))
   10 continue
```

If **ngrid** < 0, the following PLOT3D grid format is used to read in the grid(s):

```
      read(1) ngrid
      read(1) (idim(n),jdim(n),kdim(n),n=1,ngrid)
      do 20 n=1,ngrid
      read(1) (((x(i,j,k,n),i=1,idim(n)),j=1,jdim(n)),k=1,kdim(n)),
     .        (((y(i,j,k,n),i=1,idim(n)),j=1,jdim(n)),k=1,kdim(n)),
     .        (((z(i,j,k,n),i=1,idim(n)),j=1,jdim(n)),k=1,kdim(n))
   20 continue
```

It does not matter which grid format is used. What *does* matter is the particular orientation of i, j, and k relative to the body and flow field, since the matrix inversions are done in a particular order. Experience has indicated that it is usually best (from speed of convergence perspective) to have the k direction be the primary viscous direction (i.e., the primary direction of the grid lines normal to the body). If bodies exist with more than one grid line orientation, then choose one of them to serve as the indicator of the "primary" viscous direction. Also, it is usually best if the j direction is taken as the streamwise direction and the i direction as the spanwise direction. Hence, for an existing grid with i is in the streamwise direction, j in the normal direction, and k in the spanwise direction, it might be a good idea to first swap the indices: $i \rightarrow j$, $j \rightarrow k$, and $k \rightarrow i$, to put them in CFL3D's "preferred" orientation before running. The new grid created can be either CFL3D-type or PLOT3D-type. Finally, note that it is not always necessary to follow this guideline; performance is case dependent. Just be sure to satisfy the right-hand rule at all times, as discussed in the next section.

For 2-D grids (**i2d**=1), **idim** must be 2 (i.e., the grid must be described by 2 repeated planes). We recommend making the grid-distance between the planes = 1, although any distance is okay provided that it is accounted for in the **sref** term in the input file.

5.1.2 The Right-Hand Rule

The right-hand rule *must* be obeyed by the grid coordinates (x, y, z) *and* the grid indices (i, j, k). Error messages and an immediate halt to execution will result if it is not. As a reminder of this important rule, consider the system of axes in Figure 5-1. Imagine pointing the right-hand index finger along the positive x axis. Then curl the rest of the right-hand fingers toward the positive y axis. The thumb, pointing straight out, is now in the positive z direction.

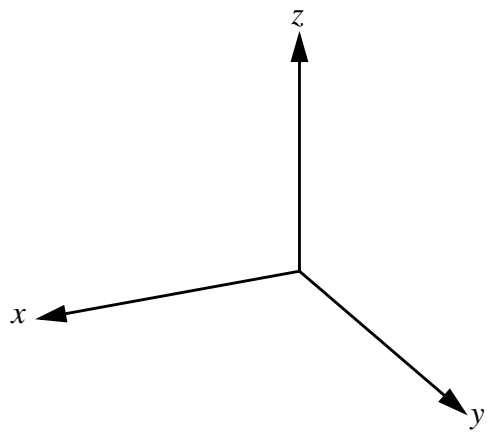


Figure 5-1. Three-dimensional system of axes showing positive x, y, and z directions.

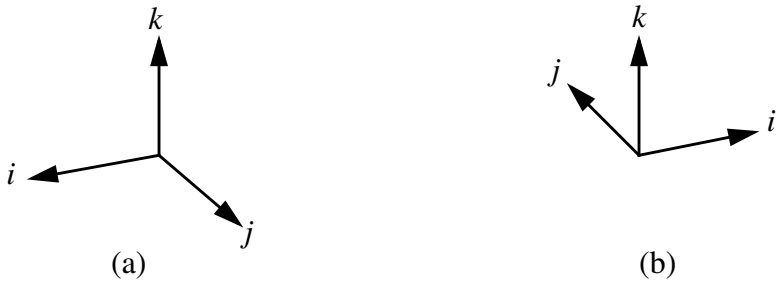


Figure 5-2. Index direction examples.

Keep in mind that the same rule applies to the indices i, j, k. Suppose the positive i, j, and k directions run along the positive x, y, and z axes, respectively, as drawn in Figure 5-1. This is the simplest case and is illustrated in Figure 5-2(a). Suppose it is necessary, however, for the positive i index direction to run in the opposite direction of positive x as drawn in Figure 5-2(b). (This means, for example, that if x varies from 0.0 to 1.0 and **idim** = 5, then at $i = 1$, $x = 1.0$ and at $i = 5$, $x = 0.0$.) One of the other index directions

must also be changed to maintain the right-hand rule. In Figure 5-2(b), the j index direction has been changed.

Also, i, j, and k do not *have* to be in the x, y, and z directions, respectively. Any permutation is valid as long as the right-hand rule is upheld. (However, keep in mind the discussion above in Section 5.1.1 as well as *Note* (6) describing the **ivisc**(m) input on page 27 when setting the index directions.)

5.1.3 Restart File

restart.bin (unit 2)

The restart file is in binary format. It must be available at the beginning of the program execution for a restart of a previous solution (**irest** > 0). *The restart file is overwritten during program execution; therefore, before resubmitting the case, be sure to save any restart files that may be needed later!* The binary restart file has the following format. (Note that this Version 5.0 format is different from the formats in Version 4.1 and earlier versions.)

General case information, the `q` array, and the turbulence quantities are written as follows (`nblk` is the total number of blocks):

```
      do nb=1,nblk
      write(3) titlw,xmachw,jdim,kdim,idim,alphw,reuew,ntr,time
c
      if (nb.eq.1) then
         write(3) (rms(n),     n=1,ntr),(clw(n),     n=1,ntr),
     .            (cdw(n),     n=1,ntr),(cdpw(n),    n=1,ntr),
     .            (cdvw(n),    n=1,ntr),(cxw(n),     n=1,ntr),
     .            (cyw(n),     n=1,ntr),(czw(n),     n=1,ntr),
     .            (cmxw(n),    n=1,ntr),(cmyw(n),    n=1,ntr),
     .            (cmzw(n),    n=1,ntr),(fmdotw(n),  n=1,ntr),
     .            (cftmomw(n), n=1,ntr),(cftpw(n),   n=1,ntr),
     .            (cftvw(n),   n=1,ntr),(cfttotw(n), n=1,ntr)
      end if
c
      write(3) ((((q(j,k,i,l,nb),j=1,jdim-1),k=1,kdim-1),
     .               i=1,idim-1),l=1,5)
c
      write(3) iv1,iv2,iv3
      if (nb.eq.1) then
         write(3) (rmstr1(n),n=1,ntr),(rmstr2(n),n=1,ntr),
     .            (nneg1(n), n=1,ntr),(nneg2(n), n=1,ntr)
      end if
c
      if (iv1.ge.2 .or. iv2.ge.2 .or. iv3.ge.2) then
        write(3) (((vist3d(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
      end if
      if (iv1.ge.4 .or. iv2.ge.4 .or. iv3.ge.4) then
         write(3) ((((tursav(j,k,i,m,nb),j=1,jdim-1),k=1,kdim-1),
     .                i=1,idim-1),m=1,2)
         write(3) (((smin(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
         if (iv1.eq.4 .or. iv2.eq.4 .or. iv3.eq.4) then
         write(3)(((xjb(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
         write(3)(((xkb(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
         write(3)(((blnum(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
```

```
          end if
          if (iv1.eq.9 .or. iv2.eq.9 .or. iv3.eq.9  .or.
     .        iv1.eq.13.or. iv2.eq.13.or. iv3.eq.13) then
          write(3) (((cmuv(j,k,i,nb),j=1,jdim-1),k=1,kdim-1),i=1,idim-1)
          end if
        end if
        end do
c
        if (dt.gt.0) then
          write(3) iflagg
          if (iflagg.eq.1 .or. iflagg.eq.3) then
            do nb=1,nblk
               write(3) jdim,kdim,idim
               write(3) ((((qc0(j,k,i,l,nb),j=1,jdim-1),k=1,kdim-1),
     .                     i=1,idim-1),l=1,5)
            end do
          end if
c
          if (iflagg.eq.2 .or. iflagg .eq.3) then
            do nb=1,nblk
               write(3) iunst
               if (iunst .ne. 0) then
                 write(3) jdim,kdim,idim
                 write(3) itrans,rfreqt,xorig,yorig,zorig,xorig0,yorig0,
     .                     zorig0,utrans,vtrans,wtrans,dxmx,dymx,dzmx,
     .                     itransmc,rfreqtmc,xorigmc,yorigmc,zorigmc,
     .                     xorig0mc,yorig0mc,zorig0mc,utransmc,vtransmc,
     .                     wtransmc,xmc,ymc,zmc,dxmxmc,dymxmc,dzmxmc,
     .                     irotat,rfreqr,thetax,thetay,thetaz,
     .                     omegax,omegay,omegaz,dthxmx,dthymx,dthzmx,
     .                     irotatmc,rfreqrmc,thetaxmc,thetaymc,thetazmc,
     .                     omegaxmc,omegaymc,omegazmc,dthxmxmc,dthymxmc,
     .                     dthzmxmc,time2,time2mc,dt
               end if
            end do
          end if
        end if
```

Where	*Indicates*
`titlw`	a real array of length 20 describing the case
`xmachw`	Mach number
`jdim,kdim,idim`	grid dimensions
`alphw`	angle of attack
`reuew`	Reynolds number
`ntr`	number of iterations
`time`	time (for time-accurate computations)
`rms`	residual
`clw`	lift coefficient
`cdw`	drag coefficient
`cyw`	force coefficient in y direction
`cmyw`	moment coefficient about y
`cdpw`	pressure drag coefficient
`cdvw`	viscous drag coefficient
`cxw`	force coefficient in x direction
`czw`	force coefficient in z direction
`cmxw`	moment coefficient about x

`cmzw`	moment coefficient about z
`fmdotw`	mass flow through control surface
`cftmomw`	magnitude of resultant momentum force on control surface
`cftpw`	magnitude of resultant pressure force on control surface
`cftvw`	magnitude of resultant viscous force on control surface
`cfttotw`	magnitude of resultant (pressure + momentum + viscous) force on control surface
`q`	nondimensional primitive variables; $\mathbf{q} = [\rho, u, v, w, p]$
`iv1,iv2,iv3`	**ivisc**(1), **ivisc**(2), and **ivisc**(3)
`rmstr1`	residual for first turbulence model equation (if used)
`rmstr2`	residual for second turbulence model equation (if used)
`nneg1`	number of points at which first turbulence model variable needed to be limited
`nneg2`	number of points at which second turbulence model variable needed to be limited
`vist3d`	nondimensional turbulent eddy viscosity μ_T
`tursav`	nondimensional turbulence quantities for field-equation turbulence models (varies with model); `tursav`(2) is also used to store nearest wall i location for the Baldwin-Barth turbulence model
`smin`	minimum distance function for field-equation turbulence models
`xjb`	nearest wall j location for Baldwin-Barth turbulence model
`xkb`	nearest wall k location for Baldwin-Barth turbulence model
`blnum`	nearest wall block number for Baldwin-Barth turbulence model
`cmuv`	variable C_μ coefficient for Girimaji turbulence model
`qc0`	**Q** values at previous time step
`iflagg`	parameter describing temporal order and whether grid is moving or stationary = 0 not second order, stationary grid = 1 second order, stationary grid = 2 not second order, moving grid = 3 second order, moving grid

The dynamic grid parameters in the restart file are not described here.

5.1.4 Grid-Overlapping and Patching Data Files

ovrlp.bin (unit 21)

The `ovrlp.bin` file is a binary file, which is only used for grid-overlapping cases. It is always listed at the top of the input deck. The file contains the interpolation stencils for overlapped grids and is created by running MaGGiE.

patch.bin (unit 22)

The `patch.bin` file is a binary file, which is only used for grid-patching cases. It is also always listed at the top of the input deck. The file contains the interpolation stencils for patched grids and is created by running ronnie.

5.2 Output Files

CFL3D produces several output files. The following sections describe the contents and formats of these files.

5.2.1 PLOT3D *Files*

The PLOT3D files are binary files (or ascii, with the modification to the code as described in the footnote at the beginning of Chapter 3) which are input to flow visualization programs like PLOT3D or FAST. Partial or entire flow-field regions may be specified by the user in the input file. The parameter **nplot3d** specifies the number of regions to be written to the files and "LT28 - PLOT3D Output Specifications" on page 39 indicates the desired index ranges. Data may also be extracted from these files for post-processing programs written by the user. In order to read these files for such a purpose, use the formats below. The number of grids to be read is indicated by `nplots` and `id`, `jd`, `kd` are the number of points to be read on each grid. Note that these files can be specified as grid-point data files (**iptype** = 0) or cell-center data files (**iptype** = 1).

plot3dg.bin (unit 3)

For 3-d grids,

```
write(3) nplots
write(3)(id(n),jd(n),kd(n),n=1,nplots)
do n=1,nplots
write(3) (((x(j,k,i,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n)),
.        (((y(j,k,i,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n)),
.        (((z(j,k,i,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n)),
.        (((iblank(j,k,i,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n))
end do
```

For 2-d grids,

```
write(3) nplots
write(3)(jd(n),kd(n),n=1,nplots)
do n=1,nplots
write(3) ((x(j,k,n),j=1,jd(n)),k=1,kd(n)),
.        ((z(j,k,n),j=1,jd(n)),k=1,kd(n)),
.        ((iblank(j,k,n),j=1,jd(n)),k=1,kd(n))
end do
```

plot3dq.bin (unit 4)

The flow-field data file has the following formats. For 3-d grids,

```
write(4) nplots
write(4)(id(n),jd(n),kd(n),n=1,nplots)
do n=1,nplots
write(4) xmach,alpha,reue,time
write(4) ((((q(j,k,i,m,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n)),m=1,5)
end do
```

For 2-d flow-field data files,

```
write(4) nplots
write(4)(jd(n),kd(n),n=1,nplots)
do n=1,nplots
write(4) xmach,alpha,reue,time
write(4) (((q(j,k,m,n),j=1,jd(n)),k=1,kd(n)),m=1,4)
end do
```

Note that five flow-field variables are output for 3-d grids, whereas four variables are output for 2-d grids. Also, note that `q` here is the array of conserved variables **Q**.

When using **iptype** = 0, be aware that the solution on block edges and corners may be inaccurate. This is because CFL3D solves for cell-center values. Values at grid points must be reconstructed from cell-center data and the reconstruction process at edges/corners is not unique.

When **iptype** = 2, CFL3D outputs turbulence quantity information rather than the conserved variables **Q**. See the note in Section 3.28 on page 39.

When **iptype** > 2, a PLOT3D function file is output in place of the **Q** file. The function file format for 3-d grids is

```
write(4) nplots
write(4)(id(n),jd(n),kd(n),1,n=1,nplots)
do n=1,nplots
write(4) (((f(j,k,i,n),i=1,id(n)),j=1,jd(n)),k=1,kd(n))
end do
```

For 2-d function files,

```
write(4) nplots
write(4)(jd(n),kd(n),1,n=1,nplots)
do n=1,nplots
write(4) ((f(j,k,n),j=1,jd(n)),k=1,kd(n))
end do
```

Note that function files are currently written for a single function only. See Section 3.28 on page 39 for information on what functions are currently available.

5.2.2 Ascii Format Files

cfl3d.out (unit 11)

The `cfl3d.out` file is the primary output file. It echoes the input file with the appropriate bookkeeping changes, such as updates in block numbers when coarser grids are generated internally. The file then gives a "blow-by-blow" message about what steps have been performed: the grid is read in, the metrics are computed, etc. until finally the other output files are written. Another important function of this file is to print any warnings and error messages. For example, it will reveal any mismatches in one-to-one block connections, signaling a problem with the grid or input file. It is a good idea to run a new job for a few iterations only and then to study this file to make sure everything looks as expected.

cfl3d.res (unit 12)

When **ihstry** = 0, the `cfl3d.res` file contains the residual, lift, drag, side force, and moment histories. When **ihstry** > 0, the residual, mass flow, pressure force, viscous force, and momentum force are given instead. This type of file has been used for all of the "iteration history" plots presented in this manual. The following sample format shows a portion of the `cfl3d.res` file (when **ihstry** = 0) for the flat plate test case discussed in Section 9.1.5 on page 173. When sub-iterations are used in a time-accurate computation, only the final residuals and forces from the last sub-iteration of each time step are output to unit 12. (For the sub-iteration residual history file, see page 76.)

```
   turbulent flat plate (plate from j=17-65, prior to 17 is symmetry)
Mach=  0.2000E+00, alpha=  0.0000E+00, ReUe=  0.6000E+07
Final res=  0.3623E-09
Final cl,cd,cy,cmy= -0.1869E-02  0.2827E-02  0.0000E+00  0.3174E-03
  800 it    log(res)        cl            cd            cy            cmy
       1  -0.15700E+02   0.25376E-12   0.67093E+00   0.00000E+00  -0.12688E-12
       2  -0.59009E+01   0.13542E-01   0.40383E-01   0.00000E+00  -0.72299E-02
                .
                .
                .
     799  -0.94402E+01  -0.18662E-02   0.28275E-02   0.00000E+00   0.31613E-03
     800  -0.94409E+01  -0.18695E-02   0.28273E-02   0.00000E+00   0.31742E-03
```

Here, `res` is the L-2 norm of the residual for the equation for density.

cfl3d.turres (unit 13)

The `cfl3d.turres` file contains the residual history for the one or two-equation turbulence models (**ivisc** > 3). The sample below shows part of the `cfl3d.turres` file for the flat plate test case discussed in Section 9.1.5 on page 173.The `log(turres1)` quantity is the log of the residual for all one-equation models or the log of the residual for the ω or ε equation for two-equation models. `Log(turres2)` is the log of the residual for the k equation for two-equation models and is not used for one-equation models. `Nneg1` and `nneg2` indicate how many field points require limiting (to avoid going negative) during that iteration. They should be zero or, at most, a small fraction of the total number of points for a converged solution. If they are large, they indicate a problem with convergence. (They may be large during start-up, as in this example, but should always go to zero or small val-

ues as the solution progresses.) When sub-iterations are used in a time-accurate computation, only the final residuals from the *last* sub-iteration of each time step are output.

```
   turbulent flat plate (plate from j=17-65, prior to 17 is symmetry)
 Mach=  0.2000E+00, alpha=  0.0000E+00, ReUe=  0.6000E+07
 Final turres1=  0.5279E-09
 Final turres2=  0.5186E-09
   800 it  log(turres1)  log(turres2)  nneg1  nneg2
        1  -0.31193E+01  -0.43162E+01      0     92
        2  -0.34342E+01  -0.53959E+01      0   2637
                .
                .
                .
      799  -0.92770E+01  -0.92830E+01      0      0
      800  -0.92774E+01  -0.92852E+01      0      0
```

cfl3d.blomax (unit 14)

The `cfl3d.blomax` file provides information about the Baldwin-Lomax turbulence model (**ivisc** = 2).

cfl3d.out15 (unit 15)

The `cfl3d.out15` file is a secondary output file. It provides more in depth information about the progression of a case. It is primarily used by the programmer for debugging purposes.

cfl3d.prout (unit 17)

The `cfl3d.prout` file contains the output as specified in the input file with the **nprint** parameter and "LT30 - Print Out Specifications" on page 41. Note that this file can be written for values at the grid points (**iptype** = 0) or cell-centers (**iptype** = 1). Keep in mind that this file can grow quite large rather quickly, so limit the regions specified for output. The `cfl3d.prout` file will also contain the control surface data if **ncs** > 0 (see Section 3.31 on page 42). (The sample below is from the flat plate test case discussed in Section 9.1.5 on page 173.) Note that `dn`, `Cf`, `Ch`, and `yplus` are only output when **iptype** = 0.

```
    turbulent flat plate (plate from j=17-65, prior to 17 is symmetry)
      Mach     alpha      beta      ReUe   Tinf,dR      time
   0.20000   0.00000   0.00000 0.600E+07 460.00000   0.00000


 BLOCK  1     IDIM,JDIM,KDIM=    2   65   97

   I  J  K     X        Y        Z      U/Uinf     V/Vinf     W/Winf     P/Pinf
T/Tinf      MACH          cp    tur. vis.
    1   1   1 -.3333E+00 0.0000E+00 0.0000E+00 0.9987E+00 0.0000E+00 -.7200E-07
0.1000E+01 0.1000E+01 0.1997E+00 0.5992E-02 0.2054E-02
    1   2   1 -.3125E+00 0.0000E+00 0.0000E+00 0.9989E+00 0.0000E+00 0.1323E-21
0.1000E+01 0.1000E+01 0.1998E+00 0.6012E-02 0.1379E-02
                .
                .
                .
    1  64   1 0.9792E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
0.9999E+00 0.1008E+01 0.0000E+00 -.3123E-02 0.1966E-09
    1  65   1 0.1000E+01 0.0000E+00 0.0000E+00 0.3675E-02 0.0000E+00 0.5962E-08
0.9999E+00 0.1008E+01 0.7320E-03 -.3607E-02 0.1963E-09
```

```
   I  J  K     X          Y          Z          dn       P/Pinf     T/Tinf
Cf           Ch         yplus
    1   2   1 -0.31250E+00  0.00000E+00  0.00000E+00  0.10000E-05  0.10002E+01
0.10000E+01  0.00000E+00  0.00000E+00  0.00000E+00
    1   3   1 -0.29167E+00  0.00000E+00  0.00000E+00  0.10000E-05  0.10002E+01
0.10000E+01  0.00000E+00  0.00000E+00  0.00000E+00
                  .
                  .
                  .
    1  63   1  0.95833E+00  0.00000E+00  0.00000E+00  0.10000E-05  0.99994E+00
0.10080E+01  0.24677E-02  0.33657E-02  0.20858E+00
    1  64   1  0.97916E+00  0.00000E+00  0.00000E+00  0.10000E-05  0.99991E+00
0.10080E+01  0.24660E-02  0.24780E-02  0.20851E+00


 BLOCK  1     IDIM,JDIM,KDIM=    2   65   97

   I  J  K    X         Y         Z       U/Uinf    V/Vinf    W/Winf    P/Pinf
T/Tinf      MACH          cp    tur. vis.
    1  49   1 0.6667E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00 0.0000E+00
0.1000E+01 0.1008E+01 0.0000E+00 0.1029E-02 0.2188E-09
    1  49   2 0.6667E+00 0.0000E+00 0.1000E-05 0.8330E-02 0.0000E+00 -.1105E-06
0.1000E+01 0.1008E+01 0.1659E-02 0.1029E-02 0.2112E-07
                  .
                  .
                  .
    1  49  96 0.6667E+00 0.0000E+00 0.9594E+00 0.9987E+00 0.0000E+00 0.2794E-03
0.1000E+01 0.1000E+01 0.1997E+00 0.2587E-02 0.7274E-02
    1  49  97 0.6667E+00 0.0000E+00 0.9837E+00 0.9987E+00 0.0000E+00 0.2672E-03
0.1000E+01 0.1000E+01 0.1997E+00 0.2609E-02 0.7274E-02

   I  J  K     X          Y          Z          dn       P/Pinf     T/Tinf
Cf           Ch         yplus
    1  49   1  0.66667E+00  0.00000E+00  0.00000E+00  0.10000E-05  0.10000E+01
0.10080E+01  0.25533E-02  0.16414E-02  0.21218E+00
    1  49  97  0.66667E+00  0.00000E+00  0.98367E+00  0.24255E-01  0.10001E+01
0.10000E+01  0.23455E-13  0.60677E-09  0.15760E-01
```

A sample of the type of control surface information that can be expected when **ncs** > 0 is shown here (this is no longer for the flat plate test case):

```
 Control Surface  1
 ==================

 Grid  1 (Block  1)  i = 25, 25  j =  1, 33  k =  1, 25  iwall =   0  Normal =   1

 x-area =   0.2687E+02 y-area =   0.1842E+00 z-area =  -0.1842E+00 total-area =
0.2687E+02

 Mass averaged properties
 P/Pinf      =   0.9446E+00         Pt/Pinf =   0.1022E+01
 T/Tinf      =   0.9840E+00         Tt/Tinf =   0.1007E+01
 Mach number =   0.3367E+00

 Mass flow / (rhoinf*vinf) =    0.42300E+02

                  x-force     y-force      z-force   resultant-force lift-force
   drag-force
Pressure force   -0.52122E+02 -0.23196E+01  0.23205E+01  0.52226E+02 0.23205E+01
-0.52122E+02
Thrust force     0.12872E+03  0.23886E+02  -0.23848E+02 0.13307E+03 -0.23848E+02
0.12872E+03
Total force      0.76598E+02  0.21566E+02  -0.21527E+02 0.82437E+02 -0.21527E+02
0.76598E+02
```

This information is printed at the end of the `cfl3d.prout` file.

cfl3d.out20 (unit 20)

The `cfl3d.out20` file is an auxiliary output file providing information when **iunst** > 0. It provides the unsteady pressures from forced oscillations. This file is primarily used by the programmer for debugging purposes. To output this file, hard-wire `irite = 1` in subroutine `resp` in `rhs.f`.

cfl3d.subit_res (unit 23)

The `cfl3d.subit_res` file is similar to the `cfl3d.res` file, except that it outputs residuals for all sub-iterations when time-accurate computations with sub-iterations are performed. This file is hard-wired with this name and is *not* in the file list at the top of the input file. However, whereas unit 12 outputs only $R(\mathbf{q}^m)$ for the equation for density (in Equation (B-12) or Equation (B-22) of Appendix B), unit 23 outputs the entire right-hand side of Equation (B-12) or Equation (B-22) for the equation for density. Therefore, the residuals in the two files do not match.

cfl3d.subit_turres (unit 24)

The `cfl3d.subit_turres` file is similar to the `cfl3d.turres` file, except that it outputs field-equation turbulence model residuals for all sub-iterations when time-accurate cases with sub-iterations are run. This file is hard-wired with this name and is *not* in the file list at the top of the input file.

cfl3d.dynamic_patch (unit 25)

The `cfl3d.dynamic_patch` file is output when using the moving grid option with dynamic patching. This file is hard-wired with this name and is *not* in the file list at the top of the input file.

5.2.3 PLOT3D Function Files

If desired, CFL3D can output PLOT3D function files (for 3-d cases) with surface grid, y^+, eddy viscosity, and normal spacing at the first grid point above all viscous walls. These files are:

surf_xyz.fmt (unit 28)

surf_y+.fmt (unit 29)

surf_vist.fmt (unit 30)

surf_dn.fmt (unit 31)

For 2-d cases, a single file is output:

surf_y+_2d.fmt (unit 28)

This file is in column format, containing all of the above information, with headers between different segments. By default, this option to write these files is currently turned *off*. To activate this option, the user must change `ifunc` from 0 to 1 in subroutine `yplusout` in `lbcx.f`.

CHAPTER 6

Boundary Conditions

The equations of motion require boundary conditions on all sides of the domain in which the solution is to be obtained, as well as on all surfaces of any objects which lie within the domain. For CFD methods, this implies that boundary conditions must be applied at each face of each computational block. Here, the term boundary conditions is used somewhat loosely as it refers to both physical conditions (e.g. solid wall, symmetry plane, etc.) and interfaces between adjacent or overlapped blocks.

In CFL3D, the boundary condition data are located in the arrays `qi0`(**jdim**,**kdim**,5,4), `qj0`(**kdim**,**idim**-1,5,4), and `qk0`(**jdim**,**idim**-1,5,4) for the i, j, and k directions, respectively. The third index in the arrays indicates the position for the five flow-field primitive variables (Equation (E-3)). The fourth dimension indicates the storage location for the boundary value positions; indices 1 and 2 are the positions for the boundary data at the left face and indices 3 and 4 are the positions for the boundary data at the right face. "Left face" here refers to the $i = 1$, $j = 1$, and $k = 1$ faces. "Right face" refers to the $i =$ **idim**, $j =$ **jdim**, and $k =$ **kdim** faces.

Note that the locations defining the boundary condition regions in "LT14 - I0 Boundary Condition Specification" through "LT19 - KDIM Boundary Condition Specification" are *grid points*, but the actual boundary conditions are applied at *cell-face centers*.

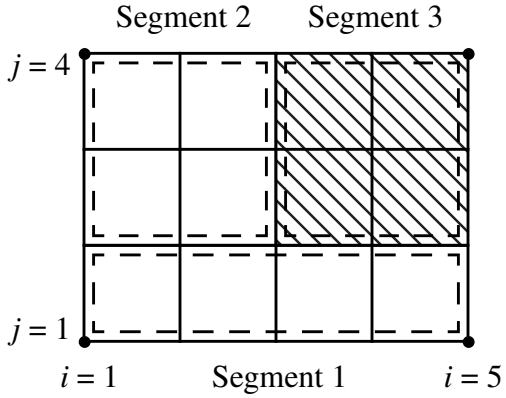


Figure 6-1. Boundary condition segments.

For example, say that Figure 6-1 represents the surface of a wing. At the unshaded cells, it is desired to apply a heated wall boundary condition, while at the shaded cells it is

desired to apply an adiabatic wall boundary condition. One way to accomplish this objective would be to divide the K0 boundary condition into three segments, as indicated by the dashed lines in the figure. The CFL3D input file would look like this:

```
Line Type
 18    K0: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND      NDATA
             1          1      2004         1         5         1         2          2
         twtype         Cq
           1.6        0.0
             1          2      2004         1         3         2         4          2
         twtype         Cq
           1.6        0.0
             1          3      2004         3         5         2         4          2
         twtype         Cq
           0.0        0.0
```

Note that, for example, the *grid points* $i = 1$ to 5, $j = 1$ to 2 define the boundary of the *cells* at which the boundary condition type for segment 1 is applied.

Two types of boundary condition representations are employed in CFL3D, namely, cell-center and cell-face. For cell-center type boundary conditions, the flow-field variables are specified at "ghost" points corresponding to two cell-center locations analytically extended outside the grid as illustrated in Figure 6-2. For example, qi0(j,k,**l**,1) contains the boundary data at the ghost point locations right next to the $i = 1$ cell-center data and qi0(j,k,**l**,2) contains the boundary data in the second set of ghost point locations at the left boundary. Likewise, qi0(j,k,**l**,3) contains the boundary data at the ghost point locations right next to the $i =$ **idim** cell-center data and qi0(j,k,**l**,4) contains the boundary data in the second set of ghost point locations at the right boundary. The same boundary data location definitions apply for the qj0 and qk0 arrays.

For cell-face type boundary conditions, the flow-field variables and their gradients are specified at the cell-face boundary as illustrated in Figure 6-3. For cell-face type boundary conditions, qi0(j,k,**l**,1) contains the flow-field variable data at the $i = 1$ grid cell-face and qi0(j,k,**l**,2) contains the flow-field variable gradients at the $i = 1$ grid cell-face. Likewise, qi0(j,k,**l**,3) contains the flow-field variable data at the $i =$ **idim** grid cell-face and qi0(j,k,**l**,4) contains the flow-field variable gradients at the $i =$ **idim** grid cell-face. The same boundary data location definitions apply for the qj0 and qk0 arrays.

Note that when running in 2-d mode (**i2d** = 1), it *does not matter* what boundary conditions are used on the $i = 1$ and $i =$ **idim** faces (provided that the boundary condition itself does not require more than one interior cell). **Bctype** is generally set to 1002 on these faces.

Boundary conditions for the eddy viscosity are stored in vi0, vj0, and vk0, while boundary conditions for the field equation turbulence models are stored in ti0, tj0, and tk0. Both of these are cell-center type, although only the first ghost points (1 and 3) are used. The field equation turbulence model boundary conditions are described in more detail in Section H.7 on page 296.

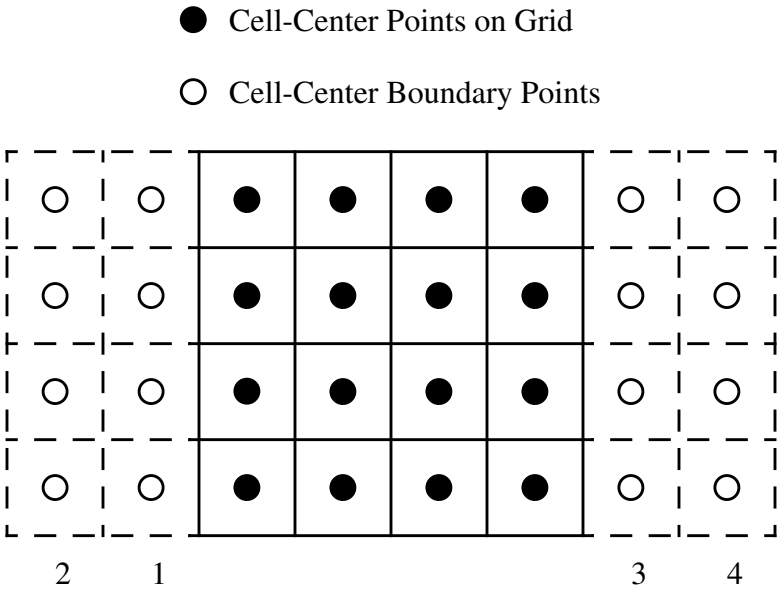


Figure 6-2. Cell-center type boundary locations.

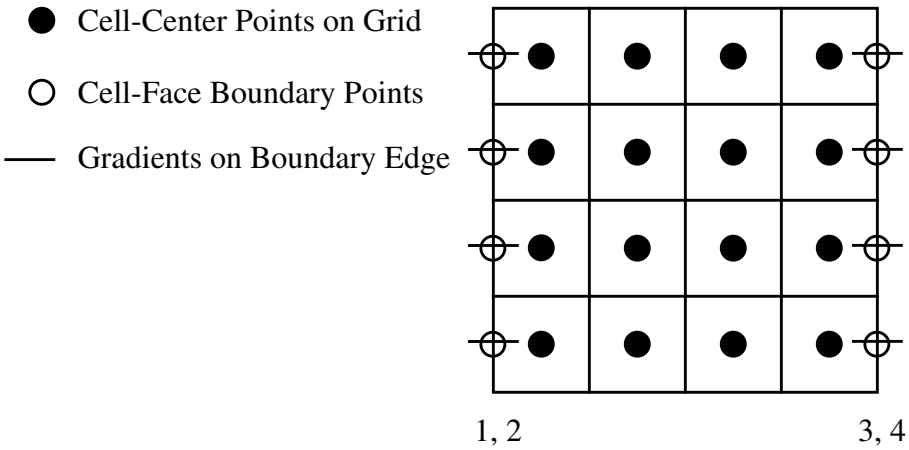


Figure 6-3. Cell-face type boundary conditions.

6.1 Physical Boundary Conditions

The 1000 series of boundary condition types are used for physical boundary conditions and are specified at any of a block's six faces in "LT14 - I0 Boundary Condition Specification" on page 32 through "LT19 - KDIM Boundary Condition Specification" on page 35. In each case, **ndata** = 0, since no additional information is required for implementation of the condition. The 1000 series boundary condition types currently supported are as follows:

bctype	boundary condition
1000	free stream
1001	general symmetry plane
1002	extrapolation
1003	inflow/outflow
1004	(no longer available, use 2004 instead)
1005	inviscid surface
1008	tunnel inflow
1011	singular axis – half-plane symmetry
1012	singular axis – full plane
1013	singular axis – partial plane

6.1.1 Free Stream

bctype 1000

The free stream boundary conditions are cell-center type boundary conditions. The five flow-field variables for both sets of ghost points are set equal to the initial values, which are:

$$\begin{aligned} \rho_{initial} &= 1.0 \\ u_{initial} &= M_\infty \cos\alpha\cos\beta \\ v_{initial} &= -M_\infty \sin\beta \\ w_{initial} &= M_\infty \sin\alpha\cos\beta \\ p_{initial} &= \rho_{initial}(a_{initial})^2/\gamma \end{aligned} \qquad \text{(6-1)}$$

where $a_{initial} = 1.0$.

6.1.2 General Symmetry Plane

bctype 1001

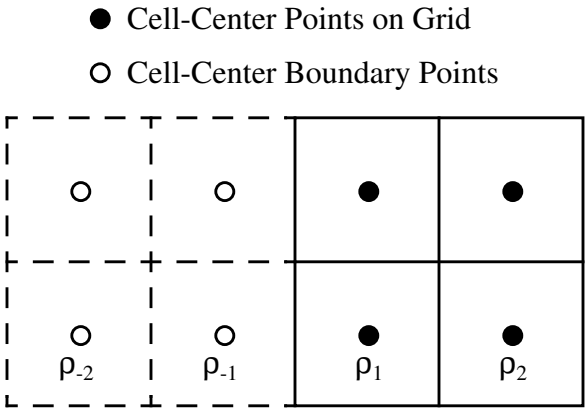


Figure 6-4. Symmetry boundary conditions (**bctype** 1001) for density.

The symmetry plane boundary conditions are cell-center type boundary conditions. As the name implies, symmetry is assumed across an axis. The ghost point density values are set equal to their "mirror image" counterparts. With the points defined as in Figure 6-4,

$$\begin{aligned} \rho_{-1} &= \rho_1 \\ \rho_{-2} &= \rho_2 \end{aligned} \tag{6-2}$$

(Note that ρ_{-1} = `qi0`(j,k,1,1) at the $i = 1$ face, for example.) The pressure values are assigned in the same way.

The velocity components at the ghost cells are obtained as follows. Consider ghost cells at an $i = 1$ face. Note that the normalized contravariant velocity $\overline{U}$ is normal to an i = constant face. Let $\overline{U}_1$ be the normalized contravariant velocity at cell center 1 in Figure 6-4. For an i symmetry plane, $\overline{U}$ must have opposite signs on each side of the plane. Thus

$$\begin{aligned} u_{-1} &= u_1 - 2\hat{\xi}_x \overline{U}_1 \\ v_{-1} &= v_1 - 2\hat{\xi}_y \overline{U}_1 \\ w_{-1} &= w_1 - 2\hat{\xi}_z \overline{U}_1 \end{aligned} \tag{6-3}$$

where $\hat{\xi}_x$, $\hat{\xi}_y$, and $\hat{\xi}_z$ are the metrics (unit normals) at the $i = 1$ face.

Since $(\hat{\xi}_x)^2 + (\hat{\xi}_y)^2 + (\hat{\xi}_z)^2 = 1$, $\overline{U}_{-1} = \overline{U}_1$. The velocity components at ghost cell center –2 are set in a similar manner using the data at cell center 2. (Note that this is now a "general" symmetry condition, applicable to any plane, not just the $x-z$ plane as in earlier versions of the code. Consequently, there is no longer a **bctype** 1006.)

6.1.3 Extrapolation

bctype 1002

The extrapolation boundary conditions are cell-center type boundary conditions. The ghost points are extrapolated from the computational domain. Based on the locations of ρ_1, ρ_{-1} and ρ_{-2} depicted in Figure 6-4, the extrapolated values would be

$$\begin{aligned} \rho_{-1} &= \rho_1 \\ \rho_{-2} &= \rho_1 \end{aligned} \tag{6-4}$$

The same zeroth order extrapolation is used for the boundary values of the other four flowfield variables.

6.1.4 Inflow/Outflow

bctype 1003

The inflow/outflow boundary conditions are cell-face type boundary conditions. In the far field, the velocity normal to the far boundary (pointing *out* of the grid) and the speed of sound are obtained from two locally 1-d Riemann invariants:

$$R \equiv \bar{u} \pm \frac{2a}{\gamma - 1} \tag{6-5}$$

where the boundary is considered a surface of constant ξ (in this example) with decreasing ξ corresponding to the interior of the domain and

$$\bar{u} = \overline{U} - \frac{\xi_t}{|\nabla\xi|} \tag{6-6}$$

$$\overline{U} = \frac{\xi_x}{|\nabla\xi|}u + \frac{\xi_y}{|\nabla\xi|}v + \frac{\xi_z}{|\nabla\xi|}w + \frac{\xi_t}{|\nabla\xi|} \tag{6-7}$$

R^- can be evaluated locally from conditions outside the computational domain and R^+ can be determined locally from inside the domain. The normal velocity and speed of sound are determined from

$$\bar{u}_{face} = \frac{1}{2}(R^+ + R^-)$$
$$a_{face} = \frac{\gamma - 1}{4}(R^+ - R^-) \quad \text{(6-8)}$$

The Cartesian velocities are determined by decomposing the normal and tangential velocity vectors:

$$u_{face} = u_{ref} + \frac{\xi_x}{|\nabla\xi|}(\bar{u}_{face} - \bar{u}_{ref})$$
$$v_{face} = v_{ref} + \frac{\xi_y}{|\nabla\xi|}(\bar{u}_{face} - \bar{u}_{ref}) \quad \text{(6-9)}$$
$$w_{face} = w_{ref} + \frac{\xi_z}{|\nabla\xi|}(\bar{u}_{face} - \bar{u}_{ref})$$

For inflow, $ref \Rightarrow \infty$. For outflow, ref represents the values from the cell inside the domain adjacent to the boundary.

The sign of the normal velocity $\bar{U}_{face} = \bar{u}_{face} + \xi_t/|\nabla\xi|$ determines whether the condition is at inflow ($\bar{U}_{face} < 0$) or outflow ($\bar{U}_{face} > 0$). The entropy p/ρ^γ is determined using the value from outside the domain for inflow and from inside the domain for outflow. The entropy and speed of sound are used to determine the density and pressure on the boundary:

$$\rho_{face} = \left[\frac{(a_{face})^2}{\gamma s_{face}}\right]^{\frac{1}{\gamma - 1}}$$
$$p_{face} = \frac{\rho_{face}(a_{face})^2}{\gamma} \quad \text{(6-10)}$$

Note that when **i2d** = –1, the 1003 boundary condition is augmented by the far-field point-vortex correction[38] (for 2-d, $x - z$ plane only).

6.1.5 Inviscid Surface

bctype 1005

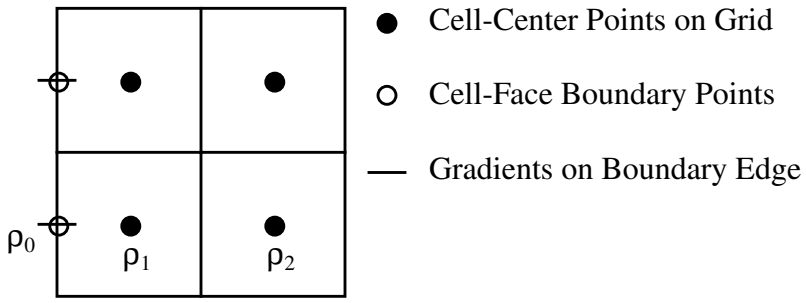


Figure 6-5. Inviscid surface boundary condition (**bctype** 1005) for density.

The inviscid surface boundary conditions are cell-face type boundary conditions. The cell-face boundary values for density are approximated with the density values of the nearest cell-center points on the grid. With the points defined as in as Figure 6-5,

$$\rho_0 = \rho_1 \tag{6-11}$$

The boundary values for pressure are obtained in the same manner.

The velocity component normal to the surface is set to zero in the following manner. Assume that the surface in Figure 6-5 is a k = constant surface, for which the metrics (unit normals) are $\hat{\zeta}_x$, $\hat{\zeta}_y$, and $\hat{\zeta}_z$. Then the normalized contravariant velocity (normal to the k direction) at cell center 1 is

$$\overline{W}_1 = u_1\hat{\zeta}_x + v_1\hat{\zeta}_y + w_1\hat{\zeta}_z + \hat{\zeta}_t \tag{6-12}$$

The velocity components at the wall are then calculated using

$$\begin{aligned} u_0 &= u_1 - \hat{\zeta}_x\overline{W}_1 \\ v_0 &= v_1 - \hat{\zeta}_y\overline{W}_1 \\ w_0 &= w_1 - \hat{\zeta}_z\overline{W}_1 \end{aligned} \tag{6-13}$$

Using $(\hat{\zeta}_x)^2 + (\hat{\zeta}_y)^2 + (\hat{\zeta}_z)^2 = 1$, it may be seen that $\overline{W}_0 = 0$ so that the normal velocity at the wall is zero.

The gradient values for all five flow-field variables needed at the cell-face are obtained using two-point differences. For example, for density,

$$\nabla\rho = 2(\rho_1 - \rho_0) \tag{6-14}$$

6.1.6 Constant Enthalpy and Entropy Inflow

bctype 1008

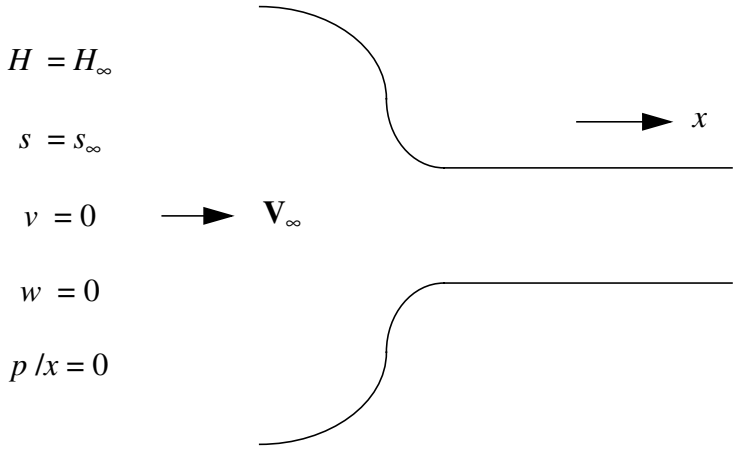


Figure 6-6. Constant enthalpy and entropy inflow boundary conditions (**bctype** 1008).

The constant enthalpy and entropy boundary condition (sometimes referred to as the wind tunnel inflow boundary condition) is a cell-center type condition. A grid set up to use the tunnel inflow condition should always have the x coordinate running along the length of the tunnel as shown in Figure 6-6. It is assumed that the entropy and the enthalpy are at the free-stream conditions. The v and w components of velocity are set to zero and the pressure gradient at the boundary is also zero. The density at the boundary is obtained from

$$s_\infty = \frac{p}{\rho^\gamma} \tag{6-15}$$

The u component of velocity is obtained from

$$H_\infty = \frac{a^2}{\gamma - 1} + \frac{u^2}{2} \tag{6-16}$$

The zero pressure gradient condition is used to set the pressure in the ghost cell equal to the nearest interior value.

For running internal (wind tunnel type) flows, **bctype** 2003 is usually preferred as the inflow boundary condition over 1008, although both may work. For the corresponding wind tunnel outflow boundary condition, **bctype** 2002 should generally be used.

6.1.7 Singular Axis Using Half-plane Symmetry

bctype 1011

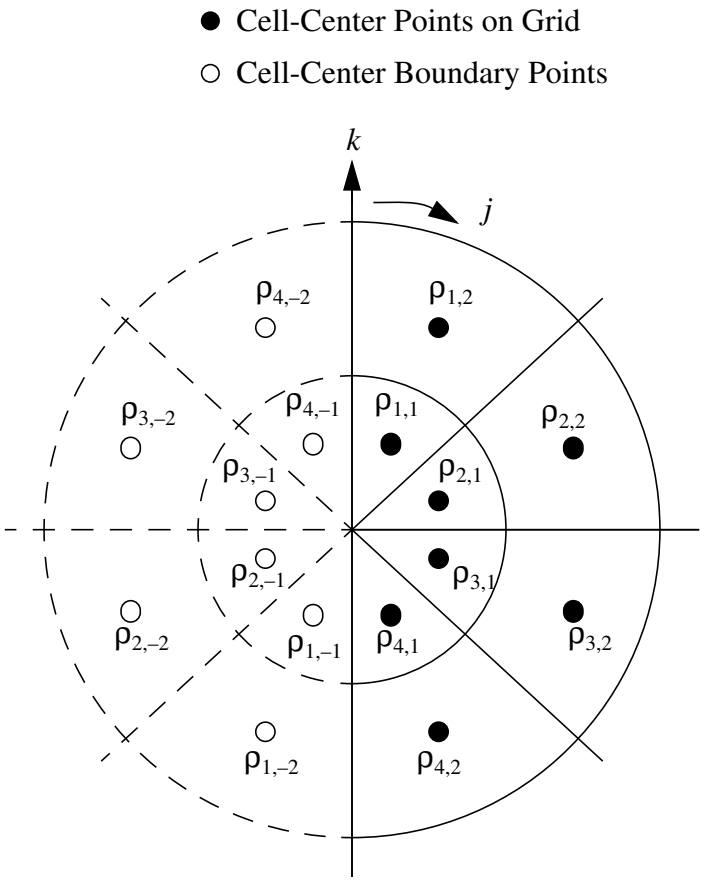


Figure 6-7. Singular axis with half-plane symmetry boundary condition (**bctype** 1011).

The singular axis using half-plane symmetry boundary conditions are cell-center type boundary conditions. As an example of this boundary condition, assume the singular axis occurs at $k = 1$. If **jdim** = 5, an i = constant plane might look like the one drawn in Figure 6-7. In the figure, the subscripts for ρ are j and then k (i.e. $\rho_{j,k}$). The boundary values for density are assigned as

$$\begin{aligned} \rho_{1,-1} &= \rho_{4,1} & \rho_{1,-2} &= \rho_{4,2} \\ \rho_{2,-1} &= \rho_{3,1} & \rho_{2,-2} &= \rho_{3,2} \\ \rho_{3,-1} &= \rho_{2,1} & \rho_{3,-2} &= \rho_{2,2} \\ \rho_{4,-1} &= \rho_{1,1} & \rho_{4,-2} &= \rho_{1,2} \end{aligned} \qquad (6\text{-}17)$$

The other four flow-field variables are assigned in a similar fashion; however, the normalized contravariant velocities and metrics are used to determine the correct signs for the

velocity components in a manner similar to boundary condition type 1001. First note that by the assumption of half-plane symmetry, the direction cosines on $j = 1$ are the same as $j =$ **jdim** apart from the sign; let these metrics be denoted $\hat{\eta}_x$, $\hat{\eta}_y$, and $\hat{\eta}_z$. Then, for example,

$$\overline{V}_{4,1} = u_{4,1}\hat{\eta}_x + v_{4,1}\hat{\eta}_y + w_{4,1}\hat{\eta}_z + \hat{\eta}_t \tag{6-18}$$

$$\begin{aligned} u_{1,-1} &= u_{4,1} - 2\hat{\eta}_x \overline{V}_{4,1} \\ v_{1,-1} &= v_{4,1} - 2\hat{\eta}_y \overline{V}_{4,1} \\ w_{1,-1} &= w_{4,1} - 2\hat{\eta}_z \overline{V}_{4,1} \end{aligned} \tag{6-19}$$

6.1.8 Singular Axis Using Full Plane, Flux Specified

bctype 1012

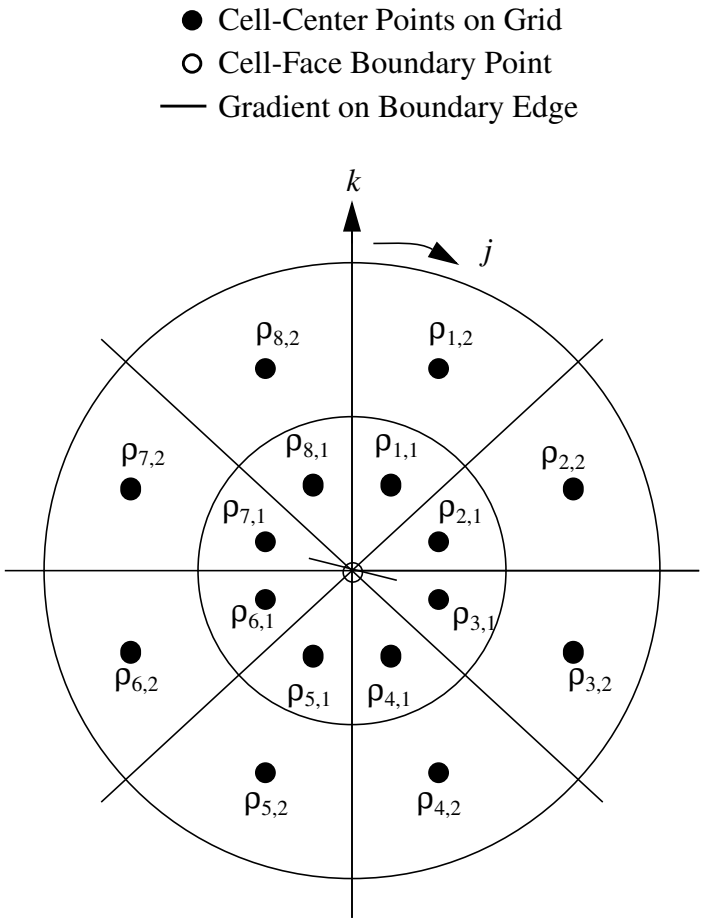


Figure 6-8. Singular axis using full plane boundary condition (**bctype** 1012).

The singular axis using full plane boundary conditions are cell-face type boundary conditions. As an example of this boundary condition, assume the singular axis occurs at $k = 1$. If **jdim** = 9, an i = constant plane might look like the one drawn in Figure 6-8. In the figure, the subscripts for ρ are j and then k (i.e. $\rho_{j,k}$).

For the cell-face boundary point value of density (ρ_0), an average value of all the values at $k = 1$ is obtained, i.e.

$$\rho_{j,0} = \frac{\sum_{j=1}^{\text{jdim}-1} \rho_{j,1}}{\text{jdim} - 1} \tag{6-20}$$

The flux value is obtained with a two point extrapolation using ρ_1 and ρ_0:

$$(\nabla\rho)_j = 2(\rho_{j,1} - \rho_{j,0}) \tag{6-21}$$

The boundary values for all five flow-field variables are assigned as described for density.

A known problem exists when using this boundary condition with the Baldwin-Lomax turbulence model. In such cases, the code employs the "wall" Baldwin-Lomax equation option rather than the "wake" Baldwin-Lomax equation option at the 1012 boundary.

6.1.9 Singular Axis Using Extrapolation (Partial Plane)

bctype 1013

The singular axis using extrapolation boundary conditions are cell-center type boundary conditions. The ghost points are extrapolated from the computational domain. This boundary condition is used with singular axes for which neither boundary condition type 1011 or 1012 is appropriate (quarter planes, for instance). For example, the density boundary values would be approximated as

$$\begin{aligned} \rho_{-1} &= \rho_1 \\ \rho_{-2} &= \rho_1 \end{aligned} \tag{6-22}$$

The same first order extrapolation is used for the boundary values of the other flow-field variables.

6.1.10 A Word About Singular Metrics

Version 5.0 will automatically detect collapsed grid lines (e.g. singular metrics, cell faces with zero area) on block boundaries. (Collapsed grid lines in the "interior" of a block

are not allowed.) The detection of singular metrics is no longer keyed to the specification of either **bctype** 1011, 1012, or 1013. While these boundary conditions are still applicable, the new singular metric detection allows any other standard boundary condition that does not rely on the grid metrics at the boundary to be used as well. Note that the following boundary conditions rely on the grid metrics at the boundary and so should not be used on a singular face/face segment: 1001 (symmetry), 1005 (inviscid), 1003 (inflow/outflow), 2003 (inflow with specified total conditions) and 2006 (radial equilibrium).

When CFL3D detects singular metrics on a particular block, a message is written to the main output file (unit 11) indicating the location. It is always a good idea to verify that any singular block faces/face segments are in fact correctly detected. Metrics are treated as singular if the total area of a face/face segment is less than a parameter `atol` (set near the top of subroutine `metric`, in module `lbcx.f`). In certain cases, `atol` may need to be changed (increased in magnitude if faces that are known to be non-singular are flagged as singular and decreased if faces that are known to be singular are not flagged as such, in which case the code will give floating point overflow in subroutine `metric`).

6.2 Physical Boundary Conditions with Auxiliary Data

For the "2000" series of boundary conditions, auxiliary data is required. Therefore, **ndata** $\neq 0$. The following sections describe the boundary condition types for constant input data (**ndata** > 0). For these conditions, the information in "LT14 - I0 Boundary Condition Specification" on page 32 through "LT19 - KDIM Boundary Condition Specification" on page 35 is immediately followed by a header line, then a single data line with the **ndata** values appropriate for the particular boundary condition. Section 6.2.8 describes how to use the same boundary condition types with variable input data.

Input values for **bctype** (i.e. boundary condition types currently supported) as follows:

bctype	boundary condition
2002	specified pressure ratio
2003	inflow with specified total conditions
2004	no-slip wall
2005	periodic in space
2006	set pressure to satisfy the radial equilibrium equation
2007	set all primitive variables
2102	pressure ratio specified as a sinusoidal function of time

6.2.1 Specified Pressure Ratio

bctype 2002

The specified pressure ratio boundary condition, generally used as the outflow boundary condition for internal flows, is a cell-center type boundary condition. A single pressure ratio, $\tilde{p}/\tilde{p}_\infty$, is specified on input. The parameter **ndata** must be set to 1 for the input pressure ratio value to be read. This pressure ratio is used to set both cell-center pressure boundary values (p_{-1} and p_{-2}). Extrapolation from inside the computational domain is used to set the boundary values for ρ, u, v, and w. See "Extrapolation" on page 84. An example of the input lines is:

```
p/pinf
 0.910
```

When using **bctype** 2002 as the outflow condition for internal (wind tunnel type) flows, the tunnel Mach number and/or mass flow should be monitored and **p/pinf** adjusted accordingly to obtain the correct conditions. This is usually an iterative process.

6.2.2 Inflow With Specified Total Conditions

bctype 2003

The inflow with specified total conditions boundary condition (sometimes referred to as an "engine inflow" boundary condition because it is often used to specify the conditions at an inflow face where an engine exhaust is located) is a cell-center type boundary condition. The following five pieces of information are provided on input (**ndata** = 5): an estimated inflow Mach number (M_e), the total pressure ratio ($\tilde{p}_t/\tilde{p}_\infty$), the total temperature ratio ($\tilde{T}_t/\tilde{T}_\infty$), and the flow directions (α and β) in degrees. These values are used as the external state in a 1-d characteristic boundary condition. An example of the input lines is:

```
  Mach        Pt/Pinf  Tt/Tinf   Alphae     Betae
0.300           4.000  1.17555      0.0       0.0
```

Boundary condition type 2003 is very similar to boundary condition type 1003 in that either interior or exterior values are chosen, depending on the sign of the characteristics at the boundary. One difference between them is that, while 1003 uses far-field reference-state levels for the exterior values, 2003 uses the total temperature, pressure, Mach number, and flow angle that are input to determine the reference exterior values. Another difference is the way the density and pressure boundary condition values are determined. Boundary condition type 1003 calculates these values as shown in Equation (6-10). Boundary condition type 2003 first determines a Mach number at the boundary using the velocities and speed of sound at the boundary (which were calculated through the characteristic method):

$$M^2 = \frac{u^2 + v^2 + w^2}{a^2} \tag{6-23}$$

It should be noted that the inflow Mach number may end up slightly different from the input M_e for a converged solution. The pressure and density boundary conditions are then determined with

$$p = \frac{\tilde{p}_t / \tilde{p}_\infty}{\gamma\left(1 + \frac{\gamma - 1}{2} M^2\right)^{\gamma/(\gamma-1)}} \tag{6-24}$$

$$\rho = \frac{\gamma p}{a^2}$$

The boundary condition velocity components are determined the same way as for boundary condition type 1003.

6.2.3 Viscous Surface

bctype 2004

The viscous surface boundary conditions are cell-face type boundary conditions. The no-slip condition ($\mathbf{V} = 0$) is applied at the surface. Two pieces of auxiliary information are supplied on input (**ndata** = 2): the wall temperature ($\tilde{T}_w / \tilde{T}_\infty$) and the mass flow ($C_q$), where $C_q = (\rho u_{normal}) / (\rho u)_\infty$. ($C_q$ is zero if there is no flow through the wall.) An example of the input lines is:

```
    Twtype            Cq
      0.95         -0.05
```

where

Twtype > 0 sets $\tilde{T}_w / \tilde{T}_\infty$ = **Twtype**
Twtype = 0 sets adiabatic wall
Twtype < 0 sets $\tilde{T}_w$ at the stagnation temperature
Cq < 0 results in mass flow in the direction of "decreasing" computational coordinate (i.e. $C_q < 0$ results in suction on minimum wall, blowing on maximum wall)
Cq > 0 results in mass flow in the direction of "increasing" computational coordinate (i.e. $C_q > 0$ results in blowing on minimum wall, suction on maximum wall)
Cq = 0 results in no flow through the wall (same as old **bctype** 1004)

Note that for a dynamic grid, **bctype** 2004 gives no-slip relative to the moving wall. To set the wall velocity to zero relative to a non-moving reference frame when the mesh is moving, use –2004 instead of 2004; this option makes sense only if the grid motion is tangential to the surface.

Also note that boundary condition type 2004 supersedes boundary condition types 1004 *and* 2004 in previous versions of CFL3D. The main reason for the replacement is that 1004 relied on global parameters **isnd** and **c2spe** to determine whether adiabatic wall or constant-temperature wall conditions were invoked. As a consequence, all 1004 segments had to be the same. Now, with 2004 (along with its additional data field **Twtype**), every no-slip wall segment can be set with different wall temperature conditions, if desired. Boundary condition type 2004 also allows for mass flow through the wall (suction or blowing) through the second additional data field **Cq**.

The no-slip wall boundary condition is implemented as follows:

- The nondimensional pressure on the body p_b is determined through linear extrapolation:

$$p_b = p_1 - (p_2 - p_1)/2 \tag{6-25}$$

  But if $p_b < 0$, then $p_b = p_1$. Here the indices 1 and 2 indicate the first and second cell-center values away from the wall, respectively.
- The nondimensional square of the speed of sound, variable `c2`, at the wall is next determined. (Note that `c2` $= (\tilde{a}_w/\tilde{a}_\infty)^2 = \tilde{T}_w/\tilde{T}_\infty$.)

  If **Twtype** > 0, `c2` = **Twtype**.

  If **Twtype** < 0, `c2` $= 1 + \dfrac{\gamma - 1}{2}(M_\infty)^2$.

  If **Twtype** = 0, `c2` $= \left(\dfrac{a_1}{a_\infty}\right)^2\left[1 + \dfrac{\gamma - 1}{2}(M_1)^2\right]$.
- The surface velocities are then determined as

$$\begin{aligned} u_w &= M_\infty C_q \frac{\xi_x}{|\nabla\xi|}\frac{(c2)}{\gamma p_b} + u_{mesh} \\ v_w &= M_\infty C_q \frac{\xi_y}{|\nabla\xi|}\frac{(c2)}{\gamma p_b} + v_{mesh} \\ w_w &= M_\infty C_q \frac{\xi_z}{|\nabla\xi|}\frac{(c2)}{\gamma p_b} + w_{mesh} \end{aligned} \tag{6-26}$$

where u_{mesh}, v_{mesh}, and w_{mesh} are the velocity components of the mesh (they equal 0 if the grid/body is not in motion).

- Since 2004 is a cell-face boundary condition type (i.e. the boundary conditions and their gradient are applied at the cell face rather than in ghost cells),

$$\rho_w = \frac{\gamma p_b}{c2}$$
$$p_w = p_b \qquad \textbf{(6-27)}$$

The gradients at the wall for all the primitive variables are determined via $\nabla\rho = 2\rho_1 - 2\rho_w$, etc.

Note that `smin` is computed from viscous walls *only*. If a wall is inviscid, then as far as `smin` is concerned, it is invisible. This is important to remember when viscous boundary conditions are turned on after running a case inviscidly for some time since `smin` may never have been computed!

6.2.4 Periodic (In Space)

bctype 2005

The periodic boundary conditions are cell-center type boundary conditions. Four pieces of additional information must be input (**ndata** = 4). The number of the grid with which the current grid is periodic and the rotation angle ($d\theta_x, d\theta_y, d\theta_z$) about one of the coordinate axes (x, y, z) are needed. Only *one* of the three angles can be used at a time and the other two angles *must* be identically zero. The angles should be determined from the right-hand rule. For example, if rotation is desired about the N axis (where N is either x, y or z, point the right-hand thumb in the direction of the +N axis. The fingers will curl in the direction of the positive angle. When setting the angle for a particular face (i.e. the $i = 1$ face), set the angle of rotation equal to the angle that the *periodic face* would have to move through to get to this face.

For a sample input, assume the current face on which the boundary condition is being set is $j = 1$. The the following input will cause the current face to be periodic with the **jdim** face in grid 2, where a rotation of +45 degrees about the x axis would map the **jdim** face of grid 2 onto the $j = 1$ face of the current grid:

```
ngridp          dthx      dthy      dthz
     2          45.0       0.0       0.0
```

At present, it is assumed that the current block and the block with which it is periodic match 1-1 at their corresponding faces after the rotation. Note that there is *not* a check for this! Also, the two blocks are assumed to be aligned similarly. That is i, j, and k on the first block must be defined *exactly* the same on the second block. For example, if the

k = **kdim** face of the current block has the periodic boundary condition applied to it, it is implicitly assumed that it is periodic with the $k = 1$ face of the specified second block *and* i and j run in the exact same directions on both blocks. Note that this means that two of the dimensions (**idim** and **jdim** in the example) must be the same on both blocks.

The periodic boundary condition also works for a grid with one cell (two grid planes) in the periodic direction if the grid is set to be periodic with itself. Note, however, that if a particular block is periodic with a *different* block, then neither block should be only one cell wide in the periodic direction.

In addition, the periodic boundary condition may be used for linear displacement, provided the rotation angles are set to zero. Alternatively, the 1-1 block connection can be used (linear displacement only!). The 1-1 internal check will flag geometric mismatches, which should be equal to the desired linear periodic displacement. This is a good check of the input which is not available with boundary condition type 2005.

Boundary condition type 2005 is a limited-use periodic (in space) boundary condition. For example, if one is solving for flow through a duct and it is known that the solution is periodic over 90 degrees (i.e. the solution is identical in each of the four quadrants of the duct), then a solution need only be obtained on a grid spanning 90 degrees, with periodic boundary conditions applied at the two edges.

For an illustration of how this boundary condition works, consider Figure 6-9. This example is for flow through a 90 degree wedge (the flow direction is in the third dimension, out of the page). The flow is periodic over 90 degrees, so periodic boundary conditions are applied at the $j = 0$ and $j =$ **jdim** faces.

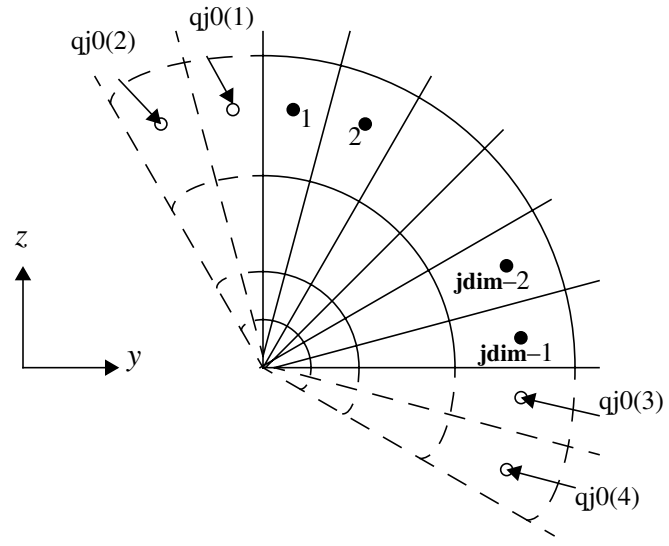


Figure 6-9. Periodic boundary condition (**bctype** 2005) example.

For density and pressure, boundary condition type 2005 sets:

```
qj0(1) = q(jdim-1)
qj0(2) = q(jdim-2)
qj0(3) = q(1)
qj0(4) = q(2)
```
(6-28)

The boundary conditions for the velocity components u, v, and w are assigned similarly, except that they are first rotated through the periodic angle, in this case $90°$. For example,

$$\begin{aligned} u_{1,\mathrm{BC}} &= \texttt{qj0}(k,i,2,1) = \texttt{q}(j,k,i,2) \quad \text{(unchanged)} \\ v_{1,\mathrm{BC}} &= \texttt{qj0}(k,i,3,1) = \texttt{q}(j,k,i,3)\cos\theta_p - \texttt{q}(j,k,i,4)\sin\theta_p \\ w_{1,\mathrm{BC}} &= \texttt{qj0}(k,i,4,1) = \texttt{q}(j,k,i,3)\sin\theta_p + \texttt{q}(j,k,i,4)\cos\theta_p \end{aligned} \tag{6-29}$$

For an example of how this boundary condition can be used for linear displacement, see the 2-d vibrating plate sample test case in Section 9.1.6 on page 177.

6.2.5 Radial Pressure Equilibrium

bctype 2006

Boundary condition type 2006 is a cell-center type boundary condition. Radial pressure equilibrium is used as a downstream boundary condition when it is desired to specify a pressure, but a large swirling component is present in the flow. Typically, this boundary condition would be used in turbomachinery applications in which a swirling flow is established by a rotor that is not corrected by the presence of stators or exit guide vanes.

Four additional pieces of information are needed for this boundary condition type (**ndata** = 4). The grid number (**ngridc**) of the grid *from* which the integration of pressure is to be continued is specified. If the integration in this grid is not continued from another grid, input 0. A value for $\tilde{p}/\tilde{p}_\infty$ is input. If **ngridc** = 0, this value will be the starting value for the integration. If **ngridc** $\neq$ 0, then the pressure value from the connecting point in grid **ngridc** is used as the starting value for the integration in the current block. The direction in which integration is to proceed is specified with the parameter **intdir**. It may have an absolute value of 1, 2, or 3 for integration in the i, j, or k directions, respectively. The sign of **intdir** indicates whether the integration proceeds in the increasing (positive) or decreasing (negative) coordinate direction. The **intdir** direction *must* be the radial direction. The fourth piece of information needed is the physical direction along which the radial axis lies, denoted with the parameter **axcoord**. It may have the values of +1, +2, or +3 for a radial axis aligned with the x, y, or z axes. The input lines for this boundary condition type would look like:

```
ngridc        P/Pinf    intdir   axcoord
```

```
0          0.9        +3         1
```

The radial equilibrium condition requires that the pressure satisfy

$$\frac{dp}{dr} = \rho\frac{(V_\theta)^2}{r} \tag{6-30}$$

where V_θ is the circumferential velocity component and r is the radius. The pressure ratio, $\tilde{p}/\tilde{p}_\infty$, is set at either the bottom or the top of the block face and then the radial equilibrium equation is integrated in either the increasing or decreasing radial direction to give the pressure at all other radii. The trapezoidal rule is used to perform the integration. The other flow-field variables (ρ, u, v, w) are extrapolated from inside the computational domain. See "Extrapolation" on page 84.

This boundary condition assumes that one coordinate direction on the block face is essentially radial and the other is essentially circumferential and that the integration is being carried out in the radial direction. Since there is no way to verify this in the code, care *must* be exercised when using this boundary condition to insure that this and the following restrictions are met. Continuation of the radial equilibrium condition through block boundaries is restricted to blocks that have the *same* orientation. For example, if the equilibrium condition is to be continued through a k boundary from an adjacent block, then i and j must run in the same direction in both blocks. This also implies that the k boundary must be $k = 1$ in one block and $k =$ **kdim** in the second (i.e. the boundary can not be $k = 1$ in both blocks). Also, the segment must run the entire length of the block face in the direction in which the integration is being carried out. For example, if the integration is being carried out in the k direction, then **ksta** must be set to 1 and **kend** must be set to **kdim**. This restriction applies only if the integration is being continued from another block.

Consider a case in which boundary condition 2006 is to be applied at an $i =$ **idim** face, with j = constant radial lines (i.e. lines of constant angle θ) and k = constant circumferential lines (i.e. lines of constant radius). Assume the block face lies in an x = constant plane, as shown in Figure 6-10.

Assuming integration in the $+k$ direction (**intdir** = 3), the pressure (`l` = 5) in the first plane of ghost cells is obtained from

$$\begin{aligned} \texttt{qi0(j,1,5,3)} &= \left(\frac{\tilde{p}}{\tilde{p}_\infty}\right)p_\infty \\ \texttt{qi0(j,k,5,3)} &= p_{k,j} = p_{k-1,j} + \frac{\rho_{k,j}[(V_\theta)^2]_{k,j}}{\bar{r}_{k,j}}\Delta r_{k,j} \qquad k = 2, \mathbf{kdim} - 1 \end{aligned} \tag{6-31}$$

where

$$[(V_\theta)^2]_{k,j} = \mathbf{V}_{k,j} \cdot \overline{\nabla\eta_j}$$
$$\overline{\nabla\eta_j} = \frac{1}{2}[\nabla\eta_{k,j+1/2} + \nabla\eta_{k,j-1/2}]_{\mathbf{idim}-1} \quad \text{(6-32)}$$

$\mathbf{V}_{k,j}$ is the velocity at the cell center (j, k, **idim** – 1) and $\bar{r}_{k,j}$ is the average radius in cell (k, j):

$$[r_{k-1/2,j} + r_{k+1/2,j}]/2 \quad \text{(6-33)}$$

(k, j) denotes a cell-center value. $(k \pm 1/2\,,\ j \pm 1/2)$ denotes a face-center value.

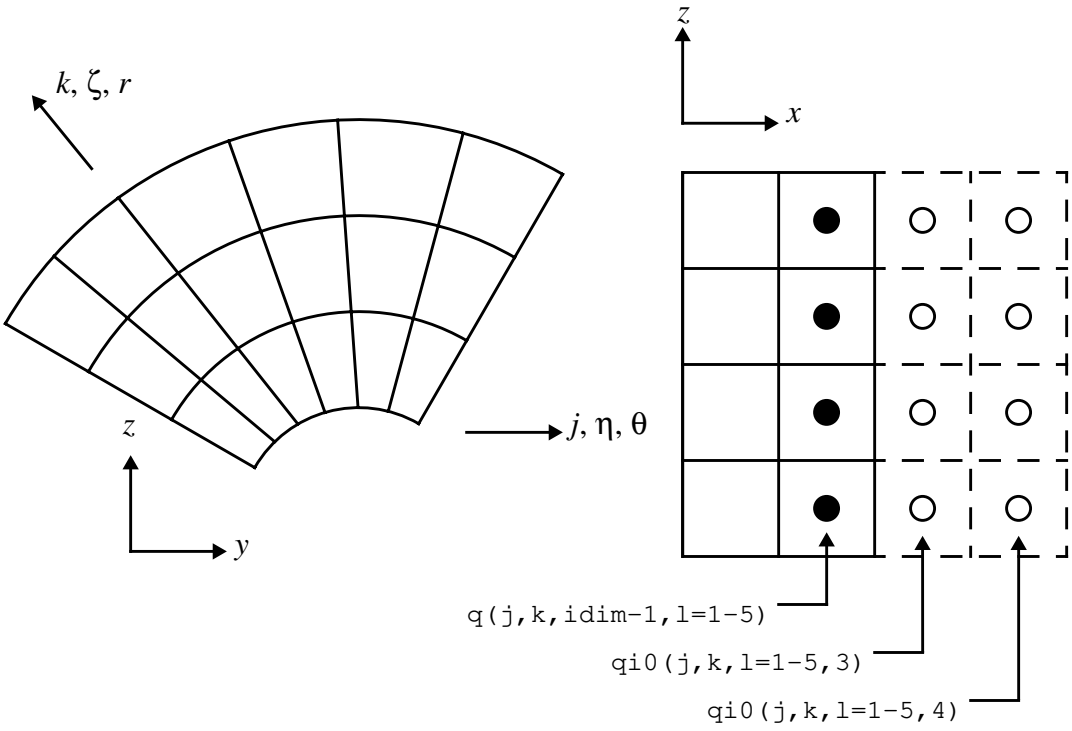


Figure 6-10. Radial pressure equilibrium boundary condition (**bctype** 2006) example.

The velocity and density in the first layer of ghost cells is determined by zeroth order extrapolation from the adjacent interior points. For the case shown in Figure 6-10, the density (`l` = 1) is obtained from

```
qi0(j,k,1,3) = q(j,k,idim-1,1)
```
(6-34)

All flow quantities in the second layer of ghost cells are obtained by zeroth order extrapolation from the first layer of ghost cells:

```
qi0(j,k,l=1-5,4) = qi0(j,k,l=1-5,3)
```
(6-35)

6.2.6 Specify All Primitive Variables

bctype 2007

Boundary condition type 2007 is a cell-center type boundary condition. It involves setting the boundary conditions with the five (**ndata** = 5) primitive variables, using standard CFL3D normalizations: $\tilde{\rho}/\tilde{\rho}_\infty$, $\tilde{u}/\tilde{a}_\infty$, $\tilde{v}/\tilde{a}_\infty$, $\tilde{w}/\tilde{a}_\infty$, and $\tilde{p}/[\tilde{\rho}_\infty(\tilde{a}_\infty)^2]$. An example of the input lines is:

```
rho/rho_inf      u/a_inf      v/a_inf      w/a_inf     p/(rho_inf*a_inf**2)
        1.0          0.3          0.0          0.0                     0.71
```

Note that the input pressure is *not* $\tilde{p}/\tilde{p}_\infty$ but rather $\tilde{p}/(\gamma\tilde{p}_\infty)$. Also note that the turbulence quantities are *not* currently allowed to be specified in the same way as the **q**'s. Instead, they are extrapolated from the interior when **bctype** 2007 is employed.

6.2.7 Specified Pressure Ratio As Sinusoidal Time Function

bctype 2102

The specified pressure ratio as a sinusoidal function of time boundary conditions are cell-center type boundary conditions. The pressure ratio, $\tilde{p}/\tilde{p}_\infty$, is specified as a sinusoidal function of time. The other flow-field variables (ρ, u, v, w) are extrapolated from inside the computational domain. Three pieces of additional information are specified on input (**ndata** = 4): the desired baseline (steady) pressure ratio ($\tilde{p}/\tilde{p}_\infty$), the amplitude of the pressure oscillation ($\Delta\tilde{p}/\tilde{p}_\infty$), the reduced frequency of the pressure oscillation (k_r), and the "grid equivalent" of the dimensional reference length used to define the reduced frequency (L_{ref}). The pressure will vary as

$$\gamma p = \frac{\tilde{p}}{\tilde{p}_\infty} + \frac{\Delta\tilde{p}}{\tilde{p}_\infty}\sin(2\pi k_r t) \qquad \textbf{(6-36)}$$

The reduced frequency is non-dimensionalized by

$$k_r = \frac{\tilde{f}\tilde{L}}{\tilde{a}_\infty} \qquad \textbf{(6-37)}$$

where $\tilde{f}$ is the frequency in cycles per second, $\tilde{L}$ is a characteristic length and $\tilde{a}_\infty$ is the free-stream speed of sound. (Note that this definition of reduced frequency differs from

the "standard" definition $k_r = \tilde{f}\tilde{L}/|\tilde{\mathbf{V}}|_\infty$.) An example of the lines in the input file for this boundary condition is:

```
p/pinf   deltap/pinf       rfreqp          lref
 0.910        0.001       175.93           1.0
```

6.2.8 Variable Data for the 2000 Series

To use this option, set **ndata** = –(the **ndata** used for the constant data value(s) as described above). Then, instead of the line containing the constant data value(s), substitute a line with the name (up to 60 characters) of a formatted file that has the appropriate array of data values. The data file will then be read with the following format:

```
read(iunit,*)                    header/title
read(iunit,*) mdim,ndim,np       mdim,ndim = cell-center dimensions of segment
                                 np = number of planes of ghost cell data
read(iunit,*) nvalues            number of data values; nvalues = abs(ndata)
read(iunit,*)((((bcdata(m,n,ip,l),m=1,mdim),n=1,ndim),ip=1,np),
                 l=1,nvalues)
```

The roles of `m,n` vary depending on which face the segment is located:

`m, n` $\rightarrow j, k$	on the i = 1 and the i = **idim** faces where `mdim` = **jend**–**jsta** and `ndim` = **kend**–**ksta**
`m, n` $\rightarrow k, i$	on the j = 1 and the j = **jdim** faces where `mdim` = **kend**–**ksta** and `ndim` = **iend**–**ista**
`m, n` $\rightarrow j, i$	on the k = 1 and the k = **kdim** faces where `mdim` = **jend**–**jsta** and `ndim` = **iend**–**ista**

Note that zeroes are *not* acceptable for `mdim` or `ndim`. Use the actual values of **jdim**-1, **kdim**-1, and/or **idim**-1 for the full face. Only boundary condition type 2007 can make use of two planes of data. For all other boundary conditions, set `np` = 1.

6.3 Block Interface Boundary Conditions

For all the types of block interface boundary conditions, set **bctype** = 0. If **bctype** ≠ 0, but block interface boundary conditions are set, the block interface boundary conditions will supersede. All block interface boundary conditions (one-to-one, patched, and overset) are cell-center type boundary conditions.

6.3.1 One-to-One Blocking

(Input Line Types Twenty-Four through Twenty-Six)

One-to-one (1-1) blocking, sometimes called C^0 continuous, means that the faces shared by two grids are exactly (to machine zero) the same. Several examples showing a variety of blocking strategies are discussed below. In the examples, the variable, **l**, represents the five flow-field variables. The sample inputs illustrate how the index ranges (**ista** to **iend**, **jsta** to **jend**, **ksta** to **kend**) in "LT25 - 1-1 Blocking Connections" on page 37 are assigned so that the correct communication between blocks is established. A good initial check to determine if the one-to-one blocking input is set up correctly is to compare the quantity of points in the range. For example, if two grids share a common portion of a j = constant face, the following must be true:

$$[\mathbf{jend} - \mathbf{jsta} + 1]_{\text{grid 1}} = [\mathbf{jend} - \mathbf{jsta} + 1]_{\text{grid 2}} \quad \textbf{(6-38)}$$

Keep in mind that j = constant faces can communicate with i = constant and/or k = constant faces as well (and vice versa), in which case the check will be

$$[\mathbf{jend} - \mathbf{jsta} + 1]_{\text{grid 1}} = [\mathbf{iend} - \mathbf{ista} + 1]_{\text{grid 2}} \quad \textbf{(6-39)}$$

or

$$[\mathbf{jend} - \mathbf{jsta} + 1]_{\text{grid 1}} = [\mathbf{kend} - \mathbf{ksta} + 1]_{\text{grid 2}} \quad \textbf{(6-40)}$$

respectively. Note that Equation (6-38) through Equation (6-40) are necessary for a one-to-one interface to be specified correctly, but they are not sufficient; the direction in which the indices are input must be correct as well. (See Example 3 on page 106.)

Example 1

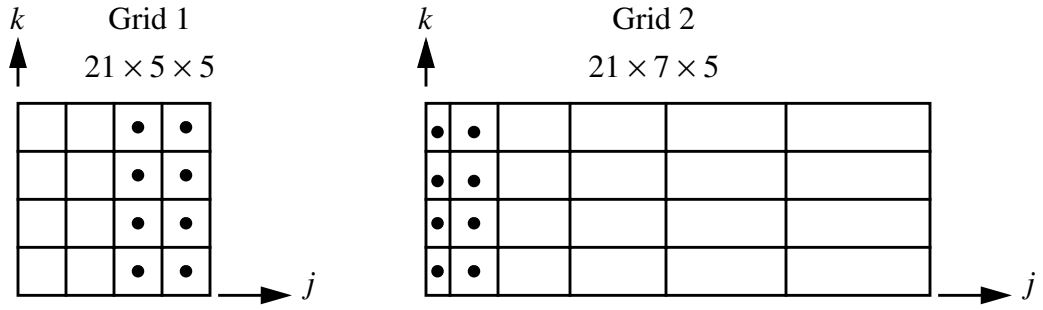


Figure 6-11. One-to-one blocking example 1.

As a simple illustration of 1-1 blocking, consider Figure 6-11. The figure shows an i = constant (not necessarily the *same* constant) face of two grids. Suppose communica-

tion is desired between the j = **jdim** face of grid 1 and the j = 1 face of grid 2. The pertinent input would look something like:

Line Type

```
 6       NGRID  NPLOT3D   NPRINT   NWREST     ICHK      I2D   NTSTEP      ITA
             2        1        0      100        0        0        1        1

 8        IDIM     JDIM     KDIM
            21        5        5
            21        7        5

13        GRID    NBCI0  NBCIDIM    NBCJ0  NBCJDIM    NBCK0  NBCKDIM    IOVRLP
             1        1        1        1        1        1        1         0
             2        1        1        1        1        1        1         0

16    J0: GRID  SEGMENT   BCTYPE     ISTA     IEND     KSTA     KEND     NDATA
             1        1     1003        1       21        1        5         0
             2        1        0        1       21        1        5         0
17  JDIM: GRID  SEGMENT   BCTYPE     ISTA     IEND     KSTA     KEND     NDATA
             1        1        0        1       21        1        5         0
             2        1     1003        1       21        1        5         0

         1-1 BLOCKING DATA:
24        NBLI
             1
25     NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
             1    1          1      5      1     21      5      5      1      3
26     NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
             1    2          1      1      1     21      1      5      1      3
```

Note that in the sample input, both grids have the same value for **idim**. This does not *have* to be true. One grid can share only a portion of a face with another grid.

The boundary conditions at the j = **jdim** face on any i plane, denoted i1, of grid 1 would be set as:

Grid 1	Grid 2
qj0(1,i1,**l**,3) =	q(1,1,i2,**l**)
qj0(2,i1,**l**,3) =	q(1,2,i2,**l**)
qj0(3,i1,**l**,3) =	q(1,3,i2,**l**)
qj0(4,i1,**l**,3) =	q(1,4,i2,**l**)
qj0(1,i1,**l**,4) =	q(2,1,i2,**l**)
qj0(2,i1,**l**,4) =	q(2,2,i2,**l**)
qj0(3,i1,**l**,4) =	q(2,3,i2,**l**)
qj0(4,i1,**l**,4) =	q(2,4,i2,**l**)

The boundary conditions at the $j = 1$ face on any i plane, denoted i2, of grid 2 would be set as:

Grid 2	Grid 1
qj0(1,i2,**l**,1) =	q(4,1,i1,**l**)
qj0(2,i2,**l**,1) =	q(4,2,i1,**l**)
qj0(3,i2,**l**,1) =	q(4,3,i1,**l**)
qj0(4,i2,**l**,1) =	q(4,4,i1,**l**)
qj0(1,i2,**l**,2) =	q(3,1,i1,**l**)
qj0(2,i2,**l**,2) =	q(3,2,i1,**l**)
qj0(3,i2,**l**,2)=	q(3,3,i1,**l**)
qj0(4,i2,**l**,2) =	q(3,4,i1,**l**)

Example 2

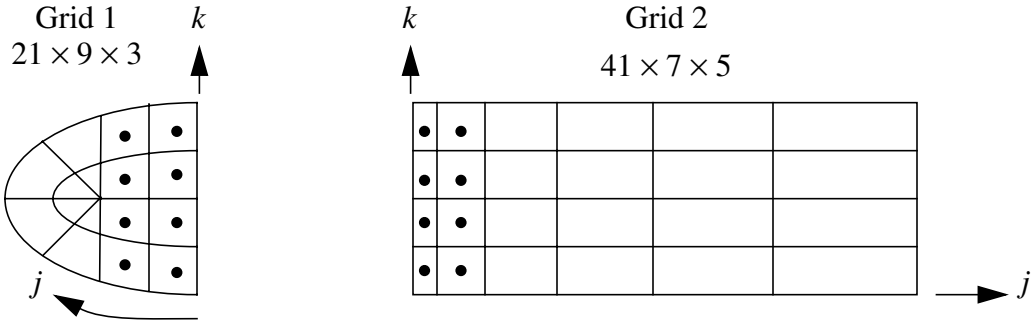


Figure 6-12. One-to-one blocking example 2.

A slightly more complicated example of 1-1 blocking is shown in Figure 6-12. Again, the figure shows an i = constant face (not necessarily the *same* constant) of two grids. In grid 1, j is now a circumferential direction. Communication is desired between the $j = 1$ *and* the $j =$ **jdim** faces of grid 1 and the $j = 1$ face of grid 2. (In this example, it is assumed that the $k = 1$ boundary of grid 1 is a solid wall flat plate, so there is no 1-1 connectivity there.) The pertinent input would look something like:

Line Type

```
 6        NGRID   NPLOT3D   NPRINT   NWREST     ICHK      I2D   NTSTEP      ITA
                2         1        0      100        0        0        1        1

 8         IDIM      JDIM      KDIM
           21         9         3
           41         7         5

13         GRID     NBCI0  NBCIDIM     NBCJ0  NBCJDIM     NBCK0  NBCKDIM    IOVRLP
           1         1        1         1        1         1        1         0
           2         1        1         2        1         1        1         0

16    J0: GRID   SEGMENT   BCTYPE     ISTA     IEND     KSTA     KEND     NDATA
           1         1        0        1       21        1        3         0
           2         1        0        1       21        1        5         0
           2         2     1001       21       41        1        5         0
17  JDIM: GRID   SEGMENT   BCTYPE     ISTA     IEND     KSTA     KEND     NDATA
```

```
          1        1        0        1       21        1        3        0
          2        1     1003        1       41        1        5        0

      1-1 BLOCKING DATA:
24     NBLI
          2
25   NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    1          1      1      1     21      1      3      1      3
          2    1          1      9      1     21      9      3      1      3
26   NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    2          1      1      3     21      1      1      1      3
          2    2          1      1      3     21      1      5      1      3
```

Note how k ranges on grid 2 to coincide with the appropriate k points on the $j = 1$ and $j =$ **jdim** faces of grid 1. This sample input also shows how grid 1 can share only a portion of the $j = 1$ face of grid 2 in the i direction.

The boundary conditions at the $j = 1$ face on any i plane, denoted i1, of grid 1 would be set as:

Grid 1	Grid 2
qj0(1,i1,**l**,1) =	q(1,2,i2,**l**)
qj0(2,i1,**l**,1) =	q(1,1,i2,**l**)
qj0(1,i1,**l**,2) =	q(2,2,i2,**l**)
qj0(2,i1,**l**,2) =	q(2,1,i2,**l**)

The boundary conditions at the $j =$ **jdim** face for the i1 plane of grid 1 would be set as:

Grid 1	Grid 2
qj0(1,i1,**l**,3) =	q(1,3,i2,**l**)
qj0(2,i1,**l**,3) =	q(1,4,i2,**l**)
qj0(1,i1,**l**,4) =	q(2,3,i2,**l**)
qj0(2,i1,**l**,4) =	q(2,4,i2,**l**)

The boundary conditions at the $j = 1$ face on any i plane, denoted i2, of grid 2 would be set as:

Grid 2	Grid 1
qj0(1,i2,**l**,1) =	q(1,2,i1,**l**)
qj0(2,i2,**l**,1) =	q(1,1,i1,**l**)
qj0(3,i2,**l**,1) =	q(8,1,i1,**l**)
qj0(4,i2,**l**,1) =	q(8,2,i1,**l**)
qj0(1,i2,**l**,2) =	q(2,2,i1,**l**)
qj0(2,i2,**l**,2) =	q(2,1,i1,**l**)
qj0(3,i2,**l**,2) =	q(7,1,i1,**l**)
qj0(4,i2,**l**,2) =	q(7,2,i1,**l**)

Example 3

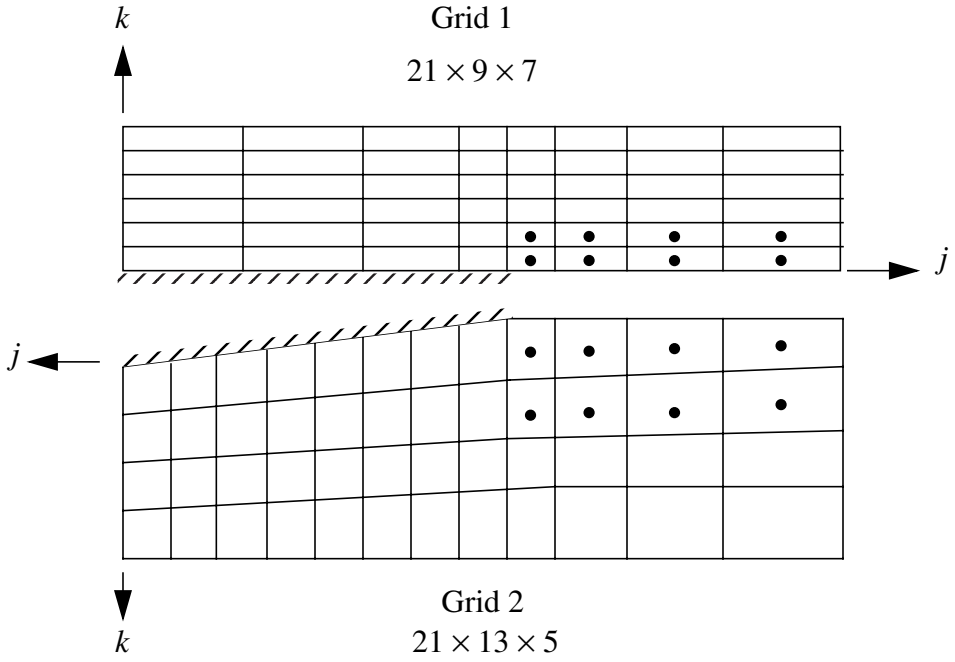


Figure 6-13. One-to-one blocking example 3.

The previous two examples show 1-1 blocking over the entire k range of two grids. In some cases, only segments of faces will utilize 1-1 communication between grids. In Figure 6-13, two grids share a portion of a face beyond boundaries defined as inviscid surfaces. Again, the figure shows an i = constant face (not necessarily the *same* constant) of two grids. Communication is desired between **jsta** = 5 and **jend** = 9 on the k = 1 face of grid 1 and **jsta** = 1 and **jend** = 5 on the k = 1 face of grid 2. The pertinent input would look something like:

Line Type

```
6       NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
            2         1         0       100         0         0         1         1

8        IDIM      JDIM      KDIM
           21         9         7
           21        13         5

13       GRID     NBCI0   NBCIDIM     NBCJ0  NBCJDIM     NBCK0  NBCKDIM    IOVRLP
            1         1         1         1         1         2         1         0
            2         1         1         1         1         2         1         0

18   K0: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
            1         1      1005         1        21         1         5         0
            1         2         0         1        21         5         9         0
            2         1         0         1        21         1         5         0
            2         2      1005         1        21         5        13         0

      1-1 BLOCKING DATA:
```

```
24      NBLI
          1
25    NUMBER GRID     :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    1            1      5      1     21      9      1      1      2
26    NUMBER GRID     :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    2            1      5      1     21      1      1      1      2
```

Note that, since the j index runs in opposite directions on the two grids, one range is increasing and one range is decreasing in the 1-1 blocking input. It does not matter which grid's range increases and which one decreases as long as they map the points in the correct order.

The boundary conditions at the k = 1 face from **jsta** to **jend**-1 on any i plane, denoted i1, of grid 1 would be set as:

Grid 1	Grid 2
qk0(5,i1,**l**,1) =	q(4,1,i2,**l**)
qk0(6,i1,**l**,1) =	q(3,1,i2,**l**)
qk0(7,i1,**l**,1) =	q(2,1,i2,**l**)
qk0(8,i1,**l**,1) =	q(1,1,i2,**l**)
qk0(5,i1,**l**,2) =	q(4,2,i2,**l**)
qk0(6,i1,**l**,2) =	q(3,2,i2,**l**)
qk0(7,i1,**l**,2) =	q(2,2,i2,**l**)
qk0(8,i1,**l**,2) =	q(1,2,i2,**l**)

The boundary conditions at the k = 1 face from **jsta** to **jend**-1 on any i plane, denoted i2, of grid 2 would be set as:

Grid 2	Grid 1
qk0(1,i2,**l**,1) =	q(8,1,i1,**l**)
qk0(2,i2,**l**,1) =	q(7,1,i1,**l**)
qk0(3,i2,**l**,1) =	q(6,1,i1,**l**)
qk0(4,i2,**l**,1)=	q(5,1,i1,**l**)
qk0(1,i2,**l**,2) =	q(8,2,i1,**l**)
qk0(2,i2,**l**,2)=	q(7,2,i1,**l**)
qk0(3,i2,**l**,2) =	q(6,2,i1,**l**)
qk0(3,i2,**l**,2) =	q(5,2,i1,**l**)

Example 4

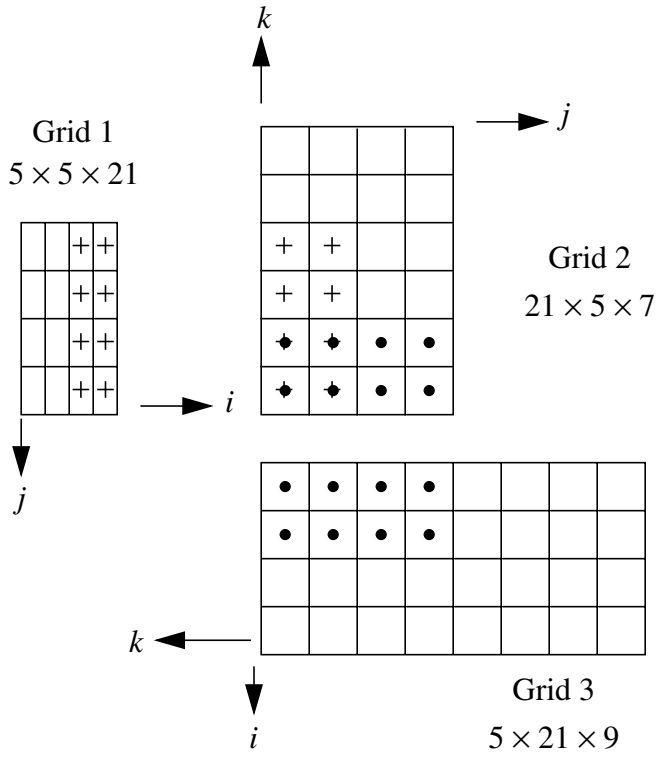


Figure 6-14. One-to-one blocking example 4.

The last example of 1-1 blocking involves three grids shown in Figure 6-14. In the figure, a k = constant (denoted k1) plane of grid 1, an i = constant (denoted i2) plane of grid 2, and a j = constant (denoted j3) plane of grid 3 are shown. Communication between grids 1 and 2 is desired between the i = **idim** face of grid 1 and the j = 1 face of grid 2. Communication between grids 2 and 3 is desired between the k = 1 face of grid 2 and the i = 1 face of grid 3. The pertinent input would look something like:

Line Type

```
6      NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
           3         1         0       100         0         0         1         1

8        IDIM      JDIM      KDIM
           5         5        21
          21         5         7
           5        21         9

13       GRID     NBCI0  NBCIDIM     NBCJ0  NBCJDIM     NBCK0  NBCKDIM    IOVRLP
           1         1         1         1         1         1         1         0
           2         1         1         2         1         1         1         0
           3         2         1         1         1         1         1         0
14  I0:  GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
           1         1      1003         1         5         1        21         0
           2         1      1003         1         5         1         7         0
           3         1      1005         1        21         1         5         0
```

```
            3        2        0        1       21        5        9        0
15IDIM: GRIDSEGMENTBCTYPE    JSTA     JEND     KSTA     KEND    NDATA
      1    1        0        1        5        1       21        0
      2    1     1001        1        5        1        7        0
      3    1     1000        1       21        1        9        0
16  J0: GRID   SEGMENT    BCTYPE     ISTA     IEND     KSTA     KEND     NDATA
            1        1     1000        1        5        1       21         0
            2        1        0        1       21        1        5         0
            2        2     1000        1       21        5        7         0
            3        1     1003        1        5        1        9         0

18  K0: GRID   SEGMENT    BCTYPE     ISTA     IEND     JSTA     JEND     NDATA
            1        1     1003        1        5        1        5         0
            2        1        0        1       21        1        5         0
            3        1     1003        1        5        1       21         0

      1-1 BLOCKING DATA:
24     NBLI
          2
25   NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    1          5      1      1      5      5     21      2      3
          2    2          1      1      1     21      5      1      1      2
26   NUMBER GRID    :   ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
          1    2         21      1      5      1      1      1      3      1
          2    3          1      1      9      1     21      5      2      3
```

Notice that the i index range for grid 2 runs in the opposite direction of the k index range for grid 3 due to the right-hand rule.

The boundary conditions at the i = **idim** face of grid 1 would be set as:

Grid 1	Grid 2
qi0(1,k1,**l**,3) =	q(1,4,i2,**l**)
qi0(2,k1,**l**,3) =	q(1,3,i2,**l**)
qi0(3,k1,**l**,3) =	q(1,2,i2,**l**)
qi0(4,k1,**l**,3) =	q(1,1,i2,**l**)
qi0(1,k1,**l**,4) =	q(2,4,i2,**l**)
qi0(2,k1,**l**,4) =	q(2,3,i2,**l**)
qi0(3,k1,**l**,4) =	q(2,2,i2,**l**)
qi0(4,k1,**l**,4) =	q(2,1,i2,**l**)

The boundary conditions from **ksta** to **kend**-1 on the j = 1 face of grid 2 would be set as:

Grid 2	Grid 1
qj0(1,i2,**l**,1) =	q(4,k1,4,**l**)
qj0(2,i2,**l**,1) =	q(3,k1,4,**l**)
qj0(3,i2,**l**,1) =	q(2,k1,4,**l**)
qj0(4,i2,**l**,1) =	q(1,k1,4,**l**)
qj0(1,i2,**l**,2) =	q(4,k1,3,**l**)
qj0(2,i2,**l**,2) =	q(3,k1,3,**l**)
qj0(3,i2,**l**,2) =	q(2,k1,3,**l**)
qj0(4,i2,**l**,2) =	q(1,k1,3,**l**)

The boundary conditions at the $k = 1$ face of grid 2 would be set as:

Grid 2	Grid 3
qk0(1,i2,**l**,1) =	q(j3,8,1,**l**)
qk0(2,i2,**l**,1) =	q(j3,7,1,**l**)
qk0(3,i2,**l**,1) =	q(j3,6,1,**l**)
qk0(4,i2,**l**,1) =	q(j3,5,1,**l**)
qk0(1,i2,**l**,2) =	q(j3,8,2,**l**)
qk0(2,i2,**l**,2) =	q(j3,7,2,**l**)
qk0(3,i2,**l**,2) =	q(j3,6,2,**l**)
qk0(4,i2,**l**,2) =	q(j3,5,2,**l**)

The boundary conditions from **ksta** to **kend**-1 on the $i = 1$ face of grid 3 would be set as:

Grid 3	Grid 2
qi0(j3,5,**l**,1) =	q(4,1,i2,**l**)
qi0(j3,6,**l**,1) =	q(3,1,i2,**l**)
qi0(j3,7,**l**,1) =	q(2,1,i2,**l**)
qi0(j3,8,**l**,1) =	q(1,1,i2,**l**)
qi0(j3,5,**l**,2) =	q(4,2,i2,**l**)
qi0(j3,6,**l**,2) =	q(3,2,i2,**l**)
qi0(j3,7,**l**,2) =	q(2,2,i2,**l**)
qi0(j3,8,**l**,2) =	q(1,2,i2,**l**)

6.3.2 Patched-Grid Interpolation

(Input Line Type Twenty-Seven)

6.3.2.1 General Information

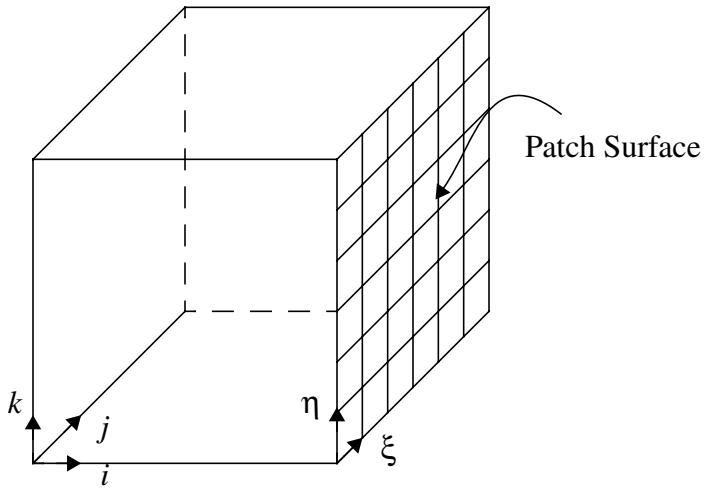


Figure 6-15. Patched-grid surface.

This section describes the "basics" of patching in CFL3D and then shows a simple static patching example to illustrate the basic concepts. Next, an example of dynamic patching is given. Patched-grid interpolation is designed for communication between grids that share a common face, but are *not* C^0 continuous. For example, a global (3-d) grid is represented by the "box" in Figure 6-15. The grid must be right-handed as shown with the i, j, and k axes drawn in the figure. Patching may occur on an i = constant, j = constant, or k = constant surface. In the figure, an i = constant surface is indicated as the patched surface. Note that the adjacent grid involved in the patch is not shown in the figure. Patching works best when the spacing of the adjacent grid in the normal direction to the patch is the same as that in the other grid. While the patch surface is shown as a plane, it may be non-planar as well. The local (2-d) indexing on the patch surface is indicated by the ξ and η axes.

The global and local indices correspond as follows (this is important when determining the input parameters for dynamic patching in Section 3.42 on page 49):

If the patch surface is on an i = constant surface (as shown in Figure 6-15), then

$$\begin{aligned} k &\Leftrightarrow \eta \\ j &\Leftrightarrow \xi \end{aligned} \qquad \text{(6-41)}$$

If the patch surface is on a j = constant surface, then

$$i \Leftrightarrow \eta$$
$$k \Leftrightarrow \xi \qquad \text{(6-42)}$$

If the patch surface is on a k = constant surface, then

$$i \Leftrightarrow \eta$$
$$j \Leftrightarrow \xi \qquad \text{(6-43)}$$

One grid may have multiple grids patched to it. For example, the flow over a nose cone illustrated in Figure 6-16 shows block 1 patched to both block 2 and block 3. Note that the orientation of grid lines in block 3 is different from that in blocks 1 and 2 to illustrate the features of the patching algorithm.

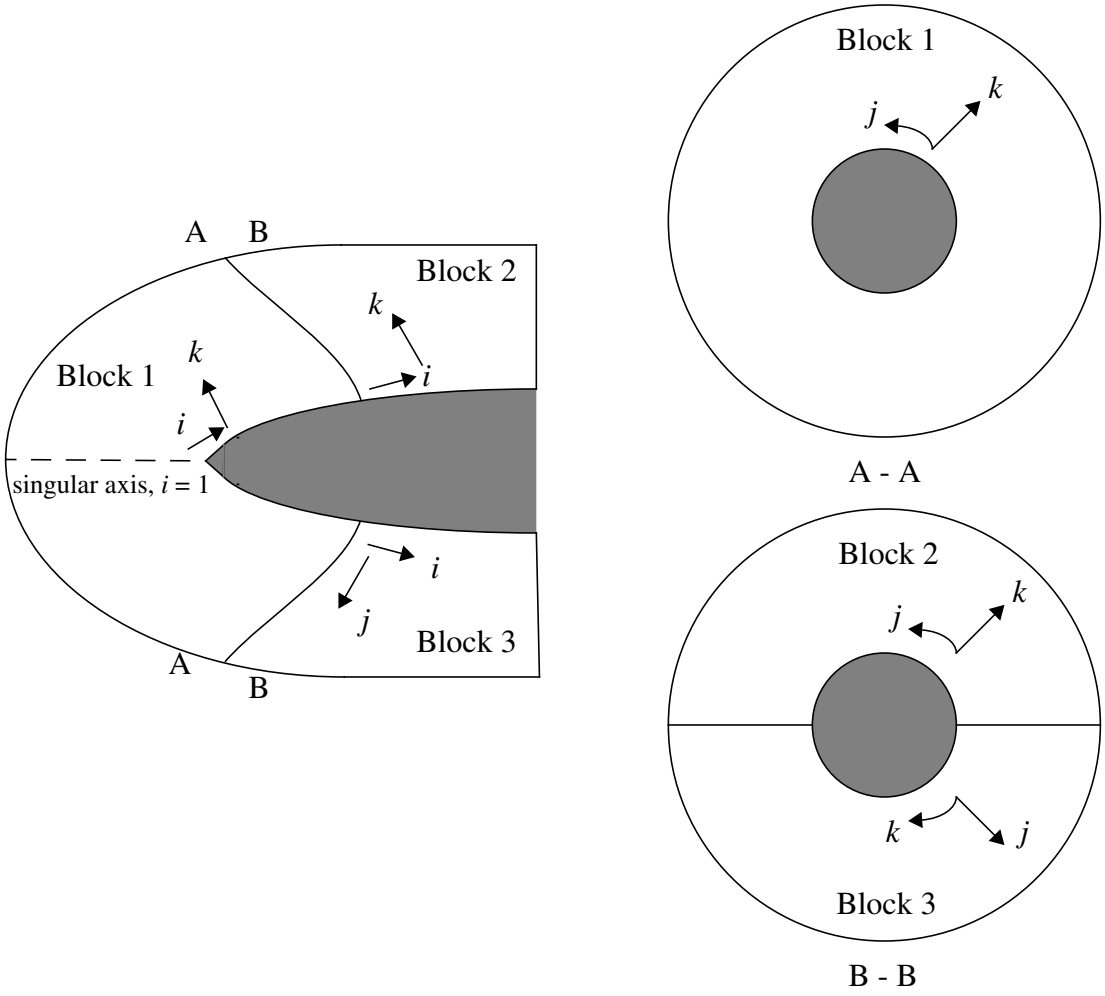


Figure 6-16. Patched-grid example.

To provide two-way communication between all blocks, a total of seven interpolations are required for this case, as follows:

Interpolation:	Description:
1	*To i* = **idim** of block 1 *from i* = 1 of blocks 2 and 3 (two "from" blocks)
2	*To i* = 1 of block 2 *from i* = **idim** of block 1
3	*To i* = 1 of block 3 *from i* = **idim** of block 1
4	*To j* = 1 of block 2 *from k* = 1 of block 3
5	*To j* = **jdim** of block 2 *from k* = **kdim** of block 3
6	*To k* = 1 of block 3 *from j* = 1 of block 2
7	*To k* = **kdim** of block 3 *from j* = **jdim** of block 2

The interpolations may be input in any order (but must be consistent once the order is chosen).

6.3.2.2 Description/Discussion of Input Parameters

Basically, the patch algorithm works as follows. The interpolations are cycled through, one at a time, so that at any given time there is one surface being interpolated to. Note, however, that there may be more than one surface being interpolated from. In order to interpolate from one (or more) block(s) to another, interpolation coefficients are required. These are found by expressing the cell-center coordinates x_c, y_c, z_c on the "to" side of the patch surface in terms of a nonlinear polynomial in ξ, η, where ξ, η are the local coordinates on the "from" side of the patch surface. Newton iteration is used to invert the polynomial to find the local coordinates of the cell center, ξ_c, η_c.

Because of the nonlinearity of the equation, problems may arise in the iteration process. The parameters **ifit**, **limit**, and **itmax** may be adjusted to try to overcome any convergence difficulties. There must be **ninter2** values for each of these three parameters. **Limit** is the maximum step size in ξ or η allowed during the search procedure. The value 1 seems to be a good general choice. **Itmax** is the maximum number of search steps allowed per grid point. A rule of thumb is **limit** $\times$ **itmax** $\approx$ the maximum dimension of i, j, or k on a patch surface.

Ifit controls the order of the polynomial fit used to relate x, y, or z to ξ and η. **Ifit** = 1 for linear in both ξ and η (bilinear). **Ifit** = 2 for quadratic in both ξ and η (degenerate biquadratic). **Ifit** = 3 for quadratic in ξ, linear in η. **Ifit** = 4 for linear in ξ, quadratic in η.

Some tips for choosing **ifit** are:

- A grid with highly curved grid lines in one of the *local* directions will need a quadratic polynomial in that direction.
- Refer to Section 6.3.2.1 and the user's own knowledge of the grid to decide which (if any) of the local directions ξ or η are highly curved.
- Use the lowest order fit which seems reasonable in each direction.

- One-to-one patching can always be done with a bilinear fit, regardless of curvature.

To and **from** are numbers containing attributes of the "to" side of the patch surface and the "from" side of the patch surface. For each of the **ninter2** interpolations there is only one value of **to**, but there may be more than one value of **from** which is set by the parameter **nfb**. (**Nfb** is the number of blocks on the "from" side of the patch surface.) The values of **to** and **from** are of the form:

to/from = Nmn

where "N" indicates the block number; "m" indicates the coordinate which is constant on the patch surface (may differ on either side of the patch):

$$m = 1 \Rightarrow i = \text{constant}$$
$$m = 2 \Rightarrow j = \text{constant}$$
$$m = 3 \Rightarrow k = \text{constant}$$

and "n" indicates on which of the two possible m = constant surfaces the patch surface occurs:

$$n = 1 \text{ for patch on } m = 1 \text{ surface}$$
$$n = 2 \text{ for patch on } m = mdim \text{ surface}$$

The following inputs pertain to the nose cone case in Figure 6-16:

Interpolation #	to	nfb	from
1	112	2	211, 311
2	211	1	112
3	311	1	112
4	221	1	331

Example

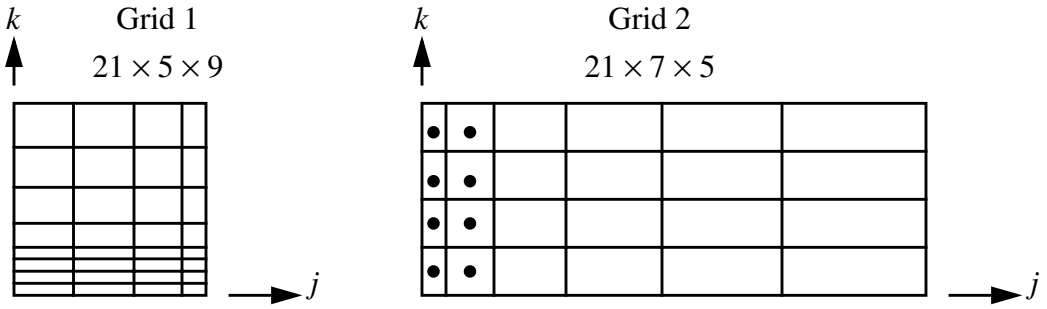


Figure 6-17. Patched-grid example.

Suppose the k dimension of grid 1 in the first example of 1-1 blocking (see Figure 6-11 on page 102) was actually **kdim** = 9. Grid 1 might look something like that drawn in Figure 6-17. As in Figure 6-11, Figure 6-17 shows an i = constant face (not necessarily the *same* constant) of two grids. Notice that the spacing in the direction normal to the interface (i.e. j) is approximately constant across the interface.

In Version 5.0 of CFL3D, patched-grid interpolation data for *static* patched interfaces (interfaces that do not change with time) must be obtained as a preprocessing step. The code ronnie is designed for this task. *Dynamic* patched interfaces, such as occur when grids slide past one another, are computed internally in CFL3D.

Note that, if two grids with very different sizes are patched together, it may be necessary to use a limiter on the gradients there. To do this, replace (hard-wire) the calls to `int2` with calls to `int3` instead.

6.3.3 Chimera Grid Interpolation

(Input Line Type Ten)

Grid overlapping, also known as the overset-grid method or chimera technique, requires neither 1-1 connectivity nor a shared interface to pass flow information from one grid to another. With this method, a variety of grid topologies can be used together. The chimera implementation used in CFL3D is based on the method of Benek et al.[12] As a simplified example of grid overlapping, consider a cross section of a polar grid and a Cartesian mesh as shown Figure 6-18. Suppose the grids overlap as in Figure 6-19. Since both grids cover the same area, computations on both grids in this area would be redundant. Therefore, certain points on the Cartesian grid will be eliminated from the computation. The polar grid is used to carve a "hole" in the Cartesian grid. In this example, the hole is defined as any cell-center point of the Cartesian grid interior to the $k = 4$ grid line of the polar mesh. The Cartesian mesh with the hole carved out is illustrated in Figure 6-20. Any cell center point of the Cartesian grid located within this hole is designated a "hole point". The first two "nonhole" cell-center points of the Cartesian grid that border a hole point both vertically and horizontally are labelled "fringe points". The remaining grid points that have not been designated as either hole or fringe points are called "field" points. Figure 6-21 depicts the hole, fringe, and boundary points for this example. Each fringe point of the Cartesian grid falls within a "target cell" of the polar grid.

A searching algorithm is used to identify the particular eight points that define the hexahedral target cell. The search begins with an initial guess for the target cell. Next, the current target cell is isoparametrically mapped into a unit cube in computational space. The same transformation into the mapped coordinate system is then applied to the fringe point; if the mapped fringe point lies in the same unit cube as the current target cell, then that target cell in fact encases the fringe point. If the mapped fringe point lies outside the unit cube, then the current target cell is not the correct choice. However, the magnitude and direction of the mapped fringe point relative to the current target cell may be used to choose a new guess for the target cell. The mapping process is repeated until the correct target cell is identified. With the correct target cell identified, the data are transferred from the target cell to the fringe point with trilinear interpolation in computational space. Outer boundary values of the polar grid are determined in a similar manner. The MultiGeometry Grid Embedder (MaGGiE) code, written specifically for CFL3D by Baysal et al.[11], is used to determine the interpolation information between grids.

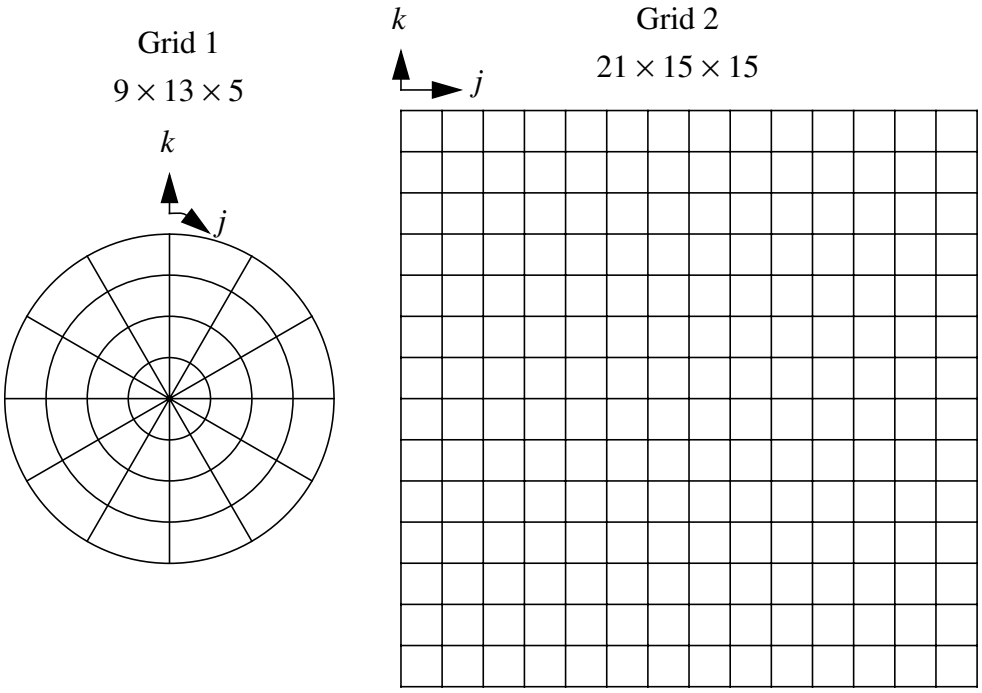


Figure 6-18. Grid-overlapping grid examples.

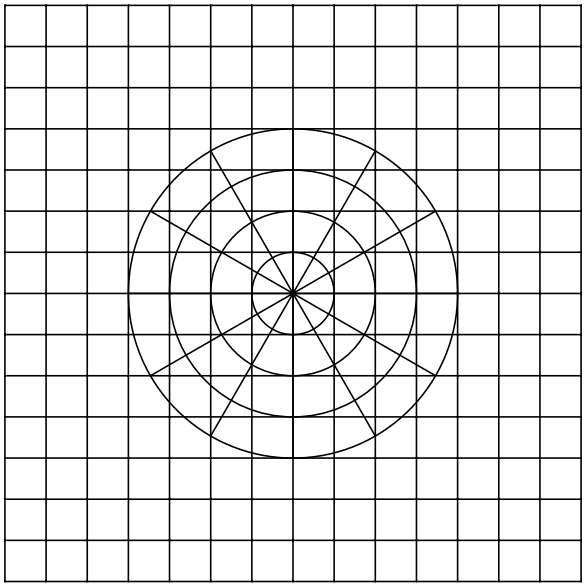


Figure 6-19. Overlapped grids.

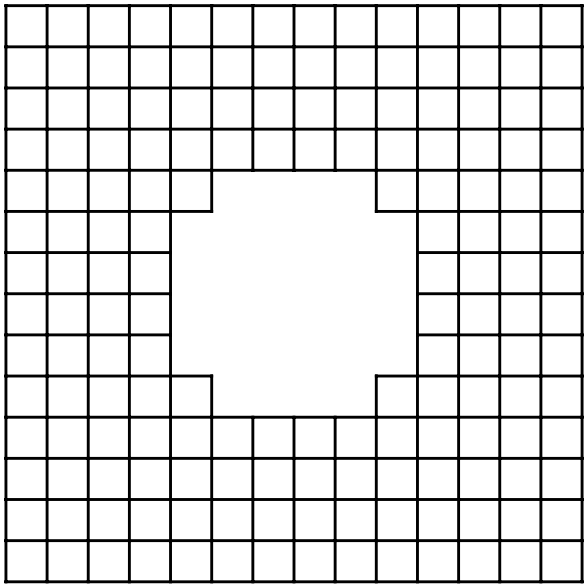


Figure 6-20. Hole in Cartesian grid in region of polar grid.

● Hole Point for Cartesian Grid

\+ Fringe Point for Cartesian Grid

○ Boundary Point for Polar Grid

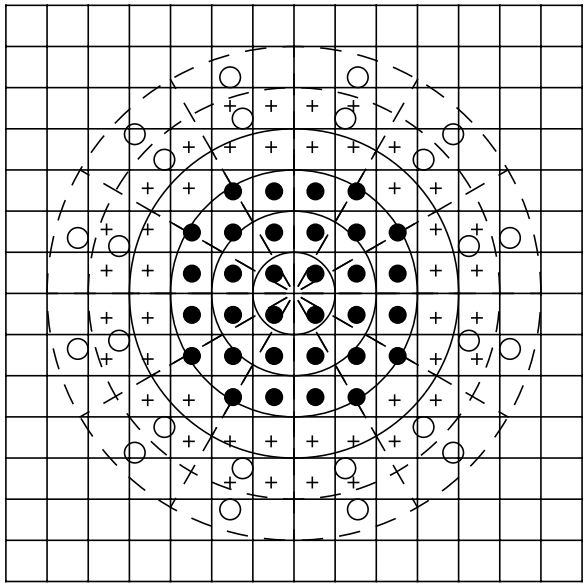


Figure 6-21. Hole, fringe, and boundary points for overlapped grid example.

6.3.4 Embedded Mesh

Regular refinement of coarse mesh

Embedded grids are useful when high gradient areas are limited to an identifiable region (see reference 25). An embedded grid can be placed in that region to resolve the flow field without refining the entire mesh. As an example, an embedded grid scheme is shown in Figure 6-22 for two dimensions. The diagram represents full refinement in both directions. The solid lines define a finer mesh embedded completely within a coarser mesh depicted by the dashed lines. In the figure, a portion of the flow field is covered by both the embedded mesh and a portion of the coarser grid. The grids are coupled together during the solution process. The cell-center variables on a coarser grid cell which underlies a finer embedded grid cell are replaced with a volume-weighted restriction of variables from the four (2-d) or eight (3-d) finer grid cells, similar to the restriction operators used in a global multigrid scheme.

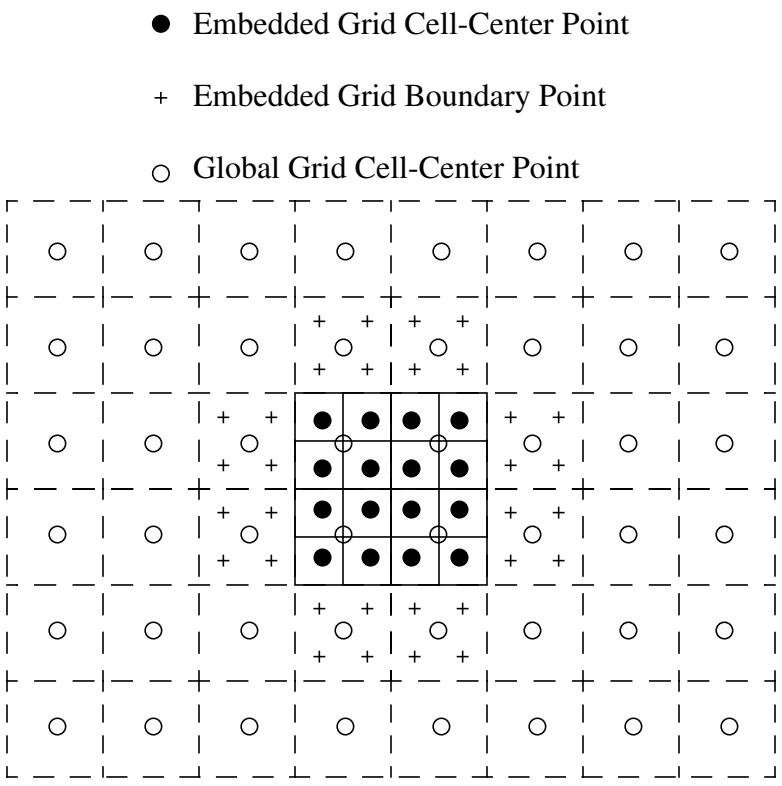


Figure 6-22. Embedded grid example.

For the embedded grid, the computational boundaries occur either at a physical boundary, such as a wing, a symmetry plane, an inflow/outflow plane along a zonal interface (one-to-one, patched, or overset), or along an interior computational surface of a coarser grid. Along an interior surface, two additional lines of data corresponding to an analytical continuation of the finer grid cell centers are constructed from linear interpolation of the coarser grid state variables. An enlarged view of the lower-left corner of the embedded

mesh of Figure 6-22 is shown in Figure 6-23. Arrows indicate which coarse grid points are used in the linear interpolation scheme to obtain a couple of the boundary points drawn. If the input parameter **iconsf** = 1 ("LT20 - Mesh Sequencing and Multigrid"), then global conservation is enforced by replacing the coarse grid flux at an embedded grid boundary with the sum of the finer grid fluxes which share the common interface. If **iconsf** = 0, the coarse grid flux is computed using the volume-weighted restriction from the fine grid, and conservation is not insured.

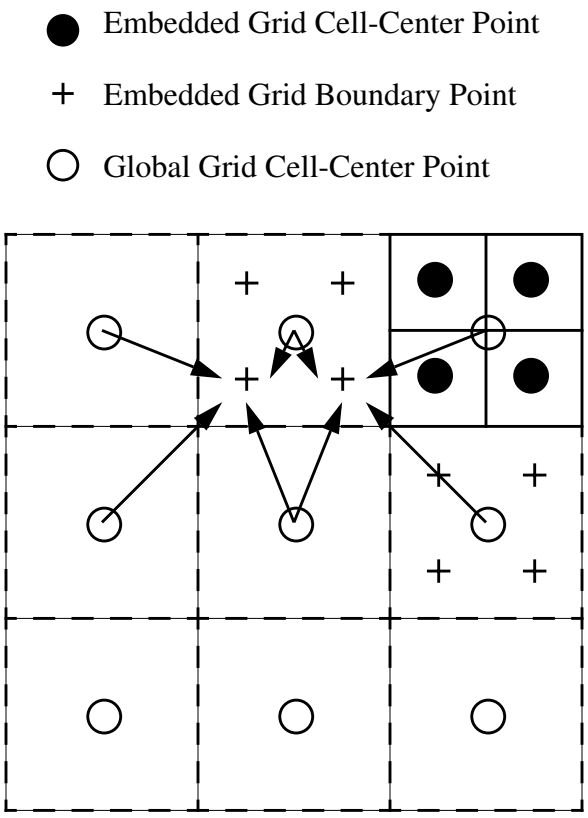


Figure 6-23. Enlarged portion of embedded mesh.

When designing an embedded mesh, it is important to remember that at least two layers of coarse grid cells should surround the embedded boundaries unless the embedded boundary is set with a physical boundary condition (e.g. solid wall, plane of symmetry, or far-field) or a zonal boundary such as a one-to-one, patched, or overset type. In the example of Figure 6-22, there are two layers of coarse grid cells above and below the embedded mesh and three layers of coarse grid cells on the left and right sides of the embedded mesh.

Suppose the grids in Figure 6-22 are in an i = constant plane. Also, **jdim** = 9 and **kdim** = 7 for the coarse grid and **jdim** = 5 and **kdim** = 5 for the embedded mesh. The pertinent input would look something like:

Line Type

6	NGRID	NPLOT3D	NPRINT	NWREST	ICHK	I2D	NTSTEP	ITA
	2	1	0	100	0	0	1	1
7	NCG	IEM	IADVANCE	IFORCE	IVISC(I)	IVISC(J)	IVISC(K)	
	0	0	0	0	0	0	0	
	0	1	0	0	0	0	0	
8	IDIM	JDIM	KDIM					
	21	9	7					
	41	5	5					
10	INEWG	IGRIDC	IS	JS	KS	IE	JE	KE
	0	0	0	0	0	0	0	0
	0	1	1	4	3	21	6	5
20	MSEQ	MGFLAG	ICONSF	MTT	NGAM			
	1	0	1	0	0			
22	NCYC	MGLEVG	NEMBL	NITFO				
	1	1	1	0				

In this example, the embedded mesh extends the entire length of the global grid in the i direction. The dimensions of the embedded mesh must satisfy a particular relationship to the data in "LT10 - Embedded Mesh Specifications". The input parameters **is**, **ie**, **js**, **je**, **ks**, and **ke** are indices in the global grid to which the embedded mesh is connected. Thus, the following must be true:

$$\mathbf{idim}_{embedded} = 2(\mathbf{ie} - \mathbf{is})_{global} + 1 \qquad \textbf{(6-44)}$$

Analogous relationships hold for the j and k directions. Note that the only exception to Equation (6-44) occurs in the i direction for 2-d cases, where **is** = 2, **ie** = 1 and $\mathbf{idim}_{embedded} = \mathbf{idim}_{global} = 2$.

There is also an option called "semi-coarsening" in which the number of points in the i direction are the same for the embedded grid and the global grid in the embedded region. This is helpful if there are sufficient points in the i direction, but grid enrichment in $j-k$ planes is desirable. There is an internal check for this option and the only change in the input sample above is to set **idim** = 21 for the embedded grid as well. Note that semi-coarsening can only be used in the i direction.

Another nice option with grid embedding is to add an embedded grid to a previously-run coarse grid problem. For instance, suppose a converged coarse grid solution is obtained and the flow field looks well resolved everywhere except in a vortex region. An embedded grid for that region can be tacked on to the original grid file and the case restarted. On the *first* run of the restart, set **inewg** = 1 for the embedded grid. This lets the code know that a solution not on the current restart file is beginning. On the next run, the embedded grid solution *will* be on the restart file, so set **inewg** = 0.

CHAPTER 7

Convergence Acceleration

In CFL3D, the convergence rate at which many problems are solved can be accelerated with the use of multigrid, mesh sequencing, or a combination of the two. The following sections describe the two techniques. The use of multigrid with grid overlapping and embedded grids is also discussed.

Multigrid is a *highly* recommended option available in CFL3D. The improvement in convergence acceleration afforded with multigrid makes it very worthwhile to learn about and utilize. Table 7–1 illustrates this fact for two cases, a 2-d airfoil and a 3-d forebody. A work unit is defined here to be the "equivalent" fine grid iteration. For example, on the finest grid level, 1 iteration = 1 work unit; on the next-to-finest grid level, 1 iteration = 1/8 (1/4 in 2-d) of a work unit; on the next coarser level, 1 iteration = 1/64 (1/16 in 2-d) of a work unit; and so forth. The residual and lift coefficient convergence histories for these two cases are shown in Figure 7-1 and Figure 7-2. The term "full multigrid" implies that mesh sequencing is used in conjunction with multigrid. See "Mesh Sequencing" on page 134. Note that there is an additional CPU penalty for the overhead associated with the multigrid scheme that is not reflected in the work unit measure, however, for most problems, the overhead is small.

Table 7–1. Convergence improvements with multigrid.

Case	Approximate Number of Work Units to Achieve the Final Lift		
	Without Multigrid	With Multigrid	With Full-Multigrid
2-d NACA 4412 Airfoil	10,000	1000	750
3-d F-18 Forebody	>> 1500*	500	500

*This case was stopped after 1500 iterations when it became clear that the lift coefficient was far from converged and to continue running the case would have been a waste of computer resources.

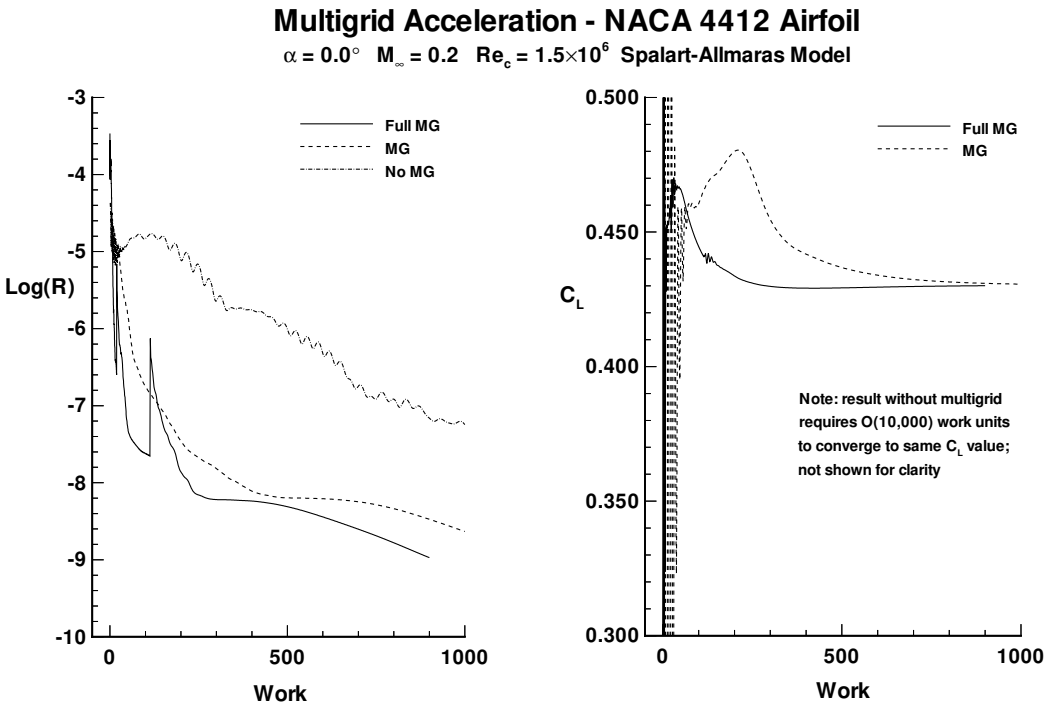


Figure 7-1. Convergence acceleration for 2-d NACA 4412 airfoil case.

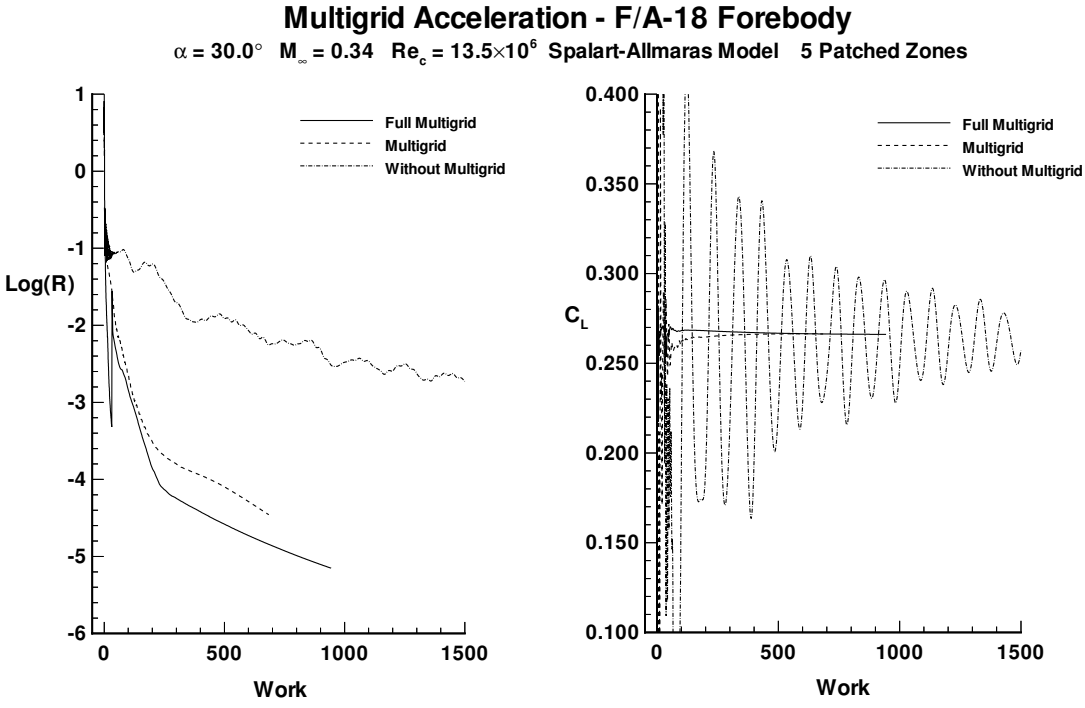


Figure 7-2. Convergence acceleration for 3-d F-18 Forebody case.

7.1 Multigrid

7.1.1 Global Grids

In a single-grid-level algorithm, the flow equations are solved only on the finest grid. The convergence rate for this algorithm is typically good during the initial stages of the computation but soon degrades significantly due to poor damping of low-frequency errors.[13] Multigrid methods, in which a sequence of coarser grids are used to eliminate the low-frequency errors, can significantly improve the convergence rate of the algorithm. The full-approximation-scheme multigrid algorithm, as developed and applied to inviscid three-dimensional flows in references 5, 6 and 7, is available in CFL3D to accelerate convergence of a solution to a steady state. A sequence of grids $G_0, G_1, \ldots, G_N$ is defined, where G_N denotes the finest grid and coarser grids are formed by successively deleting every other grid line in all three coordinate directions (except in 2-d, where the i direction is not coarsened). The fine grid serves to damp the high-frequency errors; the coarser grids damp the low-frequency errors. The coarse grids are solved with a forcing function on the right-hand side arising from restricting the residual from the finer meshes. The forcing function is the relative truncation error between the grids such that the solutions on the coarser meshes are driven by the fine grid residual. A correction is calculated on the coarser meshes and passed up to the next finest mesh using trilinear interpolation in the prolongation step.

As an example, consider a 2-d grid of dimensions 17×17 as shown in Figure 7-3. Suppose it is desired to use four levels of multigrid, i.e. the fine grid level and three coarser grid levels. The three coarser grid levels, along with their dimensions, are also shown in Figure 7-3. Figure 7-4 illustrates a typical W-cycle multigrid scheme for four grid levels. In the figure, n refers to the current iteration. The multigrid cycle begins with a Navier-Stokes calculation performed at the finest grid level specified. The fine grid residual and flow-field variables are restricted to the next grid level where they are utilized to perform a Navier-Stokes calculation at this level. The process continues until the coarsest grid level is reached. After a Navier-Stokes calculation is performed at this level, the solution is prolongated back “up” the cycle. With the W-cycle, additional calculations and the restrictions and prolongation steps are repeated at the coarser grid levels giving the “W” pattern illustrated in Figure 7-4. The other option available is a V-cycle. An example of a V-cycle for the four grid level case is illustrated in Figure 7-5. Generally, a W-cycle is more effective since additional work is done at the coarse grid level. However, this may be case dependent. Note that, if only two grid levels are specified, a V-cycle will always be performed even if a W-cycle is specified in the input file.

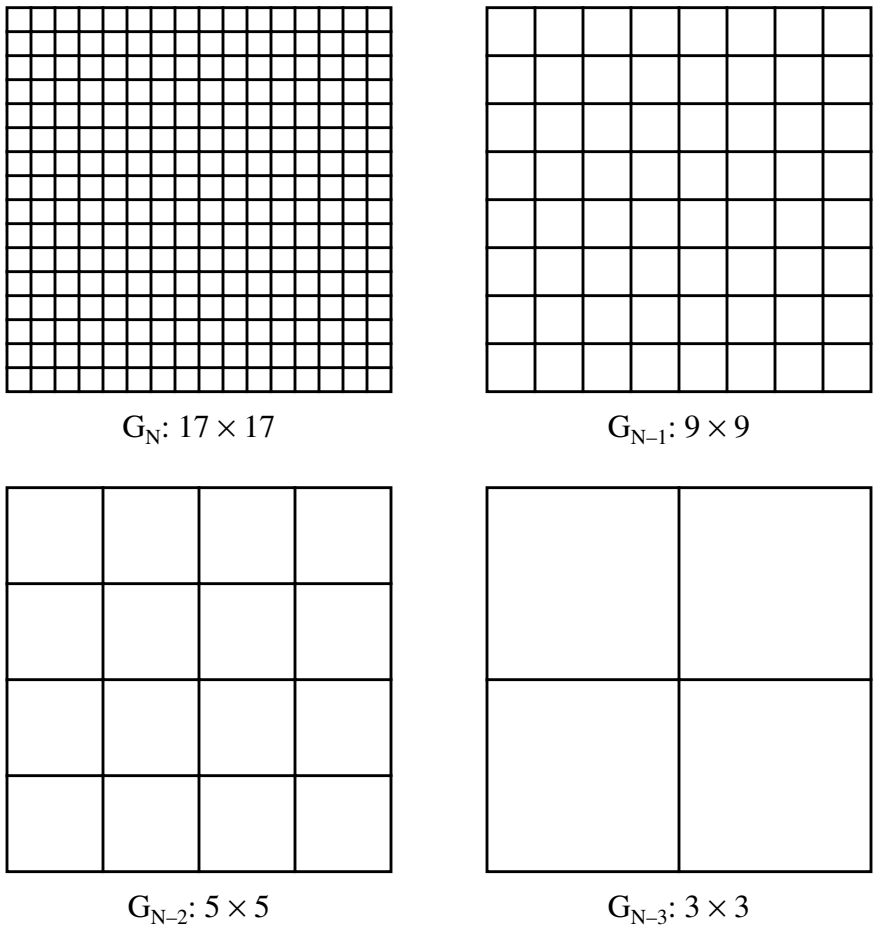


Figure 7-3. Fine grid and coarse grid levels.

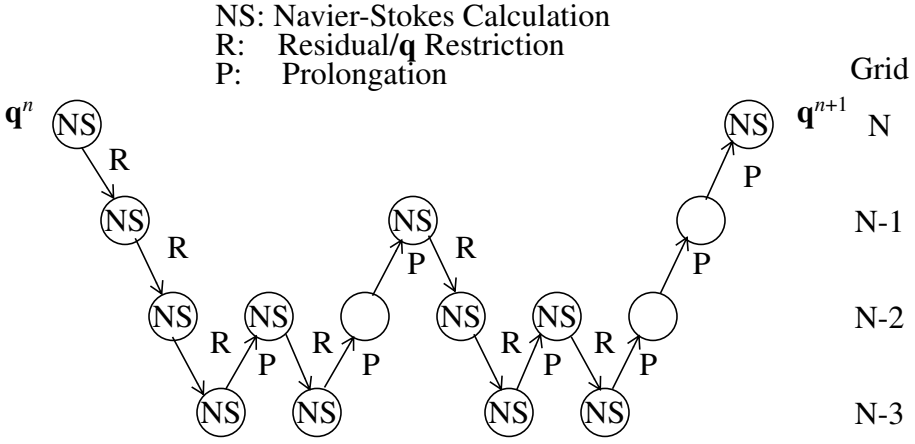


Figure 7-4. Multigrid W-cycle.

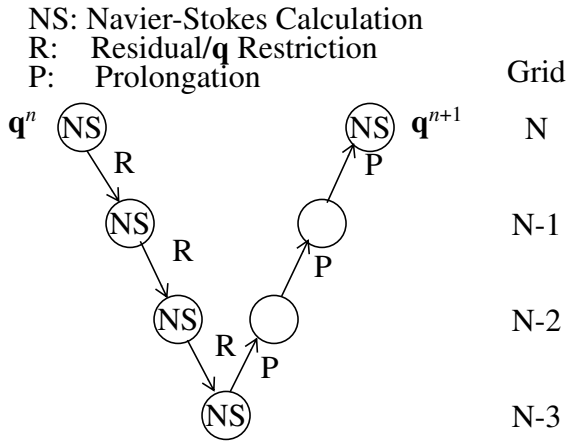


Figure 7-5. Multigrid V-cycle.

For a multigrid W-cycle, the pertinent lines of input for the example in Figure 7-3 would look something like:

Line Type

```
5       DT      IREST   IFLAGTS      FMAX     IUNST    CFLTAU
       -1.0         0         0       1.0         0      10.0
6     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
          1         0         0         0         0         1         1         1
7       NCG       IEM IADVANCE    IFORCE IVISC(I)  IVISC(J)  IVISC(K)
          3         0         0         1         0         0         0
8      IDIM      JDIM      KDIM
          2        17        17
20     MSEQ    MGFLAG    ICONSF       MTT      NGAM
          1         1         0         0         2
22     NCYC    MGLEVG     NEMGL     NITFO
         200         4         0         0
23     MIT1      MIT2      MIT3      MIT4      MIT5
          1         1         1         1         1
```

7.1.2 A Word About Grid Dimensions

When determining how many multigrid levels are available for a particular grid, use the following formula:

$$m_c = \frac{m_f - 1}{2} + 1 = \frac{m_f + 1}{2} \qquad \textbf{(7-1)}$$

where m_f is a fine-grid dimension (**idim**, **jdim**, or **kdim**) and m_c is the corresponding coarse-grid dimension. Then rename m_c as m_f and compute Equation (7-1) again to determine an even coarser grid dimension. When m_c is an even number, that is as coarse as that grid dimension can go.

For example, suppose a grid has **idim** = 37, **jdim** = 65, **kdim** = 65. The coarse grid below that would have **idim** = 19, **jdim** = 33, **kdim** = 33. The next coarser grid would have **idim** = 10, **jdim** = 17, **kdim** = 17. Since **idim** is an even number, that is as coarse as the grid can go; even though **jdim** and **kdim** can be reduced to coarser numbers, the entire grid is restricted by the **idim** = 10 value.

If the use of multigrid is desirable, it is important to plan ahead in the grid generation step of the computational problem. Choose grid dimensions that are "good" multigrid numbers, such as 129 (65, 33, 17, 9, 5, 3), 73 (37, 19), and 49 (25, 13, 7). Generally, two or three coarser grid levels are satisfactory.

It is best if all grid *segments* are also multigridable (see "LT14 - I0 Boundary Condition Specification" on page 32 through "LT19 - KDIM Boundary Condition Specification" on page 35). For example, a face with **jdim** = 65 might have a portion from j = 1 to 17 as a one-to-one interface, a portion from j = 17 to 41 as a viscous wall, and a portion from j = 41 to 65 as a patched interface. Each of these segment lengths (7, 25, and 25) is multigridable down three levels. Note that CFL3D will sometimes work fine even if some grid segments are not multigridable: the code can assign indices on the coarser levels that denote different physical locations than the indices on the fine grid and still converge on the finest level. However, this does *not* work all the time. For example, if a C-mesh has a wrap-around dimension of **jdim** = 257 and the wake extends from j = 1 to 40 and j = 218 to 257 (with one-to-one point matching), the code will create a coarser level with the wake from j = 1 to 20 and j = 109 to 129 (using Equation (7-1)). On this level, j = 20 is a *physically different point* from j = 109 (it is not even the actual trailing edge), and will yield a boundary condition error when the code is run, because these points are expected to match in a one-to-one fashion. It is currently not necessary to specify laminar regions with multigridable numbers. See *Note* (3) on page 29 in the LT9 - Laminar Region Specification description.

It is also important to consider the grid *geometry* when defining the grid dimensions. If multigrid is used, be sure to have any important geometric features (such as corners) located at multigridable points. Otherwise, on coarser levels, the geometry may change significantly, resulting in poor multigrid performance.

For a handy reference, the following table lists the grid sizes (< 1000) that are multigridable to three additional levels:

Table 7–2. Grid sizes multigridable to three additional level.

Grid:	Coarser Levels:			Grid:	Coarser Levels:			Grid:	Coarser Levels:		
9	5	3	2	345	173	87	44	673	337	169	85
17	9	5	3	353	177	89	45	681	341	171	86
25	13	7	4	361	181	91	46	689	345	173	87
33	17	9	5	369	185	93	47	697	349	175	88
41	21	11	6	377	189	95	48	705	353	177	89
49	25	13	7	385	193	97	49	713	357	179	90
57	29	15	8	393	197	99	50	721	361	181	91
65	33	17	9	401	201	101	51	729	365	183	92
73	37	19	10	409	205	103	52	737	369	185	93
81	41	21	11	417	209	105	53	745	373	187	94
89	45	23	12	425	213	107	54	753	377	189	95
97	49	25	13	433	217	109	55	761	381	191	96
105	53	27	14	441	221	111	56	769	385	193	97
113	57	29	15	449	225	113	57	777	389	195	98
121	61	31	16	457	229	115	58	785	393	197	99
129	65	33	17	465	233	117	59	793	397	199	100
137	69	35	18	473	237	119	60	801	401	201	101
145	73	37	19	481	241	121	61	809	405	203	102
153	77	39	20	489	245	123	62	817	409	205	103
161	81	41	21	497	249	125	63	825	413	207	104
169	85	43	22	505	253	127	64	833	417	209	105
177	89	45	23	513	257	129	65	841	421	211	106
185	93	47	24	521	261	131	66	849	425	213	107
193	97	49	25	529	265	133	67	857	429	215	108
201	101	51	26	537	269	135	68	865	433	217	109
209	105	53	27	545	273	137	69	873	437	219	110
217	109	55	28	553	277	139	70	881	441	221	111
225	113	57	29	561	281	141	71	889	445	223	112
233	117	59	30	569	285	143	72	897	449	225	113
241	121	61	31	577	289	145	73	905	453	227	114
249	125	63	32	585	293	147	74	913	457	229	115
257	129	65	33	593	297	149	75	921	461	231	116
265	133	67	34	601	301	151	76	929	465	233	117
273	137	69	35	609	305	153	77	937	469	235	118
281	141	71	36	617	309	155	78	945	473	237	119
289	145	73	37	625	313	157	79	953	477	239	120
297	149	75	38	633	317	159	80	961	481	241	121
305	153	77	39	641	321	161	81	969	485	243	122
313	157	79	40	649	325	163	82	977	489	245	123
321	161	81	41	657	329	165	83	985	493	247	124
329	165	83	42	665	333	167	84	993	497	249	125
337	169	85	43								

7.1.3 Overlapped Grids

With the grid-overlapping scheme, interpolation stencils are determined at the finest grid level only. Therefore, information would be missing at the coarser grid levels of the multigrid scheme. To avoid the need to obtain interpolation stencils at the coarser grid levels prior to the computation and the corresponding problems that most likely will arise, the multigrid scheme has been modified to accommodate grid-overlapping cases.[24]

First, if a fine grid boundary is supplied overlap information as the boundary condition, extrapolation is used for that boundary condition on all coarser grids. Second, for the restriction step of the multigrid scheme, all points are interpolated to coarser grids regardless of whether or not they are in a hole at the fine grid level. The flow variables in the hole are simply at free-stream conditions. Finally, for the prolongation step, only corrections on field points are used to update the finest mesh solution. The hole points are not updated since they will be overwritten with free-stream values when a Navier-Stokes calculation is made at each fine grid level. Likewise, the fringe points are not corrected since they will be updated with flow information from other grids at the same fine grid level.

Note that in Chapter 3 under "LT20 - Mesh Sequencing and Multigrid" on page 35, it is stated that the W-cycle is *not* recommended with overlapped grids. Therefore, set **ngam** = 1.

7.1.4 Embedded Grids

For time advancement, embedded grids basically become an extension of the multigrid scheme. Consider the grid system shown in Figure 7-6. The finest global grid has dimensions 9×9 and there are two coarser global grids for multigrid purposes. In addition, there is a 7×13 grid embedded in the finest global grid as shown in the figure. The multigrid (W-cycle) scheme would follow the course drawn in Figure 7-7.

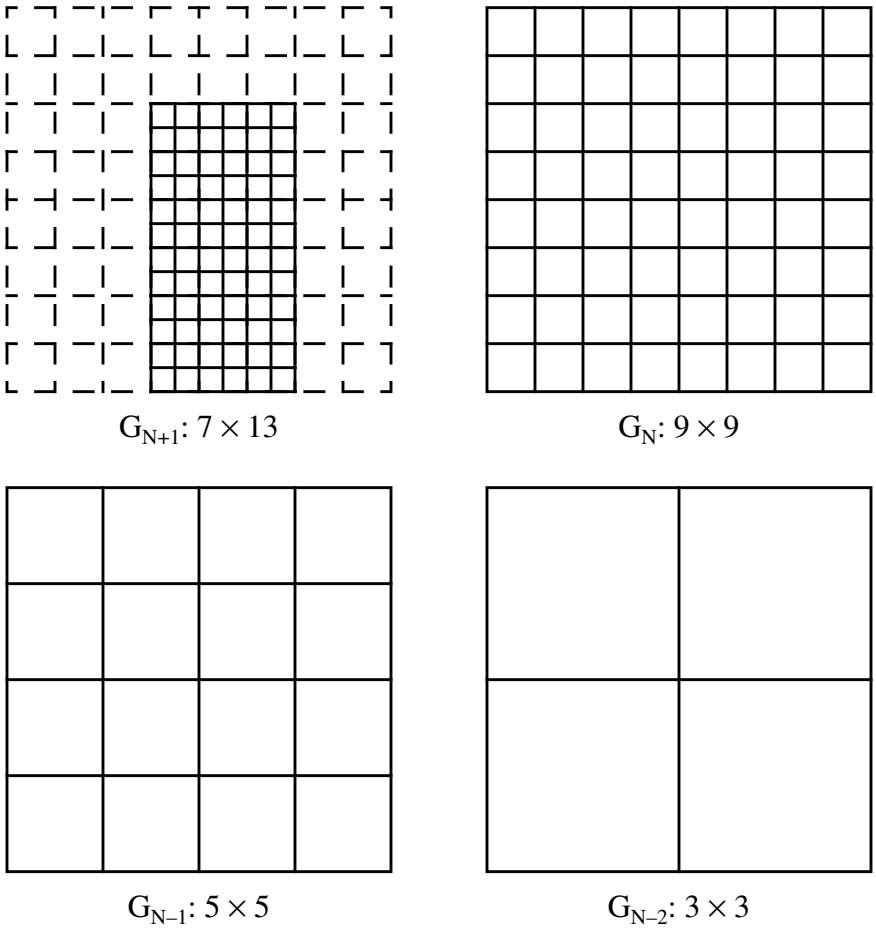


Figure 7-6. One embedded mesh in finest global grid.

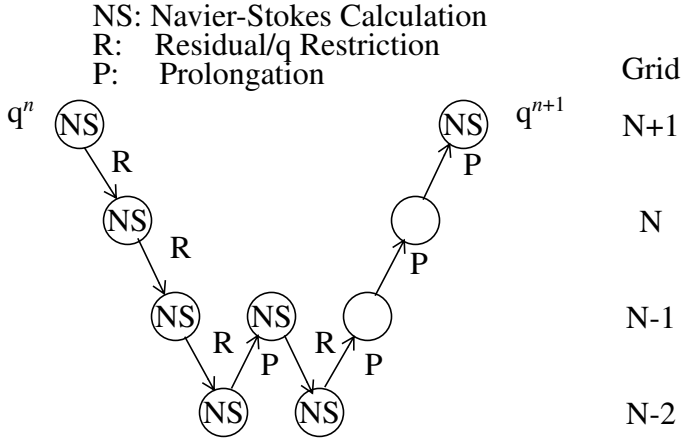


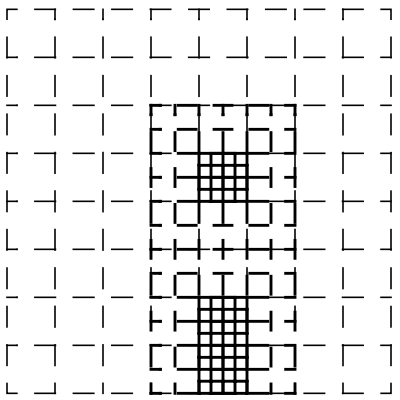
Figure 7-7. Multigrid W-cycle with one embedded grid level.

For a multigrid W-cycle, the pertinent lines of input for the example in Figure 7-6 would look something like:

Line Type

```
5       DT      IREST   IFLAGTS     FMAX     IUNST    CFLTAU
       -1.0       0         0        1.0        0      10.0
6     NGRID   NPLOT3D   NPRINT   NWREST      ICHK       I2D   NTSTEP        ITA
        2         0         0        0         0         1        1          1
7      NCG       IEM IADVANCE   IFORCE IVISC(I) IVISC(J) IVISC(K)
        2         0         0        1         0         0        0
        0         1         0        0         0         0        0
8     IDIM      JDIM      KDIM
        2         9         9
        2         7        13

10    INEWG    IGRIDC        IS       JS        KS        IE       JE        KE
        0         0         0        0         0         0        0         0
        0         1         1        4         1         2        7         7

20     MSEQ   MGFLAG    ICONSF      MTT      NGAM
        1         1         1        0         2

22     NCYC   MGLEVG     NEMGL    NITFO
       200         3         1        0
23     MIT1      MIT2      MIT3     MIT4      MIT5
        1         1         1        1         1
```

For the boundary conditions on the embedded grid, use the appropriate physical or connectivity **bctype** on any face that lies along an edge of its parent grid and **bctype** = 0 for any face that lies wholly within its parent grid. Suppose another embedded grid level is added to the grid system as in Figure 7-8. In the figure two grids are embedded in the first embedded grid. Note that only one embedded grid *level* is added by their addition.



G_{N+2}: 5×9 and 5×5

Figure 7-8. Two embedded grids within an embedded grid.

The corresponding input would resemble:

Line Type

```
5       DT      IREST   IFLAGTS      FMAX     IUNST    CFLTAU
       -1.0         1         0       1.0         0      10.0
6     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
          4         0         0         0         0         1         1         1
7       NCG       IEM IADVANCE    IFORCE IVISC(I) IVISC(J) IVISC(K)
          2         0         0         1         0         0         0
          0         1         0         0         0         0         0
          0         2         0         0         0         0         0
          0         2         0         0         0         0         0
8      IDIM      JDIM      KDIM
          2         9         9
          2         7        13
          2         5         9
          2         5         5

10    INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
          0         0         0         0         0         0         0         0
          0         1         1         4         1         2         7         7
          1         2         1         3         1         2         5         5
          1         2         1         3         9         2         5        11

20     MSEQ    MGFLAG    ICONSF       MTT      NGAM
          1         1         1         0         2

22     NCYC    MGLEVG     NEMGL     NITFO
         200         3         2         0
23     MIT1      MIT2      MIT3      MIT4      MIT5
          1         1         1         1         1
```

Note that **inewg** should be set to 0 for grids 3 and 4 if this case is restarted.

Multigrid can also be performed at the embedded grid levels themselves by setting **mgflag** = 2. The embedded grid uses the coarser grid levels in which it is embedded, so **ncg** for the embedded grid itself should remain 0. The input for this case would resemble:

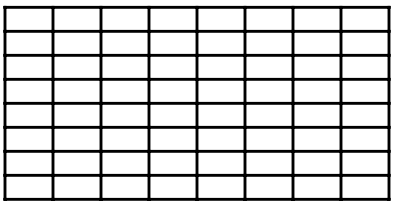
Line Type

```
5       DT      IREST   IFLAGTS      FMAX     IUNST    CFLTAU
       -1.0         0         0       1.0         0      10.0
6     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
          2         0         0         0         0         1         1         1
7       NCG       IEM IADVANCE    IFORCE IVISC(I) IVISC(J) IVISC(K)
          2         0         0         1         0         0         0
          0         1         0         0         0         0         0
8      IDIM      JDIM      KDIM
          2         9         9
          2         7        13

10    INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
          0         0         0         0         0         0         0         0
          0         1         1         4         1         2         7         7

20     MSEQ    MGFLAG    ICONSF       MTT      NGAM
          1         2         1         0         2

22     NCYC    MGLEVG     NEMGL     NITFO
         200         3         1         0
23     MIT1      MIT2      MIT3      MIT4      MIT5
          1         1         1         1         1
```

7.2 Mesh Sequencing

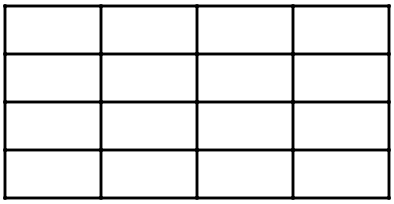
When setting up a CFD problem, initial conditions are set at every point on a grid. Usually, the closer the guess is to the final solution the quicker the case will converge. In CFL3D, the initial conditions for a problem are set at free-stream conditions for a single-grid-level case. However, if mesh sequencing is utilized, a better "guess" can be made for the initial conditions on the finer grid where the computations are most expensive.

Suppose a solution is desired on the 9×9 grid depicted in Figure 7-9. Instead of starting the solution with free-stream conditions, a solution could first be obtained on the 5×5 grid shown in Figure 7-10. While the solution on the 9×9 grid would be more accurate than that on the 5×5 grid, the coarser grid's solution would be closer to the fine grid solution than free-stream conditions. Since the coarse grid solution can be computed more quickly than the fine grid solution, it is usually beneficial to use the coarse grid solution as the initial condition for the fine grid.



9×9

Figure 7-9. Simple grid example.



5×5

Figure 7-10. Mesh sequencing sample grid.

There are two ways to implement mesh sequencing in CFL3D. The first way is to completely converge the solution on the coarse grid before mapping it up to the fine grid. For the sample grids, the pertinent input would look like:

Line Type

```
5          DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
         -1.0         0         0       1.0         0      10.0
6       NGRID   NPLOT3D    NPRINT    NWREST      ICHK      I2D   NTSTEP       ITA
```

```
          1        0        0        0        0        1        1        1
7       NCG      IEM IADVANCE   IFORCE IVISC(I) IVISC(J) IVISC(K)
          2        0        0        1        0        0        0
8      IDIM     JDIM     KDIM
          2        9        9

20     MSEQ   MGFLAG   ICONSF      MTT     NGAM
          2        1        0        0        1

22     NCYC   MGLEVG    NEMGL    NITFO
         200        2        0        0
           0        3        0        0
23     MIT1     MIT2     MIT3     MIT4     MIT5
          1        1        1        1        1
          1        1        1        1        1
```

Note that multigrid is also employed. The coarse grid solution can be examined and, if the solution on the coarse grid is not converged, the case can be restarted on that grid. After the coarse grid solution is converged, the solution is mapped to the fine grid and the calculations continue at the fine grid level. The pertinent input for this step would look something like:

Line Type

```
5         DT    IREST  IFLAGTS     FMAX    IUNST   CFLTAU
        -1.0        1        0      1.0        0     10.0
6      NGRID  NPLOT3D   NPRINT   NWREST     ICHK      I2D   NTSTEP      ITA
          1        0        0        0        0        1        1        1
7       NCG      IEM IADVANCE   IFORCE IVISC(I) IVISC(J) IVISC(K)
          2        0        0        1        0        0        0
8      IDIM     JDIM     KDIM
          2        9        9

20     MSEQ   MGFLAG   ICONSF      MTT     NGAM
          2        1        0        0        1

22     NCYC   MGLEVG    NEMGL    NITFO
           1        2        0        0
         200        2        0        0
23     MIT1     MIT2     MIT3     MIT4     MIT5
          1        1        1        1        1
          1        1        1        1        1
```

Here, one iteration is performed on the coarse grid and two-hundred iterations are performed on the fine grid. At the end of this run, only the fine grid solution is available for examination. (So be sure to save the coarse grid solution *before* this step if it will be needed later for grid refinement studies, etc.) The case can be resubmitted for additional computations on the fine grid with the following input:

Line Type

```
5         DT    IREST  IFLAGTS     FMAX    IUNST   CFLTAU
        -1.0        1        0      1.0        0     10.0
6      NGRID  NPLOT3D   NPRINT   NWREST     ICHK      I2D   NTSTEP      ITA
          1        0        0        0        0        1        1        1
7       NCG      IEM IADVANCE   IFORCE IVISC(I) IVISC(J) IVISC(K)
          2        0        0        1        0        0        0
8      IDIM     JDIM     KDIM
          2        9        9

20     MSEQ   MGFLAG   ICONSF      MTT     NGAM
```

```
             1         1         0         0         1

22       NCYC    MGLEVG     NEMGL     NITFO
          200         3         0         0
23       MIT1      MIT2      MIT3      MIT4      MIT5
             1         1         1         1         1
```

This input could be used until satisfactory convergence has been achieved.

Even a partially-converged coarse grid solution is typically a better initial guess than free-stream conditions. Therefore, the second way to set up the mesh sequencing assumes that a set number of iterations at the coarse grid level will be adequate to improve the convergence at the fine grid level. If this is the case, the coarse grid solution is mapped to the fine grid on the first submittal. For example:

Line Type

```
5          DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
          -1.0         0         0       1.0         0      10.0
6       NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
             1         0         0         0         0         1         1         1
7        NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
             2         0         0         1         0         0         0
8       IDIM      JDIM      KDIM
             2         9         9

20       MSEQ    MGFLAG    ICONSF       MTT      NGAM
             2         1         0         0         1

22       NCYC    MGLEVG     NEMGL     NITFO
          200         2         0         0
          200         3         0         0
23       MIT1      MIT2      MIT3      MIT4      MIT5
             1         1         1         1         1
             1         1         1         1         1
```

When an input like the one above is used, the coarse grid solution is not available for examination before proceeding to the fine grid. In addition, the restart file will contain the solution for the fine grid only. This is important to remember when a grid refinement study is being conducted (in which case the first approach to mesh sequencing would be beneficial). To restart from a run with the above input, use something like:

Line Type

```
5          DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
          -1.0         0         0       1.0         0      10.0
6       NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
             1         0         0         0         0         1         1         1
7        NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
             2         0         0         1         0         0         0
8       IDIM      JDIM      KDIM
             2         9         9

20       MSEQ    MGFLAG    ICONSF       MTT      NGAM
             1         1         0         0         1

22       NCYC    MGLEVG     NEMGL     NITFO
          200         3         0         0
23       MIT1      MIT2      MIT3      MIT4      MIT5
             1         1         1         1         1
```

One last note about mesh sequencing is to emphasize, as with multigrid, the advantage of "good" grid dimensions. See "A Word About Grid Dimensions" on page 127. Planning for mesh sequencing should be made at the grid generation step of the CFD problem.

CHAPTER 8

Time-Accurate Computations

Although most computations being performed today are for steady-state cases, it appears as though CFD will be used more and more for unsteady, time-accurate cases in the future. Therefore, since this realm is still somewhat in its infancy, this chapter has been written in order to explore in detail what is currently known about the subject, relative to CFL3D's capabilities.

CFL3D has been used extensively for time-accurate computations (**dt** > 0). See, for example, reference 33. Two types of sub-iterations, called "t-TS" and "τ-TS" are currently implemented in the code and are also described in this reference. This chapter describes the effects of the different types of sub-iterations, as well as the strategy for pursuing time-accurate computations in general.

When performing steady-state computations, the primary numerical accuracy issue about which the user needs to be concerned is that of spatial accuracy. Generally, the user runs a problem on a series of successively finer grids. As long as the solutions are fully converged on each grid, the user can get a clear picture of the accuracy of a solution on a given grid. In fact, the user can determine the numerical global order of accuracy by using a series of at least three grids *from the same family* and plotting some global quantity of interest as a function of a measure of the average grid spacing, such as $1/(\sqrt{\text{number of grid points}})$ for 2-d or $1/(\sqrt[3]{\text{number of grid points}})$ for 3-d, on a log-log plot. The slope of the plotted line represents the spatial order of accuracy of the scheme. When the standard $\kappa = -1/3$ scheme is employed, CFL3D has been demonstrated in the past to be globally approximately second-order accurate for most grids. However, this is problem-dependent and the accuracy can degrade somewhat on grids that are too coarse or on grids with extremely severe stretching.

For time-accurate problems, temporal accuracy becomes an additional numerical accuracy issue of concern for the user. Now, not only does the effect of the grid need to be assessed for each problem, but the effect of the time step as well. Additionally, because CFL3D is an implicit code and employs approximate factorization, linearization and factorization errors are introduced during each time step, which can degrade the accuracy of the time-accurate simulation. (In fact, if no sub-iterations are employed, the best that can possibly be hoped for is first-order temporal accuracy.) This is why sub-iterations are generally recommended for time-accurate computations. Sub-iterations "iterate away" the linearization and factorization errors. The more sub-iterations performed, the more accurate the simulation. But how many sub-iterations are enough? And which type of sub-iteration scheme works the best? Hopefully, this chapter will help to answer these questions.

8.1 General Effects of Numerical Parameters

A graphic representing the general effects of sub-iterations, time step, and grid is shown in Figure 8-1. Here, some "quantity of interest" is shown as a function of time step. For example, the quantity could represent Strouhal number for an unsteady circular-cylinder case. Say that the user ran a series of computations on a given grid, using three sub-iterations. Each successive computation used a smaller and smaller time step and each was completely converged to periodic quasi-steady-state. If the user plotted a global quantity of interest as a function of time step, a curve like the lower-most curve in the figure might be obtained. (Note that the figure shows the quantity of interest increasing with decreasing time step, but the trend could also be in the opposite direction, depending on the case and the quantity of interest chosen.)

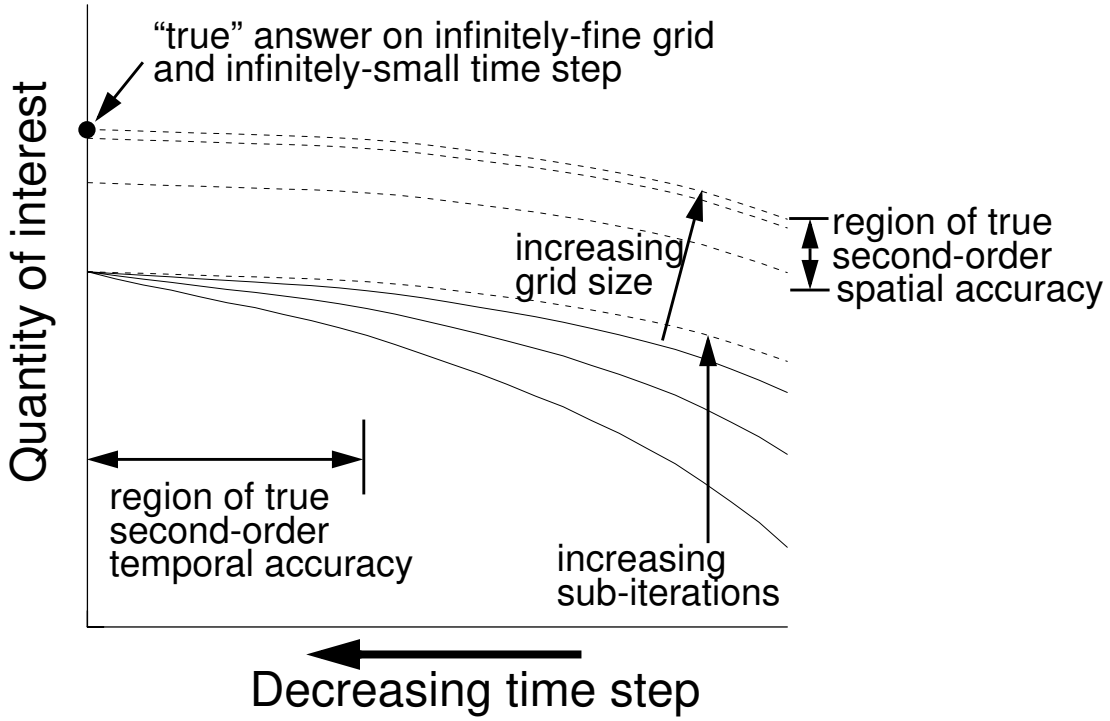


Figure 8-1. Time accuracy trends.

Now say that the user repeated the entire series of computations, except this time using six sub-iterations instead of three. A curve similar to the second curve from the bottom of the figure might be obtained. Finally, with an infinite number of sub-iterations per time step, the user would obtain the dashed curve, which represents the best possible solution *on that grid*. Note that, using a given time step, an increasing number of sub-iterations yields an increasingly better answer, but even the best answer (with infinite sub-iterations) is still in error from the answer using an infinitely-small time step.

The dashed line should behave either first or second order accurate in time (depending on the accuracy input by the user in the input file), but only for sufficiently small time

steps. This is what is graphically represented by the left horizontal arrow in the figure. If the time step is too large, the solution may not exhibit the expected temporal accuracy.

It is evident from this figure that, if an extremely small time step is taken, then sub-iterations are not as beneficial as when larger time steps are taken. The solution is already pretty good. Hence the user needs to make a trade-off between accuracy and efficiency. An extremely small time step can be taken, but at a greater cost, or a larger time step can be taken with some degradation in accuracy. And, sub-iterations may or may not contribute much toward improving accuracy, depending on the time step. Also, at one time step a certain number of sub-iterations may be enough, but at a different time step that same number may be either insufficient or overkill.

The effect of the grid size is also represented in Figure 8-1. On finer and finer grids, the location of the dashed curve will change, approaching the "true" answer on an infinitely-fine grid. (Note that the figure shows the quantity of interest increasing with finer grids, but it could also go the opposite direction, depending on the case and the quantity of interest chosen.) The standard $\kappa = -1/3$ CFL3D scheme should generally behave spatially second-order accurate on sufficiently fine grids. The filled-in circle in the figure represents the "true" answer on an infinitely fine grid with an infinitely small time step. Obviously, the user would generally like to get as close to this answer as possible, but with a reasonable expenditure of resources.

8.2 Effect of Sub-iterations With Time Step and Grid Size

The effect of sub-iterations with time step and grid size is explored in this section for a sample test case of laminar flow over a circular cylinder at Reynolds number 1200 and $M_\infty = 0.2$. The input file, for a relatively fine O-grid with t-TS multigrid sub-iterations, is given here:

```
cylnew.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    circ cylinder
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
    0.2000    00.000       0.0    0.0012     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
   +0.1000         0       000   05.0000         0        5.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         1      6100         0         1      0100        +2
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
```

```
         2         0         0         1         0         1         1
      IDIM      JDIM      KDIM
         2       193        97
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         0         0         0         0         0         0
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         0         0         0
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1         0         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1         0         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      2004         0         0         0         0         2
             TWTYPE        CQ
                 0.        0.
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         0         0        01
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
        10        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7     MIT8
        01        01        01        01        01         1         1        1
  1-1 BLOCKING DATA:
      NBLI
         1
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1            1      1      1      2      1     97      1      3
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1            1    193      1      2    193     97      1      3
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      1     01      1     01    999      1      1    999      1
  MOVIE
      0
  PRINT OUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      1     01      1     01    999      1      1    999      1
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

The initial study is to investigate the effect of number and type of sub-iterations on the sub-iteration convergence of residual and drag. These levels are printed out automatically to the file `cfl3d.subit_res` (unit 23). (See Section 5.2.2.)

For this study, the following runs were performed (the coarser grid consists of every other point from the fine grid):

Grid	Δt	Sub-iteration Type
97×49	0.02	t-TS, multigrid
97×49	0.02	t-TS, no multigrid
97×49	0.02	τ-TS, multigrid
97×49	0.02	τ-TS, no multigrid
97×49	0.10	t-TS, multigrid
97×49	0.10	t-TS, no multigrid
97×49	0.10	τ-TS, multigrid
97×49	0.10	τ-TS, no multigrid
97×49	0.50	t-TS, multigrid
97×49	0.50	t-TS, no multigrid
97×49	0.50	τ-TS, multigrid
97×49	0.50	τ-TS, no multigrid
193×97	0.10	t-TS, multigrid
193×97	0.10	t-TS, no multigrid
193×97	0.10	τ-TS, multigrid
193×97	0.10	τ-TS, no multigrid

When multigrid was employed, a 3-level V-cycle was used.

Figure 8-2 shows the residual for density and the drag coefficient as a function of **ncyc** (number of sub-iterations +1) at a time step of 0.02. This is a fairly fine time step, yielding over 1000 steps per period. At this time step, both the t-TS and τ-TS with multigrid converge the drag with **ncyc** = 4 (3 sub-iterations) and t-TS and τ-TS without multigrid require about **ncyc** = 6 (5 sub-iterations). The residual also converges quicker with multigrid, as expected. Note that the t-TS with multigrid has a slightly better residual convergence rate than τ-TS with multigrid at this time step.

At a higher time step of **dt** = 0.10, the trends in Figure 8-3 are similar: both t-TS and τ-TS with multigrid converge the quickest, requiring about **ncyc** = 5, while the non-multigrid methods require about **ncyc** = 14-16. At this time step, the residual for τ-TS with multigrid converges slightly better than t-TS with multigrid. This time step corresponds to a little over 200 steps per period.

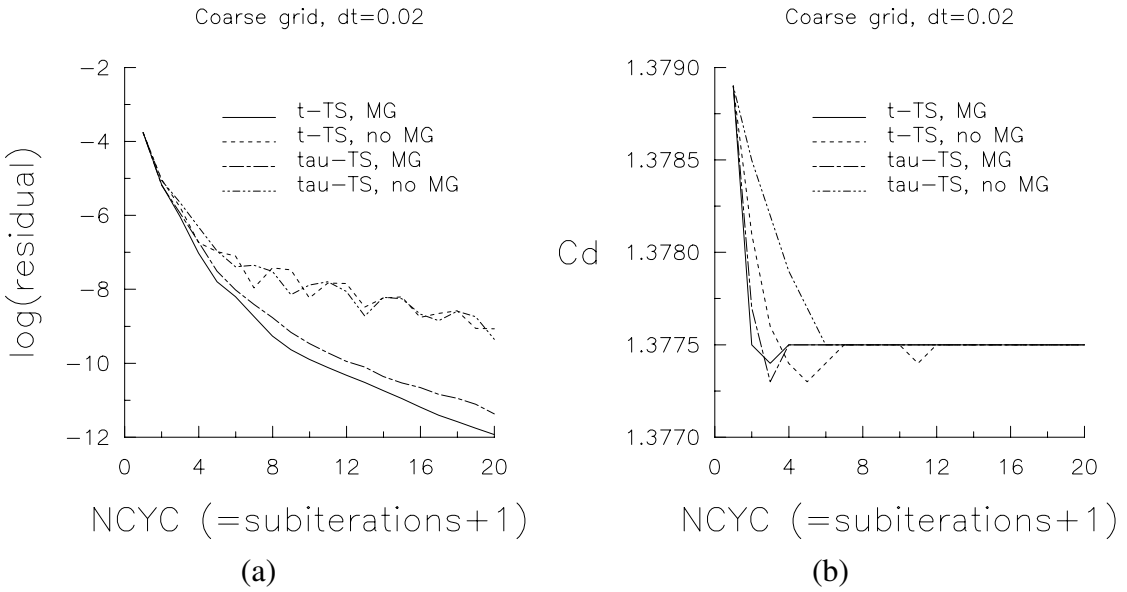


Figure 8-2. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.02$.

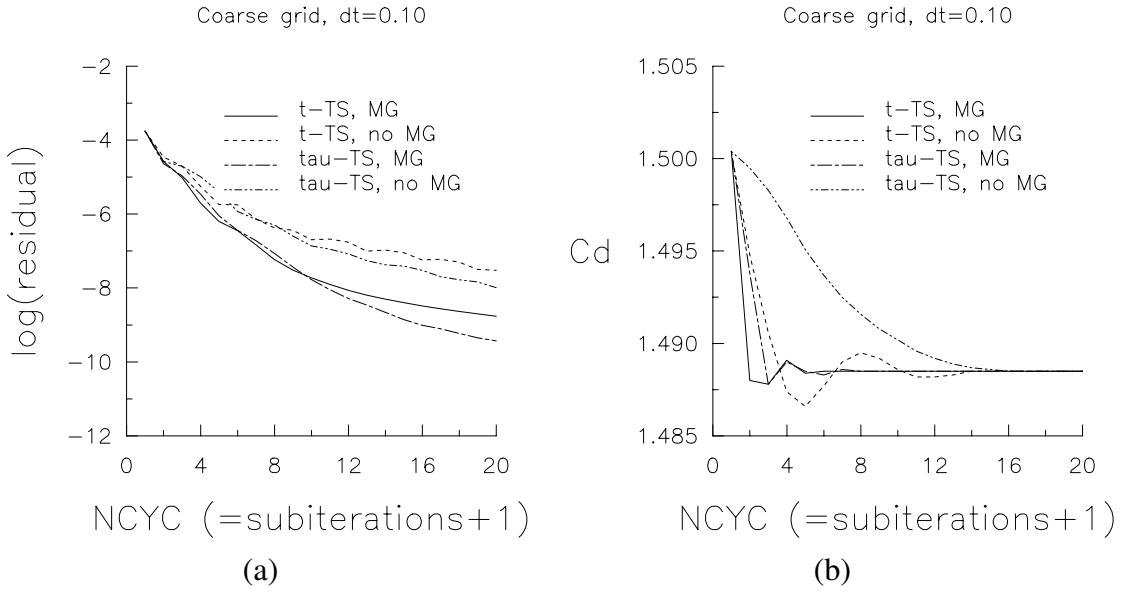


Figure 8-3. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.10$.

At the highest time step of **dt** = 0.5, Figure 8-4, the multigrid sub-iterations require about **ncyc** = 12, while the non-multigrid sub-iterations are not fully converged even after **ncyc** = 40. Also note from Figure 8-4(a), at this time step the residual for t-TS with multigrid method now does not converge as well as either τ-TS method. This time step corresponds with less than 50 steps per period.

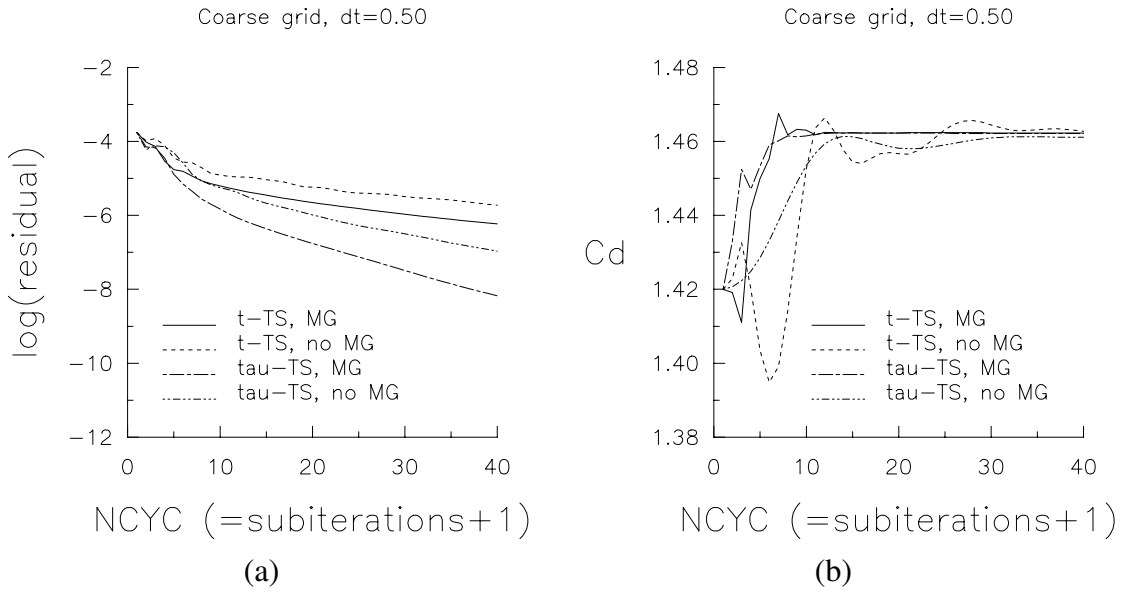


Figure 8-4. Coarse grid residual and drag coefficient histories for a single time step of $\Delta t = 0.50$.

Results for the fine 193×97 grid with **dt** = 0.1 are shown in Figure 8-5. Results are qualitatively similar to those in Figure 8-3. However, on this finer grid, more sub-iterations are required to converge the drag: 6-8 iterations for multigrid and well over 20 iterations for non-multigrid.

With a 3-level V-cycle, the multigrid method costs roughly 1.5 times as much as the non-multigrid method. (The τ-TS is only marginally more expensive than t-TS, so they may be considered essentially equivalent.) Hence, at the smallest time step of **dt** = 0.02, the user is roughly at a break-even point in terms of whether it is more efficient to use multigrid or no multigrid. However, at the larger time steps, using multigrid is clearly beneficial: for example, at **dt** = 0.1 (around 200 steps per cycle), multigrid converges the drag in approximately 0.36 the number of sub-iterations on the coarse grid at 1.4 times the cost; this means a savings of almost 50%! The savings is even greater on the fine grid. Due to this substantial savings, the remaining results will include only cases utilizing multigrid.

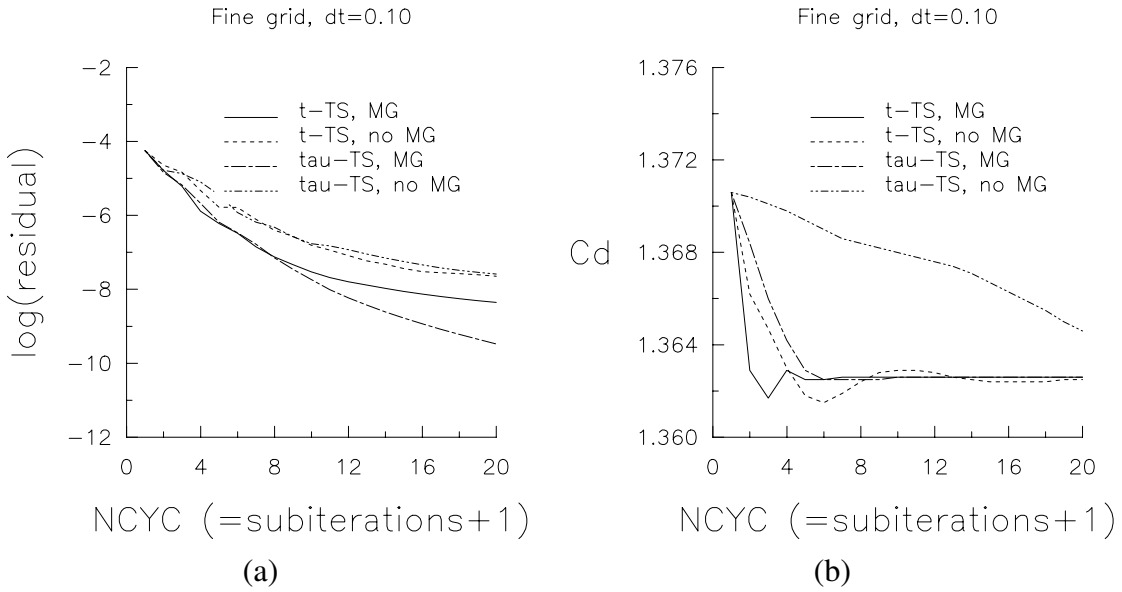


Figure 8-5. Fine grid residual and drag coefficient histories for a single time step of $\Delta t = 0.10$.

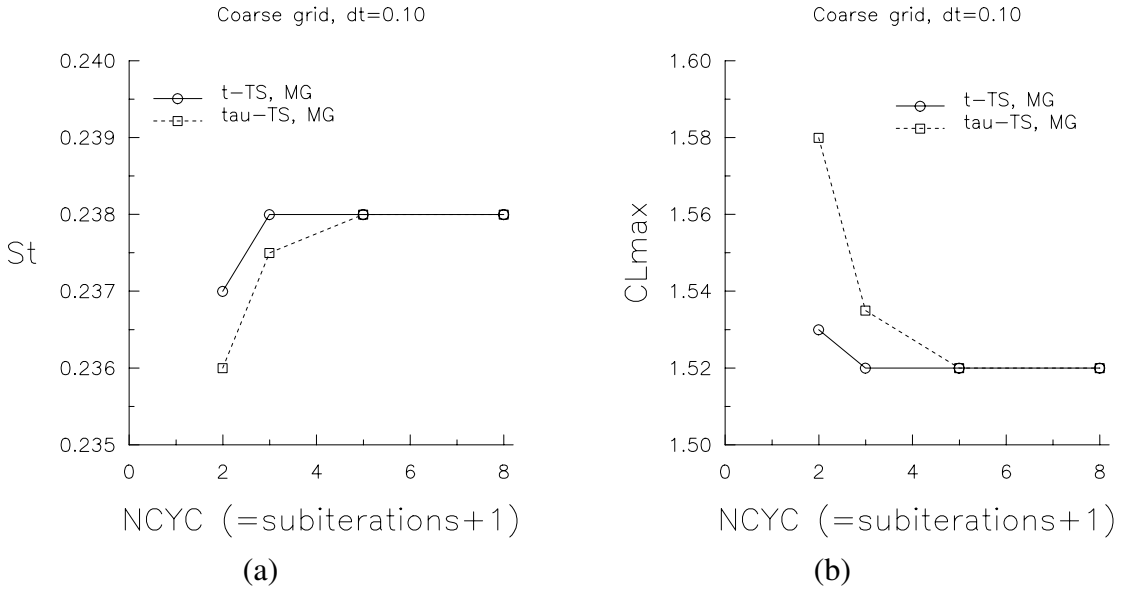


Figure 8-6. Coarse grid Strouhal number and lift coefficient histories for fully periodic solutions using $\Delta t = 0.10$.

Figure 8-2 through Figure 8-5 give a feel for the effect of sub-iterations on the convergence to the next physical time step *for one iteration*, but what is the effect of the number of sub-iterations on *global* quantities, over a long period of time? Figure 8-6 shows Strouhal number and maximum lift coefficient (in absolute value) as a function of **ncyc** for **dt** = 0.10 on the coarse grid, where the solution is obtained using the given value of **ncyc** over a long time (until periodic quasi-steady-state is reached). If at least 4 sub-iterations are run for this case (**ncyc** = 5), both t-TS and τ-TS with multigrid converge to the same result. This is consistent with the results for maximum lift coefficient shown in Figure 8-3(b). However, if less than this number of sub-iterations is run, then the t-TS method appears to give the better result.

For the higher time step of **dt** = 0.5, results are shown in Figure 8-7. If **ncyc** is less than about 7 for this time step, both t-TS and τ-TS sub-iterations with multigrid yield non-physical solutions (not shown in the figures). For example, t-TS yields a highly non-regular lift cycle, while τ-TS yields a regular lift cycle with non-zero mean. At least **ncyc** = 15-20 is required to converge the sub-iterative schemes sufficiently at this time step. This is roughly consistent with the results in Figure 8-4(b). If less than this number of sub-iterations is used, then the τ-TS method gives the better result. This is the opposite result from that given above for **dt** = 0.1, but it is consistent with the trend seen in the residual plots of Figure 8-2(a), Figure 8-3(a), and Figure 8-4(a). In other words, it appears that the t-TS method may require less sub-iterations at low time steps, while τ-TS requires less sub-iterations at higher time steps.

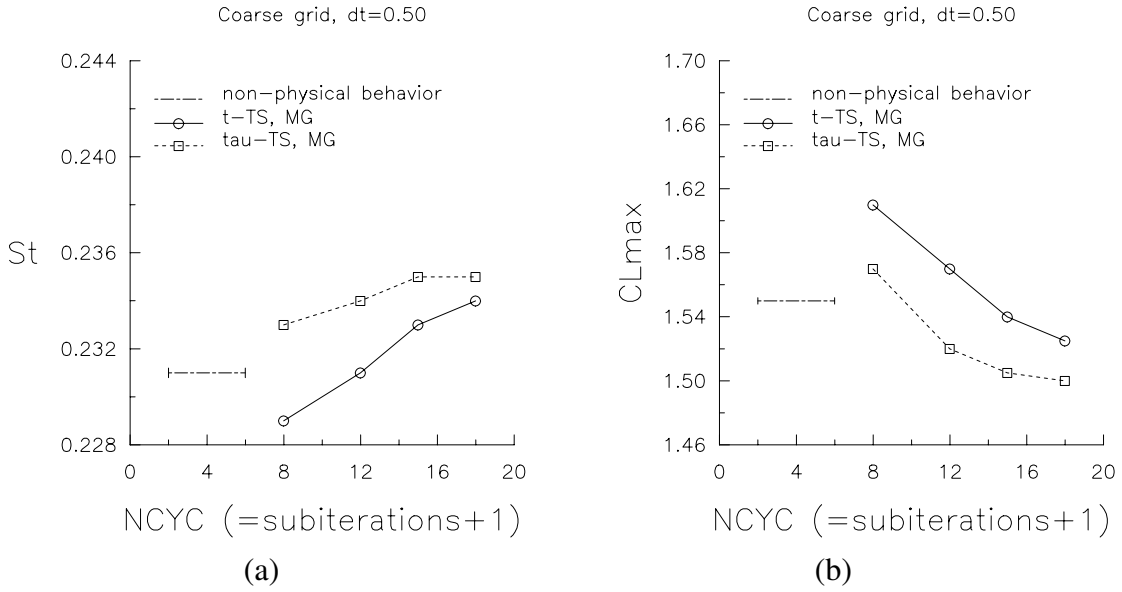


Figure 8-7. Coarse grid Strouhal number and lift coefficient histories for fully periodic solutions using $\Delta t = 0.50$.

Other time-accurate circular cylinder cases with turbulence model(s) employed (not shown), have revealed similar trends to the study shown here. One point worth mentioning in connection with these other cases is that, in some instances, the t-TS method with multigrid has been seen to either diverge or else give nonphysical answers, *regardless of the number of sub-iterations taken*, when the time step is too large. Unfortunately, determining what time step is too large remains elusive at this point. Trial and error seems to be the only way to determine it for such cases. Therefore, since the τ-TS method has not exhibited such errant behavior, it appears that, as a general rule, the safest bet is to go with the τ-TS method rather than t-TS.

From this study, combined with experience running the CFL3D code for time-accurate cases, the following conclusions are made:

1. In general, it is recommended that the user employ multigrid when using sub-iterations.
2. The larger the time step (the less steps per period), the more sub-iterations are required to converge the sub-iterative scheme.
3. The larger the grid, the more sub-iterations are required to converge the sub-iterative scheme.
4. t-TS and τ-TS are roughly equivalent in their ability to converge a quantity like "drag". However, for lowering residual, t-TS is slightly more efficient than τ-TS at small time steps, while the reverse is true at higher time steps.
5. t-TS appears to require slightly less sub-iterations at low time steps, while τ-TS appears to require slightly less sub-iterations at higher time steps.
6. Since it is not possible to know in advance what time step is "low" and what is "high" (in connection with conclusions 4. and 5.) and since t-TS has been known to *not converge* regardless of the number of sub-iterations for some cases when the time step is too high, it is recommended that τ-TS (with multigrid) be used in practice as a general rule.

Also, the following recommendation is made. When performing time-accurate computations, *always* monitor the `cfl3d.subit_res` file (unit 23) in order to insure that the sub-iterative scheme is converging sufficiently. Additionally, it is recommended that the user perform a time step study (vary **dt**), along with the usual grid density study, to determine the solution's sensitivity to time step.

8.3 Convergence Criterion for Sub-iterations

The sub-iteration equation (Equation (B-7), with $\phi' = 0$) is

$$\left[\left(\frac{1}{J\Delta\tau} + \frac{1+\phi}{J\Delta t}\right)I + \delta_\xi \mathbf{A} + \delta_\eta \mathbf{B} + \delta_\zeta \mathbf{C}\right]\Delta\mathbf{Q}^m = \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t} - \frac{(1+\phi)(\mathbf{Q}^m - \mathbf{Q}^n)}{J\Delta t} + R(\mathbf{Q}^m) \tag{8-1}$$

for $m \to \infty$, $\mathbf{Q}^m \to \mathbf{Q}^{n+1}$, and the left-hand side, which is the sub-iteration residual, $R_{subit}(\mathbf{Q}^m)$, approaches 0. Write the sub-iteration residual as

$$R_{subit}(\mathbf{Q}^m) = \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t} - \frac{(1+\phi)(\mathbf{Q}^m - \mathbf{Q}^n)}{J\Delta t} + R(\mathbf{Q}^m) \tag{8-2}$$

For $m \to \infty$,

$$R_{subit}(\mathbf{Q}^m) \sim \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t} - \frac{(1+\phi)(\mathbf{Q}^{n+1} - \mathbf{Q}^n)}{J\Delta t} + R(\mathbf{Q}^m) \tag{8-3}$$

where $\frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t} - \frac{(1+\phi)(\mathbf{Q}^{n+1} - \mathbf{Q}^n)}{J\Delta t} \Rightarrow \text{discrete} - \frac{\partial\mathbf{Q}}{\partial t}$.

Then

$$\frac{\partial\mathbf{Q}^n}{\partial t} = R(\mathbf{Q}^m) - R_{subit}(\mathbf{Q}^m) \tag{8-4}$$

where $\frac{\partial\mathbf{Q}^n}{\partial t} = R(\mathbf{Q}^m) \Rightarrow$ discrete Navier-Stokes equations. Therefore, to insure that the discrete Navier-Stokes equations are solved accurately, $R_{subit}(\mathbf{Q}^m) \ll R(\mathbf{Q}^m)$ are needed. These values are calculated in the code. $R_{subit}(\mathbf{Q}^m)$ (for $\mathbf{Q}$ = density) is output to `cfl3d_subit.res`, while $R(\mathbf{Q}^m)$ (for $\mathbf{Q}$ = density) is output to the user-specified file on unit 12 (typically called `cfl3d.res` in the sample input files).

CHAPTER 9

Test Cases

CFL3D provides multiple options for solving CFD problems. A variety of 0-equation, 1-equation, and 2-equation turbulence models are available. The code has static or dynamic mesh capabilities. If the grid has multiple zones, there are several choices for communication between the zones which can be used independently or in conjunction with one another. For convergence acceleration, multigrid and mesh sequencing are available.

The test cases described in this chapter provide a sampling of CFL3D's capabilities. After studying the test cases, the user will hopefully be able to choose the best strategy for his or her particular applications. For information on how to obtain the files needed for the test cases see "Acquiring the Code and Example Files" on page 7.

Several two-dimensional test cases discussed in this chapter involve airfoils and flat plates. The use of a single block is exemplified with a RAE 2822 airfoil case. A NACA 0012 airfoil case is used as an example for both grid patching and grid overlapping. Also included is a multielement airfoil case, involving grid overlapping. The flat plate examples include a turbulent flat plate case and a vibrating flat plate case which illustrates the dynamic mesh capabilities of CFL3D. Also included are a multistream nozzle case and a rotor-stator case.

One three-dimensional example is for an axisymmetric bump. By taking advantage of periodicity, it is solved on a grid with only two planes in the circumferential direction. Three of the three-dimensional examples are for wing topologies. A single block case is set up for an F-5 wing. A case solving for the viscous flow over the Onera M-6 wing is also set up using a single block. A delta wing case with laminar flow is also available. Keep in mind that, in order to have cases that are "quick" to run, the three-dimensional grids used in some of these examples are relatively coarse compared to what one *should* use to adequately resolve the flow.

Note: you may see slight differences in your results, due to errors in CFL3D that have been corrected since the plots in this chapter were generated.

9.1 Two-dimensional Test Cases

CFL3D solves for the primitive variables at the *cell centers* of a grid. Therefore, for two-dimensional cases, *two* grid planes are needed for one plane of cell-center points to exist. The "2-d direction" is the i direction designated by setting **idim** = 2 (and **i2d** = 1). Typically, after a 2-d grid is generated, it is simply duplicated such that identical planes

exist at $i = 1$ and $i = 2$ with a constant value in the third direction. For example, if x is the third direction, x is typically set to 0.0 at $i = 1$ and x might equal 1.0 or –1.0 at $i = 2$. When setting up the third direction by duplicating the grid plane, keep in mind that the right-hand rule *must* be satisfied. See "The Right-Hand Rule" on page 67. Also note that, while this step *is* doubling the number of grid points, only *one* plane of data is actually computed. Therefore, the number of points in *one* plane of the grid should be used when estimating the time required to run the code.

9.1.1 RAE 2822 Airfoil

This test case solves for the viscous flow over the RAE 2822 airfoil at $\alpha = 2.72°$ with $M_\infty = 0.75$. These are corrected conditions from Case 10 of Cook et al.[15] The grid consists of a single zone with 24929 points in one plane. Menter's $k - \omega$ SST model is used to solve the turbulent flow with a Reynolds number of 6.2 million. The memory requirement for this example is 5.7 million words. A typical timing for this case is 382 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of June 1996). A close-up of the grid near the airfoil is shown in Figure 9-1.

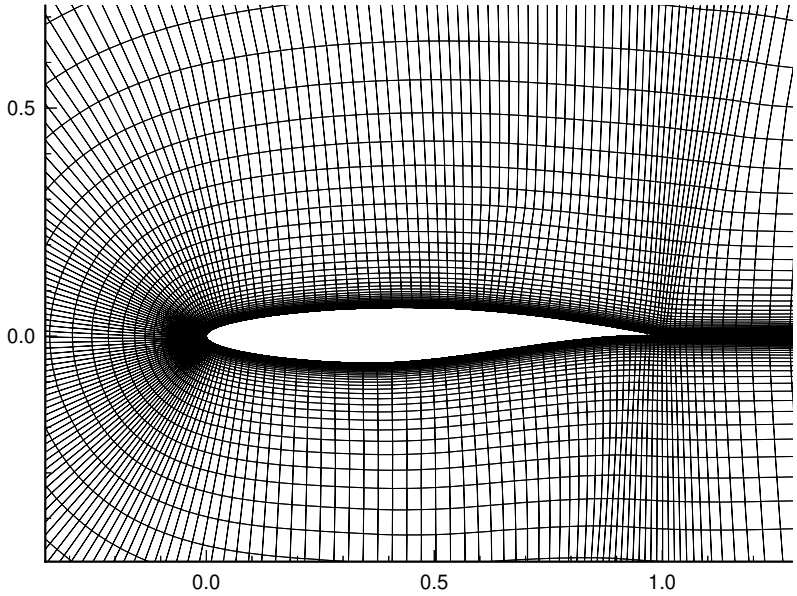


Figure 9-1. Single zone RAE 2822 airfoil grid.

Besides the CFL3D code, the following files are needed to run this test case:

File	Description
`rae10.inp`	input for CFL3D
`rae10.grd`	formatted single plane grid
`grid2dto3d.f`	converter for creating 2 grid planes

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 grid2dto3d.f
```

Step 2

Link the grid converter object file:

```
segldr -o grid2dto3d grid2dto3d.o
```

Step 3

Run the grid converter program (the binary file `rae10.bin` will be output):

```
grid2dto3d
```

In answer to the questions, type:

```
rae10.grd
rae10.bin
2
0
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < rae10.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < rae10.inp
```

The input file for this case is:

```
FILES:
rae10.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    RAE case 10, with SST model
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
    0.7500    02.720       0.0    6.2000     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
   -5.0000         0       000   05.0000         0      10.0
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         1      6100         0         1      0001         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0         1         0         0         7
      IDIM      JDIM      KDIM
         2       257        97
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         1         2        88       159         1        97
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         3         3         3
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         3         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1         0         0         0         1        41         0
         1         2      2004         0         0        41       217         2
             TWTYPE        CQ
                 0.        0.
         1         3         0         0         0       217       257         0
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         0         0        02
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
```

```
     500        03        00       000
    MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7      MIT8
      01        01        01        01        01         1         1         1
 1-1 BLOCKING DATA:
    NBLI
       1
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1      1      1      2     41      1      1      2
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1    257      1      2    217      1      1      2
 PATCH SURFACE DATA:
  NINTER
       0
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1     01      1     01    999      1      1    999      1
 MOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1     01      1     41    217      1      1      1      1
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After running this test case a result such as that shown in Figure 9-2 should be obtained. In the figure, surface pressure coefficients are plotted along with experimental data for this case. The computational surface pressures can be obtained from file `cfl3d.prout`. Experimental surface pressure coefficients from Cook et. al[15] are included with this test case for comparison purposes. The file is called `rae10.cpexp`. The residual plots shown in Figure 9-3 should also be duplicated. These convergence histories can be found in `cfl3d.res`.

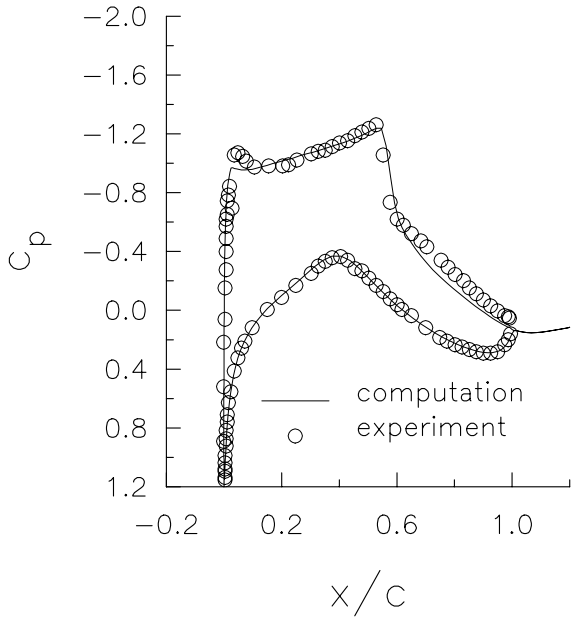


Figure 9-2. Surface pressure coefficients for RAE 2822 airfoil; $\alpha = 2.72°$, $M_\infty = 0.75$, $Re_{\tilde{L}_R} = 6.2 \times 10^6$.

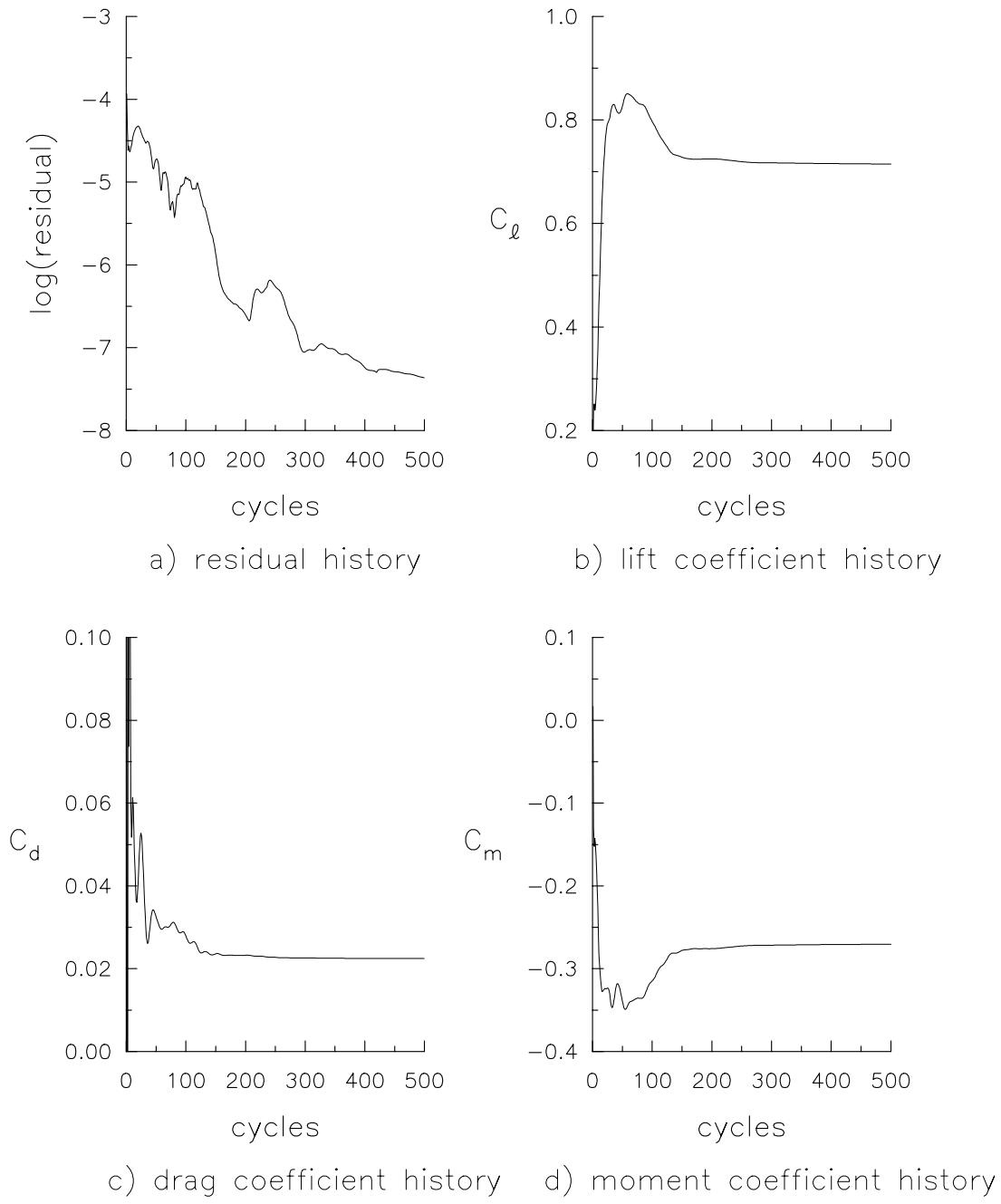


Figure 9-3. Residual and coefficient histories for RAE 2822 airfoil case;
$\alpha = 2.72°$, $M_\infty = 0.75$, $Re_{\tilde{L}_R} = 6.2 \times 10^6$

9.1.2 NACA 0012 Airfoil with Overlapped Grids

This test case solves for the inviscid flow over the NACA 0012 airfoil at $\alpha = 5°$ with $M_\infty = 0.2$. The grid has a total of 4850 points on two grid zones which communicate with one another through overlapped grid stencils. Therefore, the MaGGiE code is used in addition to CFL3D. The memory requirement for this case is 1.8 million words. A typical timing for this case is 43 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of June 1996). A close-up of the grid near the airfoil is shown in Figure 9-4.

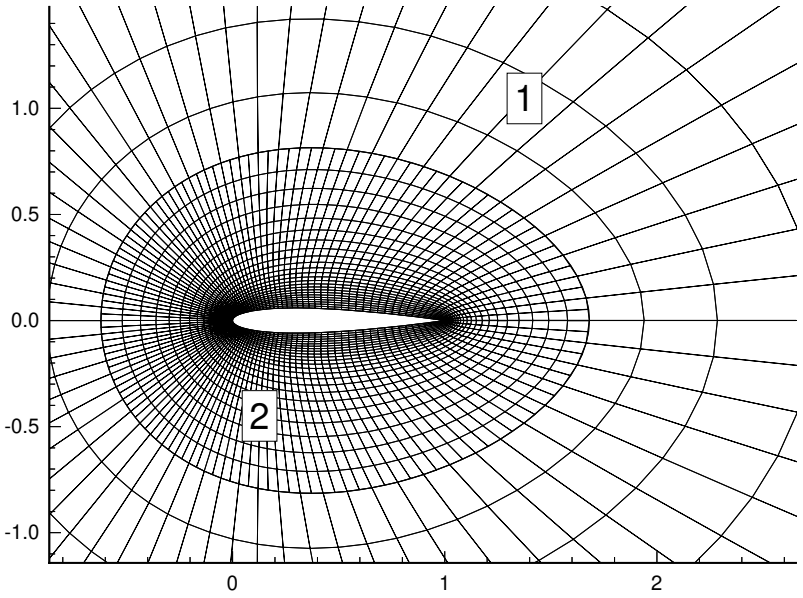


Figure 9-4. Two-zone overlapped grid system for NACA 0012 airfoil.

Besides the CFL3D and MaGGiE codes the following files are needed to run this test case:

File	Description
`0012x.inp`	input for CFL3D
`0012x.fmt`	formatted grid in PLOT3D format
`fmttobin_p3d.f`	grid converter
`mag1.h`	parameters for MaGGiE makefile
`maggie.inp`	input for MaGGiE

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 fmttobin_p3d.f
```

Step 2

Link the grid converter object file:

```
segldr -o fmttobin_p3d fmttobin_p3d.o
```

Step 3

Run the grid converter program (the binary file `0012x.bin` will be output):

```
fmttobin_p3d
```

Step 4

Use the makefile to compile, link, and create the executable for the MaGGiE code (be sure `mag1.h` is in the current directory):

```
make -f makemaggie_cray
```

Step 5

Run the MaGGiE code (the file `ovrlp.bin` will be output):

```
maggie < maggie.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 7

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < 0012x.inp
```

Step 8

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 9

Run the CFL3D code:

```
cfl3d < 0012x.inp
```

The input file for this case is:

```
I/O FILES
0012x.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
  2-block 0012 airfoil as simple chimera test
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
      .200     5.000       0.0       0.0     520.0         1         0
      SREF      CREF      BREF      XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.25000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
     -5.00         0       000      1.00         0       10.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
        -2         2         0       100         0         1         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0         0         0         0         0
         2         0         0         1         0         0         0
      IDIM      JDIM      KDIM
       002        65        25
       002       129        25
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
        00        00       000       000         0      0000
        00        00       000       000         0      0000
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         0         0         0
         1         1         1         0         0         0
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         1
         2         1         1         1         1         1         1         1
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1         0         0         0         0         0         0
         2         1         0         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1         0         0         0         0         0         0
         2         1         0         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1         0         0         0         0         0         0
         2         1      1005         0         0         0         0         0
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1      1003         0         0         0         0         0
         2         1         0         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         1         0        01
      ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
         0        .3        .3        .3         0        .3        .3        .3
```

```
     NCYC    MGLEVG     NEMGL     NITFO
      500        03        00       000
     MIT1      MIT2      MIT3      MIT4      MIT5
       01        01        01        01        01
  1-1 BLOCKING DATA:
     NBLI
        2
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1      1      1      2      1     25      1      3
     2      2            1      1      1      2      1     25      1      3
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1     65      1      2     65     25      1      3
     2      2            1    129      1      2    129     25      1      3
 PATCH SURFACE DATA:
   NINTER
        0
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1    001      1     01    999      1      1    999      1
     2      0      1    001      1     01    999      1      1    999      1
 MOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

The residual and coefficient histories for this case are plotted in Figure 9-5.

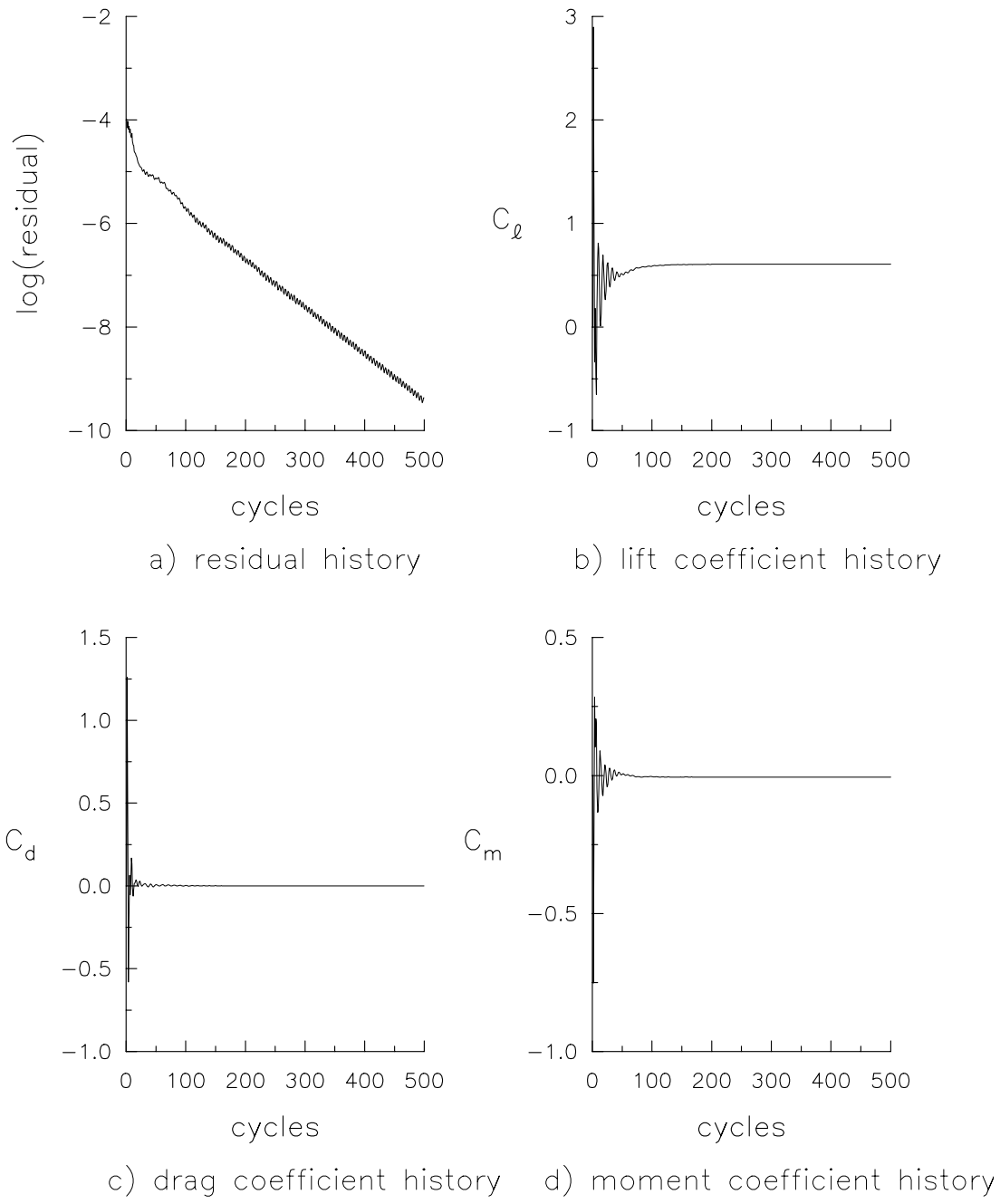


Figure 9-5. NACA 0012 with overlapped grids residual and coefficient histories; $\alpha = 5°$, $M_\infty = 0.2$.

9.1.3 NACA 0012 Airfoil with Patched Grids

This test case solves for the inviscid flow over the NACA 0012 airfoil at $\alpha = 1.25°$ with $M_\infty = 0.8$. The grid has a total of 4949 points in seven zones which communicate with one another utilizing the patching option. Therefore, the ronnie code is used in addition to CFL3D. An advantage of using patched grids is that finer grids can be placed in high gradient regions while relatively coarser grids can be placed elsewhere thus reducing the CPU time and memory needed. In this case, the finest grids are located in the regions where the upper and lower shocks are expected to occur in order to better resolve these flow phenomena. The memory requirement for this example is 1.9 million words. A typical timing is 87 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of June 1996). A close-up of the grid near the airfoil is shown in Figure 9-6. In the figure, the grids are labelled one through seven and this is the grid order in which the information is set up in the input file.

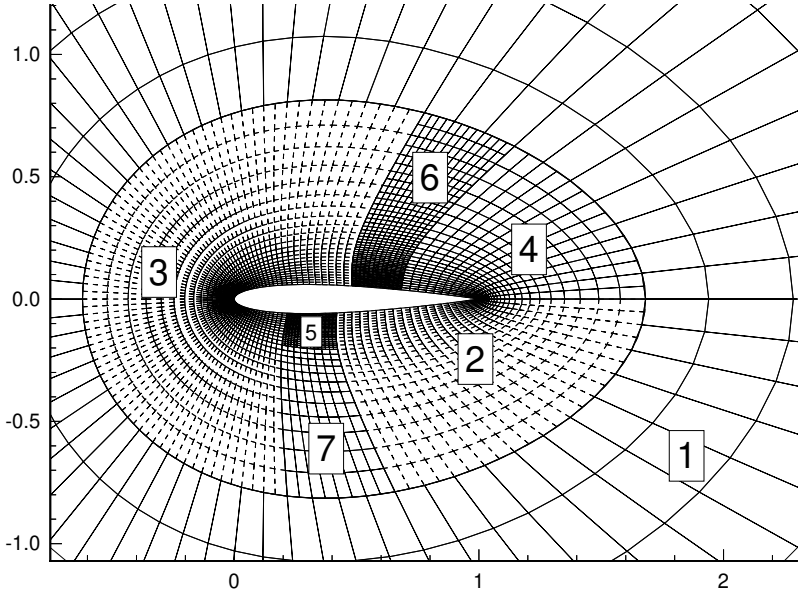


Figure 9-6. Seven-zone patched grid system for NACA 0012 airfoil.

Besides the CFL3D and ronnie codes the following files are needed to run this test case:

File	Description
`0012.inp`	input for CFL3D
`0012.fmt`	formatted grid
`fmttobin.f`	grid converter

File	Description
`ron1.h`	parameters for ronnie makefile
`ronnie.inp`	input for ronnie

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 fmttobin.f
```

Step 2

Link the grid converter object file:

```
segldr -o fmttobin fmttobin.o
```

Step 3

Run the grid converter program (the binary file `0012.bin` will be output):

```
fmttobin
```

Step 4

Use the makefile to compile, link, and create the executable for the ronnie code (be sure `ron1.h` is in the current directory):

```
make -f makeronnie_cray
```

Step 5

Run the ronnie code (the file `patch.bin_0012` will be output):

```
ronnie < ronnie.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 7

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < 0012.inp
```

Step 8

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 9

Run the CFL3D code:

```
cfl3d < 0012.inp
```

The input file for this case is:

```
I/O FILES
0012.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin_0012
restart.bin
input for 7 block patched 0012 grids - iopt = 1 in assemble.f
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
      0.80     1.25        0.0     0.000     122.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.25000       0.0       0.0
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
     -5.00         0       000      1.00         0      10.0
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D   NTSTEP        ITA
         7         7         0       100         0         1        1          1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         1         0         0         0         0         0         0
         1         0         0         1         0         0         0
         1         0         0         1         0         0         0
         1         0         0         1         0         0         0
         1         0         0         1         0         0         0
         1         0         0         1         0         0         0
         1         0         0         0         0         0         0
      IDIM      JDIM      KDIM
         2        65        13
         2        23        25
         2        79        25
         2        15        25
         2        17        25
         2        13        49
         2         9        13
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
```

```
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         0         0         0         3         3         3
         0         0         0         3         3         3
         0         0         0         3         3         3
         0         0         0         3         3         3
         0         0         0         3         3         3
         0         0         0         3         3         3
         0         0         0         3         3         3
  IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
         0         0         0     .3333     .3333     .3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         0
         2         1         1         1         1         1         1         0
         3         1         1         1         1         1         1         0
         4         1         1         1         1         1         1         0
         5         1         1         1         1         1         1         0
         6         1         1         1         1         1         1         0
         7         1         1         1         1         1         1         0
I0:    GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
         3         1      1002         0         0         0         0         0
         4         1      1002         0         0         0         0         0
         5         1      1002         0         0         0         0         0
         6         1      1002         0         0         0         0         0
         7         1      1002         0         0         0         0         0
IDIM:  GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
         3         1      1002         0         0         0         0         0
         4         1      1002         0         0         0         0         0
         5         1      1002         0         0         0         0         0
         6         1      1002         0         0         0         0         0
         7         1      1002         0         0         0         0         0
J0:    GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1         0         0         0         0         0         0
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
         4         1         0         0         0         0         0         0
         5         1         0         0         0         0         0         0
         6         1         0         0         0         0         0         0
         7         1         0         0         0         0         0         0
JDIM:  GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1         0         0         0         0         0         0
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
         4         1         0         0         0         0         0         0
         5         1         0         0         0         0         0         0
         6         1         0         0         0         0         0         0
         7         1         0         0         0         0         0         0
K0:    GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1         0         0         0         0         0         0
         2         1      1005         0         0         0         0         0
         3         1      1005         0         0         0         0         0
         4         1      1005         0         0         0         0         0
         5         1      1005         0         0         0         0         0
         6         1      1005         0         0         0         0         0
         7         1         0         0         0         0         0         0
KDIM:  GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1      1003         0         0         0         0         0
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
         4         1         0         0         0         0         0         0
         5         1         0         0         0         0         0         0
```

```
         6          1          0          0          0          0          0          0
         7          1          0          0          0          0          0          0
     MSEQ    MGFLAG    ICONSF       MTT      NGAM
        2          1          1          0        02
     ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
        0        .3        .3        .3         0        .3        .3        .3
     NCYC    MGLEVG     NEMGL     NITFO
     0300        01        00       000
     0300        02        00       000
     MIT1      MIT2      MIT3      MIT4      MIT5
       01        01        01        01        01
       01        01        01        01        01
  1-1 BLOCKING DATA:
     NBLI
        1
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1      1      1      2      1     13      1      3
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1     65      1      2     65     13      1      3
 PATCH SURFACE DATA:
   NINTER
       -1
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1    001      1     01    999      1      1    999      1
     2      0      1    001      1     01    999      1      1    999      1
     3      0      1    001      1     01    999      1      1    999      1
     4      0      1    001      1     01    999      1      1    999      1
     5      0      1    001      1     01    999      1      1    999      1
     6      0      1    001      1     01    999      1      1    999      1
     7      0      1    001      1     01    999      1      1    999      1
 MOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

The residual and force coefficient history plots for this case is shown in Figure 9-7. The sharp spike in the residual history plot depicts the iteration at which the grid levels changed for mesh sequencing. These convergence histories can be found in `cfl3d.res`.

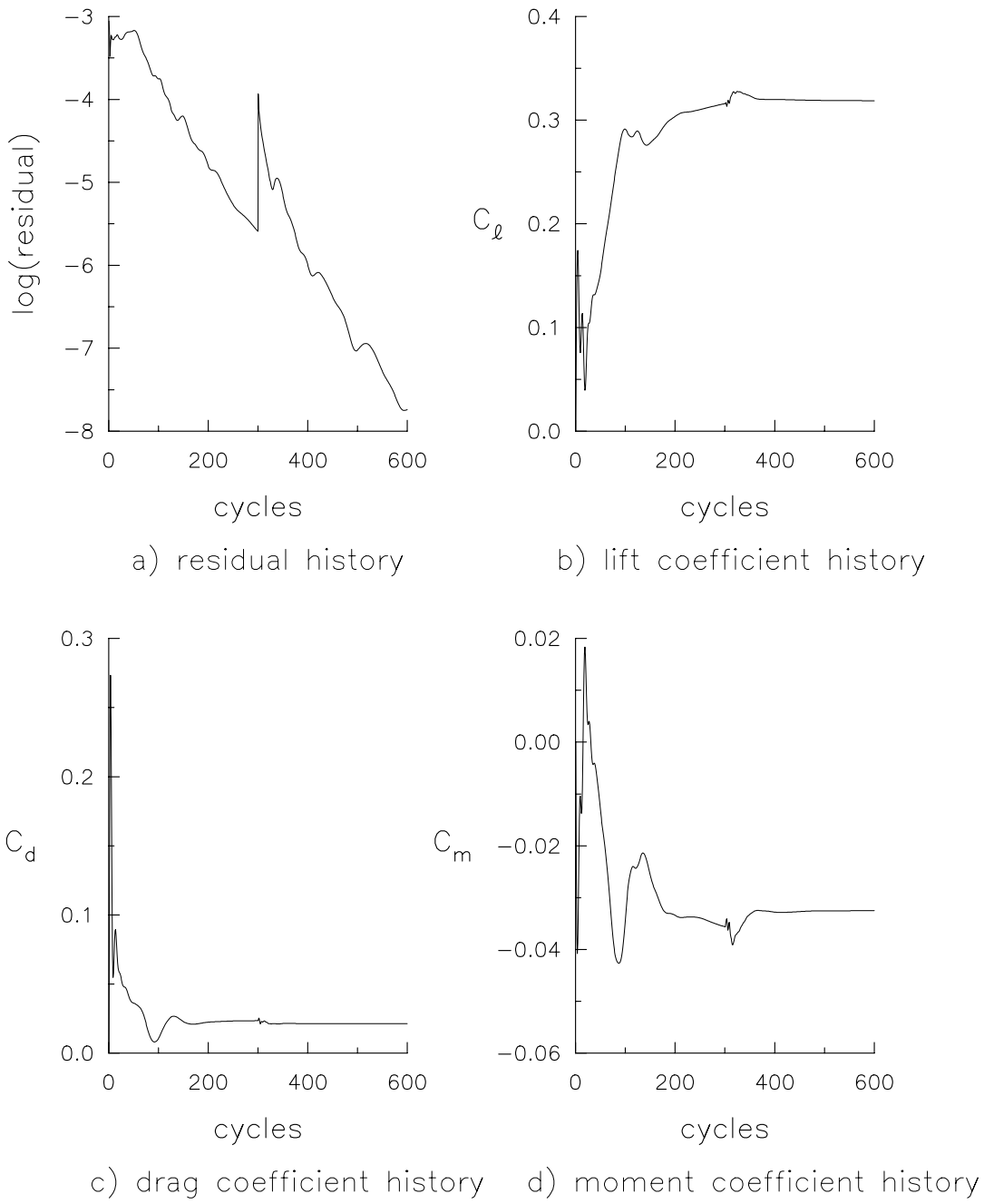


Figure 9-7. Seven-zone NACA 0012 case residual and coefficient histories; $\alpha = 1.25°$, $M_\infty = 0.8$.

9.1.4 Multielement Airfoil with Overlapped Grids

This test case solves for the viscous, turbulent flow over a three-element airfoil with $M_\infty = 0.2$, $\alpha = 8.109°$, and a Reynolds number of 9 million. The Spalart-Allmaras turbulence model is used. The grid, with a total of 59051 points, consists of three zones, one for each element. The grid zones communicate with one another utilizing the grid overlapping option. Therefore, the MaGGiE code is used in addition to CFL3D. The memory requirement for this case is 9.5 million words. A typical timing is 2849 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of June 1996). A close-up of the grid near the airfoil is shown in Figure 9-8. In the figure, the grids are labelled one through three and this is the grid order in which the information is set up in the input file.

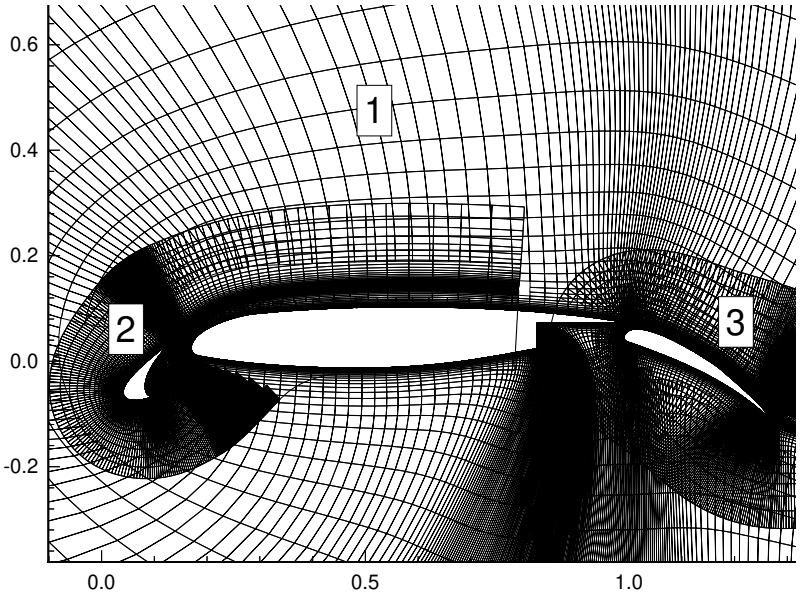


Figure 9-8. Three-zone overlapped grid system for a three-element airfoil.

Besides the CFL3D and MaGGiE codes the following files are needed to run this test case:

File	Description
`multi.inp`	input for CFL3D
`grid.fmt`	formatted grid
`fmttobin.f`	grid converter
`mag1.h`	parameters for MaGGiE makefile
`mag.inp_multi`	input for MaGGiE

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 fmttobin.f
```

Step 2

Link the grid converter object file:

```
segldr -o fmttobin fmttobin.o
```

Step 3

Run the grid converter program (the binary file `multi.bin` will be output):

```
fmttobin
```

Step 4

Use the makefile to compile, link, and create the executable for the MaGGiE code (be sure `mag1.h` is in the current directory):

```
make -f makemaggie_cray
```

Step 5

Run the MaGGiE code (the file `ovrlp.bin` will be output):

```
maggie < mag.inp_multi
```

Step 6

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 7

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < multi.inp
```

Step 8

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 9

Run the CFL3D code:

```
cfl3d < multi.inp
```

The input file for this case is:

```
I/O FILES
multi.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
  3 element airfoil - chimera-type grids - Spalart-Allmaras turb model
     XMACH      ALPHA       BETA   REUE,MIL    TINF,DR      IALPH     IHSTRY
     .2000      8.109        0.0        9.0      520.0          0          0
      SREF       CREF       BREF        XMC        YMC        ZMC
   1.00000    1.00000     1.0000    0.25000       0.00       0.00
        DT      IREST    IFLAGTS       FMAX      IUNST     CFLTAU
     -5.00          0        000       1.00          0       10.0
     NGRID    NPLOT3D     NPRINT     NWREST       ICHK        I2D    NTSTEP        ITA
         3          3          0        100          0          1         1          1
       NCG        IEM  IADVANCE     IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2          0          0        001          0          0         5
         2          0          0        001          0          0         5
         2          0          0        001          0          0         5
      IDIM       JDIM       KDIM
       002        361         65
       002        369         57
       002        297         49
    ILAMLO     ILAMHI     JLAMLO     JLAMHI     KLAMLO     KLAMHI
        00         00        000        000          0       0000
        00         00        000        000          0       0000
        00         00        000        000          0       0000
     INEWG     IGRIDC         IS         JS         KS         IE         JE         KE
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
         0          0          0          0          0          0          0          0
  IDIAG(I)   IDIAG(J)   IDIAG(K)   IFLIM(I)   IFLIM(J)   IFLIM(K)
         1          1          1          3          3          3
         1          1          1          3          3          3
         1          1          1          3          3          3
   IFDS(I)    IFDS(J)    IFDS(K)   RKAP0(I)   RKAP0(J)   RKAP0(K)
         1          1          1      .3333      .3333      .3333
         1          1          1      .3333      .3333      .3333
         1          1          1      .3333      .3333      .3333
      GRID      NBCI0    NBCIDIM      NBCJ0    NBCJDIM      NBCK0    NBCKDIM     IOVRLP
         1          1          1          1          1          3          1          1
         2          1          1          1          1          3          1          1
         3          1          1          1          1          3          1          1
I0:   GRID    SEGMENT     BCTYPE       JSTA       JEND       KSTA       KEND      NDATA
         1          1       1002          0          0          0          0          0
         2          1       1002          0          0          0          0          0
         3          1       1002          0          0          0          0          0
IDIM: GRID    SEGMENT     BCTYPE       JSTA       JEND       KSTA       KEND      NDATA
         1          1       1002          0          0          0          0          0
         2          1       1002          0          0          0          0          0
         3          1       1002          0          0          0          0          0
J0:   GRID    SEGMENT     BCTYPE       ISTA       IEND       KSTA       KEND      NDATA
         1          1       1003          0          0          0          0          0
```

```
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
JDIM: GRID    SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1003         0         0         0         0         0
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
K0:   GRID    SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1         0         0         0         1        73         0
         1         2      2004         0         0        73       289         2
               TWTYPE        CQ
                   0.        0.
         1         3         0         0         0       289       361         0
         2         1         0         0         0         1        65         0
         2         2      2004         0         0        65       305         2
               TWTYPE        CQ
                   0.        0.
         2         3         0         0         0       305       369         0
         3         1         0         0         0         1        49         0
         3         2      2004         0         0        49       249         2
               TWTYPE        CQ
                   0.        0.
         3         3         0         0         0       249       297         0
KDIM: GRID    SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
         2         1         0         0         0         0         0         0
         3         1         0         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         1         0        01
      ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
         0        .3        .3        .3         0        .3        .3        .3
      NCYC    MGLEVG     NEMGL     NITFO
      2000        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5
        01        01        01        01        01
   1-1 BLOCKING DATA:
      NBLI
         3
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           1      1      1      2     73      1      1      2
      2      2           1      1      1      2     65      1      1      2
      3      3           1      1      1      2     49      1      1      2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           1    361      1      2    289      1      1      2
      2      2           1    369      1      2    305      1      1      2
      3      3           1    297      1      2    249      1      1      2
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      1      1    001      1     01    999      1      1    999      1
      2      1      1    001      1     01    999      1      1    999      1
      3      1      1    001      1     01    999      1      1    999      1
  MOVIE
      0
  PRINT OUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After running this test case, the residual and force coefficient convergence histories should look like those in Figure 9-9. These convergence histories can be found in file `cfl3d.res`. Note the unusually high number of multigrid cycles required to converge this case. While quite large, this is the behavior typically seen (with CFL3D) for multielement airfoil cases, even when one-to-one blocking is employed.

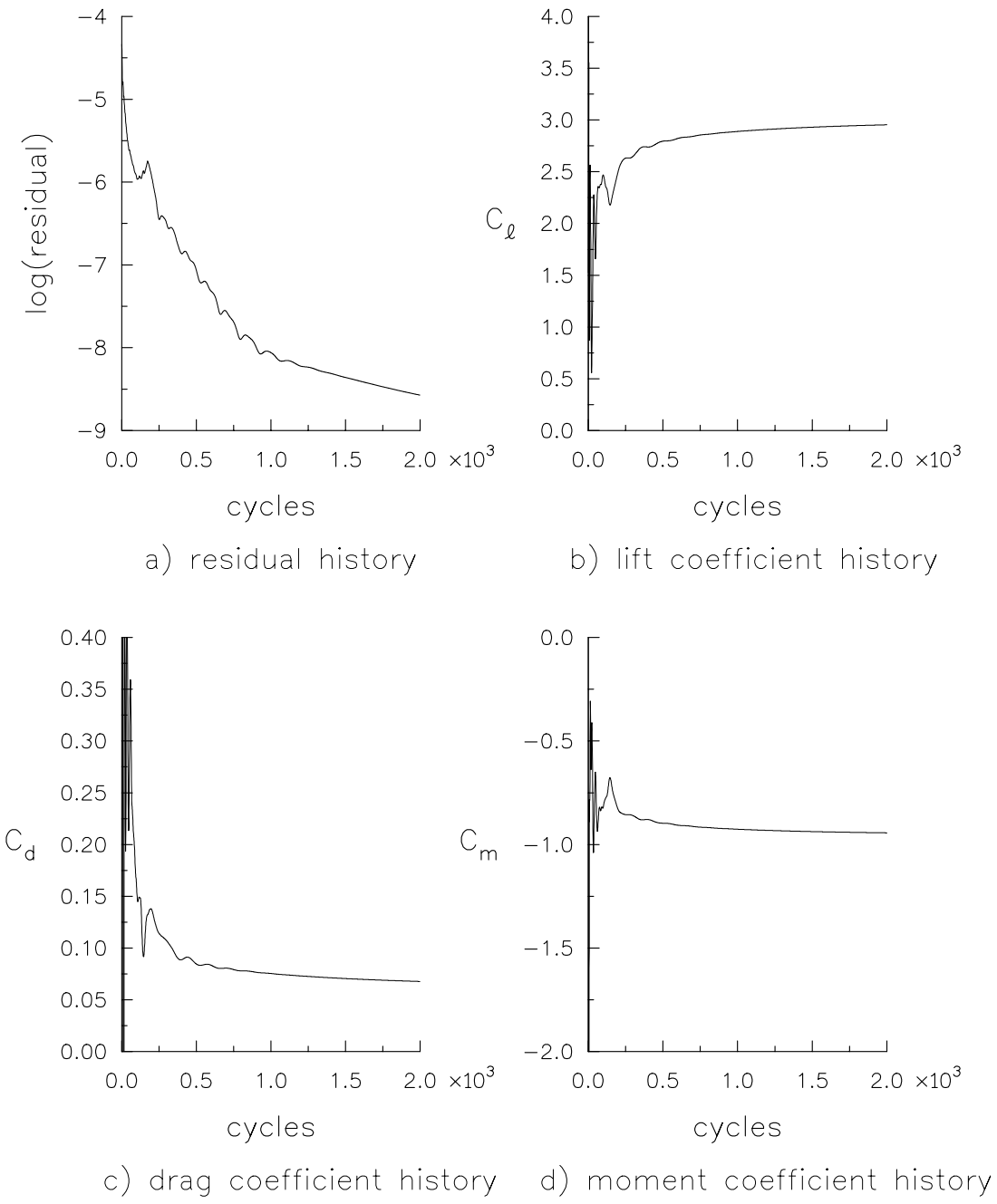


Figure 9-9. Convergence histories for three-element airfoil case;
$\alpha = 8.109°$, $Re_{\tilde{L}_R} = 9 \times 10^6$.

9.1.5 Flat Plate

The viscous, turbulent flow with a Reynolds number of 6 million over a flat plate is solved in this test case. The grid consists of a single grid zone with 6305 points. Menter's $k-\omega$ SST turbulence model is utilized in this example. The memory requirement is 2.3 million words. A typical timing for this case is 157 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of June 1996). The entire flat plate grid is illustrated in Figure 9-10.

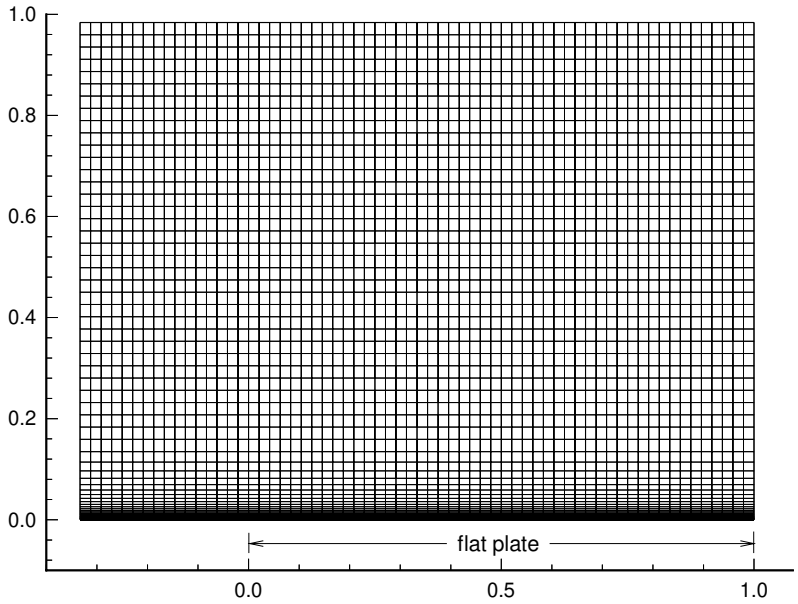


Figure 9-10. Single zone flat plate grid.

Besides the CFL3D code the following files are needed to run this test case:

File	Description
`grdflat5.inp`	input for CFL3D
`grdflat5.grd`	formatted grid
`grid2dto3d.f`	grid converter

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 grid2dto3d.f
```

Step 2

Link the grid converter object file:

```
segldr -o grid2dto3d grid2dto3d.o
```

Step 3

Run the grid converter program (the binary file `grdflat5.bin` will be output):

```
grid2dto3d
```

In answer to the questions, type:

```
grdflat5.grd
grdflat5.bin
2
0
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < grdflat5.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < grdflat5.inp
```

The input file for this case is:

```
I/O FILES
grdflat5.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
```

```
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    turbulent flat plate (plate from j=17-65, prior to 17 is symmetry)
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
    0.2000    00.000       0.0    06.000     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
    -5.000         1       000   05.0000         0       10.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         2      1200         0         1         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0       001         0         0        12
    IDIM    JDIM    KDIM
      02      65      97
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         1         2         1        17         1        97
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         0         0         0
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         2         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1008         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1001         0         0         1        17         0
         1         2      2004         0         0        17        65         2
             TWTYPE        CQ
                0.        0.
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         0         0        02
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
      0500        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7      MIT8
        01        01        01        01        01         1         1         1
   1-1 BLOCKING DATA:
      NBLI
         0
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      0      0      0      0      0      0      0      0      0
 IMOVIE
      0
  PRINT OUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      0      0      0      0      0      0      1      1      1
      1      0      0      0      0     49     49      1      0      0      0
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After this test case is run, the residual history, found in file `cfl3d.res`, should look like that plotted in Figure 9-11. In Figure 9-12, values of u^+ verses y^+ are plotted at two cross sections of the flat plate and compared with theoretical values. The u^+ and y^+ values were extracted from the PLOT3D grid and q files using a postprocessor currently not available for general use.

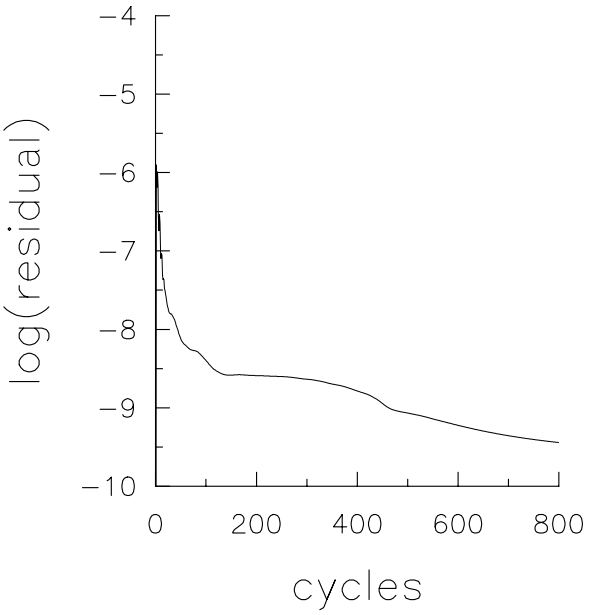


Figure 9-11. Residual history for single grid flat plate case.

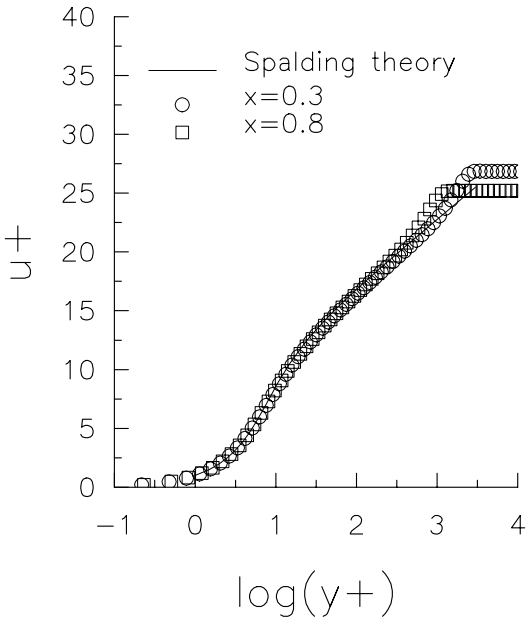


Figure 9-12. Flat plate calculation compared with experiment; $Re_{\tilde{L}_R} = 6 \times 10^6$.

This flat plate case has been studied with *all* the turbulence models currently available in CFL3D and a summary of the timings is tabulated below:

Turbulence Model	**ivisc**	Approximate CPU seconds per cycle	Percent increase over Baldwin-Lomax per cycle
Baldwin-Lomax	2	0.153	0
Baldwin-Lomax with Degani-Schiff	3	0.153	0
Baldwin-Barth	4	0.177	16
Spalart-Allmaras	5	0.182	19
Wilcox $k-\omega$	6	0.186	22
SST $k-\omega$	7	0.196	28
$k-\omega$ EASM Gatski-Speziale (Linear)	8	0.218	43
$k-\varepsilon$ EASM Girimaji (Linear)	9	0.236	54
$k-\varepsilon$ (Abid version)	10	0.197	29
$k-\varepsilon$ EASM Gatski-Speziale (Nonlinear)	11	0.236	54
$k-\omega$ EASM Gatski-Speziale (Nonlinear)	12	0.236	54
$k-\varepsilon$ EASM Girimaji (Nonlinear)	13	0.254	66

This case requires between 800 to 1800 cycles to converge, depending on the turbulence model employed and the convergence criterion chosen. Generally, $k-\varepsilon$ models take longer than $k-\omega$ models and two-equation models tend to take longer than one-equation models. (See Appendix H.) Memory requirements also depend on which turbulence model is being used, varying from 2.2 to 2.4 million words.

9.1.6 Vibrating Flat Plates

This test case solves for the unsteady, time-accurate inviscid flow through an "infinite" row of vibrating flat plates. The plates, located from $x = 0.0$ to $x = 1.0$, are vibrating up and down with a sinusoidal motion. The maximum displacement is $h = 0.001$ and the nominal distance between the plates is 1.0. The reduced frequency, defined by $k_r = \tilde{\omega}\tilde{c}/2|\tilde{\mathbf{V}}|_\infty$, where ω is in radians/second, is 4.0. (Note that the input to CFL3D defines $\tilde{\omega}$ in cycles/second; therefore, **rfreq** = 0.63662.) The plate vibration generates acoustic waves which propagate both downstream and upstream. The solution invokes periodicity at both the upper and lower boundaries (except between $x = 0.0$ and $x = 1.0$), thus mimicking an infinite row of flat plates.

This example also employs a second block downstream of $x = 2.0$. This block is a "sliding block" that is used to test the effect of a sliding block interface (such as might be used in a rotor-stator computation) on the transmission of acoustic waves. Currently, it is set to translate "up" with speed $w/a_\infty = 1/\pi = 0.31831$.

This test problem is set up to run for 961 time steps, using three multigrid sub-iterations per time step. The time step is $\pi/320 = 0.0098175$. At this time step, 160 time steps yield one complete cycle of plate oscillation, so 961 steps yield six complete cycles of plate oscillations. The code, taking advantage of the periodicity of the solution, "resets" the sliding zone (zone 2) whenever its motion exceeds **dzmax** = 1.0. At the end of time step 961, zone 2 is again physically aligned with zone 1. If the number of steps taken is such that **ntstep**-1 is not evenly divisible by 160, then zone 2 will appear displaced some distance "up" from zone 1 when looking at the solution. While this is not a problem since the solution is periodic, it is easier to visualize the whole flow field when the two zones are aligned.

The grid consists of two grid zones with a total of 5502 points. The memory requirement for this case is 2.1 million words. A typical timing for six cycles of plate oscillation is 422 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of July 1996).

Besides the CFL3D code the following files are needed to run this test case:

File	Description
`vibrate.inp`	input for CFL3D
`cartesian.f`	grid generator

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid generator code:

```
cft77 cartesian.f
```

Step 2

Link the grid converter object file:

```
segldr -o cartesian cartesian.o
```

Step 3

Run the grid generator program (the binary file `grid.bin` will be output):

```
cartesian
```

In answer to the questions, type

```
2
-6,2,0,1
.05,.05
0
2,7,0,1
.05,.05
0
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the cflx`.h` files will be output):

```
precfl3d < vibrate.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < vibrate.inp
```

The input file for this case is:

```
I/O FILES
grid.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
  Flat plate vibrating cascade with sliding interface downstream
     XMACH      ALPHA       BETA  REUE,MIL    TINF,DR      IALPH     IHSTRY
      .500       0.00        0.0      1.07      520.0          0          0
     SREF       CREF       BREF       XMC        YMC        ZMC
  1.00000    1.00000     1.0000   0.25000       0.00       0.00
       DT      IREST    IFLAGTS      FMAX      IUNST     CFLTAU
 .0098175          0        000      1.00          1         5.
```

```
      NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
          2         2         2      2000         0         1       961        -2
        NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
          2         0         0       001         0         0         0
          2         0         0       001         0         0         0
       IDIM      JDIM      KDIM
          2       161        21
          2       101        21
     ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         00        00       000       000         0      0000
         00        00       000       000         0      0000
      INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
   IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
          1         1         1         0         0         0
          1         1         1         0         0         0
    IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
          1         1         1     .3333     .3333     .3333
          1         1         1     .3333     .3333     .3333
       GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
          1         1         1         1         1         3         3         0
          2         1         1         1         1         1         1         0
I0:    GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
          1         1      1001         0         0         0         0        0
          2         1      1001         0         0         0         0        0
IDIM:  GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
          1         1      1002         0         0         0         0        0
          2         1      1002         0         0         0         0        0
J0:    GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
          1         1      1003         0         0         0         0        0
          2         1         0         0         0         0         0        0
JDIM:  GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
          1         1         0         0         0         0         0        0
          2         1      1003         0         0         0         0        0
K0:    GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
          1         1      2005         1         2         1       121        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    1       0.0        0.        0.
          1         2      1005         1         2       121       141        0
          1         3      2005         1         2       141       161        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    1       0.0        0.        0.
          2         1      2005         0         0         0         0        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    2       0.0        0.        0.
KDIM:  GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
          1         1      2005         1         2         1       121        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    1       0.0        0.        0.
          1         2      1005         1         2       121       141        0
          1         3      2005         1         2       141       161        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    1       0.0        0.        0.
          2         1      2005         0         0         0         0        4
                 NBLP     DTHTX     DTHTY     DTHTZ
                    2       0.0        0.        0.
       MSEQ    MGFLAG    ICONSF       MTT      NGAM
          1         1         1         0        02
       ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
          0        .3        .3        .3         0        .3        .3        .3
       NCYC    MGLEVG     NEMGL     NITFO
          3        03        00       000
       MIT1      MIT2      MIT3      MIT4      MIT5
         01        01        01        01        01
    1-1 BLOCKING DATA:
       NBLI
          0
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
  PATCH SURFACE DATA:
    NINTER
```

```
       0
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1    999      1     01    999      1      1    999      1
     2      0      1    999      1     01    999      1      1    999      1
 MOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      1      1      2      1    999    999      1      1    999      1
     2      1      1      2      1      1      1      1      1    999      1
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
MOVING GRID DATA - TRANSLATION
 NTRANS
      2
   LREF
    0.5
   GRID   ITRANS    RFREQ     XMAG     YMAG     ZMAG
      1        2   .31831       0.       0.  0.00100
      2        1       0.       0.       0.  0.31831
   GRID    DXMAX    DYMAX    DZMAX
      1       0.       0.    0.000
      2       0.       0.    1.000
MOVING GRID DATA - ROTATION
 NROTAT
      0
   LREF
   GRID   IROTAT    RFREQ   THXMAG   THYMAG   THZMAG    XORIG    YORIG    ZORIG
   GRID   THXMAX   THYMAX   THZMAX
 DYNAMIC PATCH INPUT DATA
   NINTER
        2
  INT  IFIT    LIMIT    ITMAX    MCXIE    MCETA      C-0    IORPH    ITOSS
    1    -1        5       20        0        0        0        0        1
    2    -1        5       20        0        0        0        0        1
  INT    TO     XIE1     XIE2     ETA1     ETA2     NFB
    1   122        0        0        0        0       2
       FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
        221        0        0        0        0       0.       0.
                  DX       DY       DZ   DTHETX   DTHETY   DTHETZ
                  0.       0.       0.       0.       0.       0.
       FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
        221        0        0        0        0       0.       0.
                  DX       DY       DZ   DTHETX   DTHETY   DTHETZ
                  0.       0.       -1.      0.       0.       0.
  INT    TO     XIE1     XIE2     ETA1     ETA2     NFB
    2   221        0        0        0        0       2
       FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
        122        0        0        0        0       0.       0.
                  DX       DY       DZ   DTHETX   DTHETY   DTHETZ
                  0.       0.       0.       0.       0.       0.
       FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
        122        0        0        0        0       0.       0.
                  DX       DY       DZ   DTHETX   DTHETY   DTHETZ
                  0.       0.      +1.       0.       0.       0.
```

The resulting residual and lift coefficient histories for this case are shown in Figure 9-13 and Figure 9-14, respectively. The oscillatory nature of the flow is clearly evident. Figure 9-15 shows a profile of the flow as defined by pressure contours.

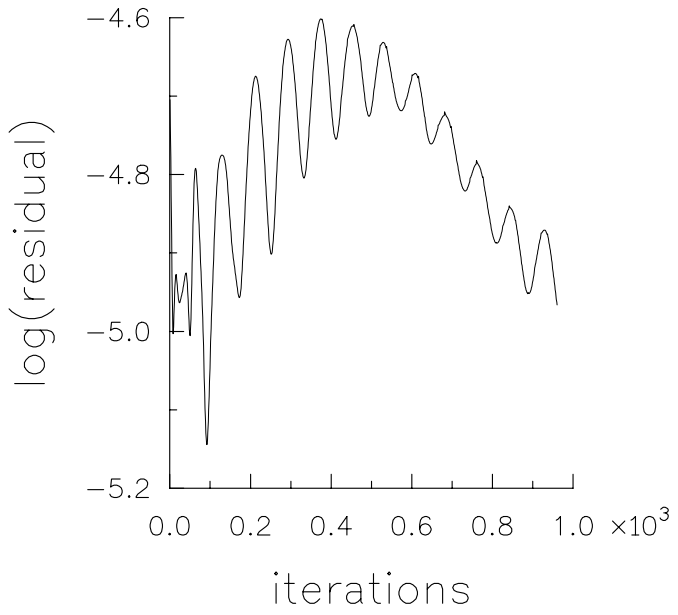


Figure 9-13. Residual history for inviscid flow through vibrating flat plates.

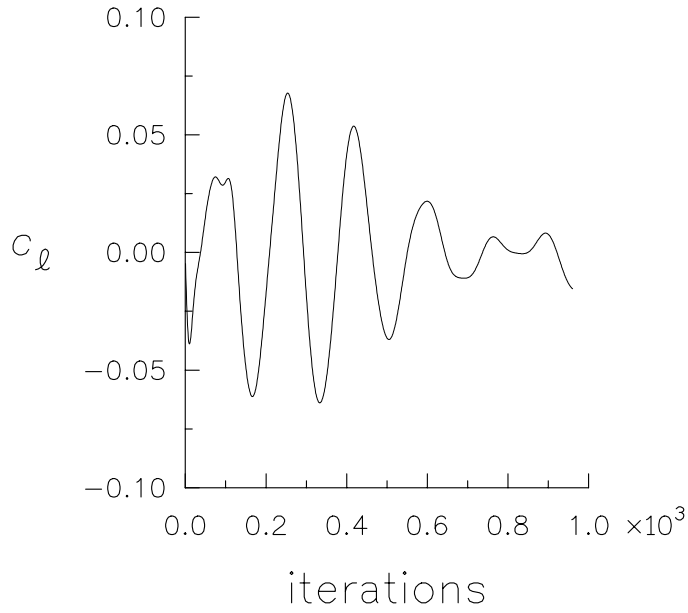


Figure 9-14. Lift coefficient history for inviscid flow through vibrating flat plates; $M_\infty = 0.5$.

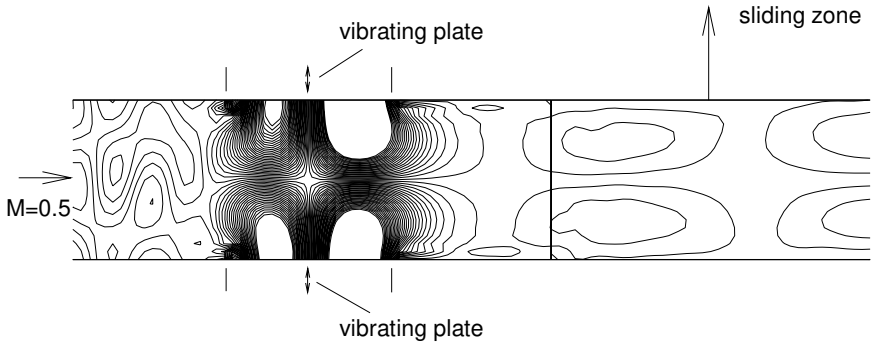


Figure 9-15. Pressure contours for inviscid flow through vibrating flat plates; $M_\infty = 0.5$.

9.1.7 Multistream Nozzle

This case simulates, in two dimensions, the flow though a converging/diverging nozzle with multiple streams. The case is meant to model the exhaust from an engine (with a hot core and cooler outer flow modeled as a "top hat" temperature profile) entering an s-shaped converging/diverging nozzle. Two additional streams are injected downstream of the throat to provide additional cooling of the exhaust.

The grid consists of thirteen patched zones with a total of 15897 points in one plane. The memory requirement for this example is 3.2 million words. A typical timing for this case is 1550 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of October 1996). A cross-section of the grid is shown in Figure 9-16.

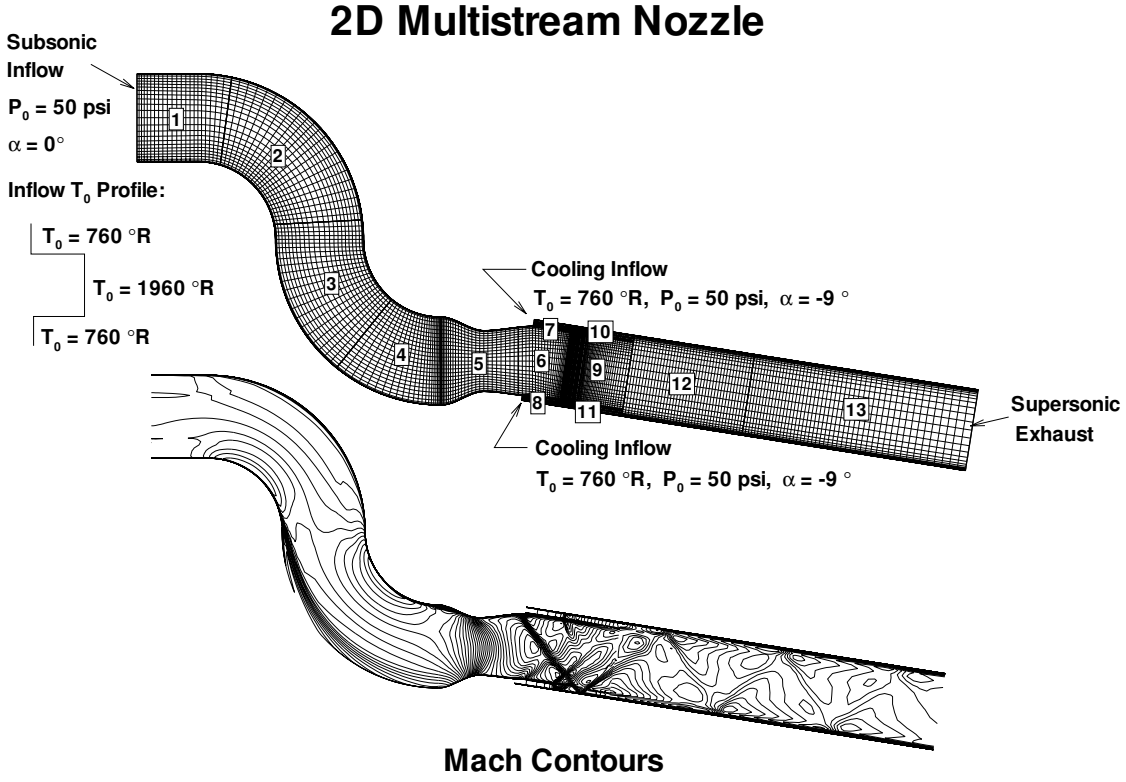


Figure 9-16. Cross-section of multistream nozzle zonal patching and Mach contours.

Boundary conditions with user-defined input, including constant data supplied via the main input deck and variable (point-to-point) data supplied via an auxiliary boundary condition data file are exemplified. Control surfaces are used to monitor mass-flow convergence (here, two control surfaces are used to measure the "difference" between mass in

and mass out). In addition, the case illustrates how one can define reference conditions for CFL3D in the case of a purely internal flow, as discussed below.

9.1.7.1 Nondimensionalization

The conditions provided for this case are that the total pressure of the primary inflow is 50 psi, with a total temperature of 1960°R in the core and 760°R in the outer region. The flow enters the s-duct with zero angle. The secondary cooling flows also have a total pressure of 50 psi and a uniform total temperature of 760°R. The cooling flow enters at –9 degrees relative to horizontal. The exhaust from the nozzle system is supersonic. The throat height is 1 foot. The primary inflow height is 1.449 feet; the core flow region (i.e. where the total temperature is 1960°R) spans the central 0.769 feet.

Although inflow stagnation conditions would be a natural reference state for this problem, the associated Mach number is zero. Since the viscous terms are scaled with the reference Mach number, another reference state is needed. A second natural reference state for nozzle flows is the sonic point. From isentropic relations, the sonic conditions can be obtained once the inflow total conditions are known. However, for this problem there are two total conditions owing to the "top hat" temperature profile. Thus, to have just one reference state, the total temperature of the inflow is area-averaged and the resulting average total temperature, together with the given total pressure, is used to determine the sonic conditions at the throat.

In what follows, stagnation conditions are denoted by 0, sonic conditions by *, and dimensional quantities by ~. First, determine the average inflow total temperature:

$$\tilde{T}_0 = \frac{0.769}{1.449(1960)} + \frac{(1.449 - 0.769)}{1.449(760)} = 1397°\text{R} \tag{9-1}$$

Next, determine the stagnation density and speed of sound:

$$\tilde{\rho}_0 = \frac{\tilde{p}_0}{\tilde{R}\tilde{T}_0} = \frac{50(144)}{1716(1397)} = 0.00300 \text{ slugs/feet}^3 \tag{9-2}$$

$$\tilde{a}_0 = \sqrt{\gamma \tilde{R} \tilde{T}_0} = \sqrt{1.4(1716)(1397)} = 1832 \text{ feet/second} \tag{9-3}$$

where $\tilde{R} = 1716$ feet^2/(second^2-°R). From the isentropic relations,

$$\frac{p}{p_0} = 0.528, \frac{T}{T_0} = 0.833, \frac{\rho}{\rho_0} = 0.634, \frac{a}{a_0} = 0.913 \tag{9-4}$$

Thus, the desired reference pressure, temperature, speed of sound and density are

$$\begin{aligned}\tilde{p} &= 50(0.528) = 26.4 \text{ psi}\\ \tilde{T} &= 1397(0.833) = 1164°\text{R}\\ \tilde{\rho} &= 0.003(0.634) = 0.0019 \text{ slug/feet}^3\\ \tilde{a} &= 1832(0.913) = 1673 \text{ feet/second}\end{aligned} \tag{9-5}$$

Assuming a molecular viscosity coefficient of 3.7×10^{-7} slugs/(feet-seconds) for a temperature of 520°R, then the power law $\mu_2/\mu_1 = (T_2/T_1)^{0.76}$ gives

$$\tilde{\mu}^* = 3.7\times10^{-7}\left(\frac{1164}{520}\right)^{0.76} = 6.82\times10^{-7} \text{ slugs/(feet-seconds)} \tag{9-6}$$

The reference Reynolds number based on throat height and the reference sonic values is

$$Re^* = \frac{\tilde{\rho}^*\tilde{a}^*(\text{throat height})}{\tilde{\mu}^*} = 0.0019(1673)\frac{1}{6.82\times10^{-7}} = 4.66\times10^6 \tag{9-7}$$

In the grid, the throat height is 12 inches, so the input parameter **reue** is

$$\textbf{reue} = \frac{Re^*\times10^{-6}}{12} = \frac{4.66}{12} = 0.388 \tag{9-8}$$

Finally, the nondimensional input values for boundary condition type 2003 are determined from the reference sonic conditions (note that in CFL3D parlance, in this problem the * conditions are the "infinity" conditions):

$$\begin{aligned}p_t = 50 \text{ psi} \rightarrow \frac{p_t}{p^*} &= \frac{p_t}{p_\infty} = \frac{50}{26.4} = 1.894\\ T_t = 760°\text{R} \rightarrow \frac{T_t}{T^*} &= \frac{T_t}{T_\infty} = \frac{760}{1164} = 0.653\\ T_t = 1960°\text{R} \rightarrow \frac{T_t}{T^*} &= \frac{T_t}{T_\infty} = \frac{1960}{1164} = 1.684\end{aligned} \tag{9-9}$$

Boundary condition type 2003 also needs an estimate of the local Mach number. For the primary inflow, the inlet height (area) to throat height (area) is 1.449/1. The isentropic relations give a corresponding Mach number of approximately 0.45. The local Mach number for the cooling inflow cannot be determined a priori; 1.0 is used in the boundary condition. The computations give the cooling inflow Mach number as approximately 0.85; 1.0 is deemed as a sufficiently close estimate since the solution does not change perceptibly if 0.85 is used instead of 1.0.

The auxiliary boundary condition data file provided for this example (`inflow.data`) contains the data for the primary inflow:

$$\begin{aligned} M &= 0.45 \\ \frac{p_t}{p_\infty} &= 1.894 \\ \frac{T_t}{T_\infty} &= \frac{0.653}{1.684} \end{aligned} \tag{9-10}$$

in top hat distribution, $\alpha = 0, \beta = 0$. The cooling inflow:

$$\begin{aligned} M &= 1.0 \\ \frac{p_t}{p_\infty} &= 1.894 \\ \frac{T_t}{T_\infty} &= 0.653 \\ \alpha &= -9 \\ \beta &= 0 \end{aligned} \tag{9-11}$$

is specified explicitly in the main data file.

9.1.7.2 Running CFL3D

Besides the CFL3D and ronnie codes the following files are needed to run this test case:

File	Description
`multistream.inp`	input for CFL3D
`grid_multistream.fmt`	formatted grid
`formtobin.f`	grid converter
`inflow.data`	auxiliary boundary condition data
`ron1.h`	parameters for ronnie makefile
`ronnie.inp`	input for ronnie

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 formtobin.f
```

Step 2

Link the grid converter object file:

```
segldr -o formtobin formtobin.o
```

Step 3

Run the grid generator program (the binary file `grid_multistream.bin` will be output):

```
formtobin
```

Step 4

Use the makefile to compile, link, and create the executable for the ronnie code (be sure `ron1.h` is in the current directory):

```
make -f makeronnie_cray
```

Step 5

Run the ronnie code (the file `patch_multistream.bin` will be output):

```
ronnie < ronnie.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 7

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < multistream.inp
```

Step 8

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 9

Run the CFL3D code (be sure the `inflow.data` file is available and correct for this case):

```
cfl3d < multistream.inp
```

The input file for this case is:

```
I/O FILES
grid_multistream.p3d
plot3dg.bin
```

```
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch_multistream.bin
restart.bin
 Multistream Nozzle (sonic conditions as reference state)
      XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
      1.000      0.00       0.0     0.388    1163.0         1         1
       SREF      CREF      BREF       XMC       YMC       ZMC
    1.00000   1.00000    1.0000   0.25000      0.00      0.00
         DT     IREST   IFLAGTS      FMAX     IUNST   CFL_TAU
    -1.0000         0       000       1.0        +1        5.
      NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
        -13        13         0       500         0         1         2        -2
        NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
          1         0         0       000         0         0        +7
       IDIM      JDIM      KDIM
          2        23        41
          2        23        41
          2        23        41
          2        23        41
          2        25        41
          2        25        41
          2        17        21
          2        17        21
          2        49        41
          2        49        21
          2        49        21
          2        41        61
          2        49        57
     ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
         00        00       000       000         0      0000
      INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
          0         0         0         0         0         0         0         0
```

```
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
         1         1         1         3         3         3
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
         1         1         1     .3333     .3333     .3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         0
         2         1         1         1         1         1         1         0
         3         1         1         1         1         1         1         0
         4         1         1         1         1         1         1         0
         5         1         1         1         1         1         1         0
         6         1         1         1         1         1         1         0
         7         1         1         1         1         1         1         0
         8         1         1         1         1         1         1         0
         9         1         1         1         1         1         1         0
        10         1         1         1         1         1         1         0
        11         1         1         1         1         1         1         0
        12         1         1         1         1         1         1         0
        13         1         1         1         1         1         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
         3         1      1002         0         0         0         0         0
         4         1      1002         0         0         0         0         0
         5         1      1002         0         0         0         0         0
         6         1      1002         0         0         0         0         0
         7         1      1002         0         0         0         0         0
         8         1      1002         0         0         0         0         0
         9         1      1002         0         0         0         0         0
        10         1      1002         0         0         0         0         0
        11         1      1002         0         0         0         0         0
        12         1      1002         0         0         0         0         0
        13         1      1002         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1002         0         0         0         0         0
         2         1      1002         0         0         0         0         0
         3         1      1002         0         0         0         0         0
         4         1      1002         0         0         0         0         0
         5         1      1002         0         0         0         0         0
         6         1      1002         0         0         0         0         0
         7         1      1002         0         0         0         0         0
         8         1      1002         0         0         0         0         0
         9         1      1002         0         0         0         0         0
        10         1      1002         0         0         0         0         0
        11         1      1002         0         0         0         0         0
```

```
         12          1       1002          0          0          0          0          0
         13          1       1002          0          0          0          0          0
J0:    GRID    SEGMENT     BCTYPE       ISTA       IEND       KSTA       KEND      NDATA
          1          1       2003          0          0          0          0         -5
Mach  Pt/Pinf  Tt/Tinf  alpha  beta
inflow.data
          2          1          0          0          0          0          0          0
          3          1          0          0          0          0          0          0
          4          1          0          0          0          0          0          0
          5          1          0          0          0          0          0          0
          6          1          0          0          0          0          0          0
          7          1       2003          0          0          0          0          5
  Mach  Pt/Pinf  Tt/Tinf  alpha  beta
  1.00   1.894    0.653     -9.    0.
          8          1       2003          0          0          0          0          5
  Mach  Pt/Pinf  Tt/Tinf  alpha  beta
  1.00   1.894    0.653     -9.    0.
          9          1          0          0          0          0          0          0
         10          1          0          0          0          0          0          0
         11          1          0          0          0          0          0          0
         12          1          0          0          0          0          0          0
         13          1          0          0          0          0          0          0
JDIM: GRID    SEGMENT     BCTYPE       ISTA       IEND       KSTA       KEND      NDATA
          1          1          0          0          0          0          0          0
          2          1          0          0          0          0          0          0
          3          1          0          0          0          0          0          0
          4          1          0          0          0          0          0          0
          5          1          0          0          0          0          0          0
          6          1          0          0          0          0          0          0
          7          1          0          0          0          0          0          0
          8          1          0          0          0          0          0          0
          9          1          0          0          0          0          0          0
         10          1          0          0          0          0          0          0
         11          1          0          0          0          0          0          0
         12          1          0          0          0          0          0          0
         13          1       1002          0          0          0          0          0
K0:    GRID    SEGMENT     BCTYPE       ISTA       IEND       JSTA       JEND      NDATA
          1          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          2          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          3          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          4          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          5          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          6          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          7          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          8          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
          9          1          0          0          0          0          0          0
         10          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
         11          1          0          0          0          0          0          0
         12          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
         13          1       2004          0          0          0          0          2
Tw       Cq
0.       0.
```

```
KDIM: GRID   SEGMENT    BCTYPE       ISTA       IEND       JSTA       JEND     NDATA
         1          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         2          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         3          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         4          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         5          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         6          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         7          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         8          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
         9          1         0          0          0          0          0         0
        10          1         0          0          0          0          0         0
        11          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
        12          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
        13          1      2004          0          0          0          0         2
Tw      Cq
0.      0.
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         1         0        02
      ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
         0        .3        .3        .3         0        .3        .3        .3
      NCYC    MGLEVG     NEMGL     NITFO
      3000        02        00         0
      MIT1      MIT2      MIT3      MIT4      MIT5
        01        01        01        01        01
  1-1 BLOCKING DATA:
     NBLI
        0
  NUMBER   GRID   :   ISTA   JSTA   KSTA   IEND   JEND   KEND   ISVA1   ISVA2
  NUMBER   GRID   :   ISTA   JSTA   KSTA   IEND   JEND   KEND   ISVA1   ISVA2
 PATCH SURFACE DATA:
   NINTER
       -1
PLOT3D OUTPUT:
 grid iptyp  ista  iend  iinc  jsta  jend  jinc  ksta  kend  kinc
    1     0     0     0     0     0     0     0     0     0     0
    2     0     0     0     0     0     0     0     0     0     0
    3     0     0     0     0     0     0     0     0     0     0
    4     0     0     0     0     0     0     0     0     0     0
    5     0     0     0     0     0     0     0     0     0     0
    6     0     0     0     0     0     0     0     0     0     0
    7     0     0     0     0     0     0     0     0     0     0
    8     0     0     0     0     0     0     0     0     0     0
    9     0     0     0     0     0     0     0     0     0     0
   10     0     0     0     0     0     0     0     0     0     0
   11     0     0     0     0     0     0     0     0     0     0
   12     0     0     0     0     0     0     0     0     0     0
   13     0     0     0     0     0     0     0     0     0     0
 MOVIE
    0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
CONTROL SURFACES:
```

```
  NCS
    4
 GRID   ISTA   IEND   JSTA   JEND   KSTA   KEND IWALL INORM
    1      0      0      1      1      0      0     0    -1
    7      0      0      1      1      0      0     0    -1
    8      0      0      1      1      0      0     0    -1
   13      0      0     49     49      0      0     0     1
```

The `inflow.data` file is:

```
 auxiliary bc data, j-face of block 1, multistream nozzle
40,  2*1
5
 40*0.4499999999999993,  40*1.893999999999998,  16*0.653000000000001,
8*1.685000000000004,  16*0.653000000000001,  40*0.,  40*0.
```

After running this test case, the residual history and mass flow convergence history shown in Figure 9-17 results. Also, a plot of Mach contours should have the flow features of those plotted in Figure 9-16.

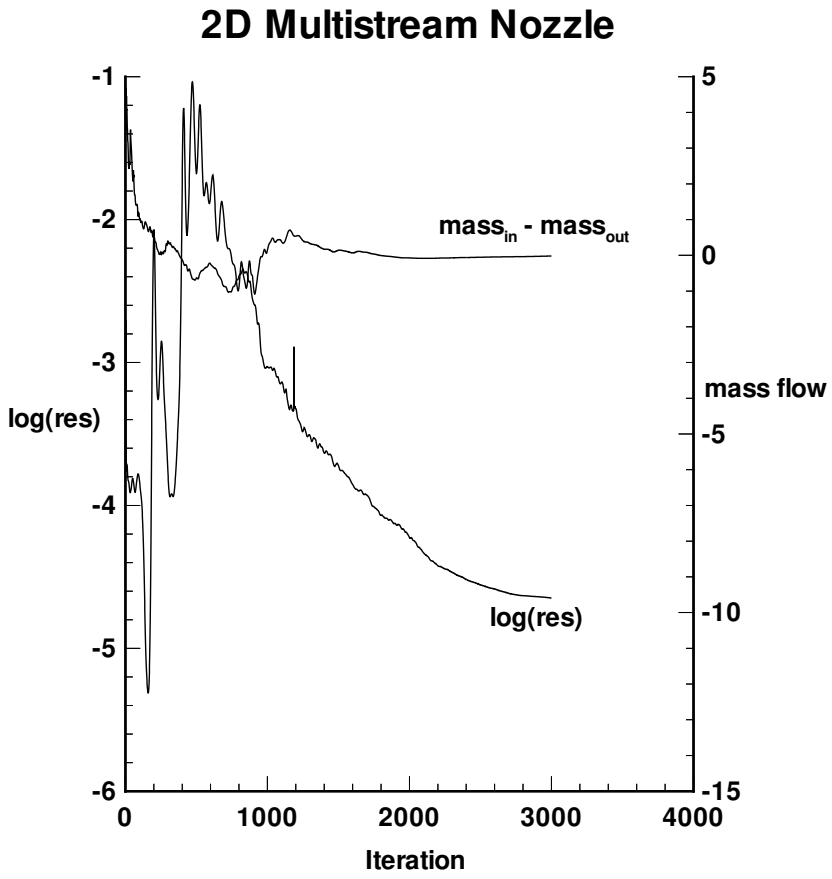


Figure 9-17. Multistream nozzle case residual and mass flow convergence history.

9.1.8 Rotor Stator

This case simulates, in two dimensions, the unsteady flow through a single stage turbine in which the ratio of stator to rotor blades is 3:4. The axial gap between the two blades is 50% of the blade chord. The case exercises a number of capabilities of CFL3D including unsteady flow, moving (translating) zones, dynamic patching between zones in relative motion, grid overlapping, and boundary conditions with user-defined input.

The original grid for this case was provided by D. J. Dorney[17] of Western Michigan University, although the grid given out for the test case contains only half the number of points of the original grid. The grid consists of fourteen zones with a total of 18374 points in one plane. A close-up of the grid near the airfoil is shown in Figure 9-18. The grid zones communicate with one another through both patching and overlapping. At a time step of 1.0, it takes 270 time steps for the eight rotor zones (containing four blades) to completely traverse the six stator zones (containing three blades). The rotor zones are reset after each complete traverse. The input file is set for 1500 time steps (using five multigrid sub-iterations per time step), which is sufficient to establish a time-periodic solution.The memory requirement for this example is 4.0 million words. A typical timing for this case (1500 time steps) is 4205 CPU seconds on a Cray YMP (NASA LaRC's Sabre as of October 1996). On a DEC Alpha workstation, the timing is 18303 CPU seconds, using single precision (as of June 1996).

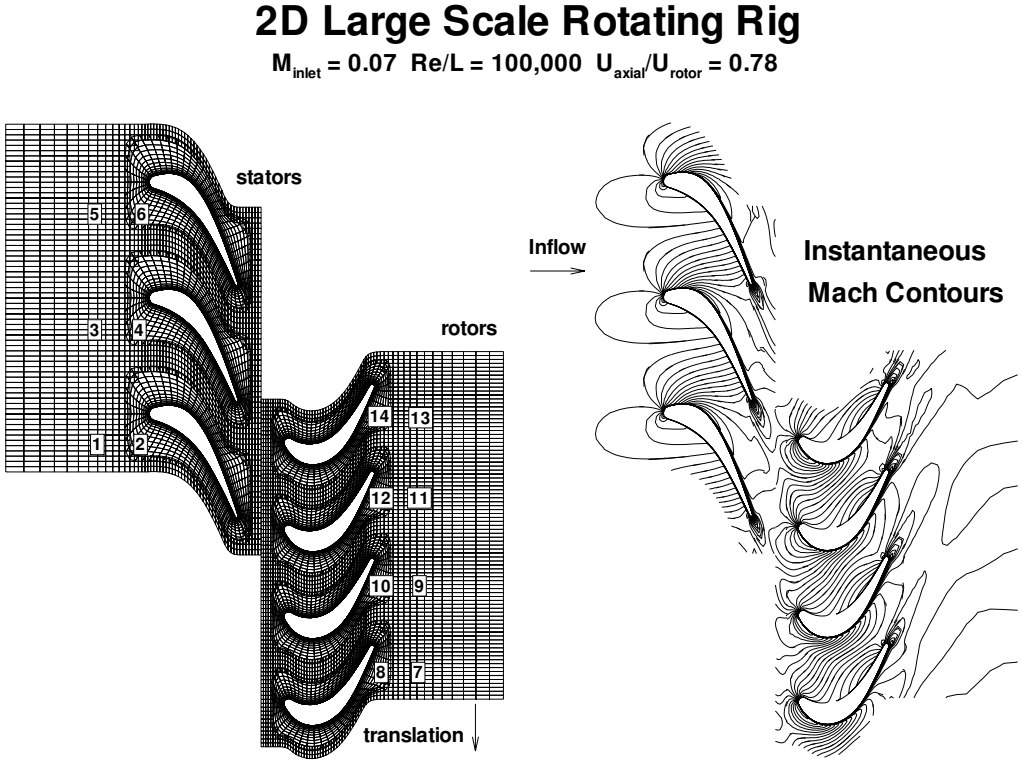


Figure 9-18. Fourteen zone rotor-stator grid system and Mach contours.

9.1.8.1 Experimental Conditions

The experimental blade count was 22 stator blades and 28 rotor blades. In order to run an exact simulation, a minimum of 11 stator blades and 14 rotor blades would be required (Dring et.al[18]). To reduce the problem size in the computation, the ratio of stator to rotor blades was reduced to 3:4 (equivalent to 21:28), and the stators were scaled by a factor of 22/21 to maintain the same pitch-to-chord ratio as in the experiment. The experimental set-up was a three-dimensional configuration; the corresponding 2-d simulation was set up from conditions at the mid-span radius of 27 inches, for a rotor speed of 410 rpm, with a nominal axial velocity of 75 feet/second. The inlet Mach number in the experiment was approximately 0.07, and the Reynolds number/inch was approximately 100,000.

9.1.8.2 Input Setup

Inlet conditions are used as the reference conditions, so **xmach** = 0.07. The grid is full scale, with dimensions in inches. Therefore, **reue** = 0.1. The inlet temperature was assumed to be $60°F$, so **tinf** = $520°R$.

Boundary condition type 2003 is used to specify total pressure and total temperature at the inlet. From isentropic flow relations or tables, for an inlet flow Mach number of 0.07,

$$\begin{aligned} M_{inlet} &= 0.07 \\ \frac{p_{t,\,inlet}}{p_\infty} &= 1.0035 \\ \frac{T_{t,\,inlet}}{T_\infty} &= 1.0010 \end{aligned} \tag{9-12}$$

Also, alphae = betae = 0 (purely axial flow is assumed).

Boundary condition type 2002 is used to specify an exit pressure. Dorney gives a ratio of static pressure to inlet total pressure = 0.963 at the rotor trailing edge plane. Assuming this value to hold at the exit as well gives

$$\frac{p_{exit}}{p_\infty} = \frac{p_{exit}}{p_{t,\,inlet}} \frac{p_{t,\,inlet}}{p_\infty} = 0.963 \times 1.0035 = 0.967 \tag{9-13}$$

Note that the inflow Mach number used in boundary condition type 2003 is an estimate; if the exit pressure were not set correctly, the computed inflow Mach number would not be close to the specified inflow value (when a time-periodic state is reached or at convergence in steady state). By specifying control surfaces at the inflow plane, the user is able to verify after the computation is complete that the average inflow Mach number is approximately 0.071; this was deemed to be close enough to the desired value. If desired,

the exit pressure could be adjusted (raised in this case) and the solution re-run until a new time-periodic solution (and a new inlet Mach number) is established.

It should be noted that the input grid is in PLOT3D format, with y as the "up" direction (**ialph** = 1; z is the spanwise, 2-d direction). However, the grid motion parameters *must* be set as if z is the up direction. Recall that if the input grid has y as the up direction, CFL3D will internally swap y and z so that the code always computes on a grid in which z is up. (See the caution in "LT35 - Translational Information and Velocities" on page 44.)

Given the rotor speed and mid-span radius, the translational velocity for a 2-d simulation corresponding to the mid-span radius is

$$\tilde{w}_{trans} = \omega r = (410/60 \times 2\pi)(27/12) = 96.6 \text{ feet/second} \quad \textbf{(9-14)}$$

This gives

$$\frac{\tilde{u}_{axial}}{\tilde{w}_{trans}} = \frac{75}{96.6} = 0.78 \quad \textbf{(9-15)}$$

The input value **wtrans** is $\tilde{w}_{trans}/\tilde{a}_{\infty}$, so with the reference Mach number 0.07, **wtrans** = 0.07/0.78 = (–) 0.0897 (the negative gives a downward rotor motion).

In order to be able to run an arbitrarily long simulation, the grid resetting option was employed. The top-to-bottom length of the grid is 24.23514 inches and the rotor and stator zones start out in alignment, so **dzmax** = 24.23514. Thus the rotor zones are reset whenever the displacement exceeds 24.23514 inches.

9.1.8.3 Running CFL3D

Besides the CFL3D code, the following files are needed to run this test case:

File	Description
`lsrr.inp`	input for CFL3D
`lsrr_coarse.p3d_fmt`	formatted single plane grid
`fmttobin_p3d.f`	converter for creating 2 grid planes
`mag1.h`	parameters for MaGGiE makefile
`maggie.inp`	input for MaGGiE

The steps for running this case on the DEC are as follows:

Step 1

Compile the grid converter code:

```
cft77 fmttobin_p3d.f
```

Step 2

Link the grid converter object file:

```
segldr -o fmttobin_p3d fmttobin_p3d.o
```

Step 3

Run the grid converter program (the binary file `lsrr_coarse.p3d` will be output):

```
fmttobin_p3d
```

Step 4

Use the makefile to compile, link, and create the executable for the MaGGiE code (be sure `mag1.h` is in the current directory):

```
make -f makemaggie_cray
```

Step 5

Run the MaGGiE code (the file `ovrlp.bin` will be output):

```
maggie < maggie.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 7

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < lsrr.inp
```

Step 8

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 9

Run the CFL3D code:

```
cfl3d < lsrr.inp
```

The input file for this case is:

```
I/O FILES
lsrr_coarse.p3d
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
  2D model of Pratt & Whitney Large Scale Rotating Rig (LSRR) - 50% Axial Gap
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH    IHSTRY
     0.070     0.000       0.0       0.1     520.0         1         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000    0.0000    0.0000
        DT     IREST   IFLAGTS      FMAX     IUNST   CFL_TAU
   +1.0000         0       000      1.00         1        5.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
       -14        14         0       100         0         1      1500        -2
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         1         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
         1         0         0         0         0         0         5
      IDIM      JDIM      KDIM
       002        55        23
       002        61        21
       002        55        23
       002        61        21
       002        55        23
       002        61        21
       002        61        23
       002        61        21
       002        61        23
       002        61        21
       002        61        23
       002        61        21
       002        61        23
       002        61        21
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
        00        00       000       000         0      0000
```

```
            00        00       000       000         0      0000
            00        00       000       000         0      0000
         INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
             0         0         0         0         0         0         0         0
      IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
             1         1         1         0         0         0
       IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
             1         1         1     .3333     .3333     .3333
          GRID    NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
             1         1         1         1         1         1         1         1
             2         1         1         1         1         1         1         1
             3         1         1         1         1         1         1         1
             4         1         1         1         1         1         1         1
             5         1         1         1         1         1         1         1
             6         1         1         1         1         1         1         1
             7         1         1         1         1         1         1         1
             8         1         1         1         1         1         1         1
             9         1         1         1         1         1         1         1
            10         1         1         1         1         1         1         1
            11         1         1         1         1         1         1         1
            12         1         1         1         1         1         1         1
            13         1         1         1         1         1         1         1
            14         1         1         1         1         1         1         1
I0:       GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
             1         1      1002         0         0         0         0         0
             2         1      1002         0         0         0         0         0
             3         1      1002         0         0         0         0         0
             4         1      1002         0         0         0         0         0
             5         1      1002         0         0         0         0         0
             6         1      1002         0         0         0         0         0
             7         1      1002         0         0         0         0         0
             8         1      1002         0         0         0         0         0
             9         1      1002         0         0         0         0         0
```

```
        10          1       1002          0          0          0          0          0
        11          1       1002          0          0          0          0          0
        12          1       1002          0          0          0          0          0
        13          1       1002          0          0          0          0          0
        14          1       1002          0          0          0          0          0
IDIM: GRID   SEGMENT    BCTYPE       JSTA       JEND       KSTA       KEND     NDATA
         1          1       1002          0          0          0          0          0
         2          1       1002          0          0          0          0          0
         3          1       1002          0          0          0          0          0
         4          1       1002          0          0          0          0          0
         5          1       1002          0          0          0          0          0
         6          1       1002          0          0          0          0          0
         7          1       1002          0          0          0          0          0
         8          1       1002          0          0          0          0          0
         9          1       1002          0          0          0          0          0
        10          1       1002          0          0          0          0          0
        11          1       1002          0          0          0          0          0
        12          1       1002          0          0          0          0          0
        13          1       1002          0          0          0          0          0
        14          1       1002          0          0          0          0          0
J0:   GRID   SEGMENT    BCTYPE       ISTA       IEND       KSTA       KEND     NDATA
         1          1       2003          0          0          0          0          5
Mach  Pt/Pinf  Tt/Tinf  alpha  beta
0.07   1.0035   1.0010     0.    0.
         2          1          0          0          0          0          0          0
         3          1       2003          0          0          0          0          5
Mach  Pt/Pinf  Tt/Tinf  alpha  beta
0.07   1.0035   1.0010     0.    0.
         4          1          0          0          0          0          0          0
         5          1       2003          0          0          0          0          5
Mach  Pt/Pinf  Tt/Tinf  alpha  beta
0.07   1.0035   1.0010     0.    0.
         6          1          0          0          0          0          0          0
         7          1          0          0          0          0          0          0
         8          1          0          0          0          0          0          0
         9          1          0          0          0          0          0          0
        10          1          0          0          0          0          0          0
        11          1          0          0          0          0          0          0
        12          1          0          0          0          0          0          0
        13          1          0          0          0          0          0          0
        14          1          0          0          0          0          0          0
JDIM: GRID   SEGMENT    BCTYPE       ISTA       IEND       KSTA       KEND     NDATA
         1          1          0          0          0          0          0          0
         2          1          0          0          0          0          0          0
         3          1          0          0          0          0          0          0
         4          1          0          0          0          0          0          0
         5          1          0          0          0          0          0          0
         6          1          0          0          0          0          0          0
         7          1       2002          0          0          0          0          1
pexit/pinf
0.967
         8          1          0          0          0          0          0          0
         9          1       2002          0          0          0          0          1
pexit/pinf
0.967
        10          1          0          0          0          0          0          0
        11          1       2002          0          0          0          0          1
pexit/pinf
0.967
        12          1          0          0          0          0          0          0
        13          1       2002          0          0          0          0          1
pexit/pinf
0.967
        14          1          0          0          0          0          0          0
K0:   GRID   SEGMENT    BCTYPE       ISTA       IEND       JSTA       JEND     NDATA
         1          1          0          0          0          0          0          0
         2          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.         0.
         3          1          0          0          0          0          0          0
         4          1       2004          0          0          0          0          2
Tw/Tinf   C_q
```

```
0.        0.
          5          1          0          0          0          0          0          0
          6          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.        0.
          7          1          0          0          0          0          0          0
          8          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.        0.
          9          1          0          0          0          0          0          0
         10          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.        0.
         11          1          0          0          0          0          0          0
         12          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.        0.
         13          1          0          0          0          0          0          0
         14          1       2004          0          0          0          0          2
Tw/Tinf   C_q
0.        0.
KDIM: GRID   SEGMENT    BCTYPE       ISTA       IEND       JSTA       JEND     NDATA
          1          1          0          0          0          0          0          0
          2          1          0          0          0          0          0          0
          3          1          0          0          0          0          0          0
          4          1          0          0          0          0          0          0
          5          1          0          0          0          0          0          0
          6          1          0          0          0          0          0          0
          7          1          0          0          0          0          0          0
          8          1          0          0          0          0          0          0
          9          1          0          0          0          0          0          0
         10          1          0          0          0          0          0          0
         11          1          0          0          0          0          0          0
         12          1          0          0          0          0          0          0
         13          1          0          0          0          0          0          0
         14          1          0          0          0          0          0          0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         1         0        01
      ISSC  EPSSC(1)  EPSSC(2)  EPSSC(3)      ISSR  EPSSR(1)  EPSSR(2)  EPSSR(3)
         0        .3        .3        .3         0        .3        .3        .3
      NCYC    MGLEVG     NEMGL     NITFO
         5        02        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5
        01        01        01        01        01
   1-1 BLOCKING DATA:
      NBLI
        14
   NUMBER   GRID   :   ISTA   JSTA   KSTA   IEND   JEND   KEND   ISVA1   ISVA2
        1      2          1      1      1      2      1     21       1       3
        2      4          1      1      1      2      1     21       1       3
        3      6          1      1      1      2      1     21       1       3
        4      8          1      1      1      2      1     21       1       3
        5     10          1      1      1      2      1     21       1       3
        6     12          1      1      1      2      1     21       1       3
        7     14          1      1      1      2      1     21       1       3
        8      1          1      1      1      2     55      1       1       2
        9      7          1      1      1      2     61      1       1       2
       10      1          1      1     23      2     55     23       1       2
       11      3          1      1     23      2     55     23       1       2
       12      7          1      1     23      2     61     23       1       2
       13      9          1      1     23      2     61     23       1       2
       14     11          1      1     23      2     61     23       1       2
   NUMBER   GRID   :   ISTA   JSTA   KSTA   IEND   JEND   KEND   ISVA1   ISVA2
        1      2          1     61      1      2     61     21       1       3
        2      4          1     61      1      2     61     21       1       3
        3      6          1     61      1      2     61     21       1       3
        4      8          1     61      1      2     61     21       1       3
        5     10          1     61      1      2     61     21       1       3
        6     12          1     61      1      2     61     21       1       3
        7     14          1     61      1      2     61     21       1       3
        8      5          1      1     23      2     55     23       1       2
        9     13          1      1     23      2     61     23       1       2
```

```
      10      3          1      1      1      2     55      1       1       2
      11      5          1      1      1      2     55      1       1       2
      12      9          1      1      1      2     61      1       1       2
      13     11          1      1      1      2     61      1       1       2
      14     13          1      1      1      2     61      1       1       2
 PATCH SURFACE DATA:
   NINTER
        0
 PLOT3D OUTPUT:
 BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      1    001      1     01    999      1       1    999      1
     2      0      1    001      1     01    999      1       1    999      1
     3      0      1    001      1     01    999      1       1    999      1
     4      0      1    001      1     01    999      1       1    999      1
     5      0      1    001      1     01    999      1       1    999      1
     6      0      1    001      1     01    999      1       1    999      1
     7      0      1    001      1     01    999      1       1    999      1
     8      0      1    001      1     01    999      1       1    999      1
     9      0      1    001      1     01    999      1       1    999      1
    10      0      1    001      1     01    999      1       1    999      1
    11      0      1    001      1     01    999      1       1    999      1
    12      0      1    001      1     01    999      1       1    999      1
    13      0      1    001      1     01    999      1       1    999      1
    14      0      1    001      1     01    999      1       1    999      1
 MOVIE
     0
 PRINT OUT:
 BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
CONTROL SURFACES:
  NCS
    7
 GRID   ISTA  IEND  JSTA  JEND  KSTA  KEND IWALL INORM
    7      1     2   999   999     0     0     0     1
    9      1     2   999   999     0     0     0     1
   11      1     2   999   999     0     0     0     1
   13      1     2   999   999     0     0     0     1
    1      1     2     1     1     0     0     0     0
    3      1     2     1     1     0     0     0     0
    5      1     2     1     1     0     0     0     0
 MOVING GRID DATA - TRANSLATION
 NTRANS
      9
   LREF
    1.0
   GRID ITRANS   RFREQ   UTRANS   VTRANS   WTRANS
      7      1      0.       0.       0. -0.0897
      8      1      0.       0.       0. -0.0897
      9      1      0.       0.       0. -0.0897
     10      1      0.       0.       0. -0.0897
     11      1      0.       0.       0. -0.0897
     12      1      0.       0.       0. -0.0897
     13      1      0.       0.       0. -0.0897
     14      1      0.       0.       0. -0.0897
      0      1      0.       0.       0. -0.0897
   GRID  DXMAX   DYMAX    DZMAX
      7     0.      0. -24.23514
      8     0.      0. -24.23514
      9     0.      0. -24.23514
     10     0.      0. -24.23514
     11     0.      0. -24.23514
     12     0.      0. -24.23514
     13     0.      0. -24.23514
     14     0.      0. -24.23514
      0     0.      0. -24.23514
 MOVING GRID DATA - ROTATION
 NROTAT
      0
   LREF
   GRID IROTAT   RFREQ   OMEGAX   OMEGAY   OMEGAZ   XORIG   YORIG   ZORIG
   GRID   THXMAX   THYMAX   THZMAX
 DYNAMIC PATCH INPUT DATA
   NINTER
```

```
      7
INT   IFIT     LIMIT     ITMAX     MCXIE     MCETA       C-0     IORPH     ITOSS
  1      1         1        30         0         0         0         0         1
  2      1         1        30         0         0         0         0         1
  3      1         1        30         0         0         0         0         1
  4      1         1        30         0         0         0         0         1
  5      1         1        30         0         0         0         0         1
  6      1         1        30         0         0         0         0         1
  7      1         1        30         0         0         0         0         1
INT     TO      XIE1      XIE2      ETA1      ETA2       NFB
  1    122         0         0         0         0         6
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      721         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      921         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1121         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1321         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      721         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.  24.23514        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      921         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.  24.23514        0.        0.        0.
INT     TO      XIE1      XIE2      ETA1      ETA2       NFB
  2    322         0         0         0         0         6
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      921         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1121         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1321         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      721         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.  24.23514        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
      921         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.  24.23514        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1121         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.  24.23514        0.        0.        0.
INT     TO      XIE1      XIE2      ETA1      ETA2       NFB
  3    522         0         0         0         0         6
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1121         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
     1321         0         0         0         0        0.        0.
       DX        DY        DZ    DTHETX    DTHETY    DTHETZ
       0.        0.        0.        0.        0.        0.
     FROM      XIE1      XIE2      ETA1      ETA2     FACTJ     FACTK
```

```
       721        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.  24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       921        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.  24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
      1121        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.  24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
      1321        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.  24.23514      0.       0.       0.
INT     TO     XIE1     XIE2     ETA1     ETA2      NFB
  4    721        0        0        0        0        4
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       122        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       522        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       322        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       122        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
INT     TO     XIE1     XIE2     ETA1     ETA2      NFB
  5    921        0        0        0        0        5
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       122        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       322        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       522        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       322        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       122        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0. -24.23514      0.       0.       0.
INT     TO     XIE1     XIE2     ETA1     ETA2      NFB
  6   1121        0        0        0        0        5
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       322        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       522        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       122        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
        0.       0.       0.       0.       0.       0.
      FROM     XIE1     XIE2     ETA1     ETA2    FACTJ    FACTK
       522        0        0        0        0       0.       0.
        DX       DY       DZ   DTHETX   DTHETY   DTHETZ
```

```
      0.       0. -24.23514       0.       0.       0.
    FROM     XIE1      XIE2     ETA1     ETA2    FACTJ    FACTK
     322        0         0        0        0       0.       0.
      DX       DY        DZ   DTHETX   DTHETY   DTHETZ
      0.       0. -24.23514       0.       0.       0.
INT   TO     XIE1      XIE2     ETA1     ETA2      NFB
  7 1321        0         0        0        0        4
    FROM     XIE1      XIE2     ETA1     ETA2    FACTJ    FACTK
     522        0         0        0        0       0.       0.
      DX       DY        DZ   DTHETX   DTHETY   DTHETZ
      0.       0.        0.       0.       0.       0.
    FROM     XIE1      XIE2     ETA1     ETA2    FACTJ    FACTK
     322        0         0        0        0       0.       0.
      DX       DY        DZ   DTHETX   DTHETY   DTHETZ
      0.       0.        0.       0.       0.       0.
    FROM     XIE1      XIE2     ETA1     ETA2    FACTJ    FACTK
     122        0         0        0        0       0.       0.
      DX       DY        DZ   DTHETX   DTHETY   DTHETZ
      0.       0.        0.       0.       0.       0.
    FROM     XIE1      XIE2     ETA1     ETA2    FACTJ    FACTK
     522        0         0        0        0       0.       0.
      DX       DY        DZ   DTHETX   DTHETY   DTHETZ
      0.       0. -24.23514       0.       0.       0.
```

The convergence histories for residual, mass flow, and rotor lift coefficient as shown in Figure 9-19 should be obtained.

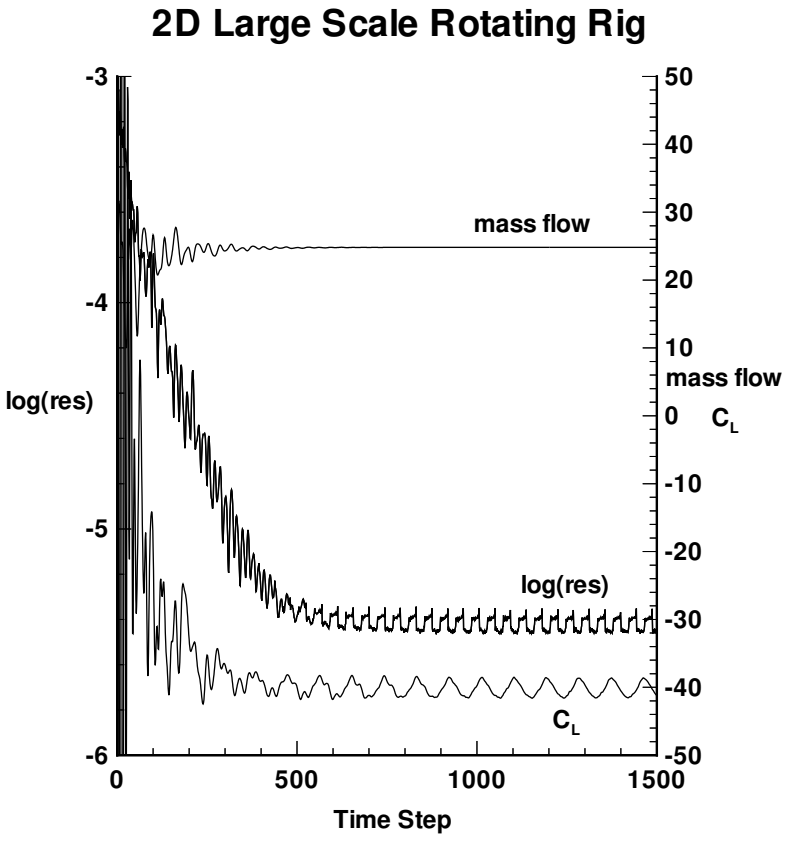


Figure 9-19. Convergence histories for rotor-stator case.

9.2 Three-dimensional Test Cases

9.2.1 Axisymmetric Bump Flow

This test case solves for the turbulent flow over an axisymmetric bump. The flow is modeled in 3-d using two computational planes (separated by an angle of 1 degree), with periodic boundary conditions; hence **bctype** is 2005 and **dthtx** is –1.0 and 1.0 on the I0 and IDIM boundaries, respectively. The grid consists of a single zone with a total of 36562 points. The memory requirement for this example is 4.9 million words. A typical timing for this case is 1026 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of October 1996). A close-up of the grid near the bump is shown in Figure 9-20.

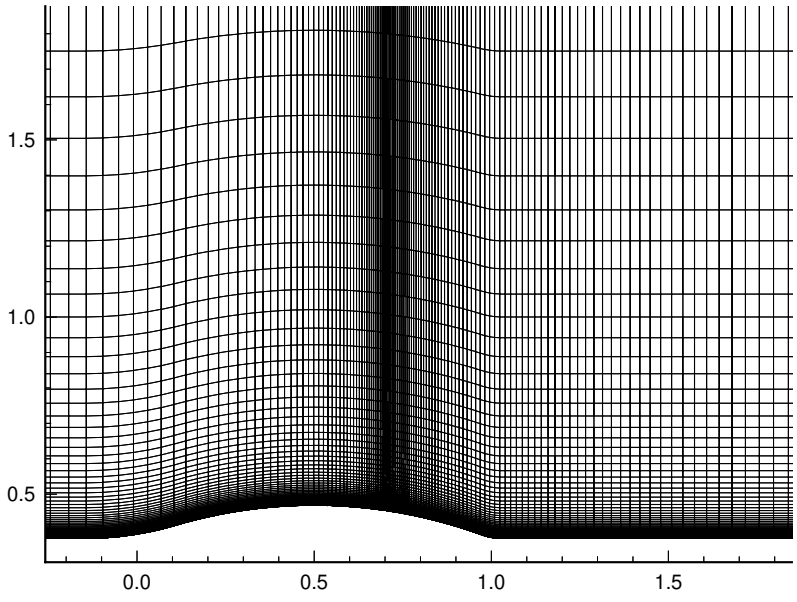


Figure 9-20. Axisymmetric bump grid.

Besides the CFL3D code, the following files are needed to run this test case:

File	Description
`bumpv5periodic.inp`	input for CFL3D
`bump.grd`	formatted single plane grid
`gridaxi.f`	converter for creating 2 grid planes

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 gridaxi.f
```

Step 2

Link the grid converter object file:

```
segldr -o gridaxi gridaxi.o
```

Step 3

Run the grid converter program (the binary file `bumpgrd.bin` will be output):

```
gridaxi
```

In answer to the question, type:

```
bump.grd
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < bumpv5periodic.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < bumpv5periodic.inp
```

The input file for this case is:

```
I/O FILES
bumpgrd.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
```

```
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
    Axisymmetric bump flow, 3-d, using 2 planes and periodic BCs
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
    0.8750    00.000       0.0    02.660     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.00000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
   -05.000         0       000    5.0000         0       10.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         1      1200         0         0         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0       001         0         0        07
    IDIM    JDIM    KDIM
      02     181     101
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
         0         0         0         0         0         0
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         3         3         3
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1    0.3333    0.3333    0.3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         1         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      2005         0         0         0         0         4
              NBLP     DTHTX     DTHTY     DTHTZ
                 1      -1.0        0.        0.
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      2005         0         0         0         0         4
              NBLP     DTHTX     DTHTY     DTHTZ
                 1      +1.0        0.        0.
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1003         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1003         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      2004         0         0         0         0         2
            TWTYPE        CQ
               0.        0.
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         0         0        02
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
      1100        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7      MIT8
        01        01        01        01        01         1         1         1
   1-1 BLOCKING DATA:
      NBLI
         0
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      0      0      0      0      0      0      0      0      0
 IMOVIE
      0
  PRINT OUT:
  BLOCK IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      1      1      1      0      0      0      1      1      1
  CONTROL SURFACE:
```

```
NCS
  0
GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After running this test case, the convergence history plots shown in Figure 9-21 should be duplicated.

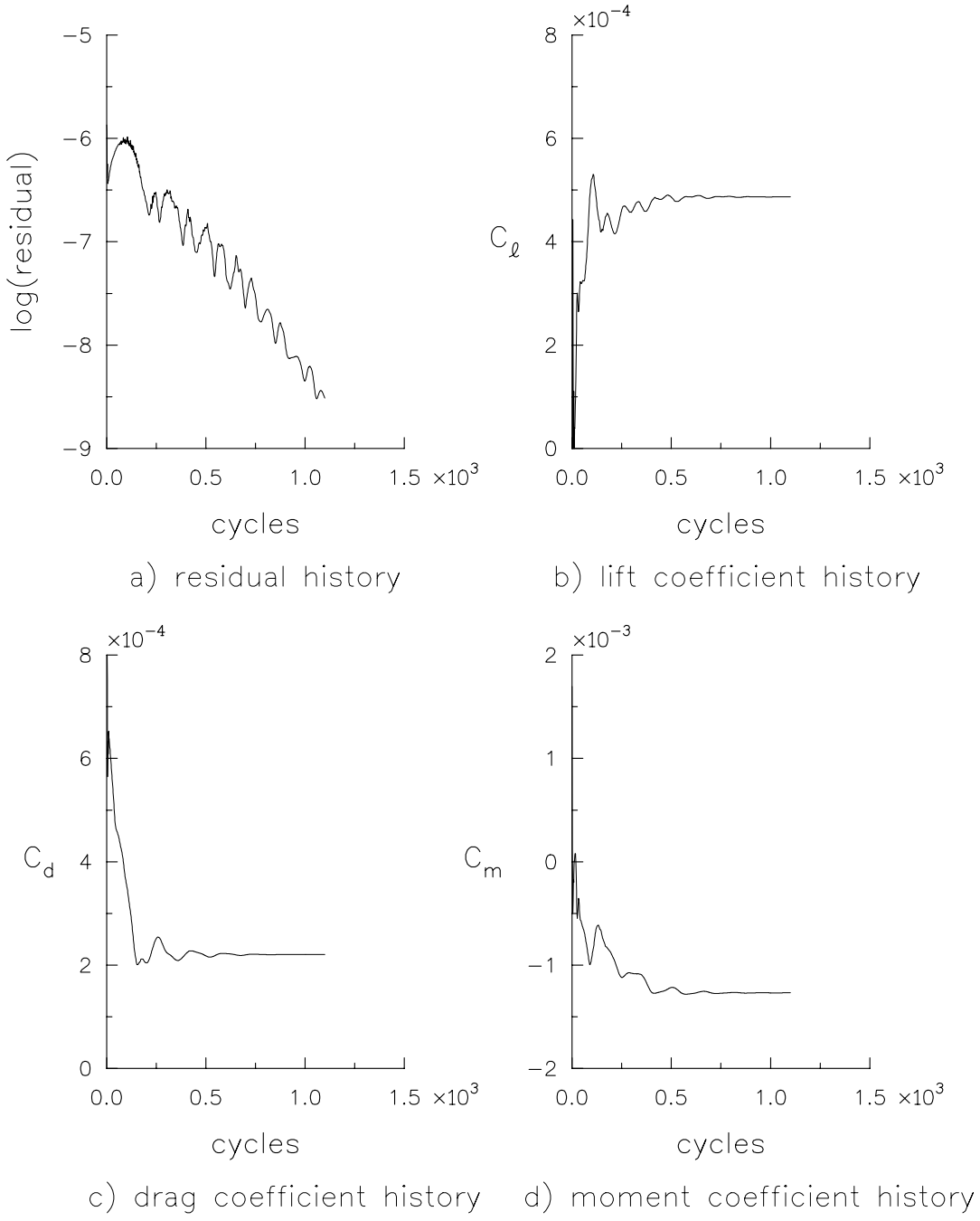


Figure 9-21. Residual and coefficient histories for axisymmetric bump flow case $M_\infty = 0.875$, $Re_{\tilde{L}_R} = 2.66 \times 10^6$.

Also, a result such as that shown in Figure 9-22 should be obtained. In the figure, surface pressure coefficients are plotted along with experimental data for this case. The computational surface pressures can be obtained from file `cfl3d.prout`. Experimental surface pressure coefficients from Bachalo et. al[8] are included with this test case for comparison purposes. The file is called `bumpcpdata.dat`.

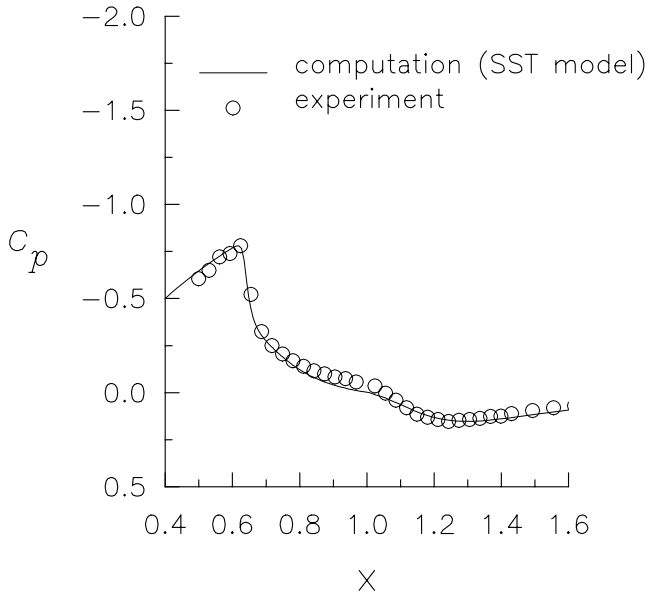


Figure 9-22. Pressure coefficients for axisymmetric bump case $M_\infty = 0.875$, $Re_{\tilde{L}_R} = 2.66 \times 10^6$.

9.2.2 F-5 Wing

The inviscid flow over an F-5 wing is solved in this test case. The grid consists of a single grid zone with a C-H mesh topology and is composed of 210,177 points. The memory requirement for this example is 10.5 million words. A typical timing for this case is 984 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of September 1996). The wing surface grid and wake, as well as the plane of symmetry grid are illustrated in Figure 9-23.

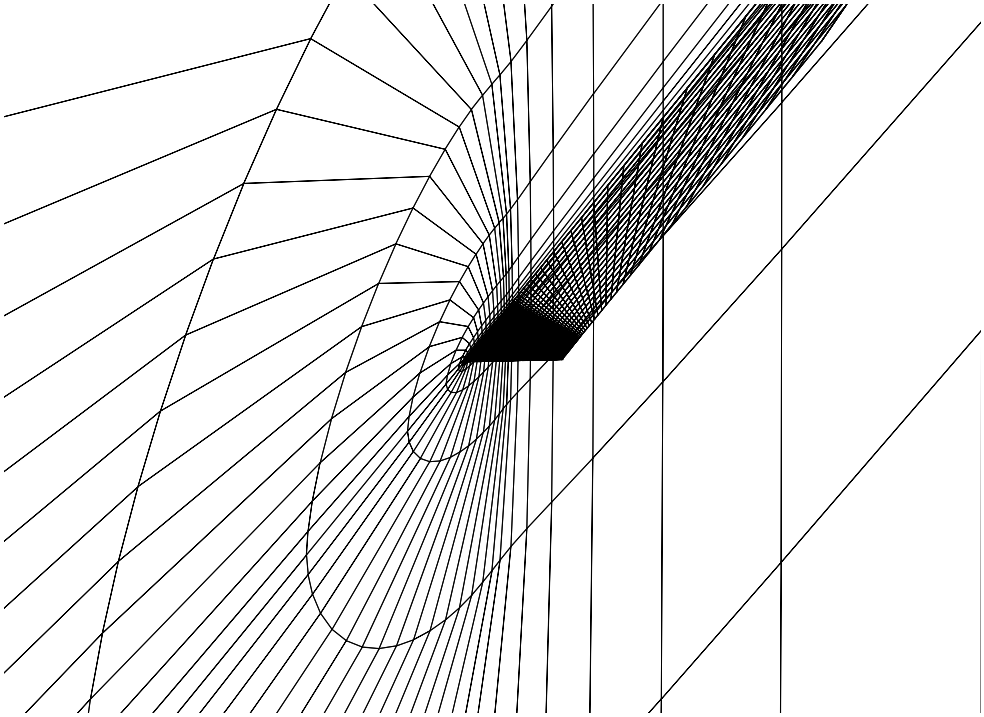


Figure 9-23. Single zone F-5 wing surface grid and plane of symmetry grid.

Besides the CFL3D code the following files are needed to run this test case:

File	Description
`f5wing.inp`	input for CFL3D
`f5grid.dat`	formatted wing section grid
`f5wing_grid.f`	grid converter

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 f5wing_grid.f
```

Step 2

Link the grid converter object file:

```
segldr -o f5wing_grid f5wing_grid.o
```

Step 3

Run the grid converter program to generate the 3-d volume grid (the binary file `f5wing.grd` will be output):

```
f5wing_grid
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < f5wing.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < f5wing.inp
```

The input file for this case is:

```
 I/O FILES:
f5wing.grd
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
     F5 Wing, cfl3d type grid
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
```

```
     0.950    00.000       0.0     0.950     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
   1.00000   1.00000    1.0000   0.25000      0.00      0.00
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
    -5.000         0       000     1.000         0       10.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         0         0       100         0         0         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0         1         0         0         0
      IDIM      JDIM      KDIM
        33       193        33
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
        00        00       000       000         0      0000
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         3         3         3
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1     .3333     .3333     .3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         4         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1001         0         0         0         0         0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND     NDATA
         1         1      1002         0         0         0         0         0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1003         0         0         0         0         0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND     NDATA
         1         1      1003         0         0         0         0         0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1         0         1        33         1        41         0
         1         2      1005         1        21        41       153         0
         1         3         0        21        33        41       153         0
         1         4         0         1        33       153       193         0
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND     NDATA
         1         1      1003         0         0         0         0         0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         3         1         0         0        02
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         0       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
       200        01        00       000
       200        02        00       000
       200        03        00       000
      MIT1      MIT2      MIT3      MIT4      MIT5
        01        01        01        01        01
        01        01        01        01        01
        01        01        01        01        01
  1-1 BLOCKING DATA:
     NBLI
        2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           1      1      1     33     41      1      1      2
      2      1          21     41      1     33     97      1      1      2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           1    193      1     33    153      1      1      2
      2      1          21    153      1     33     97      1      1      2
  PATCH SURFACE DATA:
    NINTER
        0
  PLOT3D OUTPUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
  MOVIE
      0
  PRINT OUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After this test case is run, the convergence history, found in file `cfl3d.res`, should look like that plotted in Figure 9-24. The two sharp spikes in the residual history are at the iterations at which the grid levels change in the mesh sequencing procedure.

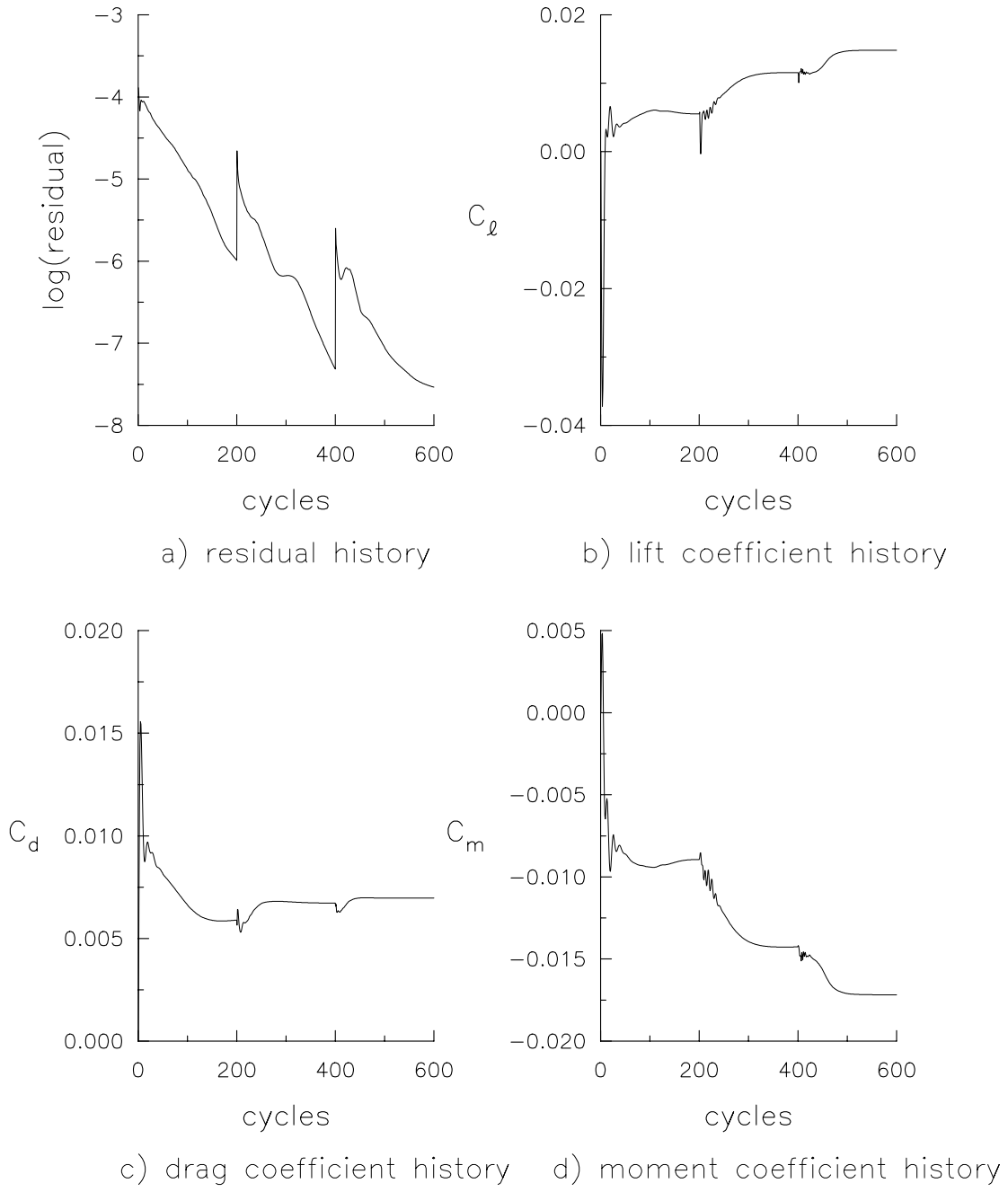


Figure 9-24. Convergence histories for single grid F-5 wing case; $\alpha = 0.0$, $M_\infty = 0.95$.

9.2.3 Onera M-6 Wing

In this case, a turbulent Navier-Stokes computation is performed over the Onera M-6 wing, on a coarse grid, using a grid in PLOT3D-type format is performed. The grid consists of a single grid zone with a C-O mesh topology and is composed of 41,225 points. (Keep in mind that one needs a grid at least double this size in each direction, e.g. $193 \times 49 \times 33$ or larger, to actually resolve the flow. A coarser grid is used here to shorten the test run.) The wing surface grid and wake, as well as the plane of symmetry grid are illustrated in Figure 9-25. The memory requirement for this example is 3.4 million words. A typical timing for this case is 453 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of September 1996).

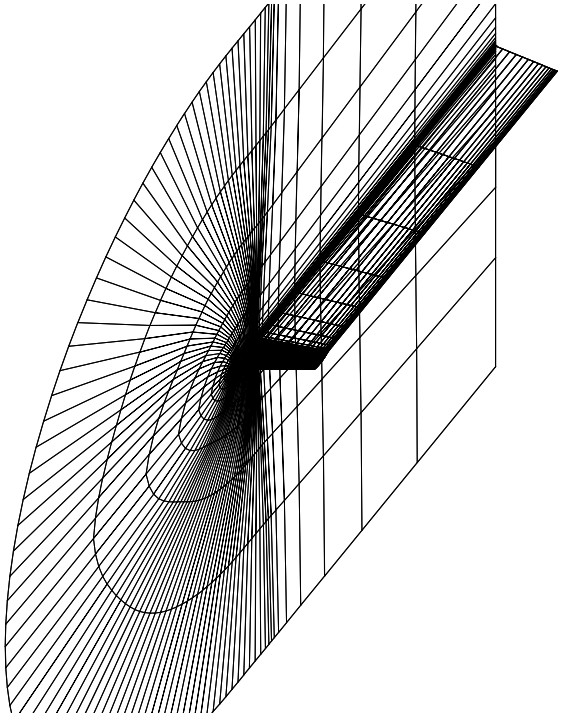


Figure 9-25. Single zone Onera wing surface grid and plane of symmetry grid.

The viscous direction in this PLOT3D-formatted grid is taken as the j direction rather than the k direction as generally recommended. (Due to the order in which CFL3D approximately factors the three index directions, the CFL3D code is usually most efficient when the primary viscous direction is taken as the k direction.) In this case, the convergence is not hurt by the altered directionality. (In some cases, however, it can be!) Note, however, that this case *is* more efficient (CPU timewise) when run on a vector machine with k as the viscous direction, due to the distribution of individual i, j, k index lengths and the way the code is vectorized. For this case, the difference on Sabre is a factor of 17% (with a CFL3D-type k viscous grid, the code runs in 374 seconds as opposed to 453 sec-

onds). It is possible to duplicate this result by changing the hard-wired parameter `iplot3d` to 0 in the `form2bin.f` file and using the input file `oneram6.inp_cfl3d` instead of `oneram6.inp_p3d`. This exercise will demonstrate the differences between PLOT3D-type and CFL3D-type grids, as well as the corresponding differences in the input files.

Besides the CFL3D code the following files are needed to run this test case:

File	Description
`oneram6.inp_p3d`	input for CFL3D
`m6_i97.fmt_p3d`	formatted grid
`form2bin.f`	grid converter

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 form2bin.f
```

Step 2

Link the grid converter object file:

```
segldr -o form2bin form2bin.o
```

Step 3

Run the grid converter program to generate the 3-d volume grid (the binary file `m6_i97.grd_p3d` will be output):

```
form2bin
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < oneram6.inp_p3d
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < oneram6.inp_p3d
```

The input file for this case is:

```
 I/O FILES:
m6_i97.grd_p3d
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
     ONERA M6 Wing, plot3d type grid, coarse grid
      XMACH      ALPHA       BETA   REUE,MIL    TINF,DR      IALPH      IHIST
     0.8400     03.060        0.0    21.660      540.0          1          0
       SREF       CREF       BREF       XMC        YMC        ZMC
    0.53080    1.00000     3.9249   0.00000       0.00       0.00
         DT      IREST    IFLAGTS      FMAX      IUNST     CFLTAU
     -5.000          0        000     1.000          0        10.
      NGRID    NPLOT3D     NPRINT     NWREST       ICHK        I2D     NTSTEP        ITA
         -1          1          0        100          0          0          0          1
        NCG        IEM   IADVANCE     IFORCE   IVISC(I)   IVISC(J)   IVISC(K)
          2          0          0         10          0          5          0
       IDIM       JDIM       KDIM
         97         25         17
     ILAMLO     ILAMHI     JLAMLO     JLAMHI     KLAMLO     KLAMHI
         00         00        000        000          0       0000
      INEWG     IGRIDC         IS         JS         KS         IE         JE         KE
          0          0          0          0          0          0          0          0
   IDIAG(I)   IDIAG(J)   IDIAG(K)   IFLIM(I)   IFLIM(J)   IFLIM(K)
          1          1          1          3          3          3
    IFDS(I)    IFDS(J)    IFDS(K)   RKAP0(I)   RKAP0(J)   RKAP0(K)
          1          1          1      .3333      .3333      .3333
       GRID      NBCI0    NBCIDIM     NBCJ0    NBCJDIM     NBCK0    NBCKDIM     IOVRLP
          1          1          1          3          1          1          1          0
I0:    GRID    SEGMENT     BCTYPE       JSTA       JEND       KSTA       KEND      NDATA
          1          1       1003          0          0          0          0          0
IDIM:  GRID    SEGMENT     BCTYPE       JSTA       JEND       KSTA       KEND      NDATA
          1          1       1003          0          0          0          0          0
J0:    GRID    SEGMENT     BCTYPE       ISTA       IEND       KSTA       KEND      NDATA
          1          1          0          1         13          0          0          0
          1          2       2004         13         85          1         17          2
                TWTYPE         CQ
                    0.         0.
          1          3          0         85         97          0          0          0
JDIM:  GRID    SEGMENT     BCTYPE       ISTA       IEND       KSTA       KEND      NDATA
          1          1       1003          0          0          0          0          0
K0:    GRID    SEGMENT     BCTYPE       ISTA       IEND       JSTA       JEND      NDATA
          1          1       1001          0          0          0          0          0
KDIM:  GRID    SEGMENT     BCTYPE       ISTA       IEND       JSTA       JEND      NDATA
          1          1          0          0          0          0          0          0
       MSEQ     MGFLAG     ICONSF        MTT       NGAM
          2          1          0          0         02
       ISSC  EPSSSC(1)  EPSSSC(2)  EPSSSC(3)       ISSR  EPSSSR(1)  EPSSSR(2)  EPSSSR(3)
          0        0.3        0.3        0.3          0        0.3        0.3        0.3
       NCYC     MGLEVG      NEMGL      NITFO
```

```
      200        02        00       000
      300        02        00       000
     MIT1      MIT2      MIT3      MIT4      MIT5
       01        01        01        01        01
       01        01        01        01        01
  1-1 BLOCKING DATA:
     NBLI
        2
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1            1      1      1     13      1     17      1      3
     2      1            1      1     17     49     25     17      1      2
NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
     1      1           97      1      1     85      1     17      1      3
     2      1           97      1     17     49     25     17      1      2
 PATCH SURFACE DATA:
   NINTER
        0
 PLOT3D OUTPUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
     1      0      0      0      0      0      0      0      0      0      0
 MOVIE
     0
 PRINT OUT:
  GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
 CONTROL SURFACE:
 NCS
   0
  GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After this test case is run, the convergence histories, found in file `cfl3d.res`, should look like those plotted in Figure 9-26. The sharp spikes in the plots indicate the iteration at which the grid level changes in the mesh sequencing process.

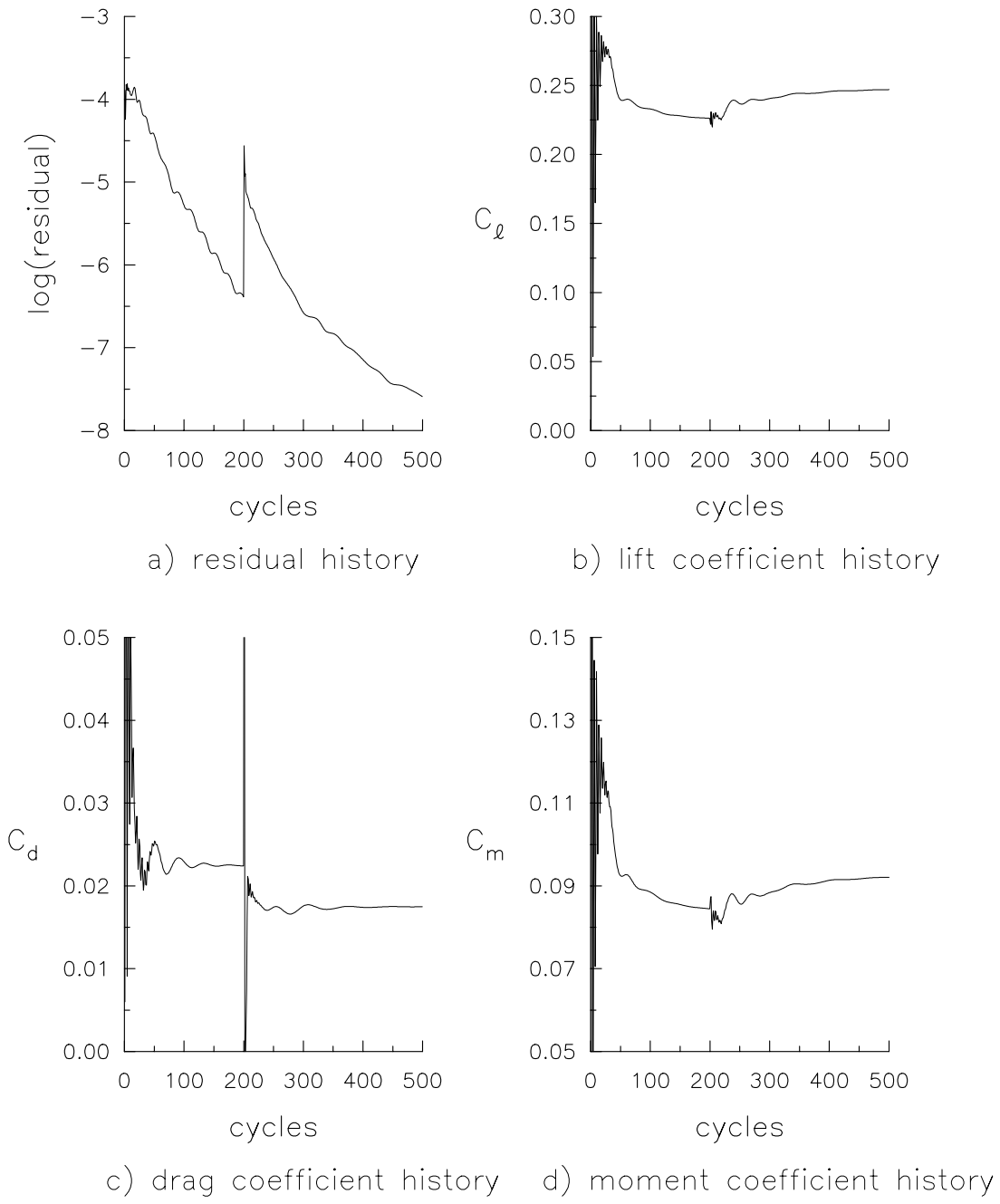


Figure 9-26. Convergence histories for single grid Onera wing case; $\alpha = 3.06$, $M_\infty = 0.84$.

9.2.4 Delta Wing

The laminar flow over a 75° swept delta wing is solved in this test case. The grid consists of a single grid zone with 156,325 points. (Note that this grid is coarser than what one would normally use to resolve this flow.) The memory requirement for this example is 8.0 million words. A typical timing for this case is 2236 CPU seconds on a CRAY YMP (NASA LaRC's Sabre as of September 1996). The surface grid ($k = 1$) and a trailing edge grid plane are shown in Figure 9-27.

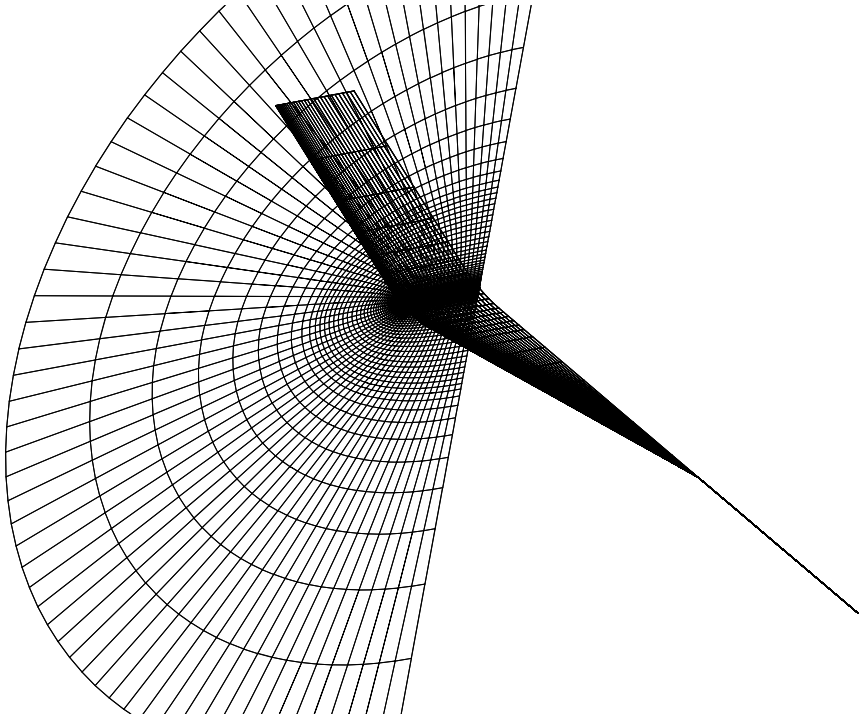


Figure 9-27. Single zone delta wing surface grid and trailing edge plane grid.

Besides the CFL3D code the following files are needed to run this test case:

File	Description
`delta.inp`	input for CFL3D
`delta.fmt`	formatted grid
`form2bin.f`	grid converter

The steps for running this case on the YMP are as follows:

Step 1

Compile the grid converter code:

```
cft77 form2bin.f
```

Step 2

Link the grid converter object file:

```
segldr -o form2bin form2bin.o
```

Step 3

Run the grid converter program to generate the 3-d volume grid (the binary file `delta.bin` will be output):

```
form2bin
```

Step 4

Use the makefile to compile, link, and create the executable for the `precfl3d` code (be sure `precfl.h` is in the current directory):

```
make -f makeprecfl3d_cray
```

Step 5

Run the `precfl3d` code (the `cflx.h` files will be output):

```
precfl3d < delta.inp
```

Step 6

Use the makefile to compile, link, and create the executable for the CFL3D code:

```
make -f makecfl3d_cray
```

Step 7

Run the CFL3D code:

```
cfl3d < delta.inp
```

The input file for this case is:

```
I/O FILES
delta.bin
plot3dg.bin
plot3dq.bin
cfl3d.out
cfl3d.res
cfl3d.turres
cfl3d.blomax
cfl3d.out15
cfl3d.prout
cfl3d.out20
ovrlp.bin
patch.bin
restart.bin
 75 Degree Swept Delta Wing - 37x65x65 - Laminar
     XMACH     ALPHA      BETA  REUE,MIL   TINF,DR     IALPH     IHIST
```

```
     0.300    20.500       0.0     0.500     460.0         0         0
      SREF      CREF      BREF       XMC       YMC       ZMC
    .13398   1.00000    .53590   0.25000        0.        0.
        DT     IREST   IFLAGTS      FMAX     IUNST    CFLTAU
     -10.0         0         0       1.0         0       10.
     NGRID   NPLOT3D    NPRINT    NWREST      ICHK       I2D    NTSTEP       ITA
         1         1         0      0100         0         0         1         1
       NCG       IEM  IADVANCE    IFORCE  IVISC(I)  IVISC(J)  IVISC(K)
         2         0         0       001         0         0         1
    IDIM    JDIM    KDIM
      37      65      65
    ILAMLO    ILAMHI    JLAMLO    JLAMHI    KLAMLO    KLAMHI
        00        00       000       000         0      0000
     INEWG    IGRIDC        IS        JS        KS        IE        JE        KE
         0         0         0         0         0         0         0         0
  IDIAG(I)  IDIAG(J)  IDIAG(K)  IFLIM(I)  IFLIM(J)  IFLIM(K)
         1         1         1         4         4         4
   IFDS(I)   IFDS(J)   IFDS(K)  RKAP0(I)  RKAP0(J)  RKAP0(K)
         1         1         1     .3333     .3333     .3333
      GRID     NBCI0   NBCIDIM     NBCJ0   NBCJDIM     NBCK0   NBCKDIM    IOVRLP
         1         1         1         1         1         3         1         0
I0:   GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1003         1        65         1        65        0
IDIM: GRID   SEGMENT    BCTYPE      JSTA      JEND      KSTA      KEND    NDATA
         1         1      1003         1        65         1        65        0
J0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1      1001         1        37         1        65        0
JDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      KSTA      KEND    NDATA
         1         1      1001         1        37         1        65        0
K0:   GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1      1011         1         9         1        65        0
         1         2      2004         9        25         1        65        2
               TWTYPE        CQ
                  -1.        0.
         1         3         0        25        37         1        65        0
KDIM: GRID   SEGMENT    BCTYPE      ISTA      IEND      JSTA      JEND    NDATA
         1         1      1003         1        37         1        65        0
      MSEQ    MGFLAG    ICONSF       MTT      NGAM
         1         1         1         0        02
      ISSC EPSSSC(1) EPSSSC(2) EPSSSC(3)      ISSR EPSSSR(1) EPSSSR(2) EPSSSR(3)
         1       0.3       0.3       0.3         0       0.3       0.3       0.3
      NCYC    MGLEVG     NEMGL     NITFO
       600        03        00      0000
      MIT1      MIT2      MIT3      MIT4      MIT5      MIT6      MIT7      MIT8
        01        01        01        01        01         1         1         1
   1-1 BLOCKING DATA:
      NBLI
         1
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           25      1      1     37     33      1      1      2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1           25     65      1     37     33      1      1      2
  PATCH SURFACE DATA:
    NINTER
         0
  PLOT3D OUTPUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
      1      0      0      0      0      0      0      0      0      0      0
  MOVIE
      0
  PRINT OUT:
   GRID IPTYPE ISTART   IEND   IINC JSTART   JEND   JINC KSTART   KEND   KINC
  CONTROL SURFACE:
  NCS
    0
   GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

After this test case is run, the convergence histories, found in file `cfl3d.res`, should look like those plotted in Figure 9-28.

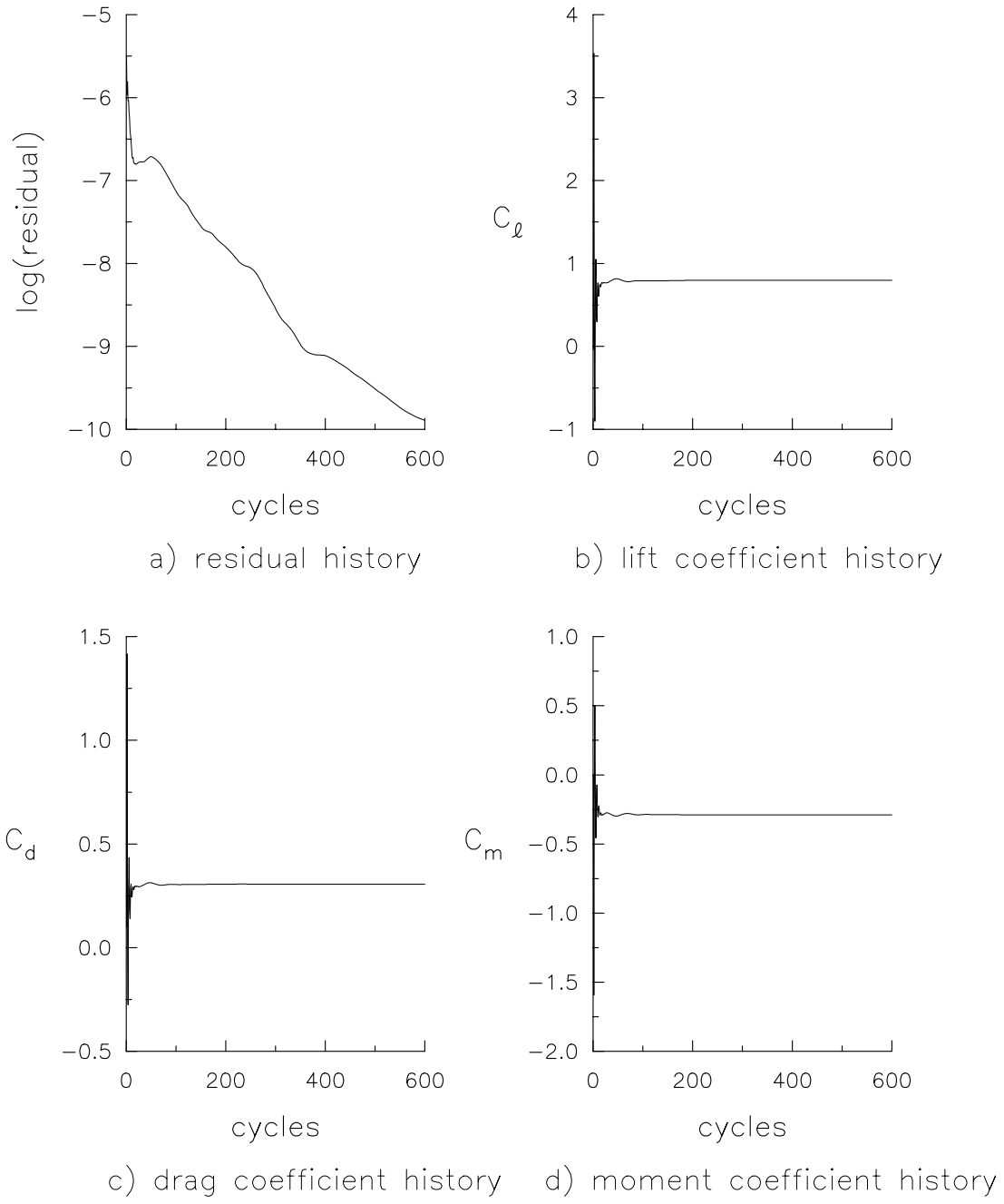


Figure 9-28. Convergence histories for single grid delta wing case; $\alpha = 20.5$.

CHAPTER 10

Troubleshooting

Mistakes made in setting up a problem for CFL3D are (hopefully) diagnosed by the code's error checks. The error messages, which are printed to unit 11 should be clear enough so that the user can easily identify and remedy the problem. This chapter contains a few general tips based on common user errors.

Unfortunately, it is sometimes the case with CFL3D (as with any CFD code) that the code blows up, even though the user has apparently done everything right. For example, if the code tries to calculate a negative square root, it will bomb with a floating point error. What to do in such cases is never clear and is often more of an art form than a science. Listed below are some things to try and/or look for when experiencing difficulty with CFL3D (when it blows up for no easily-apparent reason). These limited suggestions for what to try when problems occur are based in part on the code-developers' experiences and in part on what is heard from the user community:

1. When the code gives the message "negative volume" or blows up (end-of-file) when trying to read the grid, check the following:
   (a) Be sure the grid is written correctly, using the right-hand rule (see "The Right-Hand Rule" on page 67) for both x, y, z and i, j, k. Either CFL3D format or PLOT3D/TLNS3D format may be used. See "Grid File" on page 65 for a description of the two formats. Remember to set **ngrid** > 0 for CFL3D format and **ngrid** < 0 for PLOT3D/TLNS3D format.
   (b) Check the grid near the point where the negative volume is indicated. Make sure there are no grid lines that cross or are positioned incorrectly.
   (c) When running on a workstation, be sure to use the necessary precision for arrays (many turbulent grids are too fine to use single precision).
   (d) Be consistent with the precision between the grid and the code (i.e., the grid must be created in double precision if the CFL3D code is "made" that way).
2. Check the grid. It seems that 95% of the time, the problem is with the grid. Too much stretching too quickly, poor "quality", badly-skewed cells, etc. can all cause problems. Also be sure that the grid *and all its coarser levels* are of a reasonable size (when using multigrid, a coarser grid level should not be *too* coarse).
3. Be sure that the correct direction (either z or y) is "up" according to the type of grid being used. See "Grid File" on page 65.

4. If problems occur when starting a run for a very large grid, debugging on a coarser grid is recommended. Unfortunately, running on a coarser grid level using the **mseq** flag still requires a large percentage of the full amount of memory. Because this can be frustratingly slow for very big grids, the use of the following tools, found in the **`Tools`** subdirectory (see "The Code and Supplementary Files" on page 9), is recommended:

   `everyotherp3d.f` – reads a PLOT3D grid and writes out a coarser (every other grid point) grid

   `v5inpdoubhalf.f` – reads a CFL3D input file and creates a new input file applicable for either half or double the number of grid points

   After employing these tools to create a smaller grid and the corresponding input file, rerun `precfl3d` on the new input file and then recompile CFL3D and debug the problem on the coarser grid. The required memory will be substantially smaller than that necessary for the original grid.

5. Be sure to save a copy of the original code *before* making any modifications. If an error *is* made, the revision can be compared with the original version to aid with debugging. If additional subroutines are added, be sure they are added correctly to the makefile. If they are merely "tacked on" to the beginning of line 2 in the makefile, then they will not necessarily be recompiled correctly when `cfl1.h`, etc. are changed.

6. Peruse the output files early in the computation to make sure that they make sense. For example, if part of the grid is in motion, is it moving at the correct speed and in the expected direction? Do the results "look" as expected? Valuable CPU time can be saved by not running erroneous calculations for hundreds of iterations.

7. Check to make sure the boundary conditions are implemented correctly. This includes all 1-to-1 and patched interfaces. For 1-to-1 boundaries, look for "mismatch" in the output files for the "geometric mismatches" that may be a source of error. These numbers should all be close to zero. Small mismatches may be acceptable, but large mismatches (O(1)) most likely indicate that one of the segments on the block boundary has been specified incorrectly, or possibly backwards. For patched or overlapped interfaces, it is quite a bit more difficult to assure valid communication stencils. Check all outputs from ronnie for patched grids and from MaGGiE for overlapped grids.

8. Try lowering the CFL number significantly initially and allowing it to "creep" up as the solution progresses. Unless the grid is *really* bad, the CFL number should not have to be set below 0.1 (**dt** = -0.1). Sometimes it may be necessary to *remain* at low CFL numbers for particularly difficult problems. The optimum CFL number to run at is generally around 5 (**dt** = –5.0), for most problems on decent grids.

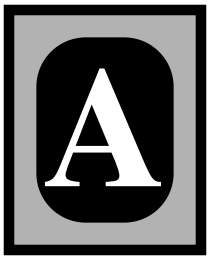
9. Make sure the executable is appropriate for the input file, particularly if any changes have been made to it *or* to the grid since the previous run. Run `precfl3d` to make sure the dimensioning is correct in the `cfl*.h` files. If `precfl3d` changes them, CFL3D must be recompiled.

10. Try employing mesh sequencing. Many people have found this to be beneficial for tough problems! For a description of mesh sequencing, see "Mesh Sequencing" on page 134.

11. Peruse all output files (there are a lot of them!) for any clues as to what might be wrong.
12. Try restarting from a less difficult, converged case on the same grid. For example, run and converge the configuration at **alpha** = 0 first; then restart at the higher, more difficult angle of attack (or Mach number or whatever).
13. When *really* desperate, try running with first order in space for awhile (**nitfo** = **ncyc**), then restart with third order from that. Or, try using second order (**rkap0** = -1) rather than third order (**rkap0** = +0.333333).
14. If multigrid is being used, try turning multigrid off for a short time. If multigrid is *not* being used, turn it *on*! In general, for steady-state problems, multigrid should definitely be used. See "Multigrid" on page 125 for instructions.
15. Be sure to use the latest version of the code (as of November 1996, Version 5.0).
16. There only seems to be sporadic success when varying **idiag**, **ifds**, or **iflim**. In general, if trouble arises when the default values are used for these variables, then trouble will occur even if they are altered. There are of course exceptions. For example, often hypersonic flow cases run more successfully with flux-vector splitting (**idiag** = 0). Also, sometimes, if a particular configuration is marginally stable (i.e., on the brink of going unsteady), then flux-vector splitting, which has more inherent dissipation than flux-difference splitting, may yield a steady solution while flux-difference splitting may go unsteady. What this means, however, is unclear, since it may be that the real physics of the flow *should* go unsteady. Presumably in such a case, if the grid is refined extensively, even flux-vector splitting should probably go unsteady.
17. There is little recent experience with this, but experimentation can be made with **ngam** (type of multigrid cycle), **mitL** (number of iterations on each multigrid level), and **mglevg** (the number of multigrid levels).
18. Another feature that is rarely used, but could be experimented with is residual correction and/or smoothing with multigrid. Set **issc** and/or **issr** to 1. (Zero is the default.) These parameters have been known, in isolated instances (particularly for hypersonic cases), to cause a "bombing" solution to work. In fact, some users turn residual correction and smoothing *on* as default. This is *not* recommended however because, for the majority of cases the code developers have seen, these parameters seem to hurt more than help. It probably depends a lot on the type of configuration or case being run.
19. Try running a different turbulence model that is more robust (Baldwin-Lomax and Spalart-Allmaras are probably the most robust of the models). Then restart the case from this converged solution using the desired model.
20. Monitor `cfl3d.turres`. Make sure there are either no or relatively few `nneg` values. Make sure the residual of the turbulent equations is not going *up* gradually over time rather than down. If there are problems, try setting `factor` lower in the appropriate turbulent model subroutine (such as subroutine `spalart` when using the Spalart-Allmaras turbulence model).
21. Sometimes the order of the indices in the grid can make a difference, since the code performs the approximate factorization (AF) in a particular order. Experience has shown that it is usually best to let the *primary* viscous direction (if there is one), such

as the direction normal to the wing surface, be the k direction. Eventually, at steady-state, the answer should be the same regardless of the index directions, but how *well* the code converges to the steady state can unfortunately sometimes be influenced by the choice.

22. When restarting and switching turbulence models, it is sometimes necessary, at least temporarily, to lower the CFL number or time step to get the new solution going. After a time, the CFL number or time step may then be "bumped up" to the desired level.
23. Hypersonic cases can often be difficult to start. Some suggestions are:
    (a) Start with a very low CFL number (on the order of 0.1) and run for a while, then later ramp it up.
    (b) Use mesh sequencing.
    (c) Use flux-vector splitting (**ifds** = 0), at least in the beginning.
    (d) Restart from a previous similar solution.
    (e) Experiment with varying **mitL** (number of iterations on each multigrid level).
24. If the residuals near patched boundaries are high (particularly when the grid sizes are very different near the patch), try replacing calls to `int2` with calls to `int3` instead (`int3` employs gradient limiting on the patch stencil).



APPENDIX A

Governing Equations

The computational algorithm employed in CFL3D for the three-dimensional Navier-Stokes code CFL3D is described in Thomas et al.[37] The governing equations, which are the thin-layer approximations to the three-dimensional time-dependent compressible Navier-Stokes equations, can be written in terms of generalized coordinates as

$$\frac{\partial \hat{\mathbf{Q}}}{\partial t} + \frac{\partial(\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)}{\partial \xi} + \frac{\partial(\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)}{\partial \eta} + \frac{\partial(\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)}{\partial \zeta} = 0 \tag{A-1}$$

A general, three-dimensional transformation between the Cartesian variables (x, y, z) and the generalized coordinated (ξ, η, ζ) is implied. (See Appendix F for details.) The variable J represents the Jacobian of the transformation:

$$J = \frac{\partial(\xi, \eta, \zeta, t)}{\partial(x, y, z, t)} \tag{A-2}$$

In Equation (A-1), $\mathbf{Q}$ is the vector of conserved variables, density, momentum, and total energy per unit volume, such that

$$\hat{\mathbf{Q}} = \frac{\mathbf{Q}}{J} = \frac{1}{J}\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix} \tag{A-3}$$

The inviscid flux terms are

$$\hat{\mathbf{F}} = \frac{\mathbf{F}}{J} = \frac{1}{J}\begin{bmatrix} \rho U \\ \rho U u + \xi_x p \\ \rho U v + \xi_y p \\ \rho U w + \xi_z p \\ (e + p)U - \xi_t p \end{bmatrix} \tag{A-4}$$

$$\hat{\mathbf{G}} = \frac{\mathbf{G}}{J} = \frac{1}{J}\begin{bmatrix} \rho V \\ \rho Vu + \eta_x p \\ \rho Vv + \eta_y p \\ \rho Vw + \eta_z p \\ (e+p)V - \eta_t p \end{bmatrix} \tag{A-5}$$

$$\hat{\mathbf{H}} = \frac{\mathbf{H}}{J} = \frac{1}{J}\begin{bmatrix} \rho W \\ \rho Wu + \zeta_x p \\ \rho Wv + \zeta_y p \\ \rho Ww + \zeta_z p \\ (e+p)W - \zeta_t p \end{bmatrix} \tag{A-6}$$

The contravariant velocities are given by

$$\begin{aligned} U &= \xi_x u + \xi_y v + \xi_z w + \xi_t \\ V &= \eta_x u + \eta_y v + \eta_z w + \eta_t \\ W &= \zeta_x u + \zeta_y v + \zeta_z w + \zeta_t \end{aligned} \tag{A-7}$$

The viscous flux terms are

$$\hat{\mathbf{F}}_v = \frac{\mathbf{F}_v}{J} = \frac{1}{J}\begin{bmatrix} 0 \\ \xi_x \tau_{xx} + \xi_y \tau_{xy} + \xi_z \tau_{xz} \\ \xi_x \tau_{xy} + \xi_y \tau_{yy} + \xi_z \tau_{yz} \\ \xi_x \tau_{xz} + \xi_y \tau_{yz} + \xi_z \tau_{zz} \\ \xi_x b_x + \xi_y b_y + \xi_z b_z \end{bmatrix} \tag{A-8}$$

$$\hat{\mathbf{G}}_v = \frac{\mathbf{G}_v}{J} = \frac{1}{J}\begin{bmatrix} 0 \\ \eta_x \tau_{xx} + \eta_y \tau_{xy} + \eta_z \tau_{xz} \\ \eta_x \tau_{xy} + \eta_y \tau_{yy} + \eta_z \tau_{yz} \\ \eta_x \tau_{xz} + \eta_y \tau_{yz} + \eta_z \tau_{zz} \\ \eta_x b_x + \eta_y b_y + \eta_z b_z \end{bmatrix} \tag{A-9}$$

$$\hat{\mathbf{H}}_v = \frac{\mathbf{H}_v}{J} = \frac{1}{J}\begin{bmatrix} 0 \\ \zeta_x\tau_{xx} + \zeta_y\tau_{xy} + \zeta_z\tau_{xz} \\ \zeta_x\tau_{xy} + \zeta_y\tau_{yy} + \zeta_z\tau_{yz} \\ \zeta_x\tau_{xz} + \zeta_y\tau_{yz} + \zeta_z\tau_{zz} \\ \zeta_x b_x + \zeta_y b_y + \zeta_z b_z \end{bmatrix} \tag{A-10}$$

The shear stress and hear flux terms are defined in tensor notations (summation convention implied) as

$$\tau_{x_i x_j} = \frac{M_\infty}{Re_{\tilde{L}_R}}\left[\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \lambda\frac{\partial u_k}{\partial x_k}\delta_{ij}\right] \tag{A-11}$$

$$b_{x_i} = u_j\tau_{x_i x_j} - \dot{q}_{x_i} \tag{A-12}$$

$$\dot{q}_{x_i} = -\left[\frac{M_\infty\mu}{Re_{\tilde{L}_R}Pr(\gamma-1)}\right]\frac{\partial a^2}{\partial x_i} \tag{A-13}$$

The pressure is obtained by the equation of state for a perfect gas

$$p = (\gamma-1)\left[e - \frac{\rho}{2}(u^2+v^2+w^2)\right] \tag{A-14}$$

The above equations have been nondimensionalized in terms of the free-stream[1] density, $\tilde{\rho}_\infty$, the free-stream speed of sound, $\tilde{a}_\infty$, and the free-stream molecular viscosity, $\tilde{\mu}_\infty$. (See Chapter 4.) The chain rule is used to evaluate derivatives with respect to (x, y, z) in terms of (ξ, η, ζ). Consistent with the thin-layer assumption, only those derivatives in the direction normal to the wall (ζ) are retained in the shear stress and heat flux terms. Equation (A-1) is closed by the Stokes hypothesis for bulk viscosity ($\lambda + 2\mu/3=0$) and Sutherland's law for molecular viscosity.[45]

The CFL3D code also has the capability to solve the Euler equations, which are obtained when the $\hat{\mathbf{F}}_v$, $\hat{\mathbf{G}}_v$, and $\hat{\mathbf{H}}_v$ terms are omitted from Equation (A-1).

The numerical algorithm uses a semi-discrete finite-volume formulation, resulting in a consistent approximation to the conservation laws in integral form

$$\frac{\partial}{\partial t}\iiint_V \mathbf{Q}\,dV + \iint_S \vec{\mathrm{f}}\cdot\vec{\mathrm{n}}\,dS = 0 \tag{A-15}$$

[1] See the note on page 3 about the usage of the phrase *free stream*.

where $\vec{f}$ denotes the net flux through a surface S with unit normal $\vec{n}$ containing the (time-invariant) volume V. Integration of Equation (A-15) over a control volume bounded by lines of constant ξ, η, and ζ gives the semi-discrete form

$$\left(\frac{\partial \hat{\mathbf{Q}}}{\partial t}\right)_{i,j,k} + (\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)_{i+1/2,j,k} - (\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)_{i-1/2,j,k} + (\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)_{i,j+1/2,k} - (\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)_{i,j-1/2,k} + (\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)_{i,j,k+1/2} - (\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)_{i,j,k-1/2} = 0 \tag{A-16}$$

where, for convenience,

$$\begin{aligned} \Delta\xi &= \xi_{i+1/2,j,k} - \xi_{i-1/2,j,k} = 1 \\ \Delta\eta &= \eta_{i,j+1/2,k} - \eta_{i,j-1/2,k} = 1 \\ \Delta\zeta &= \zeta_{i,j,k+1/2} - \zeta_{i,j,k-1/2} = 1 \end{aligned} \tag{A-17}$$

The discrete values $\hat{\mathbf{Q}}_{i,j,k}$ are regarded as average values taken over a unit computational cell; similarly, discrete values of $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$, and $\hat{\mathbf{H}}$ are regarded as face-average values. The convective and pressure terms are differenced using either the upwind flux-difference-splitting technique of Roe[31] or the flux-vector-splitting technique of van Leer.[39] The MUSCL (Monotone Upstream-centered Scheme for Conservation Laws) approach of van Leer[40] is used to determine state-variable interpolations at the cell interfaces. The shear stress and heat transfer terms are centrally differenced.

APPENDIX B

Time Advancement

For a nondeforming mesh, Equation (A-1) can be written as

$$\frac{1}{J}\frac{\partial \mathbf{Q}}{\partial t} = R(\mathbf{Q}) \tag{B-1}$$

where

$$R = -\left[\frac{\partial(\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)}{\partial \xi} + \frac{\partial(\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)}{\partial \eta} + \frac{\partial(\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)}{\partial \zeta}\right] \tag{B-2}$$

The time term can be discretized with backward differencing:

$$\frac{(1+\phi)(\mathbf{Q}^{n+1} - \mathbf{Q}^n) - \phi(\mathbf{Q}^n - \mathbf{Q}^{n-1})}{J\Delta t} = R(\mathbf{Q}^{n+1}) \tag{B-3}$$

where the superscripts indicate time level. When $\phi = 0$ the method is first-order temporally accurate; when $\phi = 1/2$ the method is second-order accurate. This equation is implicit because the right-hand side is a function of the unknown flow variables at time level $n+1$.

The CFL3D code is advanced in time with an implicit approximate-factorization method. The implicit derivatives are written as spatially first-order accurate, which results in block-tridiagonal inversions for each sweep. However, for solutions that utilize FDS the block-tridiagonal inversions are usually further simplified with a diagonal algorithm (with a spectral radius scaling of the viscous terms).

Because of the method which the left-hand side is treated for computational efficiency in steady-state simulations (approximate factorization, first-order accuracy), second-order temporal accuracy is forfeited for unsteady computations. One method for recovering the desired accuracy is through the use of sub-iterations. Two different sub-iteration strategies have been implemented in CFL3D. The first method is termed "pseudo time sub-iteration (τ-TS)". The method is also often referred to as the "dual time stepping" method. The other method, termed "physical time sub-iteration (t-TS)," follows Pulliam.[28]

For the τ-TS method, a pseudo time term is added to the time-accurate Navier-Stokes equations.

$$\frac{1}{J}\frac{\partial \mathbf{Q}}{\partial \tau} + \frac{(1+\phi)(\mathbf{Q}^{n+1} - \mathbf{Q}^{n}) - \phi(\mathbf{Q}^{n} - \mathbf{Q}^{n-1})}{J\Delta t} = R(\mathbf{Q}^{n+1}) \tag{B-4}$$

This equation is then discretized and iterated in m, where m is the sub-iteration counter.

$$\frac{(1+\phi')(\mathbf{Q}^{m+1} - \mathbf{Q}^{m}) - \phi'(\mathbf{Q}^{m} - \mathbf{Q}^{m-1})}{J\Delta\tau} + \frac{(1+\phi)(\mathbf{Q}^{m+1} - \mathbf{Q}^{n}) - \phi(\mathbf{Q}^{n} - \mathbf{Q}^{n-1})}{J\Delta t} = R(\mathbf{Q}^{m+1}) \tag{B-5}$$

In Equation (B-4), ϕ and ϕ' govern the order of accuracy of the physical and pseudo time terms, respectively. In practice, the pseudo time term is treated as first order (i.e., $\phi' = 0$), but the general form is shown here for completeness. As $m \to \infty$, the pseudo time term vanishes if the sub-iterations converge and $\mathbf{Q}^{m+1} \to \mathbf{Q}^{n+1}$. If R is linearized with

$$R(\mathbf{Q}^{m+1}) \cong R(\mathbf{Q}^{m}) + \frac{\partial R}{\partial \mathbf{Q}}\Delta\mathbf{Q}^{m} \tag{B-6}$$

and the quantity $-(1+\phi)\mathbf{Q}^{m}/(J\Delta t)$ is added to both sides of Equation (B-4)), then Equation (B-4) becomes

$$\left[\left(\frac{1+\phi'}{J\Delta\tau} + \frac{1+\phi}{J\Delta t}\right)I + \delta_\xi \mathbf{A} + \delta_\eta \mathbf{B} + \delta_\zeta \mathbf{C}\right]\Delta\mathbf{Q}^{m} = \frac{\phi'\Delta\mathbf{Q}^{m-1}}{J\Delta\tau} + \frac{\phi\Delta\mathbf{Q}^{n-1}}{J\Delta t} - \frac{(1+\phi)(\mathbf{Q}^{m} - \mathbf{Q}^{n})}{J\Delta t} + R(\mathbf{Q}^{m}) \tag{B-7}$$

where

$$\Delta\mathbf{Q}^{m} = \mathbf{Q}^{m+1} - \mathbf{Q}^{m} \tag{B-8}$$

$$\mathbf{A} = \frac{\partial(\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)}{\partial \mathbf{Q}} \tag{B-9}$$

$$\mathbf{B} = \frac{\partial(\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)}{\partial \mathbf{Q}} \tag{B-10}$$

$$\mathbf{C} = \frac{\partial(\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)}{\partial \mathbf{Q}} \tag{B-11}$$

Equation (B-7) is approximately factored and written in primitive variable form; it is solved as a series of sweeps in each coordinate direction as

$$\left[\left(\frac{(1+\phi')\mathbf{M}}{J\Delta\tau}+\frac{(1+\phi)\mathbf{M}}{J\Delta t}\right)+\delta_\xi\mathbf{A}^*\right]\Delta\mathbf{q}' = \frac{\phi'\mathbf{M}\Delta\mathbf{q}^{m-1}}{J\Delta\tau}+\frac{\phi\mathbf{M}\Delta\mathbf{q}^{n-1}}{J\Delta t}- \frac{(1+\phi)\mathbf{M}(\mathbf{q}^m-\mathbf{q}^n)}{J\Delta t}+R(\mathbf{q}^m) \tag{B-12}$$

$$\left[\left(\frac{(1+\phi')\mathbf{M}}{J\Delta\tau}+\frac{(1+\phi)\mathbf{M}}{J\Delta t}\right)+\delta_\eta\mathbf{B}^*\right]\Delta\mathbf{q}'' = \left(\frac{(1+\phi')\mathbf{M}}{J\Delta\tau}+\frac{(1+\phi)\mathbf{M}}{J\Delta t}\right)\Delta\mathbf{q}' \tag{B-13}$$

$$\left[\left(\frac{(1+\phi')\mathbf{M}}{J\Delta\tau}+\frac{(1+\phi)\mathbf{M}}{J\Delta t}\right)+\delta_\zeta\mathbf{C}^*\right]\Delta\mathbf{q}^m = \left(\frac{(1+\phi')\mathbf{M}}{J\Delta\tau}+\frac{(1+\phi)\mathbf{M}}{J\Delta t}\right)\Delta\mathbf{q}'' \tag{B-14}$$

$$\mathbf{q}^{m+1} = \mathbf{q}^m + \Delta\mathbf{q}^m \tag{B-15}$$

where the primitive variables are

$$\mathbf{q} = \begin{bmatrix}\rho\\ u\\ v\\ w\\ p\end{bmatrix} \tag{B-16}$$

$$\mathbf{M} = \frac{\partial\mathbf{Q}}{\partial\mathbf{q}} \tag{B-17}$$

$$\mathbf{A}^* = \frac{\partial(\hat{\mathbf{F}}-\hat{\mathbf{F}}_v)}{\partial\mathbf{q}} \tag{B-18}$$

$$\mathbf{B}^* = \frac{\partial(\hat{\mathbf{G}}-\hat{\mathbf{G}}_v)}{\partial\mathbf{q}} \tag{B-19}$$

$$\mathbf{C}^* = \frac{\partial(\hat{\mathbf{H}}-\hat{\mathbf{H}}_v)}{\partial\mathbf{q}} \tag{B-20}$$

The quantity $\Delta\tau$ is based on a constant CFL number set by the input parameter **cfl_tau** (See “LT5 - Time Step Parameters” on page 21). Multigrid is used to drive $\Delta\mathbf{q}^m$ to zero in a reasonable number of sub-iterations.

In the t-TS method, Equation (B-3) is merely iterated in m, where m is the sub-iteration counter:

$$\frac{(1+\phi)(\mathbf{Q}^{m+1}-\mathbf{Q}^{n})-\phi(\mathbf{Q}^{n}-\mathbf{Q}^{n-1})}{J\Delta t} = R(\mathbf{Q}^{m+1}) \tag{B-21}$$

The quantity $-(1+\phi)\mathbf{Q}^{m}/(J\Delta t)$ is added to both sides, the residual is linearized, and the equation is approximately factored and written in primitive variable form as

$$\left[\frac{(1+\phi)\mathbf{M}}{J\Delta t}+\delta_{\xi}\mathbf{A}^{*}\right]\Delta\mathbf{q}' = \frac{\phi\mathbf{M}\Delta\mathbf{q}^{n-1}}{J\Delta t}-\frac{(1+\phi)\mathbf{M}(\mathbf{q}^{m}-\mathbf{q}^{n})}{J\Delta t}+R(\mathbf{q}^{m}) \tag{B-22}$$

$$\left[\frac{(1+\phi)\mathbf{M}}{J\Delta t}+\delta_{\eta}\mathbf{B}^{*}\right]\Delta\mathbf{q}'' = \frac{(1+\phi)\mathbf{M}}{J\Delta t}\Delta\mathbf{q}' \tag{B-23}$$

$$\left[\frac{(1+\phi)\mathbf{M}}{J\Delta t}+\delta_{\zeta}\mathbf{C}^{*}\right]\Delta\mathbf{q}^{m} = \frac{(1+\phi)\mathbf{M}}{J\Delta t}\Delta\mathbf{q}'' \tag{B-24}$$

$$\mathbf{q}^{m+1} = \mathbf{q}^{m}+\Delta\mathbf{q}^{m} \tag{B-25}$$

As $m\to\infty$, $\mathbf{q}^{m+1}\to\mathbf{q}^{n+1}$. When only one series of sweeps is performed, $\mathbf{q}^{m} = \mathbf{q}^{n}$ and the standard time-accurate CFL3D scheme is recovered (i.e., no sub-iterations). Unlike the τ-TS method, this sub-iteration procedure (Equation (B-22) through Equation (B-25)) utilizes only one time step: the physical time step Δt (= constant).

Prior to the execution of Equation (B-25) in the code, the corrections are constrained in order to maintain the positivity of the thermodynamic variables ρ and p. For example, the update to pressure is taken as

$$p^{n+1} = p^{n}+\Delta p\left[1+\phi_{c}\left(\alpha_{c}+\left|\frac{\Delta p}{p^{n}}\right|\right)\right]^{-1} \tag{B-26}$$

whenever $\Delta p/p^{n}\le\alpha_{c}$. Currently, $\alpha_{c} = -0.2$ and $\phi_{c} = 2.0$.

In the limit of $\Delta p/p^{n}\to-\infty$, $p^{n+1}\to p^{n}/2$. This modification improves the robustness of the method by allowing it to proceed through local transients encountered during the convergence process which would otherwise terminate the calculation.

When running steady-state computations (**dt** < 0), the time step advanced locally in each cell is related to the input CFL number by

$$\Delta t = \frac{\mathrm{CFL}}{|\nabla\xi|t_{1}+|\nabla\eta|t_{2}+|\nabla\zeta|t_{3}} \tag{B-27}$$

where

$$
\begin{aligned}
t_1 &= |\overline{U}| + a + 2|\nabla\xi|(\mu + \mu_T)\max\left(\frac{4}{3}, \frac{\gamma}{Pr}\right)\frac{M}{Re_{\tilde{L}_R}}\frac{1}{\rho} \\
t_2 &= |\overline{V}| + a + 2|\nabla\eta|(\mu + \mu_T)\max\left(\frac{4}{3}, \frac{\gamma}{Pr}\right)\frac{M}{Re_{\tilde{L}_R}}\frac{1}{\rho} \\
t_3 &= |\overline{W}| + a + 2|\nabla\zeta|(\mu + \mu_T)\max\left(\frac{4}{3}, \frac{\gamma}{Pr}\right)\frac{M}{Re_{\tilde{L}_R}}\frac{1}{\rho}
\end{aligned}
\qquad \textbf{(B-28)}
$$

where $\overline{U} = U/|\nabla\xi|$, $\overline{V} = V/|\nabla\eta|$, $\overline{W} = W/|\nabla\zeta|$ and U, V, and W are defined in Equation (A-7) in Appendix A. The viscous scaling terms (the last term in each equation of Equation (B-28)) are only used when the solution includes viscous terms. They arise from a spectral radius scaling (see Coakley[14]).

APPENDIX C

Spatial Discretization

C.1 Inviscid Fluxes

The spatial derivatives of the convective and pressure terms are written conservatively as a flux balance across a cell as, for example,

$$(\delta_\xi \hat{\mathbf{F}})_i = \hat{\mathbf{F}}_{i+\frac{1}{2}} - \hat{\mathbf{F}}_{i-\frac{1}{2}} \tag{C-1}$$

where the i index denotes a cell-center location and $i \pm 1/2$ corresponds to a cell-interface location. The interface flux is determined from a state-variable interpolation and a locally one-dimensional flux model. For flux-vector splitting (FVS), Equation (C-1) is split into forward and backward-moving pieces as:

$$(\delta_\xi \hat{\mathbf{F}})_i = (\delta_\xi^- \hat{\mathbf{F}}^+ + \delta_\xi^+ \hat{\mathbf{F}}^-)_i = [\hat{\mathbf{F}}^+(\mathbf{q}_L) + \hat{\mathbf{F}}^-(\mathbf{q}_R)]_{i+\frac{1}{2}} - [\hat{\mathbf{F}}^+(\mathbf{q}_L) + \hat{\mathbf{F}}^-(\mathbf{q}_R)]_{i-\frac{1}{2}} \tag{C-2}$$

For flux-difference splitting (FDS), the interface flux is written as an exact solution to an approximate Riemann problem as:

$$\begin{aligned}(\delta_\xi \hat{\mathbf{F}})_i = {} & \frac{1}{2}[\hat{\mathbf{F}}(\mathbf{q}_L) + \hat{\mathbf{F}}(\mathbf{q}_R) - |\tilde{\mathbf{A}}_{\text{inv}}|(\mathbf{q}_R - \mathbf{q}_L)]_{i+\frac{1}{2}} \\ & - \frac{1}{2}[\hat{\mathbf{F}}(\mathbf{q}_L) + \hat{\mathbf{F}}(\mathbf{q}_R) - |\tilde{\mathbf{A}}_{\text{inv}}|(\mathbf{q}_R - \mathbf{q}_L)]_{i-\frac{1}{2}}\end{aligned} \tag{C-3}$$

Further details on the FVS and FDS methods are given below. In both cases, interpolated values $\mathbf{q}_L$ and $\mathbf{q}_R$ at each interface are required. The state variable interpolations determine the resulting accuracy of the scheme. They are constructed from interpolation of the primitive variables. For first-order fully-upwind differencing:

$$\begin{aligned}(\mathbf{q}_L)_{i+\frac{1}{2}} &= \mathbf{q}_i \\ (\mathbf{q}_R)_{i+\frac{1}{2}} &= \mathbf{q}_{i+1}\end{aligned} \tag{C-4}$$

Higher order accuracy is given by the family of interpolations:

$$
\begin{aligned}
(\mathbf{q}_L)_{i+\frac{1}{2}} &= \mathbf{q}_i + \frac{1}{4}[(1-\kappa)\Delta_- + (1+\kappa)\Delta_+]_i \\
(\mathbf{q}_R)_{i+\frac{1}{2}} &= \mathbf{q}_{i+1} - \frac{1}{4}[(1-\kappa)\Delta_+ + (1+\kappa)\Delta_-]_{i+1}
\end{aligned}
\tag{C-5}
$$

where

$$
\begin{aligned}
\Delta_+ &\equiv \mathbf{q}_{i+1} - \mathbf{q}_i \\
\Delta_- &\equiv \mathbf{q}_i - \mathbf{q}_{i-1}
\end{aligned}
\tag{C-6}
$$

The parameter $\kappa \in [-1,1]$ forms a family of difference schemes. $\kappa = -1$ corresponds to second-order fully-upwind differencing, $\kappa = 1/3$ to third-order upwind-biased differencing, and $\kappa = 1$ to central differencing. Note, however, that setting $\kappa = 1$ in CFL3D will result in odd-even point decoupling, since there is no artificial dissipation in the code. Also note that "third order" for $\kappa = 1/3$ is only obtained for one-dimensional flows. For two and three dimensions, the formal order of accuracy is second order.

In practice, the gradients of density and pressure are biased by an average value in order to improve the robustness of the calculation at higher Mach numbers and in the early transient stages of a solution.

$$
\begin{aligned}
\Delta_+^{\mathrm{b}} &= \frac{\Delta_+}{\mathbf{q}_i + \frac{1}{2}\Delta_+} \\
\Delta_-^{\mathrm{b}} &= \frac{\Delta_-}{\mathbf{q}_i - \frac{1}{2}\Delta_-}
\end{aligned}
\tag{C-7}
$$

And,

$$
\begin{aligned}
(\mathbf{q}_L)_{i+\frac{1}{2}} &= \mathbf{q}_i + \frac{\mathbf{q}_i}{4}[(1-\kappa)\Delta_-^{\mathrm{b}} + (1+\kappa)\Delta_+^{\mathrm{b}}]_i \\
(\mathbf{q}_R)_{i+\frac{1}{2}} &= \mathbf{q}_{i+1} - \frac{\mathbf{q}_{i+1}}{4}[(1-\kappa)\Delta_+^{\mathrm{b}} + (1+\kappa)\Delta_-^{\mathrm{b}}]_{i+1}
\end{aligned}
\tag{C-8}
$$

Note that Equation (C-8) is not performed for the gradients of velocity; this is essentially an extrapolation of the log of the density and pressure (first-order expansion).

C.1.1 Flux Limiting

For solutions with discontinuities (such as shock waves), schemes of order higher than one generally require a flux limiter to avoid numerical oscillations in the solution. CFL3D has several limiter options.

The smooth limiter (**iflim** = 1) is implemented via

$$\begin{aligned}(\mathbf{q}_L)_{i+\frac{1}{2}} &= \mathbf{q}_i + \left\{\frac{s}{4}[(1-\kappa s)\Delta_- + (1+\kappa s)\Delta_+]\right\}_i \\ (\mathbf{q}_R)_{i+\frac{1}{2}} &= \mathbf{q}_{i+1} - \left\{\frac{s}{4}[(1-\kappa s)\Delta_+ + (1+\kappa s)\Delta_-]\right\}_{i+1}\end{aligned} \tag{C-9}$$

where

$$s = \frac{2\Delta_+\Delta_- + \varepsilon}{(\Delta_+)^2 + (\Delta_-)^2 + \varepsilon} \tag{C-10}$$

and ε is a small number ($\varepsilon = 1 \times 10^{-6}$) preventing division by zero in regions of null gradient.

The min-mod limiter (**iflim** = 2) is implemented via

$$\begin{aligned}(\mathbf{q}_L)_{i+\frac{1}{2}} &= \mathbf{q}_i + \frac{1}{4}[(1-\kappa)\bar{\Delta}_- + (1+\kappa)\bar{\Delta}_+]_i \\ (\mathbf{q}_R)_{i+\frac{1}{2}} &= \mathbf{q}_{i+1} - \frac{1}{4}[(1-\kappa)\bar{\Delta}_+ + (1+\kappa)\bar{\Delta}_-]_{i+1}\end{aligned} \tag{C-11}$$

where

$$\begin{aligned}\bar{\Delta}_- &= \text{minmod}(\Delta_-, b\Delta_+) \\ \bar{\Delta}_+ &= \text{minmod}(\Delta_+, b\Delta_-)\end{aligned} \tag{C-12}$$

$$\text{minmod}(x, y) = \max\{0, \min[x\ \text{sign}(y), by\text{sign}(x)]\}\text{sign}(x) \tag{C-13}$$

The parameter b is a compression parameter, $b = (3-\kappa)/(1-\kappa)$.

The smooth limiter tuned to $\kappa = 1/3$ (**iflim** = 3) is implemented as follows.

$$(\mathbf{q}_L)_{i+\frac{1}{2}} = \mathbf{q}_i + \frac{1}{2}(\delta_L \mathbf{q})_i$$
$$(\mathbf{q}_R)_{i+\frac{1}{2}} = \mathbf{q}_{i+1} - \frac{1}{2}(\delta_R \mathbf{q})_{i+1} \tag{C-14}$$

where

$$(\delta_L \mathbf{q})_i = \mathrm{I}(\mathbf{q}_{i+1} - \mathbf{q}_i, \mathbf{q}_i - \mathbf{q}_{i-1})$$
$$(\delta_R \mathbf{q})_i = \mathrm{I}(\mathbf{q}_i - \mathbf{q}_{i-1}, \mathbf{q}_{i+1} - \mathbf{q}_i) \tag{C-15}$$

$$\mathrm{I}(x, y) = \frac{x(y^2 + 2\varepsilon^2) + y(2x^2 + \varepsilon^2)}{2x^2 - xy + 2y^2 + 3\varepsilon^2} \tag{C-16}$$

and I is designed to recover the state variable to third-order accuracy in the one-dimensional case in smooth regions of the flow and interpolate without oscillations near discontinuities. The parameter ε^2 is a small constant of order Δx^3 which is used to improve the accuracy near smooth extremum and reduce the nonlinearity of the interpolation in regions of small gradient.

C.1.2 Flux-Vector Splitting

The flux-vector splitting (FVS) method of van Leer[39] is implemented as follows. The generalized inviscid fluxes $\hat{\mathbf{F}}$, $\hat{\mathbf{G}}$, and $\hat{\mathbf{H}}$, representing the pressure and convection terms, are upwind differenced by splitting into forward and backward contributions and differencing accordingly. For example, the flux difference in the ξ direction is

$$\delta_\xi \hat{\mathbf{F}} = \delta_\xi^- \hat{\mathbf{F}}^+ + \delta_\xi^+ \hat{\mathbf{F}}^- \tag{C-17}$$

where δ_ξ^- and δ_ξ^+ denote general backward and forward divided difference operators, respectively. For flux-vector splitting, $\hat{\mathbf{F}}$ is split according to the contravariant Mach number in the ξ direction, defined as

$$M_\xi = \frac{\overline{U}}{a} \tag{C-18}$$

where

$$\overline{U} = \frac{U}{|\nabla \xi|} \tag{C-19}$$

and

$$U = \xi_x u + \xi_y v + \xi_z w + \xi_t \tag{C-20}$$

For locally supersonic flow, where $|M_\xi| \geq 1$,

$$\begin{aligned} \hat{\mathbf{F}}^+ &= \hat{\mathbf{F}} \qquad \hat{\mathbf{F}}^- = 0 \qquad M_\xi \geq +1 \\ \hat{\mathbf{F}}^- &= \hat{\mathbf{F}} \qquad \hat{\mathbf{F}}^+ = 0 \qquad M_\xi \leq -1 \end{aligned} \tag{C-21}$$

and for locally subsonic flow, where $|M_\xi| < 1$,

$$\hat{\mathbf{F}}^\pm = \frac{|\nabla\xi|}{J}\begin{bmatrix} \mathrm{F}_1^\pm \\ \mathrm{F}_2^\pm \\ \mathrm{F}_3^\pm \\ \mathrm{F}_4^\pm \\ \mathrm{F}_5^\pm \end{bmatrix} = \frac{|\nabla\xi|}{J}\begin{bmatrix} f_{\text{mass}}^\pm \\ f_{\text{mass}}^\pm[\hat{\xi}_x(-\bar{U} \pm 2a)/\gamma + u] \\ f_{\text{mass}}^\pm[\hat{\xi}_y(-\bar{U} \pm 2a)/\gamma + v] \\ f_{\text{mass}}^\pm[\hat{\xi}_z(-\bar{U} \pm 2a)/\gamma + w] \\ f_{\text{energy}}^\pm \end{bmatrix} \tag{C-22}$$

where the direction cosines of the cell interfaces are

$$\hat{\xi}_x = \frac{\xi_x}{|\nabla\xi|} \qquad \hat{\xi}_y = \frac{\xi_y}{|\nabla\xi|} \qquad \hat{\xi}_z = \frac{\xi_z}{|\nabla\xi|} \tag{C-23}$$

and

$$f_{\text{mass}}^\pm = \pm\frac{\rho a}{4}(M_\xi \pm 1)^2 \tag{C-24}$$

$$f_{\text{energy}}^\pm = f_{\text{mass}}^\pm\left[\frac{(1-\gamma)\bar{U}^2 \pm 2(\gamma-1)\bar{U}a + 2a^2}{(\gamma^2-1)} + \frac{(u^2+v^2+w^2)}{2} - \frac{\hat{\xi}_t}{\gamma}(-\bar{U} \pm 2a)\right] \tag{C-25}$$

and

$$\hat{\xi}_t = \frac{\xi_t}{|\nabla\xi|} \tag{C-26}$$

For an implicit scheme, the Jacobians of the right-hand-side flux terms must be determined (see Appendix B). They appear as terms in the left-hand-side matrix multiplying $\Delta\mathbf{q}$, and the resulting system of equations is solved with a full 5 by 5 block-tridiagonal

inversion procedure. For flux-vector splitting, the split flux Jacobians (with respect to the primitive variables) for the ξ direction, i.e. $\frac{\partial \hat{\mathbf{F}}^{\pm}}{\partial \mathbf{q}}$, are

$$\begin{aligned}
\frac{\partial F_1^{\pm}}{\partial \rho} &= \frac{1}{4}(\overline{U} \pm a)\left[-1 \pm \frac{3}{2a}(\overline{U} \pm a)\right] \\
\frac{\partial F_1^{\pm}}{\partial u} &= \pm\frac{\rho}{2a}(\overline{U} \pm a)\hat{\xi}_x \\
\frac{\partial F_1^{\pm}}{\partial v} &= \pm\frac{\rho}{2a}(\overline{U} \pm a)\hat{\xi}_y \\
\frac{\partial F_1^{\pm}}{\partial w} &= \pm\frac{\rho}{2a}(\overline{U} \pm a)\hat{\xi}_z \\
\frac{\partial F_1^{\pm}}{\partial p} &= \frac{-\gamma}{4a^2}(\overline{U} \pm a)\left[-1 \pm \frac{1}{2a}(\overline{U} \pm a)\right]
\end{aligned} \qquad \text{(C-27)}$$

$$\begin{aligned}
\frac{\partial F_2^{\pm}}{\partial \rho} &= \pm\frac{1}{4}(\overline{U} \pm a)\left[\frac{3\overline{U}}{2a\gamma}(-\overline{U} \pm a)\hat{\xi}_x + u\left(\frac{3}{2a}(\overline{U} \pm a) \mp 1\right)\right] \\
\frac{\partial F_2^{\pm}}{\partial u} &= \pm\frac{\rho}{4a}(\overline{U} \pm a)\left[\hat{\xi}_x\left(\frac{3\hat{\xi}_x}{\gamma}(-\overline{U} \pm a) + 2u\right) + (\overline{U} \pm a)\right] \\
\frac{\partial F_2^{\pm}}{\partial v} &= \pm\frac{\rho}{4a}(\overline{U} \pm a)\hat{\xi}_y\left[\frac{3\hat{\xi}_x}{\gamma}(-\overline{U} \pm a) + 2u\right] \\
\frac{\partial F_2^{\pm}}{\partial w} &= \pm\frac{\rho}{4a}(\overline{U} \pm a)\hat{\xi}_z\left[\frac{3\hat{\xi}_x}{\gamma}(-\overline{U} \pm a) + 2u\right] \\
\frac{\partial F_2^{\pm}}{\partial p} &= \pm\frac{1}{4a^2}(\overline{U} \pm a)\left\{\hat{\xi}_x\left[2a - \frac{\overline{U}}{2a}(-\overline{U} \pm a) \pm \gamma u\left(1 \mp \frac{1}{2a}(\overline{U} \pm a)\right)\right]\right\}
\end{aligned} \qquad \text{(C-28)}$$

$$
\begin{aligned}
\frac{\partial F_3^{\pm}}{\partial \rho} &= \pm\frac{1}{4}(\overline{U}\pm a)\left[\frac{3\overline{U}}{2a\gamma}(-\overline{U}\pm a)\hat{\xi}_y + v\left(\frac{3}{2a}(\overline{U}\pm a)\mp 1\right)\right] \\
\frac{\partial F_3^{\pm}}{\partial u} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\hat{\xi}_x\left[\frac{3\hat{\xi}_y}{\gamma}(-\overline{U}\pm a)+2v\right] \\
\frac{\partial F_3^{\pm}}{\partial v} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\left[\hat{\xi}_y\left(\frac{3\hat{\xi}_y}{\gamma}(-\overline{U}\pm a)+2v\right)+(\overline{U}\pm a)\right] \\
\frac{\partial F_3^{\pm}}{\partial w} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\hat{\xi}_z\left[\frac{3\hat{\xi}_y}{\gamma}(-\overline{U}\pm a)+2v\right] \\
\frac{\partial F_3^{\pm}}{\partial p} &= \pm\frac{1}{4a^2}(\overline{U}\pm a)\left\{\hat{\xi}_y\left[2a-\frac{\overline{U}}{2a}(-\overline{U}\pm a)\pm\gamma v\left(1\mp\frac{1}{2a}(\overline{U}\pm a)\right)\right]\right\}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial F_4^{\pm}}{\partial \rho} &= \pm\frac{1}{4}(\overline{U}\pm a)\left[\frac{3\overline{U}}{2a\gamma}(-\overline{U}\pm a)\hat{\xi}_z + w\left(\frac{3}{2a}(\overline{U}\pm a)\mp 1\right)\right] \\
\frac{\partial F_4^{\pm}}{\partial u} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\hat{\xi}_x\left[\frac{3\hat{\xi}_z}{\gamma}(-\overline{U}\pm a)+2w\right] \\
\frac{\partial F_4^{\pm}}{\partial v} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\hat{\xi}_y\left[\frac{3\hat{\xi}_z}{\gamma}(-\overline{U}\pm a)+2w\right] \\
\frac{\partial F_4^{\pm}}{\partial w} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\left[\hat{\xi}_z\left(\frac{3\hat{\xi}_z}{\gamma}(-\overline{U}\pm a)+2w\right)+(\overline{U}\pm a)\right] \\
\frac{\partial F_4^{\pm}}{\partial p} &= \pm\frac{1}{4a^2}(\overline{U}\pm a)\left\{\hat{\xi}_z\left[2a-\frac{\overline{U}}{2a}(-\overline{U}\pm a)\pm\gamma w\left(1\mp\frac{1}{2a}(\overline{U}\pm a)\right)\right]\right\}
\end{aligned}
$$

(C-29)

(C-30)

$$
\begin{aligned}
\frac{\partial F_5^{\pm}}{\partial \rho} &= \pm\frac{1}{4}(\overline{U}\pm a)\left\{(-\overline{U}\pm a)\left[\frac{3\overline{U}}{2a}\left(\frac{\overline{U}}{\gamma+1}-\frac{\hat{\xi}_t}{\gamma}\right)-\frac{a}{\gamma^2-1}\right]+\frac{u^2+v^2+w^2}{2}\left[\frac{3}{2a}(\overline{U}\pm a)\mp 1\right]\right\} \\
\frac{\partial F_5^{\pm}}{\partial u} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\left\{\hat{\xi}_x\left[\frac{2a^2}{\gamma-1}+\left(\frac{4\overline{U}}{\gamma+1}-\frac{3\hat{\xi}_t}{\gamma}\right)(-\overline{U}\pm a)+u^2+v^2+w^2\right]+u(\overline{U}\pm a)\right\} \\
\frac{\partial F_5^{\pm}}{\partial v} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\left\{\hat{\xi}_y\left[\frac{2a^2}{\gamma-1}+\left(\frac{4\overline{U}}{\gamma+1}-\frac{3\hat{\xi}_t}{\gamma}\right)(-\overline{U}\pm a)+u^2+v^2+w^2\right]+v(\overline{U}\pm a)\right\} \\
\frac{\partial F_5^{\pm}}{\partial w} &= \pm\frac{\rho}{4a}(\overline{U}\pm a)\left\{\hat{\xi}_z\left[\frac{2a^2}{\gamma-1}+\left(\frac{4\overline{U}}{\gamma+1}-\frac{3\hat{\xi}_t}{\gamma}\right)(-\overline{U}\pm a)+u^2+v^2+w^2\right]+w(\overline{U}\pm a)\right\}
\end{aligned}
$$

(C-31)

$$\frac{\partial F_5^{\pm}}{\partial p} = \pm\frac{\gamma}{4a}(\bar{U}\pm a)\left\{\left[\left(\frac{\bar{U}}{\gamma+1}-\frac{\hat{\xi}_t}{\gamma}\right)\left(2a-\frac{\bar{U}}{2a}(-\bar{U}\pm a)\right)+\frac{a}{\gamma^2-1}(\bar{U}\pm 3a)\right.\right.$$
$$\left.\left.+\frac{u^2+v^2+w^2}{2}\left(-\frac{1}{2a}(\bar{U}\pm a)\pm 1\right)\right]\right\}$$

C.1.3 Flux-Difference Splitting

In the flux-difference splitting (FDS) method of Roe[31], the interface flux in the ξ direction is written

$$\hat{\mathbf{F}}_{i+\frac{1}{2}} = \frac{1}{2}[\hat{\mathbf{F}}(\mathbf{q}_L)+\hat{\mathbf{F}}(\mathbf{q}_R)-|\tilde{\mathbf{A}}_{\text{inv}}|(\mathbf{q}_R-\mathbf{q}_L)]_{i+\frac{1}{2}} \tag{C-32}$$

where $\tilde{\mathbf{A}}_{\text{inv}}$ is $\mathbf{A}_{\text{inv}}$ evaluated with Roe-averaged variables defined below. ($\mathbf{A}_{\text{inv}}$ is the inviscid part of the $\mathbf{A}$ matrix defined in Appendix B.) Hence,

$$|\tilde{\mathbf{A}}_{\text{inv}}| = |\mathbf{A}_{\text{inv}}(\tilde{\mathbf{q}})| \tag{C-33}$$

and

$$\begin{aligned}\tilde{\mathbf{q}} &= \tilde{\mathbf{q}}(\mathbf{q}_L,\mathbf{q}_R)\\ \mathbf{A}_{\text{inv}} &= \frac{\partial\hat{\mathbf{F}}}{\partial\mathbf{Q}} = \mathbf{T}\Lambda\mathbf{T}^{-1} = \mathbf{T}(\Lambda^+ + \Lambda^-)\mathbf{T}^{-1}\\ \Lambda^{\pm} &= \frac{\Lambda\pm|\Lambda|}{2}\\ |\mathbf{A}_{\text{inv}}| &= \mathbf{T}|\Lambda|\mathbf{T}^{-1}\end{aligned} \tag{C-34}$$

The diagonal matrix Λ is the matrix of eigenvalues of $\mathbf{A}_{\text{inv}}$, $\mathbf{T}$ is the matrix of right eigenvectors as columns, and $\mathbf{T}^{-1}$ is the matrix of left eigenvectors as rows. They are all evaluated at Roe-averaged values such that

$$\hat{\mathbf{F}}(\mathbf{Q}_R)-\hat{\mathbf{F}}(\mathbf{Q}_L) = \tilde{\mathbf{A}}_{\text{inv}}(\mathbf{Q}_R-\mathbf{Q}_L) \tag{C-35}$$

is satisfied exactly. The term $|\tilde{\mathbf{A}}_{\text{inv}}|(\mathbf{Q}_R-\mathbf{Q}_L)$ can be written

$$\left|\tilde{\mathbf{A}}_{\text{inv}}\right|(\mathbf{Q}_R-\mathbf{Q}_L)\equiv\left|\tilde{\mathbf{A}}_{\text{inv}}\right|\Delta\mathbf{Q}\;=\;\begin{bmatrix}\alpha_4\\ \tilde{u}\alpha_4+\hat{\xi}_x\alpha_5+\alpha_6\\ \tilde{v}\alpha_4+\hat{\xi}_y\alpha_5+\alpha_7\\ \tilde{w}\alpha_4+\hat{\xi}_z\alpha_5+\alpha_8\\ \tilde{H}\alpha_4+(\tilde{\bar{U}}-\hat{\xi}_t)\alpha_5+\tilde{u}\alpha_6+\tilde{v}\alpha_7+\tilde{w}\alpha_8-\dfrac{\tilde{a}^2\alpha_1}{\gamma-1}\end{bmatrix} \tag{C-36}$$

where

$$\begin{aligned}
\alpha_1 &= \left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}\right|\left(\Delta\rho-\frac{\Delta p}{\tilde{a}^2}\right)\\
\alpha_2 &= \frac{1}{2\tilde{a}^2}\left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}+\tilde{a}\right|(\Delta p+\tilde{\rho}\tilde{a}\Delta\bar{U})\\
\alpha_3 &= \frac{1}{2\tilde{a}^2}\left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}-\tilde{a}\right|(\Delta p-\tilde{\rho}\tilde{a}\Delta\bar{U})\\
\alpha_4 &= \alpha_1+\alpha_2+\alpha_3\\
\alpha_5 &= \tilde{a}(\alpha_2-\alpha_3)\\
\alpha_6 &= \left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}\right|(\tilde{\rho}\Delta u-\hat{\xi}_x\tilde{\rho}\Delta\bar{U})\\
\alpha_7 &= \left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}\right|(\tilde{\rho}\Delta v-\hat{\xi}_y\tilde{\rho}\Delta\bar{U})\\
\alpha_8 &= \left|\frac{\nabla\xi}{J}\right|\left|\tilde{\bar{U}}\right|(\tilde{\rho}\Delta w-\hat{\xi}_z\tilde{\rho}\Delta\bar{U})
\end{aligned} \tag{C-37}$$

Here, the notation ~ denotes the following Roe-averaged evaluations:

$$
\begin{aligned}
\tilde{\rho} &= \sqrt{\rho_L \rho_R} \\
\tilde{u} &= \frac{u_L + u_R\sqrt{\rho_R/\rho_L}}{1+\sqrt{\rho_R/\rho_L}} \\
\tilde{v} &= \frac{v_L + v_R\sqrt{\rho_R/\rho_L}}{1+\sqrt{\rho_R/\rho_L}} \\
\tilde{w} &= \frac{w_L + w_R\sqrt{\rho_R/\rho_L}}{1+\sqrt{\rho_R/\rho_L}} \\
\tilde{H} &= \frac{H_L + H_R\sqrt{\rho_R/\rho_L}}{1+\sqrt{\rho_R/\rho_L}} \\
\tilde{a}^2 &= (\gamma-1)\tilde{H} - \frac{\tilde{u}^2+\tilde{v}^2+\tilde{w}^2}{2}
\end{aligned}
\qquad \text{(C-38)}
$$

The terms $\hat{\xi}_x$, $\hat{\xi}_y$, $\hat{\xi}_z$, $\hat{\xi}_t$ and $\overline{U}$ are defined in Section C.1.2 and

$$
\tilde{\overline{U}} = \frac{1}{|\nabla\xi|}(\xi_x\tilde{u} + \xi_y\tilde{v} + \xi_z\tilde{w} + \xi_t) \qquad \text{(C-39)}
$$

For flux-difference splitting, the true Jacobians are expensive to compute. Two alternatives are employed in CFL3D. When the system of equations is solved with a full 5 by 5 block-tridiagonal inversion procedure (**idiag** = 0), the following approximations for the left-hand-side flux-difference splitting Jacobians are employed:

$$
\begin{aligned}
(\mathbf{A}^+_{\text{inv}})_{i+\frac{1}{2}} &= \frac{1}{2}\left[(\mathbf{A}_{\text{inv}})_i + \left|\tilde{\mathbf{A}}_{\text{inv}}\right|_{i+\frac{1}{2}}\right] \\
(\mathbf{A}^-_{\text{inv}})_{i+\frac{1}{2}} &= \frac{1}{2}\left[(\mathbf{A}_{\text{inv}})_{i+1} - \left|\tilde{\mathbf{A}}_{\text{inv}}\right|_{i+\frac{1}{2}}\right]
\end{aligned}
\qquad \text{(C-40)}
$$

Flux-difference splitting also has an option to employ a diagonal approximation for the left-hand side (**idiag** = 1). In this method, each of the spatial factors is approximated with a diagonal inversion as

$$
\left[\frac{\mathrm{I}}{J\Delta t} + \delta_\xi\frac{\partial \mathbf{F}}{\partial \mathbf{Q}}\right]\Delta\mathbf{Q} \approx \mathbf{T}\left[\frac{\mathrm{I}}{J\Delta t} + \delta^-_\xi\Lambda^+ + \delta^+_\xi\Lambda^-\right]\mathbf{T}^{-1}\Delta\mathbf{Q} \qquad \text{(C-41)}
$$

Because of the repeated eigenvalues of Λ, only three scalar tridiagonal LU decompositions are needed in each direction, resulting in a significant savings in run time.

C.2 Viscous Fluxes

The viscous terms, which represent shear stress and heat transfer effects, are discretized with second-order central differences. The second derivatives are treated as differences across cell interfaces of first-derivative terms. Hence, in the ξ direction for example, the viscous terms are discretized as

$$\delta_\xi(\hat{\mathbf{F}}_v)_i = (\hat{\mathbf{F}}_v)_{i+\frac{1}{2}} - (\hat{\mathbf{F}}_v)_{i-\frac{1}{2}} \tag{C-42}$$

The term $\hat{\mathbf{F}}_v$ is given in Equation (A-8) of Appendix A. Using the thin-layer approximation, it can be written as:

$$\hat{\mathbf{F}}_v = \frac{M_\infty \mu}{Re_{\tilde{L}_R} J} \begin{bmatrix} 0 \\ \phi_1 \dfrac{\partial u}{\partial \xi} - \xi_x \phi_2 \\ \phi_1 \dfrac{\partial v}{\partial \xi} - \xi_y \phi_2 \\ \phi_1 \dfrac{\partial w}{\partial \xi} - \xi_z \phi_2 \\ \phi_1 \left[\delta_\xi \left(\dfrac{|\mathbf{V}|^2}{2} \right) + \dfrac{1}{Pr(\gamma - 1)} \delta_\xi (a^2) \right] + (U - \xi_t)\phi_2 \end{bmatrix} \tag{C-43}$$

where

$$\phi_1 = \xi_x^2 + \xi_y^2 + \xi_z^2 \qquad \phi_2 = \left(\xi_x \frac{\partial u}{\partial \xi} + \xi_y \frac{\partial v}{\partial \xi} + \xi_z \frac{\partial w}{\partial \xi} \right) \Big/ 3 \tag{C-44}$$

Implemented in a finite-volume approach, Equation (C-43) requires an approximation to the volume at the cell interface $(1/J)_{i+1/2}$, which is calculated by averaging the neighboring values.

When the system of equations is solved with a full 5 by 5 block-tridiagonal inversion procedure (**idiag** = 0), the viscous Jacobians (left-hand-side implicit terms) are employed only for the viscous terms in the ζ direction (corresponding with the k index of the grid). This is a hold-over from the early days of the CFL3D code, when viscous terms were employed in only one direction. Hence, keep in mind that if the **idiag** = 0 option is used when viscous terms are included in either the j or i directions, the convergence may suffer due to the lack of the appropriate left-hand-side terms. The missing left-hand-side terms have no effect on *converged* solutions, however.

However, almost all runs with CFL3D employ flux-difference splitting with the diagonal left-hand-side option (**idiag** = 1), because flux-difference splitting is generally more accurate for Navier-Stokes computations (see van Leer et al[41]), and **idiag** = 1 is significantly less expensive than **idiag** = 0. For the diagonal method, a spectral radius scaling for the viscous Jacobian matrices is used, similar to that developed by Coakley.[14] In this reference, the true left-hand-side viscous matrix term is replaced by the matrix $\tilde{\nu}\mathbf{I}$, where $\mathbf{I}$ is the identity matrix, $\tilde{\nu} = \tilde{\mu}_{\max}/\tilde{\rho}$, and $\tilde{\mu}_{\max}$ is the largest eigenvalue of the one-dimensional Navier-Stokes equation, $\tilde{\mu}_{\max} = \max(4\tilde{\mu}/3, \gamma\tilde{\mu}/Pr)$. In CFL3D, nondimensional $\mu_{\max}$ is taken as $2(1+\mu_t)$, where μ_t is the nondimensional turbulent eddy viscosity.

APPENDIX D

Multigrid Algorithm

The full-approximation storage (FAS) multigrid algorithm is used to accelerate convergence to steady state (or to accelerate convergence of sub-iterations during a time-accurate computation). A sequence of grids $G_0, G_1, \ldots, G_N$ is defined, where G_N denotes the finest grid, and coarser grids are formed by successively deleting every other grid line in all three coordinate directions. The fine grid serves to damp the high-frequency errors; the coarser grids damp the low-frequency errors. The coarse grids are solved with a forcing function on the right-hand side, arising from restricting the residual from the finer meshes. The forcing function is the relative truncation error between the grids, such that the solution on the coarser meshes are driven by the fine grid residual. The resulting scheme on mesh G_i is given as

$$N_i \Delta \mathbf{q}_i^c = -[L_i(\mathbf{q}_i^c) - \tau_i] \equiv -R_i \tag{D-1}$$

where $\mathbf{q}_i^c$ is the current approximation to the solution on mesh G_i, N_i is the spatially-factored implicit matrix, and τ_i is the relative truncation error (where τ_N is defined to be zero). The relative truncation error is calculated as

$$\tau_i = L_i(I_{i+1}^i \mathbf{q}_{i+1}^c) - \hat{I}_{i+1}^i R_{i+1} \tag{D-2}$$

The operator I_{i+1}^i is a volume-weighted restriction operator that transfers values on the finer grid to the coarser grid

$$I_{i+1}^i \mathbf{q}_{i+1} = \frac{\sum \frac{\mathbf{q}_{i+1}}{J_{i+1}}}{\sum \frac{1}{J_{i+1}}} \tag{D-3}$$

where the summation is taken over the eight finer grid cells that make up the coarser grid cell. The restriction operator $\hat{I}_{i+1}^i$ represents a summation over the finer grid cells which make up the coarser cell

$$\hat{I}_{i+1}^i R_{i+1} = \sum R_{i+1} \tag{D-4}$$

The corrections V_i on the coarser meshes are used to update the finer mesh

$$\begin{aligned} V_i &= \mathbf{q}_i^c - I_{i+1}^i \mathbf{q}_{i+1}^c \\ \mathbf{q}_i^c &\leftarrow \mathbf{q}_i^c + I_{i-1}^i V_{i-1} \end{aligned} \tag{D-5}$$

where the prolongation operator corresponds to trilinear interpolation.

When correction smoothing (**issc** = 1) is employed, the corrections V_i are smoothed on G_i, before prolongation, with a Laplacian-type operator factored into three sweeps. Equation (D-5) is replaced with

$$\begin{aligned} V_i &= \mathbf{q}_i^c - I_{i+1}^i \mathbf{q}_{i+1}^c \\ (I - \varepsilon_\xi^c \delta_{\xi\xi})(I - \varepsilon_\eta^c \delta_{\eta\eta})(I - \varepsilon_\zeta^c \delta_{\zeta\zeta})\tilde{V}_i &= V_i \\ \mathbf{q}_i^c &\leftarrow \mathbf{q}_i^c + I_{i-1}^i \tilde{V}_{i-1} \end{aligned} \tag{D-6}$$

The ε_ξ^c, ε_η^c, and ε_ζ^c coefficients are user-input values. When used, typical values are 0.3. The correction smoothing overcomes an odd-even decoupling sometimes encountered with the FAS algorithm on highly-stretched grids.

When residual smoothing (**issr** = 1) is employed, the same Laplacian-type operator is used to smooth the $\Delta\mathbf{q}_i^c$ values, just prior to updating the primitive variables on a given grid level.

$$(I - \varepsilon_\xi^r \delta_{\xi\xi})(I - \varepsilon_\eta^r \delta_{\eta\eta})(I - \varepsilon_\zeta^r \delta_{\zeta\zeta})\tilde{\mathbf{q}}_i^c = \Delta\mathbf{q}_i^c \tag{D-7}$$

The ε_ξ^r, ε_η^r, and ε_ζ^r coefficients are again user-input values. When used, typical values are 0.3.

APPENDIX E

Primitive Variables

While CFL3D solves the governing equations in conservation law form, only the primitive variables are stored in permanent memory. This necessitates the use of the transformation matrix:

$$\mathbf{M} = \frac{\partial \mathbf{Q}}{\partial \mathbf{q}} \tag{E-1}$$

where

$$\mathbf{Q} = \left[\rho \quad \rho u \quad \rho v \quad \rho w \quad e \right]^{\mathrm{T}} \tag{E-2}$$

$$\mathbf{q} = \left[\rho \quad u \quad v \quad w \quad p \right]^{\mathrm{T}} \tag{E-3}$$

From Equation (A-14),

$$e = \frac{p}{\gamma - 1} + \frac{\rho}{2}(u^2 + v^2 + w^2) \tag{E-4}$$

Therefore,

$$\mathbf{M} = \begin{bmatrix} \frac{\partial \rho}{\partial \rho} & \frac{\partial \rho}{\partial u} & \frac{\partial \rho}{\partial v} & \frac{\partial \rho}{\partial w} & \frac{\partial \rho}{\partial p} \\ \frac{\partial \rho u}{\partial \rho} & \frac{\partial \rho u}{\partial u} & \frac{\partial \rho u}{\partial v} & \frac{\partial \rho u}{\partial w} & \frac{\partial \rho u}{\partial p} \\ \frac{\partial \rho v}{\partial \rho} & \frac{\partial \rho v}{\partial u} & \frac{\partial \rho v}{\partial v} & \frac{\partial \rho v}{\partial w} & \frac{\partial \rho v}{\partial p} \\ \frac{\partial \rho w}{\partial \rho} & \frac{\partial \rho w}{\partial u} & \frac{\partial \rho w}{\partial v} & \frac{\partial \rho w}{\partial w} & \frac{\partial \rho w}{\partial p} \\ \frac{\partial e}{\partial \rho} & \frac{\partial e}{\partial u} & \frac{\partial e}{\partial v} & \frac{\partial e}{\partial w} & \frac{\partial e}{\partial p} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ u & \rho & 0 & 0 & 0 \\ v & 0 & \rho & 0 & 0 \\ w & 0 & 0 & \rho & 0 \\ \frac{u^2 + v^2 + w^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma - 1} \end{bmatrix} \tag{E-5}$$

The inverse of $\mathbf{M}$ is

$$\mathbf{M}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -\frac{u}{\rho} & \frac{1}{\rho} & 0 & 0 & 0 \\ -\frac{v}{\rho} & 0 & \frac{1}{\rho} & 0 & 0 \\ -\frac{w}{\rho} & 0 & 0 & \frac{1}{\rho} & 0 \\ \frac{\gamma-1}{2}(u^2+v^2+w^2) & -u(\gamma-1) & -v(\gamma-1) & -w(\gamma-1) & 1 \end{bmatrix} \quad \text{(E-6)}$$

APPENDIX F

Generalized Coordinates

F.1 Navier-Stokes Equations in Cartesian Coordinates

The compressible three-dimensional Navier-Stokes equations, excluding body forces and external heat sources, in Cartesian coordinates are

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} + \frac{\partial \mathbf{h}}{\partial z} = 0 \tag{F-1}$$

where

$$\mathbf{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix} \tag{F-2}$$

$$\mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p - \tau_{xx} \\ \rho uv - \tau_{xy} \\ \rho uw - \tau_{xz} \\ (e+p)u - u\tau_{xx} - v\tau_{xy} - w\tau_{xz} + q_x \end{bmatrix} \tag{F-3}$$

$$\mathbf{g} = \begin{bmatrix} \rho v \\ \rho uv - \tau_{xy} \\ \rho v^2 + p - \tau_{yy} \\ \rho vw - \tau_{yz} \\ (e+p)v - u\tau_{xy} - v\tau_{yy} - w\tau_{yz} + q_y \end{bmatrix} \tag{F-4}$$

$$\mathbf{h} = \begin{bmatrix} \rho w \\ \rho uw - \tau_{xz} \\ \rho vw - \tau_{yz} \\ \rho w^2 + p - \tau_{zz} \\ (e+p)w - u\tau_{xz} - v\tau_{yz} - w\tau_{zz} + q_z \end{bmatrix} \tag{F-5}$$

and

$$\tau_{xx} = \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z}\right) \tag{F-6}$$

$$\tau_{yy} = \frac{2}{3}\mu\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z}\right) \tag{F-7}$$

$$\tau_{zz} = \frac{2}{3}\mu\left(2\frac{\partial w}{\partial z} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) \tag{F-8}$$

$$\tau_{xy} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) = \tau_{yx} \tag{F-9}$$

$$\tau_{xz} = \mu\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) = \tau_{zx} \tag{F-10}$$

$$\tau_{yz} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) = \tau_{zy} \tag{F-11}$$

The pressure is defined by the equation of state for an ideal gas:

$$p = (\gamma - 1)\left[e - \frac{\rho}{2}(u^2 + v^2 + w^2)\right] \tag{F-12}$$

F.2 Transformation to Generalized Coordinates

Now apply the generalized coordinate transformation

$$\begin{aligned} \xi &= \xi(x, y, z, t) \\ \eta &= \eta(x, y, z, t) \\ \zeta &= \zeta(x, y, z, t) \end{aligned} \tag{F-13}$$

From the chain-rule for a function of multiple variables,

$$\begin{aligned}
\frac{\partial}{\partial x} &= \xi_x\frac{\partial}{\partial \xi} + \eta_x\frac{\partial}{\partial \eta} + \zeta_x\frac{\partial}{\partial \zeta} + t_x\frac{\partial}{\partial t} \\
\frac{\partial}{\partial y} &= \xi_y\frac{\partial}{\partial \xi} + \eta_y\frac{\partial}{\partial \eta} + \zeta_y\frac{\partial}{\partial \zeta} + t_y\frac{\partial}{\partial t} \\
\frac{\partial}{\partial z} &= \xi_z\frac{\partial}{\partial \xi} + \eta_z\frac{\partial}{\partial \eta} + \zeta_z\frac{\partial}{\partial \zeta} + t_z\frac{\partial}{\partial t} \\
\frac{\partial}{\partial t} &= \xi_t\frac{\partial}{\partial \xi} + \eta_t\frac{\partial}{\partial \eta} + \zeta_t\frac{\partial}{\partial \zeta} + t_t\frac{\partial}{\partial t}
\end{aligned} \tag{F-14}$$

where ξ_x, η_x, ζ_x, ξ_y, η_y, ζ_y, ξ_z, η_z, ζ_z, ξ_t, η_t, ζ_t are the metrics.

F.2.1 Obtaining the Metrics

The metrics are determined as follows. (For a description of the geometrical evaluation of the metrics, see “Geometrical Evaluation of the Metrics” on page 263.) The derivatives of the generalized coordinates can be written

$$\begin{aligned}
d\xi &= \xi_x dx + \xi_y dy + \xi_z dz + \xi_t dt \\
d\eta &= \eta_x dx + \eta_y dy + \eta_z dz + \eta_t dt \\
d\zeta &= \zeta_x dx + \zeta_y dy + \zeta_z dz + \zeta_t dt \\
dt &= t_x dx + t_y dy + t_z dz + t_t dt
\end{aligned} \tag{F-15}$$

or, in matrix form (noting that $t_x = t_y = t_z = 0$ and $t_t = 1$),

$$\begin{bmatrix} d\xi \\ d\eta \\ d\zeta \\ dt \end{bmatrix} = \begin{bmatrix} \xi_x & \xi_y & \xi_z & \xi_t \\ \eta_x & \eta_y & \eta_z & \eta_t \\ \zeta_x & \zeta_y & \zeta_z & \zeta_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \\ dt \end{bmatrix} \tag{F-16}$$

Similarly, the derivatives of the Cartesian coordinates can be written

$$\begin{bmatrix} dx \\ dy \\ dz \\ dt \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta & x_\zeta & x_t \\ y_\xi & y_\eta & y_\zeta & y_t \\ z_\xi & z_\eta & z_\zeta & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d\xi \\ d\eta \\ d\zeta \\ dt \end{bmatrix} \tag{F-17}$$

Therefore, it is evident that

$$\begin{bmatrix} \xi_x & \xi_y & \xi_z & \xi_t \\ \eta_x & \eta_y & \eta_z & \eta_t \\ \zeta_x & \zeta_y & \zeta_z & \zeta_t \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x_\xi & x_\eta & x_\zeta & x_t \\ y_\xi & y_\eta & y_\zeta & y_t \\ z_\xi & z_\eta & z_\zeta & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \tag{F-18}$$

To determine the inverse matrix represented on the right-hand side of Equation (F-18), first the cofactor matrix is found to be:

$$\begin{bmatrix} (y_\eta z_\zeta - y_\zeta z_\eta) & -(y_\xi z_\zeta - y_\zeta z_\xi) & (y_\xi z_\eta - y_\eta z_\xi) & 0 \\ -(x_\eta z_\zeta - x_\zeta z_\eta) & (x_\xi z_\zeta - x_\zeta z_\xi) & -(x_\xi z_\eta - x_\eta z_\xi) & 0 \\ (x_\eta y_\zeta - x_\zeta y_\eta) & -(x_\xi y_\zeta - x_\zeta y_\xi) & (x_\xi y_\eta - x_\eta y_\xi) & 0 \\ \mathrm{CF}_{41} & \mathrm{CF}_{42} & \mathrm{CF}_{43} & \mathrm{CF}_{44} \end{bmatrix} \tag{F-19}$$

where

$$\begin{aligned} \mathrm{CF}_{41} &= -x_t(y_\eta z_\zeta - y_\zeta z_\eta) - y_t(x_\zeta z_\eta - x_\eta z_\zeta) - z_t(x_\eta y_\zeta - x_\zeta y_\eta) \\ \mathrm{CF}_{42} &= x_t(y_\xi z_\zeta - y_\zeta z_\xi) + y_t(x_\zeta z_\xi - x_\xi z_\zeta) + z_t(x_\xi y_\zeta - x_\zeta y_\xi) \\ \mathrm{CF}_{43} &= -x_t(y_\xi z_\eta - y_\eta z_\xi) - y_t(x_\eta z_\xi - x_\xi z_\eta) - z_t(x_\xi y_\eta - x_\eta y_\xi) \\ \mathrm{CF}_{44} &= x_\xi(y_\eta z_\zeta - y_\zeta z_\eta) + x_\eta(y_\zeta z_\xi - y_\xi z_\zeta) + x_\zeta(y_\xi z_\eta - y_\eta z_\xi) \end{aligned} \tag{F-20}$$

The transpose of the cofactor matrix (i.e. the adjoint matrix) is

$$\begin{bmatrix} (y_\eta z_\zeta - y_\zeta z_\eta) & -(x_\eta z_\zeta - x_\zeta z_\eta) & (x_\eta y_\zeta - x_\zeta y_\eta) & \mathrm{CF}_{41} \\ -(y_\xi z_\zeta - y_\zeta z_\xi) & (x_\xi z_\zeta - x_\zeta z_\xi) & -(x_\xi y_\zeta - x_\zeta y_\xi) & \mathrm{CF}_{42} \\ (y_\xi z_\eta - y_\eta z_\xi) & -(x_\xi z_\eta - x_\eta z_\xi) & (x_\xi y_\eta - x_\eta y_\xi) & \mathrm{CF}_{43} \\ 0 & 0 & 0 & \mathrm{CF}_{44} \end{bmatrix} \tag{F-21}$$

The determinant of the inverse matrix represented on the right-hand side of Equation (F-18) is

$$\begin{vmatrix} x_\xi & x_\eta & x_\zeta & x_t \\ y_\xi & y_\eta & y_\zeta & y_t \\ z_\xi & z_\eta & z_\zeta & z_t \\ 0 & 0 & 0 & 1 \end{vmatrix} = x_\xi(y_\eta z_\zeta - y_\zeta z_\eta) - x_\eta(y_\xi z_\zeta - y_\zeta z_\xi) + x_\zeta(y_\xi z_\eta - y_\eta z_\xi) \tag{F-22}$$

The Jacobian of the transformation is defined to be

$$J = \frac{\partial(\xi, \eta, \zeta, t)}{\partial(x, y, z, t)} = \begin{vmatrix} \xi_x & \xi_y & \xi_z & \xi_t \\ \eta_x & \eta_y & \eta_z & \eta_t \\ \zeta_x & \zeta_y & \zeta_z & \zeta_t \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad \text{(F-23)}$$

Therefore,

$$\frac{1}{J} = \frac{\partial(x, y, z, t)}{\partial(\xi, \eta, \zeta, t)} = \begin{vmatrix} x_\xi & x_\eta & x_\zeta & x_t \\ y_\xi & y_\eta & y_\zeta & y_t \\ z_\xi & z_\eta & z_\zeta & z_t \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad \text{(F-24)}$$

Note that $\mathrm{CF}_{44} = 1/J$. Now, from the property of an invertible matrix $\mathbf{D}$,

$$\mathbf{D}^{-1} = \frac{1}{det(\mathbf{D})} adj(\mathbf{D}) \quad \text{(F-25)}$$

let

$$\mathbf{D} = \begin{bmatrix} x_\xi & x_\eta & x_\zeta & x_t \\ y_\xi & y_\eta & y_\zeta & y_t \\ z_\xi & z_\eta & z_\zeta & z_t \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(F-26)}$$

It follows from Equation (F-18) that

$$\mathbf{D}^{-1} = \begin{bmatrix} \xi_x & \xi_y & \xi_z & \xi_t \\ \eta_x & \eta_y & \eta_z & \eta_t \\ \zeta_x & \zeta_y & \zeta_z & \zeta_t \\ 0 & 0 & 0 & 1 \end{bmatrix} = J \begin{bmatrix} (y_\eta z_\zeta - y_\zeta z_\eta) & -(x_\eta z_\zeta - x_\zeta z_\eta) & (x_\eta y_\zeta - x_\zeta y_\eta) & \mathrm{CF}_{41} \\ -(y_\xi z_\zeta - y_\zeta z_\xi) & (x_\xi z_\zeta - x_\zeta z_\xi) & -(x_\xi y_\zeta - x_\zeta y_\xi) & \mathrm{CF}_{42} \\ (y_\xi z_\eta - y_\eta z_\xi) & -(x_\xi z_\eta - x_\eta z_\xi) & (x_\xi y_\eta - x_\eta y_\xi) & \mathrm{CF}_{43} \\ 0 & 0 & 0 & \frac{1}{J} \end{bmatrix} \quad \text{(F-27)}$$

Therefore, the metrics are

$$
\begin{aligned}
\xi_x &= J(y_\eta z_\zeta - y_\zeta z_\eta) \\
\xi_y &= J(x_\zeta z_\eta - x_\eta z_\zeta) \\
\xi_z &= J(x_\eta y_\zeta - x_\zeta y_\eta) \\
\xi_t &= -x_t\xi_x - y_t\xi_y - z_t\xi_z \\
\eta_x &= J(y_\zeta z_\xi - y_\xi z_\zeta) \\
\eta_y &= J(x_\xi z_\zeta - x_\zeta z_\xi) \\
\eta_z &= J(x_\zeta y_\xi - x_\xi y_\zeta) \\
\eta_t &= -x_t\eta_x - y_t\eta_y - z_t\eta_z \\
\zeta_x &= J(y_\xi z_\eta - y_\eta z_\xi) \\
\zeta_y &= J(x_\eta z_\xi - x_\xi z_\eta) \\
\zeta_z &= J(x_\xi y_\eta - x_\eta y_\xi) \\
\zeta_t &= -x_t\zeta_x - y_t\zeta_y - z_t\zeta_z
\end{aligned}
\qquad \text{(F-28)}
$$

F.2.2 Applying the Transformation

Now, to apply the transformation to the Navier-Stokes equations represented in Equation (F-1), substitute Equation (F-14) into Equation (F-1) and multiply by $1/J$:

$$
\begin{aligned}
&\frac{1}{J}\frac{\partial \mathbf{Q}}{\partial t} + \frac{1}{J}\frac{\partial \mathbf{Q}}{\partial \xi}\xi_t + \frac{1}{J}\frac{\partial \mathbf{Q}}{\partial \eta}\eta_t + \frac{1}{J}\frac{\partial \mathbf{Q}}{\partial \zeta}\zeta_t + \\
&\frac{1}{J}\frac{\partial \mathbf{f}}{\partial t}t_x + \frac{1}{J}\frac{\partial \mathbf{f}}{\partial \xi}\xi_x + \frac{1}{J}\frac{\partial \mathbf{f}}{\partial \eta}\eta_x + \frac{1}{J}\frac{\partial \mathbf{f}}{\partial \zeta}\zeta_x + \\
&\frac{1}{J}\frac{\partial \mathbf{g}}{\partial t}t_y + \frac{1}{J}\frac{\partial \mathbf{g}}{\partial \xi}\xi_y + \frac{1}{J}\frac{\partial \mathbf{g}}{\partial \eta}\eta_y + \frac{1}{J}\frac{\partial \mathbf{g}}{\partial \zeta}\zeta_y + \\
&\frac{1}{J}\frac{\partial \mathbf{h}}{\partial t}t_z + \frac{1}{J}\frac{\partial \mathbf{h}}{\partial \xi}\xi_z + \frac{1}{J}\frac{\partial \mathbf{h}}{\partial \eta}\eta_z + \frac{1}{J}\frac{\partial \mathbf{h}}{\partial \zeta}\zeta_z = 0
\end{aligned}
\qquad \text{(F-29)}
$$

(Remember, however, that $t_x = t_y = t_z = 0$). Since

$$
\frac{\partial}{\partial \xi}\left(\mathbf{f}\frac{\xi_x}{J}\right) = \mathbf{f}\frac{\partial}{\partial \xi}\left(\frac{\xi_x}{J}\right) + \frac{\xi_x}{J}\frac{\partial \mathbf{f}}{\partial \xi} \qquad \text{(F-30)}
$$

Then,

$$
\frac{\xi_x}{J}\frac{\partial \mathbf{f}}{\partial \xi} = \frac{\partial}{\partial \xi}\left(\mathbf{f}\frac{\xi_x}{J}\right) - \mathbf{f}\frac{\partial}{\partial \xi}\left(\frac{\xi_x}{J}\right) \qquad \text{(F-31)}
$$

So, Equation (F-29) becomes

$$\frac{\partial}{\partial t}\left(\frac{\mathbf{Q}}{J}\right)-\mathbf{Q}\frac{\partial}{\partial t}\left(\frac{1}{J}\right) \tag{F-32}$$
$$+\frac{\partial}{\partial\xi}\left(\frac{\mathbf{Q}}{J}\xi_t\right)-\mathbf{Q}\frac{\partial}{\partial\xi}\left(\frac{\xi_t}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\mathbf{Q}}{J}\eta_t\right)-\mathbf{Q}\frac{\partial}{\partial\eta}\left(\frac{\eta_t}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\mathbf{Q}}{J}\zeta_t\right)-\mathbf{Q}\frac{\partial}{\partial\zeta}\left(\frac{\zeta_t}{J}\right)$$
$$+\frac{\partial}{\partial\xi}\left(\frac{\mathbf{f}}{J}\xi_x\right)-\mathbf{f}\frac{\partial}{\partial\xi}\left(\frac{\xi_x}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\mathbf{f}}{J}\eta_x\right)-\mathbf{f}\frac{\partial}{\partial\eta}\left(\frac{\eta_x}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\mathbf{f}}{J}\zeta_x\right)-\mathbf{f}\frac{\partial}{\partial\zeta}\left(\frac{\zeta_x}{J}\right)$$
$$+\frac{\partial}{\partial\xi}\left(\frac{\mathbf{g}}{J}\xi_y\right)-\mathbf{g}\frac{\partial}{\partial\xi}\left(\frac{\xi_y}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\mathbf{g}}{J}\eta_y\right)-\mathbf{g}\frac{\partial}{\partial\eta}\left(\frac{\eta_y}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\mathbf{g}}{J}\zeta_y\right)-\mathbf{g}\frac{\partial}{\partial\zeta}\left(\frac{\zeta_y}{J}\right)$$
$$+\frac{\partial}{\partial\xi}\left(\frac{\mathbf{h}}{J}\xi_z\right)-\mathbf{h}\frac{\partial}{\partial\xi}\left(\frac{\xi_z}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\mathbf{h}}{J}\eta_z\right)-\mathbf{h}\frac{\partial}{\partial\eta}\left(\frac{\eta_z}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\mathbf{h}}{J}\zeta_z\right)-\mathbf{h}\frac{\partial}{\partial\zeta}\left(\frac{\zeta_z}{J}\right)=0$$

Regrouping,

$$\frac{\partial}{\partial t}\left(\frac{\mathbf{Q}}{J}\right)+\frac{\partial}{\partial\xi}\left[\frac{1}{J}(\mathbf{f}\xi_x+\mathbf{g}\xi_y+\mathbf{h}\xi_z+\mathbf{Q}\xi_t)\right] \tag{F-33}$$
$$+\frac{\partial}{\partial\eta}\left[\frac{1}{J}(\mathbf{f}\eta_x+\mathbf{g}\eta_y+\mathbf{h}\eta_z+\mathbf{Q}\eta_t)\right]$$
$$+\frac{\partial}{\partial\zeta}\left[\frac{1}{J}(\mathbf{f}\zeta_x+\mathbf{g}\zeta_y+\mathbf{h}\zeta_z+\mathbf{Q}\zeta_t)\right]$$
$$-\mathbf{f}\left[\frac{\partial}{\partial\xi}\left(\frac{\xi_x}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\eta_x}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\zeta_x}{J}\right)\right]$$
$$-\mathbf{g}\left[\frac{\partial}{\partial\xi}\left(\frac{\xi_y}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\eta_y}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\zeta_y}{J}\right)\right]$$
$$-\mathbf{h}\left[\frac{\partial}{\partial\xi}\left(\frac{\xi_z}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\eta_z}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\zeta_z}{J}\right)\right]$$
$$-\mathbf{Q}\left[\frac{\partial}{\partial t}\left(\frac{1}{J}\right)+\frac{\partial}{\partial\xi}\left(\frac{\xi_t}{J}\right)+\frac{\partial}{\partial\eta}\left(\frac{\eta_t}{J}\right)+\frac{\partial}{\partial\zeta}\left(\frac{\zeta_t}{J}\right)\right]=0$$

Consider the fifth major term in Equation (F-33):

$$\frac{\partial}{\partial\xi}\left(\frac{\xi_x}{J}\right)=\frac{\partial}{\partial\xi}(y_\eta z_\zeta-y_\zeta z_\eta)=y_\eta z_{\xi\zeta}+y_{\xi\eta}z_\zeta-y_\zeta z_{\xi\eta}-y_{\xi\zeta}z_\eta$$
$$\frac{\partial}{\partial\eta}\left(\frac{\eta_x}{J}\right)=\frac{\partial}{\partial\eta}(y_\zeta z_\xi-y_\xi z_\zeta)=y_\zeta z_{\xi\eta}+y_{\eta\zeta}z_\xi-y_\xi z_{\eta\zeta}-y_{\xi\eta}z_\zeta \tag{F-34}$$
$$\frac{\partial}{\partial\zeta}\left(\frac{\zeta_x}{J}\right)=\frac{\partial}{\partial\zeta}(y_\xi z_\eta+y_\eta z_\xi)=y_\xi z_{\eta\zeta}+y_{\xi\zeta}z_\eta-y_\eta z_{\xi\zeta}-y_{\eta\zeta}z_\xi$$

The summation of the terms in Equation (F-34) is zero. The same can be shown for the sixth, seventh and eighth major terms in Equation (F-33). Therefore, Equation (F-33) can be written

$$\frac{\partial}{\partial t}\left(\frac{\mathbf{Q}}{J}\right) + \frac{\partial}{\partial \xi}\left[\frac{1}{J}(\mathbf{f}\xi_x + \mathbf{g}\xi_y + \mathbf{h}\xi_z + \mathbf{Q}\xi_t)\right] + \frac{\partial}{\partial \eta}\left[\frac{1}{J}(\mathbf{f}\eta_x + \mathbf{g}\eta_y + \mathbf{h}\eta_z + \mathbf{Q}\eta_t)\right] + \frac{\partial}{\partial \zeta}\left[\frac{1}{J}(\mathbf{f}\zeta_x + \mathbf{g}\zeta_y + \mathbf{h}\zeta_z + \mathbf{Q}\zeta_t)\right] = 0 \tag{F-35}$$

When summed, the expression in the second major term in Equation (F-35) can be written

$$\mathbf{f}\xi_x + \mathbf{g}\xi_y + \mathbf{h}\xi_z + \mathbf{Q}\xi_t = \begin{bmatrix} \rho U \\ \rho U u + p\xi_x \\ \rho U v + p\xi_y \\ \rho U w + p\xi_z \\ (e+p)U - \xi_t p \end{bmatrix} - \begin{bmatrix} 0 \\ \xi_x\tau_{xx} + \xi_y\tau_{xy} + \xi_z\tau_{xz} \\ \xi_x\tau_{xy} + \xi_y\tau_{yy} + \xi_z\tau_{yz} \\ \xi_x\tau_{xz} + \xi_y\tau_{yz} + \xi_z\tau_{zz} \\ \xi_x b_x + \xi_y b_y + \xi_z b_z \end{bmatrix} \tag{F-36}$$

where

$$U = \xi_x u + \xi_y v + \xi_z w + \xi_t \tag{F-37}$$

$$\begin{aligned} b_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + \dot{q}_x \\ b_y &= u\tau_{xy} + v\tau_{yy} + w\tau_{yz} + \dot{q}_y \\ b_z &= u\tau_{xz} + v\tau_{yz} + w\tau_{zz} + \dot{q}_z \end{aligned} \tag{F-38}$$

Equation (F-38) can be written compactly using indicial notation as

$$b_{x_i} = u_j\tau_{x_i x_j} - \dot{q}_{x_i} \tag{F-39}$$

with $i = 1, 2, 3$ and $j = 1, 2, 3$ where 1 indicates the x direction, 2 indicates the y direction, and 3 indicates the z direction.

F.3 Navier-Stokes Equation in Generalized Coordinates

Now, let

$$\mathbf{F} = \begin{bmatrix} \rho U \\ \rho Uu + p\xi_x \\ \rho Uv + p\xi_y \\ \rho Uw + p\xi_z \\ (e+p)U - \xi_t p \end{bmatrix} \qquad \mathbf{F}_v = \begin{bmatrix} 0 \\ \xi_x\tau_{xx} + \xi_y\tau_{xy} + \xi_z\tau_{xz} \\ \xi_x\tau_{xy} + \xi_y\tau_{yy} + \xi_z\tau_{yz} \\ \xi_x\tau_{xz} + \xi_y\tau_{yz} + \xi_z\tau_{zz} \\ \xi_x b_x + \xi_y b_y + \xi_z b_z \end{bmatrix} \tag{F-40}$$

Combining the other terms in a similar manner and letting $\hat{\mathbf{Q}} = \mathbf{Q}/J$, $\hat{\mathbf{F}} = \mathbf{F}/J$, $\hat{\mathbf{F}}_v = \mathbf{F}_v/J$, $\hat{\mathbf{G}} = \mathbf{G}/J$, $\hat{\mathbf{G}}_v = \mathbf{G}_v/J$, $\hat{\mathbf{H}} = \mathbf{H}/J$, $\hat{\mathbf{H}}_v = \mathbf{H}_v/J$, Equation (F-35) can be written

$$\frac{\partial \hat{\mathbf{Q}}}{\partial t} + \frac{\partial(\hat{\mathbf{F}} - \hat{\mathbf{F}}_v)}{\partial \xi} + \frac{\partial(\hat{\mathbf{G}} - \hat{\mathbf{G}}_v)}{\partial \eta} + \frac{\partial(\hat{\mathbf{H}} - \hat{\mathbf{H}}_v)}{\partial \zeta} = 0 \tag{F-41}$$

The terms are as shown in Appendix A, Equations (A-3) through (A-14).

F.4 Geometrical Evaluation of the Metrics

By analogy with the integral form of the equations, a geometrical interpretation of the metric terms can be made. The vector $\nabla \mathrm{k}/J$ is the directed area of the cell interface normal to a k = constant coordinate direction (k = ξ, η, or ζ). (See Figure F-1.) The unit vector $(\mathrm{k}_x, \mathrm{k}_y, \mathrm{k}_z)/|\nabla \mathrm{k}|$ is composed of the direction cosines of the cell interface and $|\nabla \mathrm{k}|/J$ is the area of the cell interface. Note that

$$\begin{aligned} \nabla \mathrm{k} &= \mathrm{k}_x \hat{i} + \mathrm{k}_y \hat{j} + \mathrm{k}_z \hat{k} \\ |\nabla \mathrm{k}| &= \sqrt{\mathrm{k}_x^2 + \mathrm{k}_y^2 + \mathrm{k}_z^2} \end{aligned} \tag{F-42}$$

The directed areas are calculated as one-half the vector product of the two diagonal vectors connecting opposite vertex points of a cell face, taken such that the directed area is parallel to the direction of increasing k.

Likewise, the normalized contravariant velocity, $\overline{U} = U/|\nabla \xi|$, for example, represents the relative velocity normal to a ξ interface, where $-\hat{\xi}_t = -\xi_t/|\nabla \xi|$ is the grid speed normal to the interface. The volume of the cell is $1/J$ and is determined by sum-

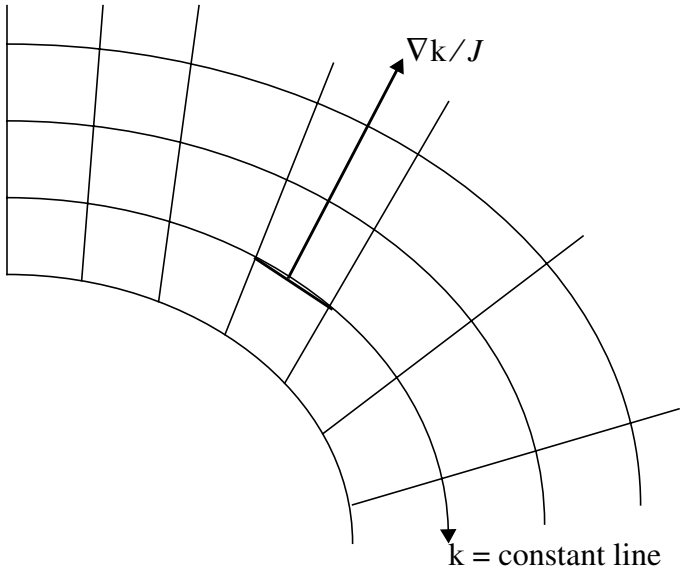


Figure F-1. Directed area schematic.

ming the volumes of the six pentahedra forming the hexagonal cell. Each pentahedron is defined by one of the six cell faces and the average point in the volume. The net effect is that the difference equations are satisfied identically when evaluated at free-stream conditions on arbitrary meshes.

In the code, the metric arrays are set up as follows using the ξ direction as an example.

`si(j,k,i,1)` $= \xi_x/|\nabla\xi| \equiv \hat{\xi}_x$ $= x$ component of unit normal to i face

`si(j,k,i,2)` $= \xi_y/|\nabla\xi| \equiv \hat{\xi}_y$ $= y$ component of unit normal to i face

`si(j,k,i,3)` $= \xi_z/|\nabla\xi| \equiv \hat{\xi}_z$ $= z$ component of unit normal to i face

`si(j,k,i,4)` $= |\nabla\xi|/J$ $= i$ face area

`si(j,k,i,5)` $= -\xi_t/|\nabla\xi|$ $= i$ face normal velocity

APPENDIX G

Force and Moment Calculations

G.1 Forces

The forces are calculated in CFL3D as follows. Let $\tilde{\vec{F}}_l$ be the total dimensional force acting on the surface element l, with area $\tilde{s}_l$, normalized by $\tilde{q}_\infty \tilde{s}_{ref}$, where

$$\tilde{q}_\infty = \frac{1}{2}\tilde{\rho}_\infty |\tilde{\mathbf{V}}|_\infty^2 \tag{G-1}$$

and $\tilde{s}_{ref}$ is the reference area. In what follows, $\tilde{s}_{ref}$ and $\tilde{s}_l$ are taken in terms of grid dimensions. Then the dimensionless force (force coefficient) acting on element l is

$$\vec{F}_l = \frac{\tilde{\vec{F}}_l}{\tilde{q}_\infty \tilde{s}_{ref}} \tag{G-2}$$

$\vec{F}_l$ is composed of pressure and viscous components, $\vec{F}_l^p$ and $\vec{F}_l^v$, respectively. The total force coefficient is computed by summing the contributions from all specified surface elements:

$$\vec{F} = \sum_l \vec{F}_l \tag{G-3}$$

G.1.1 Pressure Component

$\vec{F}_l^p$ is normal to the surface.

$$\tilde{s}_{ref}\vec{F}_l^p = \frac{\tilde{p} - \tilde{p}_\infty}{\frac{1}{2}\tilde{\rho}_\infty |\tilde{\mathbf{V}}|_\infty^2}\tilde{s}_l\vec{n} = 2\frac{\tilde{p}/(\tilde{\rho}_\infty \tilde{a}_\infty^2) - \tilde{p}_\infty/(\tilde{\rho}_\infty \tilde{a}_\infty^2)}{|\tilde{\mathbf{V}}|_\infty^2/\tilde{a}_\infty^2}\tilde{s}_l\vec{n} = 2\frac{p - \frac{1}{\gamma}}{M_\infty^2}\tilde{s}_l\vec{n} \tag{G-4}$$

Therefore

$$\tilde{s}_{ref}\vec{\tilde{F}}_l^p = \frac{2}{\gamma M_\infty^2}(\gamma p - 1)\tilde{s}_l\vec{\hat{n}} \tag{G-5}$$

The x, y, and z components of $\vec{\tilde{F}}_l^p$ are obtained by multiplying Equation (G-5) by the appropriate direction cosine. For example, if element l lies on an i = constant surface,

$$\vec{\hat{n}} = \nabla\hat{\xi} \tag{G-6}$$

$$\begin{aligned}
\tilde{s}_{ref}(\vec{\tilde{F}}_l^p)_x &= \frac{2}{\gamma M_\infty^2}(\gamma p - 1)\tilde{s}_l\hat{\xi}_x \\
\tilde{s}_{ref}(\vec{\tilde{F}}_l^p)_y &= \frac{2}{\gamma M_\infty^2}(\gamma p - 1)\tilde{s}_l\hat{\xi}_y \\
\tilde{s}_{ref}(\vec{\tilde{F}}_l^p)_z &= \frac{2}{\gamma M_\infty^2}(\gamma p - 1)\tilde{s}_l\hat{\xi}_z
\end{aligned} \tag{G-7}$$

where $\hat{\xi}_x = \xi_x/|\nabla\xi|$, $\hat{\xi}_y = \xi_y/|\nabla\xi|$, and $\hat{\xi}_z = \xi_z/|\nabla\xi|$.

G.1.2 Viscous Component

$\vec{\tilde{F}}_l^v$ is tangential to the surface.

$$\tilde{s}_{ref}\vec{\tilde{F}}_l^v = \frac{\tilde{\vec{\tau}}}{\frac{1}{2}\tilde{\rho}_\infty|\tilde{\mathbf{V}}|_\infty^2}\tilde{s}_l = 2\frac{\tilde{\vec{\tau}}/(\tilde{\rho}_\infty\tilde{a}_\infty^2)}{|\tilde{\mathbf{V}}|_\infty^2/\tilde{a}_\infty^2}\tilde{s}_l = \frac{2}{M_\infty^2}\vec{\hat{\tau}}\tilde{s}_l \tag{G-8}$$

Consider the flow near a surface element l of an i = constant surface. (See Figure G-1.)

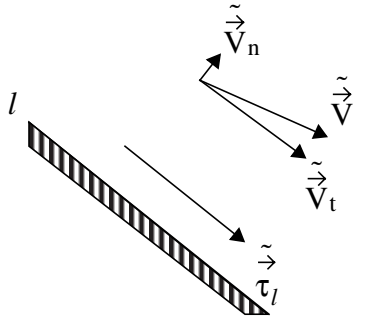


Figure G-1. Viscous force component example.

In the figure,

$$\begin{aligned} \tilde{\vec{V}} &= \text{velocity vector} \\ \tilde{\vec{V}}_n &= \text{normal velocity vector} \\ \tilde{\vec{V}}_t &= \text{tangential velocity vector (0 on the surface)} \end{aligned} \tag{G-9}$$

and

$$\tilde{\vec{V}}_t = \tilde{\vec{V}} - \tilde{\vec{V}}_n \tag{G-10}$$

$$\tilde{\vec{\tau}}_l = \tilde{\mu}\frac{\partial \tilde{\vec{V}}_t}{\partial \tilde{n}} = \tilde{\mu}\frac{\partial\left(\tilde{\vec{V}} - \tilde{\vec{V}}_n\right)}{\partial \tilde{n}} \tag{G-11}$$

where

$$\tilde{\vec{V}}_n = \left(\tilde{\vec{V}} \cdot \vec{n}\right)\vec{n} \tag{G-12}$$

($\tilde{\vec{V}} \cdot \vec{n}$ is the normalized contravariant velocity and $\vec{n}$ is the unit surface normal.) The x component of $\tilde{\vec{\tau}}_l$ is $\tilde{\vec{\tau}}_l \cdot \vec{i}$:

$$(\tilde{\vec{\tau}}_l)_x = \tilde{\mu}\frac{\partial\left(\tilde{\vec{V}} \cdot \vec{i} - \tilde{\vec{V}}_n \cdot \vec{i}\right)}{\partial \tilde{n}} = \tilde{\mu}\frac{\partial\left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{n}\right)\hat{\xi}_x\right]}{\partial \tilde{n}} \tag{G-13}$$

Similarly,

$$(\tilde{\vec{\tau}}_l)_y = \tilde{\mu}\frac{\partial\left[\tilde{v} - \left(\tilde{\vec{V}} \cdot \vec{n}\right)\hat{\xi}_y\right]}{\partial \tilde{n}} \tag{G-14}$$

$$(\tilde{\vec{\tau}}_l)_z = \tilde{\mu}\frac{\partial\left[\tilde{w} - \left(\tilde{\vec{V}} \cdot \vec{n}\right)\hat{\xi}_z\right]}{\partial \tilde{n}} \tag{G-15}$$

Consider the nondimensionalization of the x component of $\tilde{\vec{\tau}}_l$:

$$(\tilde{\vec{\tau}}_l)_x = \frac{(\tilde{\vec{\tau}}_l)_x}{\tilde{\rho}_\infty \tilde{a}_\infty^2} = \frac{1}{\tilde{\rho}_\infty \tilde{a}_\infty^2} \cdot \tilde{\mu} \frac{\partial \left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{\hat{n}} \right) \hat{\xi}_x \right]}{\partial \tilde{n}} \tag{G-16}$$

(See Equation (G-8).) Then

$$(\tilde{\vec{\tau}}_l)_x = \frac{\tilde{\mu}_\infty}{\tilde{\rho}_\infty |\tilde{\mathbf{V}}|_\infty \tilde{L}_R} \frac{|\tilde{\mathbf{V}}|_\infty}{\tilde{a}_\infty} \frac{\tilde{\mu}}{\tilde{\mu}_\infty} \frac{\partial \left\{ \left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{\hat{n}} \right) \hat{\xi}_x \right] / \tilde{a}_\infty \right\}}{\partial (\tilde{n} / \tilde{L}_R)} \tag{G-17}$$

Therefore,

$$(\tilde{\vec{\tau}}_l)_x = \frac{M_\infty}{Re_{\tilde{L}_R}} \mu \frac{\partial [u - (\vec{V} \cdot \vec{\hat{n}}) \hat{\xi}_x]}{\partial n} \tag{G-18}$$

with similar expressions for $(\tilde{\vec{\tau}}_l)_y$ and $(\tilde{\vec{\tau}}_l)_z$. The derivative is evaluated using the cell-center and wall values of a cell volume like that shown in Figure G-2.

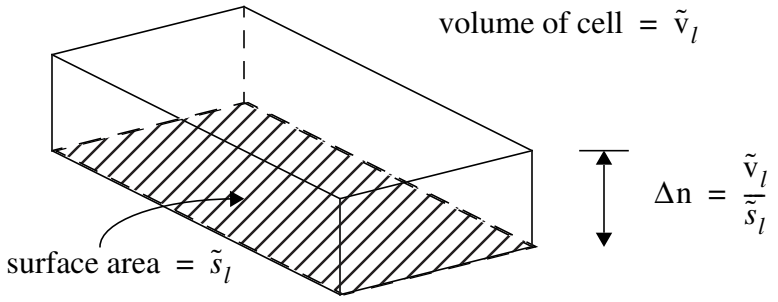


Figure G-2. Cell volume example.

That is, for the x component,

$$\frac{\partial \left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{\hat{n}} \right) \hat{\xi}_x \right]}{\partial n} = \frac{\left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{\hat{n}} \right) \hat{\xi}_x \right]_c - 0}{\frac{1}{2} \Delta n} \tag{G-19}$$

where the subscript c denotes the cell-center value and $\left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{\hat{n}} \right) \hat{\xi}_x \right] \equiv 0$ on a solid wall with the no-slip assumption. So

$$(\vec{\tau}_l)_x = \frac{M_\infty}{Re_{\tilde{L}_R}}\mu\frac{\left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{n}\right)\hat{\xi}_x\right]_c}{\frac{1}{2}\Delta n} = \frac{2M_\infty}{Re_{\tilde{L}_R}}\mu\left(\frac{\tilde{s}_l}{\tilde{v}_l}\right)\left[\tilde{u} - \left(\tilde{\vec{V}} \cdot \vec{n}\right)\hat{\xi}_x\right]_c \quad \text{(G-20)}$$

Thus, for the x component of the viscous force,

$$\tilde{s}_{ref}(\vec{F}_l^v)_x = \frac{2}{M_\infty^2}(\vec{\tau}_l)_x\tilde{s}_l = \frac{4}{M_\infty Re_{\tilde{L}_R}}\mu[u - (\vec{V} \cdot \vec{n})\hat{\xi}_x]_c\frac{\tilde{s}_l^2}{\tilde{v}_l} \quad \text{(G-21)}$$

Similarly,

$$\tilde{s}_{ref}(\vec{F}_l^v)_y = \frac{4}{M_\infty Re_{\tilde{L}_R}}\mu[v - (\vec{V} \cdot \vec{n})\hat{\xi}_y]_c\frac{\tilde{s}_l^2}{\tilde{v}_l} \quad \text{(G-22)}$$

$$\tilde{s}_{ref}(\vec{F}_l^v)_z = \frac{4}{M_\infty Re_{\tilde{L}_R}}\mu[w - (\vec{V} \cdot \vec{n})\hat{\xi}_z]_c\frac{\tilde{s}_l^2}{\tilde{v}_l} \quad \text{(G-23)}$$

G.2 Moments

The moments due to the forces acting on an element are determined as follows. Let

$$\begin{aligned}(\vec{F}_l)_x &= (\vec{F}_l^p + \vec{F}_l^v)_x \\ (\vec{F}_l)_y &= (\vec{F}_l^p + \vec{F}_l^v)_y \\ (\vec{F}_l)_z &= (\vec{F}_l^p + \vec{F}_l^v)_z\end{aligned} \quad \text{(G-24)}$$

Figure G-2 illustrates the directions of the moments. All moments are positive if counter-clockwise when viewed from the positive axis. The conventions (assuming x points downstream and z points up) are

M_x: rolling moment; positive for counter-clockwise roll when viewed from downstream.

M_y: pitching moment; positive for pitch up.

M_z: yawing moment; positive for counter-clockwise yaw when viewed from above. Therefore,

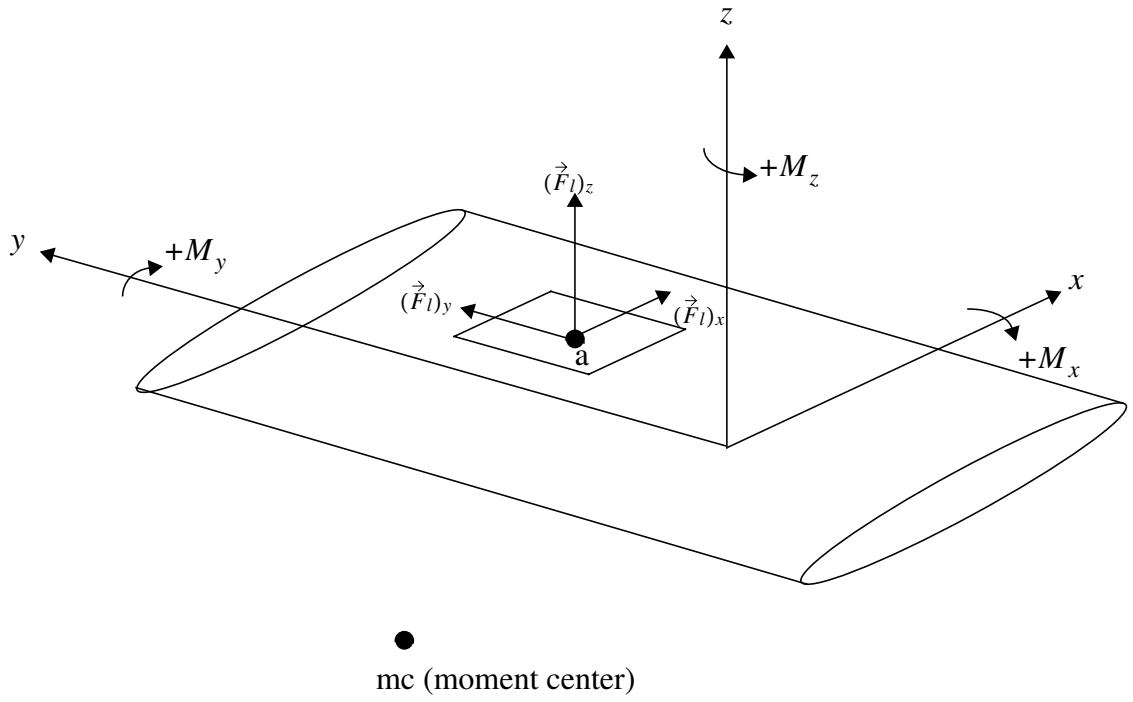


Figure G-3. Moment directions.

$$
\begin{aligned}
(\vec{M}_l)_x &= [+(\vec{F}_l)_x(\tilde{y}_{\mathrm{a}} - \tilde{y}_{\mathrm{mc}}) - (\vec{F}_l)_y(\tilde{z}_{\mathrm{a}} - \tilde{z}_{\mathrm{mc}})]/\tilde{b}_{ref} \\
(\vec{M}_l)_y &= [-(\vec{F}_l)_z(\tilde{x}_{\mathrm{a}} - \tilde{x}_{\mathrm{mc}}) + (\vec{F}_l)_x(\tilde{z}_{\mathrm{a}} - \tilde{z}_{\mathrm{mc}})]/\tilde{c}_{ref} \\
(\vec{M}_l)_z &= [+(\vec{F}_l)_y(\tilde{x}_{\mathrm{a}} - \tilde{x}_{\mathrm{mc}}) - (\vec{F}_l)_x(\tilde{y}_{\mathrm{a}} - \tilde{y}_{\mathrm{mc}})]/\tilde{b}_{ref}
\end{aligned}
\qquad \textbf{(G-25)}
$$

Note that the reference length used to nondimensionalize $(\vec{M}_l)_y$ is $\tilde{c}_{ref}$, while $(\vec{M}_l)_z$ is made dimensionless with $\tilde{b}_{ref}$. This is the default for CFL3D-type grids or PLOT3D-type grids with **ialph** = 0 (see "LT3 - Flow Conditions" on page 19). However, if a PLOT3D-type grid is used with **ialph** = 1, then $(\vec{M}_l)_y$ is made nondimensional with $\tilde{b}_{ref}$ and $(\vec{M}_l)_z$ is made dimensionless with $\tilde{c}_{ref}$. $(\vec{M}_l)_x$ is always made dimensionless with $\tilde{b}_{ref}$. By switching the reference lengths based on **ialph**, the moment coefficient that is normally associated with the pitching moment is always based on $\tilde{c}_{ref}$, while the moment coefficient that is normally associated with the yawing moment is always based on $\tilde{b}_{ref}$. Because $(\vec{M}_l)_x$ always uses $\tilde{b}_{ref}$, the moment coefficient associated with the rolling moment is based on $\tilde{b}_{ref}$.

APPENDIX H

Turbulence Model Equations

CFL3D has several turbulence model capabilities. Appendix H provides the derivation for the advanced turbulence model equations. Be aware that while some variables in this appendix are consistent with the rest of the manual (and are listed in the Nomenclature section), many are introduced, defined, and used only in these sections and may or may not appear in the Nomenclature listing. Also, for simplicity's sake in this Appendix, $Re = Re_{\tilde{L}_R}$.

H.1 Equations of Motion

Following Wilcox[46], Favre[19] averaging can be used with the Navier-Stokes equations to account for turbulent fluctuations. The resulting equations of motion can be written using the summation convention as follows. The full Navier-Stokes equations are shown here, but in CFL3D, they are solved as the thin-layer approximation in pre-selected coordinate direction(s). The ~ indicates a dimensional quantity.

$$\frac{\partial \tilde{\rho}}{\partial \tilde{t}} + \frac{\partial}{\partial \tilde{x}_j}(\tilde{\rho}\tilde{u}_j) = 0 \tag{H-1}$$

$$\frac{\partial}{\partial \tilde{t}}(\tilde{\rho}\tilde{u}_i) + \frac{\partial}{\partial \tilde{x}_j}(\tilde{\rho}\tilde{u}_j\tilde{u}_i) = -\frac{\partial \tilde{p}}{\partial \tilde{x}_i} + \frac{\partial \tilde{\tau}_{ji}}{\partial \tilde{x}_j} \tag{H-2}$$

$$\frac{\partial}{\partial \tilde{t}}(\tilde{\rho}\tilde{E}) + \frac{\partial}{\partial \tilde{x}_j}(\tilde{\rho}\tilde{u}_j\tilde{H}) = \frac{\partial}{\partial \tilde{x}_j}[\tilde{u}_i\tilde{\tau}_{ij} - \tilde{q}_j + \tilde{\psi}_j] \tag{H-3}$$

where

$$\tilde{p} = (\gamma - 1)\left[\tilde{\rho}\tilde{E} - \frac{1}{2}\tilde{\rho}(\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2)\right] \tag{H-4}$$

$$\tilde{E} = \tilde{e} + \frac{1}{2}(\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2) \tag{H-5}$$

$$\tilde{H} = \tilde{E} + \frac{\tilde{p}}{\tilde{\rho}} \tag{H-6}$$

$$\tilde{q}_j = -\frac{1}{\gamma - 1}\left(\frac{\tilde{\mu}}{\mathrm{Pr}} + \frac{\tilde{\mu}_\mathrm{T}}{\mathrm{Pr}_\mathrm{T}}\right)\frac{\partial \tilde{a}^2}{\partial \tilde{x}_j} \tag{H-7}$$

$$\tilde{a}^2 = \frac{\gamma \tilde{p}}{\tilde{\rho}} \tag{H-8}$$

Note that the kinetic energy of the fluctuating turbulent field $\tilde{k}$ is ignored in the definition of $\tilde{E}$ in CFL3D (it is assumed that $\tilde{k} << \tilde{e} + \frac{1}{2}(\tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2)$). Define

$$\begin{aligned} \tilde{S}_{ij} &= \frac{1}{2}\left(\frac{\partial \tilde{u}_i}{\partial \tilde{x}_j} + \frac{\partial \tilde{u}_j}{\partial \tilde{x}_i}\right) \\ \tilde{W}_{ij} &= \frac{1}{2}\left(\frac{\partial \tilde{u}_i}{\partial \tilde{x}_j} - \frac{\partial \tilde{u}_j}{\partial \tilde{x}_i}\right) \end{aligned} \tag{H-9}$$

Also, the magnitude of vorticity is

$$\tilde{\Omega} = \sqrt{2\tilde{W}_{ij}\tilde{W}_{ij}} \tag{H-10}$$

The shear stress terms $\tilde{\tau}_{ij}$ is composed of a laminar and a turbulent component as

$$\tilde{\tau}_{ij} = \tilde{\tau}^\mathrm{L}_{ij} + \tilde{\tau}^\mathrm{T}_{ij} \tag{H-11}$$

where

$$\tilde{\tau}^\mathrm{L}_{ij} = 2\tilde{\mu}\left(\tilde{S}_{ij} - \frac{1}{3}\frac{\partial \tilde{u}_\mathrm{k}}{\partial \tilde{x}_\mathrm{k}}\delta_{ij}\right) \tag{H-12}$$

For all *eddy-viscosity* models in CFL3D, the following approximations are made:

$$\tilde{\tau}^\mathrm{T}_{ij} = 2\tilde{\mu}_\mathrm{T}\left(\tilde{S}_{ij} - \frac{1}{3}\frac{\partial \tilde{u}_\mathrm{k}}{\partial \tilde{x}_\mathrm{k}}\delta_{ij}\right) \tag{H-13}$$

$$\tilde{\psi}_j = 0 \tag{H-14}$$

Currently, for the *nonlinear* models in CFL3D,

$$\tilde{\tau}_{ij}^{T} = 2\tilde{\mu}_T\left(\tilde{S}_{ij} - \frac{1}{3}\frac{\partial \tilde{u}_k}{\partial \tilde{x}_k}\delta_{ij}\right) - \frac{2}{3}\tilde{\rho}\tilde{k}\delta_{ij} + \tag{H-15}$$
$$2\tilde{\mu}_T K_1 \frac{\tilde{k}}{\tilde{\varepsilon}}(\tilde{S}_{ik}\tilde{W}_{kj} - \tilde{W}_{ik}\tilde{S}_{kj}) -$$
$$2\tilde{\mu}_T K_2 \frac{\tilde{k}}{\tilde{\varepsilon}}\left(\tilde{S}_{ik}\tilde{S}_{kj} - \frac{1}{3}\tilde{S}_{kl}\tilde{S}_{lk}\delta_{ij}\right)$$

$$\tilde{\psi}_j = \left(\tilde{\mu} + \frac{\tilde{\mu}_T}{\sigma_k}\right)\frac{\partial \tilde{k}}{\partial \tilde{x}_j} \tag{H-16}$$

where $\tilde{k}/\tilde{\varepsilon}$ is replaced by $1/\tilde{\omega}$ when the $k-\omega$ equations rather than the $k-\varepsilon$ equations are employed.

The Navier-Stokes equations are nondimensionalized and written in generalized coordinates, as described Appendix F. For *eddy-viscosity* models, the end result is that the turbulent Navier-Stokes equations are identical to the laminar equations with the exception that

$$\tilde{\mu} \text{ is replaced by } \tilde{\mu} + \tilde{\mu}_T$$

and

$$\frac{\tilde{\mu}}{Pr} \text{ is replaced by } \frac{\tilde{\mu}}{Pr} + \frac{\tilde{\mu}_T}{Pr_T}$$

where $\tilde{\mu}_T$ is the eddy viscosity value obtained by whatever turbulence model is used, and $Pr = 0.72$, $Pr_T = 0.9$. For the *nonlinear* models, both the above substitutions must be made *and* the following additions as well. The term

$$-\frac{2}{3}\tilde{\rho}\tilde{k}\delta_{ij} + 2\tilde{\mu}_T K_1 \frac{\tilde{k}}{\tilde{\varepsilon}}(\tilde{S}_{ik}\tilde{W}_{kj} - \tilde{W}_{ik}\tilde{S}_{kj}) - 2\tilde{\mu}_T K_2 \frac{\tilde{k}}{\tilde{\varepsilon}}\left(\tilde{S}_{ik}\tilde{S}_{kj} - \frac{1}{3}\tilde{S}_{kl}\tilde{S}_{lk}\delta_{ij}\right)$$

is added to $\tilde{\tau}_{ij}$ in the momentum and energy equations and

$$\frac{\partial}{\partial \tilde{x}_j}\left[\left(\tilde{\mu} + \frac{\tilde{\mu}_T}{\sigma_k}\right)\frac{\partial \tilde{k}}{\partial \tilde{x}_j}\right]$$

is added to the energy equation. These additions are made in subroutines `gfluxv`, `hfluxv`, and `ffluxv`.

H.2 Nondimensionalizations

The turbulence equations are nondimensionalized by the same reference quantities as the Navier-Stokes equations: $\tilde{a}_\infty$, $\tilde{\rho}_\infty$, $\tilde{\mu}_\infty$, and $\tilde{L}_R = \tilde{L}/L_{ref}$. The nondimensionalized variables used with the turbulence models are:

$$
\begin{aligned}
\mathrm{k} &= \frac{\tilde{\mathrm{k}}}{\tilde{a}_\infty^2} & \omega &= \frac{\tilde{\mu}_\infty \tilde{\omega}}{\tilde{\rho}_\infty \tilde{a}_\infty^2} & \mathrm{P_k} &= \frac{\tilde{\mathrm{P}}_\mathrm{k} \tilde{L}_R^2}{\tilde{\mu}_\infty \tilde{a}_\infty^2} \\
\varepsilon &= \frac{\tilde{\mu}_\infty \tilde{\varepsilon}}{\tilde{\rho}_\infty \tilde{a}_\infty^4} & \rho &= \frac{\tilde{\rho}}{\tilde{\rho}_\infty} & \mathrm{P}_\omega &= \frac{\tilde{\mathrm{P}}_\omega \tilde{L}_R^2}{\tilde{\rho}_\infty \tilde{a}_\infty^2} \\
u &= \frac{\tilde{u}}{\tilde{a}_\infty} & p &= \frac{\tilde{p}}{\tilde{\rho}_\infty \tilde{a}_\infty^2} & & \\
x &= \frac{\tilde{x}}{\tilde{L}_R} & \mu &= \frac{\tilde{\mu}}{\tilde{\mu}_\infty} & \mathrm{P}_\varepsilon &= \frac{\tilde{\mathrm{P}}_\varepsilon \tilde{L}_R^2}{\tilde{\rho}_\infty \tilde{a}_\infty^4} \\
\tau_{ij} &= \frac{\tilde{\tau}_{ij} \tilde{L}_R}{\tilde{\mu}_\infty \tilde{a}_\infty} & t &= \frac{\tilde{t} \tilde{a}_\infty}{\tilde{L}_R} & \Omega &= \frac{\tilde{\Omega} \tilde{L}_R}{\tilde{a}_\infty}
\end{aligned}
\qquad \text{(H-17)}
$$

H.3 Zero-equation Models

H.3.1 Baldwin-Lomax Model

(**ivisc** = 2)

The Baldwin-Lomax[10] model is not a field-equation model; it is an algebraic model. Because it is the original model employed in CFL3D, its implementation differs in many respects from the more advanced models. First of all, it does not use the *minimum distance function* as its length scale, like the other models. Instead, it uses a *directed distance* from the $i = 1$, $j = 1$, $k = 1$, $i =$ **idim**, $j =$ **jdim**, or $k =$ **kdim** point along a constant index line. For example, if the "body" is at $k = 1$, then the directed distance to a given point in the field is the directed (normal) distance from k_{given} to $k = 1$, keeping i and j fixed. See Figure H-1(a). The directed distance is

$$d = \vec{r}_k \cdot \hat{\mathrm{n}} - \vec{r}_0 \cdot \hat{\mathrm{n}} \qquad \text{(H-18)}$$

where $\vec{r}_k$ is the vector from the origin to k_{given} and $\vec{r}_0$ is the vector from the origin to the $k = 1$ point. Note that if grid lines curve significantly d can become negative as in Figure H-1(b). If the grid lines do this, CFL3D currently does not allow the distance to go

negative. Instead, it computes d out to its maximum, then sets all distances thereafter to that maximum value. However, if the grid behaves like the grid in Figure H-1(b), the Baldwin-Lomax model, which is inherently dependent on the grid structure, is probably not a good choice anyway. The use of any of the other field-equation models is recommended instead.

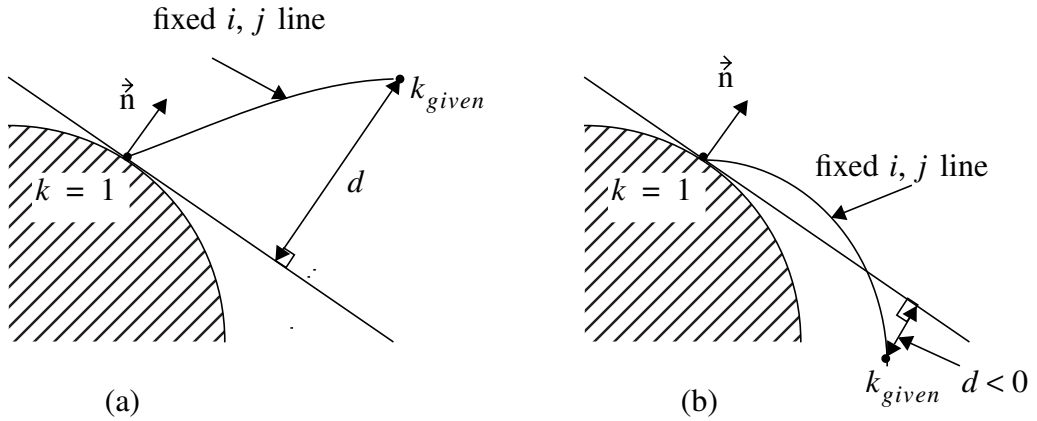


Figure H-1. Directed distance schematic.

In the current implementation in CFL3D, if the grid contains multiple blocks, Baldwin-Lomax should only be implemented on blocks that contain bodies. This restriction applies to the Baldwin-Lomax model only. It does *not* apply to the more general field-equation models.

For the Baldwin-Lomax model,

$$\begin{aligned} \mu_T &= \mu_{T,\,inner} \qquad y \le y_{crossover} \\ \mu_T &= \mu_{T,\,outer} \qquad y > y_{crossover} \end{aligned} \tag{H-19}$$

where $y_{crossover}$ is the location along the constant index line where $\mu_{T,\,inner}$ exceeds $\mu_{T,\,outer}$, marching away from the wall. The inner eddy viscosity is

$$\mu_{T,\,inner} = \rho l^2 \Omega \left(\frac{Re}{M_\infty} \right) \tag{H-20}$$

where Ω is the magnitude of the vorticity and

$$l = y[1 - \exp(-y^+/26)] \tag{H-21}$$

Since it now appears to be fairly widely accepted that the definition of y^+ for the damping term needs to be modified in Baldwin-Lomax to give better answers when there are large temperature gradients near the wall,

$$y^+ = \frac{\sqrt{\tilde{\rho}\tilde{\tau}_w}}{\tilde{\mu}}\tilde{y} \tag{H-22}$$

is used rather than

$$y^+ = \frac{\sqrt{\tilde{\rho}_w\tilde{\tau}_w}}{\tilde{\mu}_w}\tilde{y} \tag{H-23}$$

Nondimensionally, this becomes

$$y^+ = \frac{\sqrt{\rho\tau_w}}{\mu}y\left(\frac{Re}{M_\infty}\right)^{\frac{1}{2}} \tag{H-24}$$

The outer eddy viscosity is

$$\mu_{T,\,outer} = 0.0168(1.6)\rho F_{wake}F_{kleb}\left(\frac{Re}{M}\right) \tag{H-25}$$

where

$$\begin{aligned} F_{wake} &= \min[y_{max}F_{max},\, 1.0y_{max}u_{dif}^2/F_{max}] \\ F(y) &= y\Omega[1-\exp(-y^+/26)] \end{aligned} \tag{H-26}$$

In wakes, $\exp(-y^+/26)$ is set to zero. F_{max} is the maximum value of $F(y)$ that occurs in a profile and y_{max} is the value at which F_{max} occurs. Also,

$$\begin{aligned} F_{kleb} &= \left[1+5.5\left(\frac{0.3y}{y_{max}}\right)^6\right]^{-1} \\ u_{dif} &= (\sqrt{u^2+v^2+w^2})_{max} - (\sqrt{u^2+v^2+w^2})_{min} \end{aligned} \tag{H-27}$$

The second term in u_{dif} is taken to be zero, except in wakes. The region for the search for F_{max} is currently bound by

$$0.2\eta_{max}+1 < \eta < 0.8\eta_{max}+1 \tag{H-28}$$

where η, in this case, is the index direction away from the body.

H.3.2 Baldwin-Lomax with Degani-Schiff Modification

(**ivisc** = 3)

The Degani-Schiff[16] modification to the Baldwin-Lomax model is an algorithmic change which attempts to select the *first* occurrence of F_{max} in a search from the wall outwards. This can be important when there is a vortex somewhere above the body surface. If the code is not forced to select the F_{max} in the boundary layer, it may choose a length scale corresponding to the distance to the vortex, since often F can be larger in the vortex.

The test in CFL3D is very simple-minded and can often fail. However, it is quite difficult to find a test that works for all circumstances; so, for lack of anything better, the following is used. Marching outward away from the body, F_{max} is updated index by index. Then, if

$$F < 0.9F_{max}, \tag{H-29}$$

the code stops searching.

H.4 One- and Two-equation Field-equation Models

All of the one- and two-equation model equations can be written in the general form

$$\frac{\partial}{\partial t}(\mathrm{X}) + u_j\frac{\partial}{\partial x_j}(\mathrm{X}) = \mathrm{S_P} + \mathrm{S_D} + D \tag{H-30}$$

where $\mathrm{S_P}$ is a "production" source term(s), $\mathrm{S_D}$ is a "destruction" source term(s), and D represents diffusion terms of the form $\frac{\partial}{\partial x_j}\left[(\ ) \cdot \frac{\partial \mathrm{X}}{\partial x_j}\right]$. Note that $\mathrm{S_P}$ and $\mathrm{S_D}$ have been grouped together rather loosely. In Spalart's model in CFL3D, for example, part of Spalart's production term is grouped in $\mathrm{S_D}$ for convenience. In Menter's model in CFL3D, the cross-derivative term is grouped in $\mathrm{S_D}$. In CFL3D, the $\mathrm{S_P}$ terms are treated explicitly while the $\mathrm{S_D}$ terms are linearized and treated implicitly. This is Spalart's "third strategy."[35]

All of the field-equation models are solved *uncoupled* from the Navier-Stokes equations. All of the models are solved in essentially the same fashion. Details are given in "Solution Method" on page 299 in the form of an example.

Also, all of the one- and two-equation models are based on incompressible turbulence equations. No compressibility corrections have been added. Hence, for problems where the turbulent Mach number, $M_T = \sqrt{2\tilde{k}/\tilde{a}^2}$, is high, these turbulence models may not be applicable. Note that for most subsonic through low supersonic aerodynamic applications, the incompressible forms of the turbulence models are generally expected to be valid.

All of the field equation models except for Wilcox $k-\omega$ make use of the *minimum distance function*, or the distance to the nearest wall, `smin`. This distance differs from the *directed distance* used by the Baldwin-Lomax model in that it does *not* follow grid lines and can be computed across zone boundaries. Hence, it is more easily applicable to multiple zone applications. The minimum distance function represents the distance to the nearest viscous wall and takes into account grid skewness when computing the nearest wall-point location (in subroutine `findmin_new`). The exception is the Baldwin-Barth model which uses a minimum distance algorithm (subroutine `findmin`) that does not take grid skewness into account. (The reason for this exception is that the Baldwin-Barth model requires other variables not currently implemented in subroutine `findmin_new`.)

H.4.1 Baldwin-Barth Model

(**ivisc** = 4)

The Baldwin-Barth[9] model solves a single field equation for a turbulent Reynolds number term R:

$$\frac{\partial R}{\partial t} + u_j\frac{\partial R}{\partial x_j} = (C_{\varepsilon_2}f_2 - C_{\varepsilon_1})\sqrt{RP} + \frac{M_\infty}{Re}\left(\nu + \frac{\nu_T}{\sigma_\varepsilon}\right)\frac{\partial^2 R}{\partial x_j^2} - \frac{M_\infty}{Re}\frac{1}{\sigma_\varepsilon}\frac{\partial}{\partial x_j}\left(\nu_T\frac{\partial R}{\partial x_j}\right) \quad \text{(H-31)}$$

where

$$\frac{1}{\sigma_\varepsilon} = (C_{\varepsilon_2} - C_{\varepsilon_1})\frac{\sqrt{C_\mu}}{\kappa^2} \quad \text{(H-32)}$$

$$\nu_T = C_\mu R D_1 D_2 \quad \text{(H-33)}$$

$$D_1 = 1 - \exp\left(-\frac{y^+}{A^+}\right) \quad \text{(H-34)}$$

$$D_2 = 1 - \exp\left(-\frac{y^+}{A_2^+}\right) \quad \text{(H-35)}$$

$$f_2 = \frac{C_{\varepsilon_1}}{C_{\varepsilon_2}} + \left(1 - \frac{C_{\varepsilon_1}}{C_{\varepsilon_2}}\right)\left(\frac{1}{\kappa y^+} + D_1 D_2\right)\left\{\sqrt{D_1 D_2} + \frac{y^+}{\sqrt{D_1 D_2}}\left[\frac{1}{A^+}\exp\left(-\frac{y^+}{A^+}\right)D_2 + \frac{1}{A_2^+}\exp\left(-\frac{y^+}{A_2^+}\right)D_1\right]\right\} \quad \text{(H-36)}$$

$$\mu_T = \rho \nu_T \quad \text{(H-37)}$$

$$\kappa = 0.41 \qquad C_{\varepsilon_1} = 1.2 \qquad C_{\varepsilon_2} = 2.0$$
$$C_\mu = 0.09 \qquad A^+ = 26 \qquad A_2^+ = 10 \quad \text{(H-38)}$$

The production term P is given by

$$P = 2\nu_T S_{ij} S_{ij} - \frac{2}{3}\nu_T\left(\frac{\partial u_k}{\partial x_k}\right)^2 \quad \text{(H-39)}$$

but is approximated as

$$P \cong \nu_T \Omega^2 \quad \text{(H-40)}$$

Hence, using the general form in Equation (H-30) with $X = R$,

$$S_P \cong (C_{\varepsilon_2} f_2 - C_{\varepsilon_1})\sqrt{C_\mu D_1 D_2}\,\Omega R \quad \text{(H-41)}$$

$$D = \frac{M_\infty}{Re}\left(\nu + \frac{\nu_T}{\sigma_\varepsilon}\right)\frac{\partial^2 R}{\partial x_j^2} - \frac{M_\infty}{Re}\frac{1}{\sigma_\varepsilon}\frac{\partial}{\partial x_j}\left(\nu_T \frac{\partial R}{\partial x_j}\right) \quad \text{(H-42)}$$

Since the Baldwin-Barth model requires y^+ (rather than just the minimum distance to the wall), additional information about the nearest wall point needs to be stored as well as the minimum distance function d. For simplicity, it is currently assumed that in the regions where y^+ has an effect, these additional wall values are in the same grid zone as the point in question.

H.4.2 Spalart-Allmaras Model

(**ivisc** = 5)

The Spalart-Allmaras[34] model solves a single field equation for a variable $\hat{\nu}$ related to the eddy viscosity through

$$\mu_T = \rho \hat{\nu} f_{v_1} \tag{H-43}$$

where

$$f_{v_1} = \frac{\chi^3}{\chi^3 + C_{v_1}^3} \tag{H-44}$$

$$\chi \equiv \frac{\hat{\nu}}{\nu} \tag{H-45}$$

The equation is

$$\begin{aligned} \frac{\partial \hat{\nu}}{\partial t} + u_j \frac{\partial \hat{\nu}}{\partial x_j} &= C_{b_1}[1 - f_{t_2}]\Omega \hat{\nu} \\ &+ \frac{M_\infty}{Re}\left\{ C_{b_1}[(1 - f_{t_2}) f_{v_2} + f_{t_2}]\frac{1}{\kappa^2} - C_{w_1} f_w \right\}\left(\frac{\hat{\nu}}{d}\right)^2 \\ &- \frac{M_\infty}{Re}\frac{C_{b_2}}{\sigma}\hat{\nu}\frac{\partial^2 \hat{\nu}}{\partial x_j^2} \\ &+ \frac{M_\infty}{Re}\frac{1}{\sigma}\frac{\partial}{\partial x_j}\left[(\nu + (1 + C_{b_2})\hat{\nu})\frac{\partial \hat{\nu}}{\partial x_j}\right] \end{aligned} \tag{H-46}$$

(Note that Spalart's trip function is not used.) Also,

$$f_{t_2} = C_{t_3} \exp(-C_{t_4}\chi^2) \tag{H-47}$$

$$d = \text{distance to the closest wall = minimum distance function} \tag{H-48}$$

$$f_w = g\left[\frac{1 + C_{w_3}^6}{g^6 + C_{w_3}^6}\right]^{\frac{1}{6}} = \left[\frac{g^{-6} + C_{w_3}^{-6}}{1 + C_{w_3}^{-6}}\right]^{-\frac{1}{6}} \tag{H-49}$$

$$g = r + C_{w_2}(r^6 - r) \tag{H-50}$$

$$r = \frac{\hat{\nu}}{\hat{S}\left(\frac{Re}{M_\infty}\right)\kappa^2 d^2} \tag{H-51}$$

where

$$\hat{S} = \Omega + \frac{\hat{\nu} f_{v_2}}{\left(\frac{Re}{M_\infty}\right)\kappa^2 d^2} \tag{H-52}$$

$$f_{v_3} = \text{no longer used} \tag{H-53}$$

$$f_{v_2} = 1 - \frac{\chi}{1 + \chi f_{v_1}} \tag{H-54}$$

CFL3D currently uses Spalart's Version Ia[34]. The fv3 term was employed as a smooth fix to prevent $\hat{S}$ from going negative prior to 12/97, but was removed after an error was discovered ($\hat{S}$ and fv2 were also different). The constants are

$$C_{b_1} = 0.1355 \qquad \sigma = \frac{2}{3} \qquad C_{b_2} = 0.622 \qquad \kappa = 0.41 \qquad C_{w_2} = 0.3$$
$$C_{w_3} = 2.0 \qquad C_{v_1} = 7.1 \qquad C_{t_3} = 1.2 \qquad C_{t_4} = 0.5 \tag{H-55}$$
$$C_{w_1} = \frac{C_{b_1}}{\kappa^2} + \frac{(1 + C_{b_2})}{\sigma} \quad \text{(note typo in ref. 34)}$$

For the general form in Equation (H-30):

$$X = \hat{\nu} \tag{H-56}$$

$$S_P = C_{b_1}[1 - f_{t_2}]\Omega\hat{\nu} \tag{H-57}$$

$$S_D = \frac{M_\infty}{Re}\left\{C_{b_1}[(1 - f_{t_2})f_{v_2} + f_{t_2}]\frac{1}{\kappa^2} - C_{w_1} f_w\right\}\left(\frac{\hat{\nu}}{d}\right)^2 \tag{H-58}$$

$$D = -\frac{M_\infty}{Re}\frac{C_{b_2}}{\sigma}\hat{\nu}\frac{\partial^2\hat{\nu}}{\partial x_j^2} + \frac{M_\infty}{Re}\frac{1}{\sigma}\frac{\partial}{\partial x_j}\left[(\nu + (1 + C_{b_2})\hat{\nu})\frac{\partial\hat{\nu}}{\partial x_j}\right] \tag{H-59}$$

Note that in CFL3D, the S_P and S_D terms are grouped differently than Spalart's. Part of Spalart's production term is grouped into S_D because it has the common factor M_∞/Re, like the other destruction terms.

H.4.3 Wilcox k-Omega Model

(**ivisc** = 6)

For Wilcox's[46] model,

$$\frac{\partial k}{\partial t} + u_j \frac{\partial k}{\partial x_j} = \frac{1}{\rho} P_k \left(\frac{M_\infty}{Re}\right) - \beta' k\omega \left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-60}$$

$$\frac{\partial \omega}{\partial t} + u_j \frac{\partial \omega}{\partial x_j} = \frac{1}{\rho} P_\omega \left(\frac{M_\infty}{Re}\right) - \beta \omega^2 \left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\omega}\right)\frac{\partial \omega}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-61}$$

In this model, C_μ is incorporated into the definition of ω, so

$$\mu_T = \frac{\rho k}{\omega} \tag{H-62}$$

The production terms are approximated by

$$P_k = \mu_T \Omega^2 \tag{H-63}$$

$$P_\omega = \gamma \rho \Omega^2 \tag{H-64}$$

The constants are

$$\gamma = \frac{\beta}{C_\mu} - \frac{\kappa^2}{\sigma_\omega \sqrt{C_\mu}} \qquad \beta' = C_\mu = 0.09$$
$$\sigma_k = 1/0.5 \qquad \beta = 0.075 \tag{H-65}$$
$$\sigma_\omega = 1/0.5 \qquad \kappa = 0.41$$

The minimum distance function d is not required by this model. However, at the present time it is still computed and stored by CFL3D. For the general form in Equation (H-30),

$$X_k = k \tag{H-66}$$

$$S_{P,k} = \frac{1}{\rho}\mu_T \Omega^2 \left(\frac{M_\infty}{Re}\right) \tag{H-67}$$

$$S_{D,k} = -\beta' k\omega \left(\frac{Re}{M_\infty}\right) \tag{H-68}$$

$$D_k = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-69}$$

and

$$X_\omega = \omega \tag{H-70}$$

$$S_{P,\omega} = \gamma\Omega^2\left(\frac{M_\infty}{Re}\right) \tag{H-71}$$

$$S_{D,\omega} = -\beta\omega^2\left(\frac{Re}{M_\infty}\right) \tag{H-72}$$

$$D_\omega = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\omega}\right)\frac{\partial\omega}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-73}$$

H.4.4 Menter's k-Omega SST Model

(**ivisc** = 7)

Menter's k-ω model is assessed in reference 27. Note that there are two corrections to this reference. Equation (1) should be

$$\begin{aligned}\sigma\mu_t &= \mu + \sigma\mu_t \\ \sigma^*\mu_t &= \mu + \sigma^*\mu_t\end{aligned} \tag{H-74}$$

and Equation (17) should be

$$\Gamma = \max(2\Gamma_3, \Gamma_1) \tag{H-75}$$

The two equations for this model are

$$\frac{\partial k}{\partial t} + u_j\frac{\partial k}{\partial x_j} = \frac{1}{\rho}P_k\left(\frac{M_\infty}{Re}\right) - \beta' k\omega\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-76}$$

$$\begin{aligned}\frac{\partial\omega}{\partial t} + u_j\frac{\partial\omega}{\partial x_j} = {} & \frac{1}{\rho}P_\omega\left(\frac{M_\infty}{Re}\right) - \beta\omega^2\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\omega}\right)\frac{\partial\omega}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \\ & + 2(1 - F_1)\sigma_{\omega_2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j}\left(\frac{M_\infty}{Re}\right)\end{aligned} \tag{H-77}$$

In this model, C_μ is incorporated into the definition of ω. The eddy viscosity is given by

$$\mu_T = \min\left[\frac{\rho k}{\omega}, \frac{a_1\rho k}{\Omega F_2}\left(\frac{Re}{M}\right)\right] \tag{H-78}$$

The production terms are approximated by

$$P_k = \mu_T \Omega^2 \tag{H-79}$$

$$P_\omega = \gamma \rho \Omega^2 \tag{H-80}$$

The constants are calculated from $\phi = F_1\phi_1 + (1 - F_1)\phi_2$, where the ϕ's are the constants:

$$\begin{aligned}
\gamma_1 &= \frac{\beta_1}{C_\mu} - \frac{\kappa^2}{\sigma_{\omega_1}\sqrt{C_\mu}} \qquad & \gamma_2 &= \frac{\beta_2}{C_\mu} - \frac{\kappa^2}{\sigma_{\omega_2}\sqrt{C_\mu}} \\
\sigma_{k_1} &= 1/0.85 & \sigma_{k_2} &= 1.0 \\
\sigma_{\omega_1} &= 1/0.5 & \sigma_{\omega_2} &= 1/0.856 \\
\beta_1 &= 0.075 & \beta_2 &= 0.0828 \\
\kappa &= 0.41 & a_1 &= 0.31 \\
\beta' &= C_\mu = 0.09
\end{aligned} \tag{H-81}$$

$$\begin{aligned}
F_1 &= \tanh(\Gamma^4) \\
\Gamma &= \min[\max(\Gamma_1, \Gamma_3), \Gamma_2] \\
\Gamma_1 &= \frac{500\nu}{d^2\Omega}\left(\frac{M_\infty}{Re}\right)^2 \qquad \Gamma_2 = \frac{4\rho\sigma_2 k}{d^2(CD_{k-\omega})} \qquad \Gamma_3 = \frac{\sqrt{k}}{C_\mu \omega d}\left(\frac{M_\infty}{Re}\right) \\
CD_{k-\omega} &= \max\left(\rho\frac{2\sigma_2}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}, 1\times10^{-20}\right) \\
F_2 &= \tanh\Pi \\
\Pi &= \max(2\Gamma_3, \Gamma_1)
\end{aligned} \tag{H-82}$$

For the general form in Equation (H-30),

$$X_k = k \tag{H-83}$$

$$S_{P,k} = \frac{1}{\rho}\mu_T \Omega^2 \left(\frac{M_\infty}{Re}\right) \tag{H-84}$$

$$S_{D,k} = -\beta' k\omega \left(\frac{Re}{M_\infty}\right) \tag{H-85}$$

$$D_k = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-86}$$

and

$$X_\omega = \omega \tag{H-87}$$

$$S_{P,\omega} = \gamma\Omega^2\left(\frac{M_\infty}{Re}\right) \tag{H-88}$$

$$S_{D,\omega} = -\beta\omega^2\left(\frac{Re}{M_\infty}\right) + 2(1-F_1)\sigma_{\omega_2}\frac{1}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \tag{H-89}$$

$$D_\omega = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\omega}\right)\frac{\partial \omega}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-90}$$

H.4.5 Abid k-Epsilon Model

(**ivisc** = 10)

For the Abid[2] k-ε model, the equations are

$$\frac{\partial k}{\partial t} + u_j\frac{\partial k}{\partial x_j} = \frac{1}{\rho}P_k\left(\frac{M_\infty}{Re}\right) - \varepsilon\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-91}$$

$$\frac{\partial \varepsilon}{\partial t} + u_j\frac{\partial \varepsilon}{\partial x_j} = \frac{1}{\rho}P_\varepsilon\left(\frac{M_\infty}{Re}\right) - C_{\varepsilon_2}\frac{\varepsilon^2}{k}f_2\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-92}$$

$$\mu_T = C_\mu f_\mu \frac{\rho k^2}{\varepsilon} \tag{H-93}$$

The production terms are approximated by

$$\begin{aligned} P_k &= \mu_T\Omega^2 \\ P_\varepsilon &= C_{\varepsilon_1}\frac{\varepsilon}{k}\mu_T\Omega^2 \end{aligned} \tag{H-94}$$

The constants are

$$\begin{aligned} C_{\varepsilon_1} &= 1.45 \qquad \sigma_k = 1.0 \\ C_{\varepsilon_2} &= 1.83 \qquad \sigma_\varepsilon = 1.4 \\ C_\mu &= 0.09 \end{aligned} \tag{H-95}$$

The damping functions are given by

$$f_\mu = [1 + 4(Re_\tau^{-0.75})]\tanh(0.008Re_k)$$
$$f_2 = \left[1 - \exp\left(-\frac{Re_k}{12}\right)\right] \quad \text{(H-96)}$$

where

$$Re_\tau = \frac{\rho k^2}{\mu\varepsilon}$$
$$Re_k = \frac{\rho\sqrt{k}d}{\mu}\left(\frac{Re}{M}\right) \quad \text{(H-97)}$$

where d is the distance to the nearest wall. For the general form in Equation (H-30),

$$X_k = k \quad \text{(H-98)}$$

$$S_{P,k} = \frac{1}{\rho}\mu_T\Omega^2\left(\frac{M_\infty}{Re}\right) \quad \text{(H-99)}$$

$$S_{D,k} = -\varepsilon\left(\frac{Re}{M_\infty}\right) \quad \text{(H-100)}$$

$$D_k = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \quad \text{(H-101)}$$

and

$$X_\varepsilon = \varepsilon \quad \text{(H-102)}$$

$$S_{P,\varepsilon} = \frac{1}{\rho}C_{\varepsilon_1}\frac{\varepsilon}{k}\mu_T\Omega^2\left(\frac{M_\infty}{Re}\right) \quad \text{(H-103)}$$

$$S_{D,\varepsilon} = -C_{\varepsilon_2}\frac{\varepsilon^2}{k}f_2\left(\frac{Re}{M_\infty}\right) \quad \text{(H-104)}$$

$$D_\varepsilon = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \quad \text{(H-105)}$$

H.4.6 EASM Gatski-Speziale k-Omega Model

(**ivisc** = 8 and 12)

EASM stands for Explicit Algebraic Stress Model.[3] **Ivisc** = 8 is the "linear" k – ω version, treated as an eddy viscosity model, whereas **ivisc** = 12 is the fully nonlinear k – ω version. The two versions obtain μ_T with identical methods. However, the nonlinear model includes nonlinear terms added to the Navier-Stokes equations, whereas the linear model does not. (Its effect is felt only through the μ_T term.)

The k – ω equations are of the same form as in the Wilcox k – ω model (Equation (H-60) and Equation (H-61)). However, C_μ is not incorporated into the definition of ω (it is now a variable coefficient), so,

$$\mu_T = C_\mu \rho \frac{k}{\omega} \tag{H-106}$$

The constants are different as well:

$$\gamma = \beta - \frac{\kappa^2}{\sigma_\omega \sqrt{C_\mu^*}} \qquad \begin{aligned} \sigma_k &= 1.4 \\ \sigma_\omega &= 2.2 \\ \beta' &= 1.0 \\ \beta &= 0.83 \\ \kappa &= 0.41 \end{aligned} \tag{H-107}$$

The production terms are

$$\begin{aligned} P_k &= \tau_{ij}^T \frac{\partial u_i}{\partial x_j} \\ P_\omega &= \gamma \frac{\omega}{k} \tau_{ij}^T \frac{\partial u_i}{\partial x_j} \end{aligned} \tag{H-108}$$

where τ_{ij}^T is given (dimensionally) in Equation (H-15). The variable C_μ and the coefficients K_1 and K_2 in Equation (H-15) are determined as follows:

$$C_\mu = \frac{3(1+\eta^2) + 0.2(\eta^6 + \zeta^6)}{3 + \eta^2 + 6\zeta^2\eta^2 + 6\zeta^2 + \eta^6 + \zeta^6} \alpha_1 \tag{H-109}$$

$$\eta = \frac{\alpha_2}{\omega} (S_{ij} S_{ij})^{\frac{1}{2}} \left(\frac{M_\infty}{Re} \right) \tag{H-110}$$

$$\zeta = \frac{\alpha_3}{\omega} (W_{ij} W_{ij})^{\frac{1}{2}} \left(\frac{M_\infty}{Re} \right) \tag{H-111}$$

$$K_1 = \alpha_3$$
$$K_2 = 2\alpha_2 \qquad \text{(H-112)}$$

where

$$\alpha_1 = \left(\frac{4}{3} - C_2\right)\frac{g}{2}$$
$$\alpha_2 = (2 - C_3)\frac{g}{2} \qquad \text{(H-113)}$$
$$\alpha_3 = (2 - C_4)\frac{g}{2}$$
$$g = \left(\frac{C_1}{2} + C_5 - 1\right)^{-1}$$

(Note that η, ζ, and α are just used here to denote terms in the equations and do *not* reflect the definitions in the Nomenclature list.) Currently, the pressure-strain correlation is modeled with the Speziale-Sarkar-Gatski (SSG) correlation, which uses:

$$C_1 = 6.8 \qquad C_3 = 1.25 \qquad C_5 = 1.88$$
$$C_2 = 0.36 \qquad C_4 = 0.4 \qquad C_\mu{}^* = 0.081 \qquad \text{(H-114)}$$

To improve convergence, the μ_T terms in τ_{ij}^T multiplying the nonlinear terms

$$S_{ik}W_{kj} - W_{ik}S_{kj} \text{ and } S_{ik}S_{kj} - \frac{1}{3}S_{kl}S_{lk}\delta_{ij}$$

are replaced by

$$\mu_T{}' = C_\mu{}'\rho\frac{k}{\omega} \qquad \text{(H-115)}$$

where

$$C_\mu{}' = \frac{3(1+\eta^2)}{3 + \eta^2 + 6\zeta^2\eta^2 + 6\zeta^2 + \eta^6 + \zeta^6}\alpha_1 \qquad \text{(H-116)}$$

Also, the μ_T terms in the D_k and D_ω diffusion terms are replaced by

$$\mu_T{}^* = C_\mu{}^*\rho\frac{k}{\omega} \qquad \text{(H-117)}$$

The minimum distance function d is not required by this model. However, at the present time it is still computed and stored by CFL3D. For the general form in Equation (H-30),

$$X_k = k \tag{H-118}$$

$$S_{P,k} = \frac{1}{\rho}\tau_{ij}^T \frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \tag{H-119}$$

$$S_{D,k} = -\beta' k\omega\left(\frac{Re}{M_\infty}\right) \tag{H-120}$$

$$D_k = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T^*}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-121}$$

and

$$X_\omega = \omega \tag{H-122}$$

$$S_{P,\omega} = \frac{1}{\rho}\gamma\frac{\omega}{k}\tau_{ij}^T \frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \tag{H-123}$$

$$S_{D,\omega} = -\beta\omega^2\left(\frac{Re}{M_\infty}\right) \tag{H-124}$$

$$D_\omega = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T^*}{\sigma_\omega}\right)\frac{\partial \omega}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-125}$$

H.4.7 EASM Gatski-Speziale k-Epsilon Model

(**ivisc** = 11)

The $k-\varepsilon$ version of the Gatski-Speziale EASM model[3] is only implemented as a non-linear model. Its equations are identical to those of the Abid $k-\varepsilon$ model (Equation (H-91) and Equation (H-92)), with

$$\mu_T = C_\mu \frac{\rho k^2}{\varepsilon} \tag{H-126}$$

(There is no f_μ term.) The constants are

$$C_{\varepsilon_1} = C_{\varepsilon_2} - \frac{\kappa^2}{\sigma_\varepsilon \sqrt{C_\mu *}} \qquad \begin{aligned} C_{\varepsilon_2} &= 1.83 \\ \sigma_k &= 1.0 \\ \sigma_\varepsilon &= 1.3 \\ \kappa &= 0.41 \end{aligned} \tag{H-127}$$

The production terms are

$$\begin{aligned} P_k &= \tau_{ij}^T \frac{\partial u_i}{\partial x_j} \\ P_\varepsilon &= C_{\varepsilon_1} \frac{\varepsilon}{k} \tau_{ij}^T \frac{\partial u_i}{\partial x_j} \end{aligned} \tag{H-128}$$

The variable C_μ, and all constants are identical to those given for the EASM k – ω model in "EASM Gatski-Speziale k-Omega Model" on page 286, except that ω is replaced by ε/k. The damping functions are given by

$$f_2 = \left[1 - \exp\left(-\frac{Re_k}{12}\right)\right] \tag{H-129}$$

where

$$Re_k = \frac{\rho \sqrt{k} d}{\mu}\left(\frac{Re}{M_\infty}\right) \tag{H-130}$$

The μ_T terms in τ_{ij}^T multiplying the nonlinear terms are replaced in exactly the same way as for the EASM k – ω model. Also, the μ_T terms in the D_k and D_ε diffusion terms are replaced by

$$\mu_T * = C_\mu * \frac{\rho k^2}{\varepsilon} \tag{H-131}$$

For the general form in Equation (H-30),

$$X_k = k \tag{H-132}$$

$$S_{P,k} = \frac{1}{\rho} \tau_{ij}^T \frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \tag{H-133}$$

$$S_{D,k} = -\varepsilon\left(\frac{Re}{M_\infty}\right) \tag{H-134}$$

$$D_{\mathrm{k}} = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_{\mathrm{T}}^*}{\sigma_{\mathrm{k}}}\right)\frac{\partial \mathrm{k}}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-135}$$

and

$$X_\varepsilon = \varepsilon \tag{H-136}$$

$$S_{\mathrm{P},\varepsilon} = \frac{1}{\rho}C_{\varepsilon_1}\frac{\varepsilon}{\mathrm{k}}\tau_{ij}^{\mathrm{T}}\frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \tag{H-137}$$

$$S_{\mathrm{D},\varepsilon} = -C_{\varepsilon_2}\frac{\varepsilon^2}{\mathrm{k}}f_2\left(\frac{Re}{M_\infty}\right) \tag{H-138}$$

$$D_\varepsilon = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_{\mathrm{T}}^*}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-139}$$

H.4.8 EASM Girimaji k-Epsilon Model

(**ivisc** = 9 and 13)

The Girimaji[21] version of the EASM $\mathrm{k}-\varepsilon$ model is implemented both as a "linear" (eddy-viscosity) (**ivisc** = 9) and "nonlinear" (**ivisc** = 13) version. The $\mathrm{k}-\varepsilon$ equations are

$$\frac{\partial \mathrm{k}}{\partial t} + u_j\frac{\partial \mathrm{k}}{\partial x_j} = \frac{1}{\rho}\mathrm{P_k}\left(\frac{M_\infty}{Re}\right) - \varepsilon\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_{\mathrm{T}}}{\sigma_{\mathrm{k}}}\right)\frac{\partial \mathrm{k}}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-140}$$

$$\frac{\partial \varepsilon}{\partial t} + u_j\frac{\partial \varepsilon}{\partial x_j} = \frac{1}{\rho}\mathrm{P}_\varepsilon\left(\frac{M_\infty}{Re}\right) - C_{\varepsilon_2}\frac{\varepsilon^2}{\mathrm{k}}\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_{\mathrm{T}}}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) + C_{\varepsilon_2}\frac{\varepsilon}{\mathrm{k}}\frac{2\mu}{\rho}\left(\frac{\partial\sqrt{\mathrm{k}}}{\partial x_j}\right)^2\left(\frac{M_\infty}{Re}\right) \tag{H-141}$$

$$\mu_{\mathrm{T}} = -G_1 f_\mu\frac{\rho \mathrm{k}^2}{\varepsilon} \tag{H-142}$$

The extra $(\partial\sqrt{\mathrm{k}}/\partial x_j)^2$ term in the ε equation was added to replace the f_2 damping function term on the ε^2/k term, which is needed because $\varepsilon^2/\mathrm{k} \to \infty$ at the wall. The constants are

$$C_{\varepsilon_1} = 1.44 \qquad \sigma_k = 1.0$$
$$C_{\varepsilon_2} = 1.83 \qquad \sigma_\varepsilon = 1.3 \tag{H-143}$$

The production terms are approximated by

$$P_k = \tau_{ij}^T \frac{\partial u_i}{\partial x_j}$$
$$P_\varepsilon = C_{\varepsilon_1} \frac{\varepsilon}{k} \tau_{ij}^T \frac{\partial u_i}{\partial x_j} \tag{H-144}$$

The damping functions f_μ is

$$f_\mu = \tanh\left[0.015 \frac{\sqrt{k} d \rho}{\mu} \left(\frac{Re}{M_\infty}\right)\right] \tag{H-145}$$

τ_{ij}^T is given (dimensionally) in Equation (H-15). However, K_1 and K_2 in that equation are now given by

$$K_1 = \frac{G_2}{G_1}$$
$$K_2 = -\frac{G_3}{G_1} \tag{H-146}$$

and

$$\text{for} \quad \eta_1 = 0 \qquad G_1 = \frac{L_1^0 L_2}{(L_1^0)^2 + 2\eta_2 (L_4)^2}$$

$$\text{for} \quad L_1^1 = 0 \qquad G_1 = \frac{L_1^0 L_2}{(L_1^0)^2 + \frac{2}{3}\eta_1 (L_3)^2 + 2\eta_2 (L_4)^2}$$

$$\text{for} \quad D > 0 \qquad G_1 = -\frac{p}{3} + \left(-\frac{b}{2} + \sqrt{D}\right)^{\frac{1}{3}} + \left(-\frac{b}{2} - \sqrt{D}\right)^{\frac{1}{3}} \tag{H-147}$$

$$\text{for} \quad D < 0, b < 0 \qquad G_1 = -\frac{p}{3} + 2\sqrt{-\frac{a}{3}} \cos\left(\frac{\theta}{3}\right)$$

$$\text{for} \quad D < 0, b > 0 \qquad G_1 = -\frac{p}{3} + 2\sqrt{-\frac{a}{3}} \cos\left(\frac{\theta}{3} + \frac{2\pi}{3}\right)$$

where

$$\eta_1 = \left(\frac{k}{\varepsilon}\right)^2 S_{ij}S_{ij}\left(\frac{M_\infty}{Re}\right)^2$$
$$\eta_2 = \left(\frac{k}{\varepsilon}\right)^2 W_{ij}W_{ij}\left(\frac{M_\infty}{Re}\right)^2 \quad \text{(H-148)}$$

$$p = -\frac{2L_1^0}{\eta_1 L_1^1} \qquad r = -\frac{L_1^0 L_2}{(\eta_1 L_1^1)^2} \quad \text{(H-149)}$$

$$q = \frac{1}{(\eta_1 L_1^1)^2}\left[(L_1^0)^2 + \eta_1 L_1^1 L_2 - \frac{2}{3}\eta_1 (L_3)^2 + 2\eta_2 (L_4)^2\right] \quad \text{(H-150)}$$

$$a = \left(q - \frac{p^2}{3}\right) \qquad b = \frac{1}{27}(2p^3 - 9pq + 27r)$$
$$D = \frac{b^2}{4} + \frac{a^3}{27} \qquad \cos\theta = \frac{-b/2}{\sqrt{-a^3/27}} \quad \text{(H-151)}$$

$$G_2 = \frac{-L_4 G_1}{L_0^1 - \eta_1 L_1^1 G_1} \qquad G_3 = \frac{+2L_3 G_1}{L_0^1 - \eta_1 L_1^1 G_1} \quad \text{(H-152)}$$

$$L_0^1 = \frac{C_1^0}{2} - 1 \qquad L_2 = \frac{C_2}{2} - \frac{2}{3}$$
$$L_1^1 = C_1^1 + 2 \qquad L_3 = \frac{C_3}{2} - 1 \quad \text{(H-153)}$$
$$L_4 = \frac{C_4}{2} - 1$$

$$C_1^0 = 3.4 \qquad C_2 = 0.36$$
$$C_1^1 = 1.8 \qquad C_3 = 1.25 \quad \text{(H-154)}$$
$$C_4 = 0.4$$

For the general form in Equation (H-30),

$$X_k = k \quad \text{(H-155)}$$

$$S_{P,k} = \frac{1}{\rho}\tau_{ij}^T \frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) \quad \text{(H-156)}$$

$$S_{D,k} = -\varepsilon\left(\frac{Re}{M_\infty}\right) \tag{H-157}$$

$$D_k = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-158}$$

and

$$X_\varepsilon = \varepsilon \tag{H-159}$$

$$S_{P,\varepsilon} = \frac{1}{\rho}C_{\varepsilon_1}\frac{\varepsilon}{k}\tau_{ij}^T\frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right) + C_{\varepsilon_2}\frac{\varepsilon}{k}\frac{2\mu}{\rho}\left(\frac{\partial\sqrt{k}}{\partial x_j}\right)^2\left(\frac{M_\infty}{Re}\right) \tag{H-160}$$

$$S_{D,\varepsilon} = -C_{\varepsilon_2}\frac{\varepsilon^2}{k}\left(\frac{Re}{M_\infty}\right) \tag{H-161}$$

$$D_\varepsilon = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \tag{H-162}$$

H.5 Generalized Coordinate Form

For the transformation from Cartesian coordinates to generalized coordinates, for example,

$$u_j\frac{\partial k}{\partial x_j} = U\frac{\partial k}{\partial \xi} + V\frac{\partial k}{\partial \eta} + W\frac{\partial k}{\partial \zeta} \tag{H-163}$$

where

$$\begin{aligned} U &= \xi_x u + \xi_y v + \xi_z w + \xi_t \\ V &= \eta_x u + \eta_y v + \eta_z w + \eta_t \\ W &= \zeta_x u + \zeta_y v + \zeta_z w + \zeta_t \end{aligned} \tag{H-164}$$

In the diffusion-type terms, cross-derivative terms are neglected. For example,

$$\frac{\partial}{\partial x_j}\left(\mu\frac{\partial k}{\partial x_j}\right) \cong \xi_x\frac{\partial}{\partial\xi}\left(\xi_x\mu\frac{\partial k}{\partial\xi}\right) + \xi_y\frac{\partial}{\partial\xi}\left(\xi_y\mu\frac{\partial k}{\partial\xi}\right) + \xi_z\frac{\partial}{\partial\xi}\left(\xi_z\mu\frac{\partial k}{\partial\xi}\right) \tag{H-165}$$
$$+ \eta_x\frac{\partial}{\partial\eta}\left(\eta_x\mu\frac{\partial k}{\partial\eta}\right) + \eta_y\frac{\partial}{\partial\eta}\left(\eta_y\mu\frac{\partial k}{\partial\eta}\right) + \eta_z\frac{\partial}{\partial\eta}\left(\eta_z\mu\frac{\partial k}{\partial\eta}\right)$$
$$+ \zeta_x\frac{\partial}{\partial\zeta}\left(\zeta_x\mu\frac{\partial k}{\partial\zeta}\right) + \zeta_y\frac{\partial}{\partial\zeta}\left(\zeta_y\mu\frac{\partial k}{\partial\zeta}\right) + \zeta_z\frac{\partial}{\partial\zeta}\left(\zeta_z\mu\frac{\partial k}{\partial\zeta}\right)$$

An example of the discretization is given for the first term on the right-hand side of Equation (H-165):

$$\xi_x\frac{\partial}{\partial\xi}\left(\xi_x\mu\frac{\partial k}{\partial\xi}\right) \rightarrow \xi_{x_i}\xi_{x_{i+\frac{1}{2}}}\mu_{i+\frac{1}{2}}\left(\frac{\partial k}{\partial\xi}\right)_{i+\frac{1}{2}} - \xi_{x_i}\xi_{x_{i-\frac{1}{2}}}\mu_{i-\frac{1}{2}}\left(\frac{\partial k}{\partial\xi}\right)_{i-\frac{1}{2}} \tag{H-166}$$

where

$$\left(\frac{\partial k}{\partial\xi}\right)_{i+\frac{1}{2}} = k_{i+1} - k_i \qquad \left(\frac{\partial k}{\partial\xi}\right)_{i-\frac{1}{2}} = k_i - k_{i-1} \tag{H-167}$$

The locations of the indices are shown in Figure H-2.

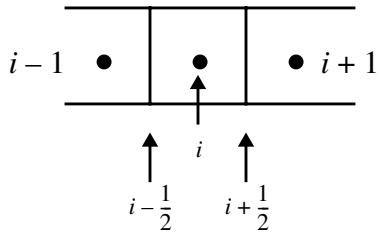


Figure H-2. Definition of left and right states for generalized coordinate transformation.

H.6 Initial Conditions

For the Baldwin-Barth and Spalart-Allmaras models, the levels of the turbulence quantities R and $\hat{\nu}$ are initialized at their free-stream levels everywhere in the flow field. For the two-equation models, it helps the solution start-up phase to initialize the k and ω (or k and ε) values to crude profiles that simulate typical profiles in a boundary layer. Currently, for $k-\omega$:

$$k_{IC} = \max(k_\infty, -C_1 d^2 + C_2 d)$$
$$\omega_{IC} = \max\left(-12444d + 0.54, \frac{C_\mu k_{IC}}{v3d}\right) \quad \text{(H-168)}$$

where d is the distance to the nearest wall and, for EASM models, $C_\mu = 0.09$; otherwise, $C_\mu = 1.0$. Also,

$$v3d = \frac{100 k_{IC}}{t_{max}}$$
$$t_{max} = -C_1 S_{max}^2 + C_2 S_{max} \quad \text{(H-169)}$$
$$S_{max} = \frac{C_2}{2C_1}$$
$$C_1 = 45.8 \qquad C_2 = 1.68$$

For $k-\varepsilon$:

$$k_{IC} = \min\{zk4, \max[zk1, \min(zk2, zk3)]\} \quad \text{(H-170)}$$

$$zk1 = k_\infty$$
$$zk2 = 10^{-471d + 0.47}$$
$$zk3 = 10^{-37.5d - 3.7} \quad \text{(H-171)}$$
$$zk4 = 6.7d$$

$$\varepsilon_{IC} = \min\{ep4, \max[ep1, \min(ep2, ep3)]\} \quad \text{(H-172)}$$

$$ep1 = \varepsilon_\infty$$
$$ep2 = 10^{-555d - 6}$$
$$ep3 = 10^{-280d - 9.2} \quad \text{(H-173)}$$
$$ep4 = \min(1\times10^{20}, 10^{13333d - 9.8})$$

H.7 Boundary Conditions

H.7.1 Free-stream Levels

For the Baldwin-Barth model,

$$R_\infty = 0.1 \tag{H-174}$$

For the Spalart-Allmaras model,

$$\hat{\nu}_\infty = 1.341946 \tag{H-175}$$

For the standard $k - \omega$ or Menter's SST models (C_μ is incorporated into the definition of ω),

$$\begin{aligned} k_\infty &= 9 \times 10^{-9} \\ \omega_\infty &= 1 \times 10^{-6} \end{aligned} \tag{H-176}$$

For the EASM ($k - \omega$) models,

$$\begin{aligned} k_\infty &= 9 \times 10^{-9} \\ \omega_\infty &= 9 \times 10^{-8} \end{aligned} \tag{H-177}$$

For the $k - \varepsilon$ models,

$$\begin{aligned} k_\infty &= 1 \times 10^{-9} \\ \varepsilon_\infty &= 1 \times 10^{-17} \end{aligned} \tag{H-178}$$

The end result for all models is

$$\mu_{T,\infty} \cong 0.009 \tag{H-179}$$

in the free stream. Note that $\omega_\infty = 9 \times 10^{-8}$ and $\varepsilon_\infty = 1 \times 10^{-17}$ for the EASM models are based on $C_\mu = 0.09$, which is only approximate since C_μ is variable.

H.7.2 Boundary Conditions at Solid Walls

With the parameters known at the cell-centers near the solid wall boundary (see the wall region schematic in Figure H-3), the following boundary conditions are applied.

For Baldwin-Barth,

$$R_w = 0 \tag{H-180}$$

For Spalart-Allmaras,

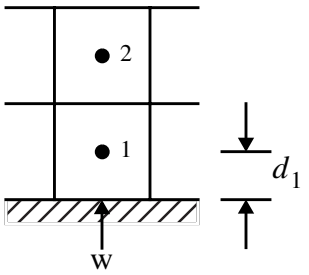


Figure H-3. Parameter locations at wall boundary.

$$\hat{\nu} = 0 \tag{H-181}$$

For the $k-\omega$ models,

$$k_w = 0 \tag{H-182}$$

$$\omega_w = \frac{60\mu_1}{\rho_1 \beta (d_1)^2}\left(\frac{M_\infty}{Re}\right)^2 \tag{H-183}$$

where

$\beta = 0.83$ for the EASM $k-\omega$ models and

$\beta = 0.075$ otherwise.

For the k-ε models,

$$k_w = 0 \tag{H-184}$$

$$\varepsilon_w = \frac{2\mu_1}{\rho_1}\left(\frac{\partial\sqrt{k}}{\partial n}\right)_w^2\left(\frac{M_\infty}{Re}\right)^2 \tag{H-185}$$

where n is the direction normal to the wall.

Note that the actual wall boundary conditions for the turbulence quantities (unlike the primitive variables) are applied at ghost cells. Hence,

$$\chi_{BC} = 2\chi_w - \chi_1 \tag{H-186}$$

where χ represents R, $\hat{\nu}$, k, ω, or ε.

Version 4.1 of CFL3D applied the above boundary condition for ω (Equation (H-183)) at the ghost cell center rather than at the wall. This is not correct, but supposedly any wall boundary condition for ω that is bigger by some factor than the "analytical" value $6\mu_1/(\rho\beta y^2)$ gives basically the same result. (See reference 26.) The current version of CFL3D (Version 5.0) obtains the ghost cell value using linear interpolation from ω_w and ω_1 (or ε_w and ε_1).

H.8 Solution Method

The one or two equations are solved, decoupled, using implicit approximate factorization (AF). The S_P terms are treated explicitly, lagged in time while the S_D and D terms are treated implicitly (they are linearized and a term is brought to the left-hand-side of the equations). This procedure is described by Spalart and Allmaras[35] (strategy #3). The advective terms are discretized using first-order upwinding. Treating the destruction terms implicitly helps increase the diagonal dominance of the left-hand-side matrix. Most of the S_D terms are linearized in a simplified fashion by assuming no coupling between the variables. For example,

$$\begin{aligned}\beta' k\omega^{(n+1)} &\cong \beta' k\omega^{(n)} + \frac{\partial}{\partial k}(\beta' k\omega)\Delta k \\ &\cong \beta' k\omega^{(n)} + \beta'\omega\Delta k\end{aligned} \qquad \text{(H-187)}$$

For the S_D terms in the $k-\varepsilon$ equations, however, a more sophisticated approach was taken. When linearizing, the k and ε variables are assumed to be coupled, with the eddy viscosity μ_T assumed to be fixed. For example, in the k equation, $S_D = -\varepsilon(Re/M_\infty)$ is linearized as follows:

$$\begin{aligned}\varepsilon^{(n+1)} &\cong \varepsilon^{(n)} + \frac{\partial}{\partial k}(\varepsilon)\Delta k \\ &\cong \varepsilon^{(n)} + \frac{\partial}{\partial k}\left(\frac{C_\mu f_\mu \rho k^2}{\mu_T}\right)\Delta k \\ &\cong \varepsilon^{(n)} + 2\frac{C_\mu f_\mu \rho k}{\mu_T}\Delta k \\ &\cong \varepsilon^{(n)} + 2\frac{\varepsilon}{k}\Delta k\end{aligned} \qquad \text{(H-188)}$$

The $S_D = -C_{\varepsilon_2}\frac{\varepsilon^2}{k}f_2(Re/M_\infty)$ term in the ε equation is treated similarly, for consistency:

$$\frac{\varepsilon^{2(n+1)}}{k} \cong \frac{\varepsilon^{2(n)}}{k} + \frac{\partial}{\partial\varepsilon}\left(\frac{\varepsilon^2}{k}\right)\Delta\varepsilon \qquad \text{(H-189)}$$

$$\cong \frac{\varepsilon^{2(n)}}{k} + \frac{\partial}{\partial\varepsilon}\left[\left(\frac{C_\mu f_\mu \rho}{\mu_T}\right)^{\frac{1}{2}}\varepsilon^{\frac{3}{2}}\right]\Delta\varepsilon$$

$$\cong \frac{\varepsilon^{2(n)}}{k} + \frac{3}{2}\left(\frac{C_\mu f_\mu \rho\varepsilon}{\mu_T}\right)^{\frac{1}{2}}\Delta\varepsilon$$

$$\cong \frac{\varepsilon^{2(n)}}{k} + \frac{3}{2}\frac{\varepsilon}{k}\Delta\varepsilon$$

Menter's SST $k-\omega$ model has an additional cross-diffusion term in the ω equation. This term is treated implicitly (See reference 26).

$$C_\omega^{(n+1)} \cong C_\omega^{(n)} + \frac{\partial}{\partial\omega}C_\omega\Delta\omega \qquad \text{(H-190)}$$

$$\cong C_\omega^{(n)} - \frac{|C_\omega|}{\omega}\Delta\omega$$

(The negative sign insures that, when taken to the left-hand side, this term increases the diagonal dominance of the implicit matrix.)

H.8.1 Example

This example shows the solution method for the k equation in Menter's SST model. All other equations are solved in a similar fashion.

$$\frac{\partial k}{\partial t} + u_j\frac{\partial k}{\partial x_j} = \frac{1}{\rho}P_k\left(\frac{M_\infty}{Re}\right) - \beta' k\omega\left(\frac{Re}{M_\infty}\right) + \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]\left(\frac{M_\infty}{Re}\right) \qquad \text{(H-191)}$$

Written in generalized coordinates, with the production term treated explicitly (lagged in time), this equation becomes:

$$\frac{\partial k^{(n+1)}}{\partial t} = -U\frac{\partial k^{(n+1)}}{\partial \xi} - V\frac{\partial k^{(n+1)}}{\partial \eta} - W\frac{\partial k^{(n+1)}}{\partial \zeta} \tag{H-192}$$

$$+ \frac{1}{\rho}P_k\left(\frac{M_\infty}{Re}\right)^{(n)} - \beta' k\omega\left(\frac{Re}{M_\infty}\right)^{(n+1)}$$

$$+ \frac{1}{\rho}\left[\frac{\partial}{\partial \xi}\left\{(\xi_x^2 + \xi_y^2 + \xi_z^2)\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial \xi}\right\}\right.$$

$$+ \frac{\partial}{\partial \eta}\left\{(\eta_x^2 + \eta_y^2 + \eta_z^2)\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial \eta}\right\}$$

$$\left.+ \frac{\partial}{\partial \zeta}\left\{(\zeta_x^2 + \zeta_y^2 + \zeta_z^2)\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial \zeta}\right\}\right]\left(\frac{M_\infty}{Re}\right)^{(n+1)}$$

$$= RHS_k$$

where the superscripts denote the time level. Note that the terms $(\xi_x^2 + \xi_y^2 + \xi_z^2)$ etc. are not strictly correct as written. This is really short-hand notation. The correct way to expand the diffusion terms is given in “Generalized Coordinate Form” on page 294. This section also indicates how these metric terms are discretized.

$$\begin{aligned} U &= \xi_x u + \xi_y v + \xi_z w + \xi_t \\ V &= \eta_x u + \eta_y v + \eta_z w + \eta_t \\ W &= \zeta_x u + \zeta_y v + \zeta_z w + \zeta_t \end{aligned} \tag{H-193}$$

Define

$$\Delta k = k^{(n+1)} - k^{(n)} \tag{H-194}$$

Then

$$\frac{\Delta k}{\Delta t} = RHS_k \tag{H-195}$$

(H-196)

Linearize the right-hand-side terms that are taken at time level $(n+1)$:

$$-U\frac{\partial k^{(n+1)}}{\partial \xi} \cong -U\frac{\partial k^{(n)}}{\partial \xi} - U\delta_\xi^{upwind}(\Delta k) \tag{H-197}$$

$$-\left(\frac{Re}{M_\infty}\right)\beta' k\omega^{(n+1)} \cong -\left(\frac{Re}{M_\infty}\right)\beta' k\omega^{(n)} - \left(\frac{Re}{M_\infty}\right)\beta'\omega(\Delta k) \tag{H-198}$$

$$\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\xi}\left[\chi_\xi\frac{\partial k}{\partial\xi}\right]^{(n+1)} \cong \left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\xi}\left[\chi_\xi\frac{\partial k}{\partial\xi}\right]^{(n)} + \left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\xi}\{\chi_\xi\delta_\xi(\Delta k)\} \tag{H-199}$$

where χ_ξ represents $(\xi_x^2 + \xi_y^2 + \xi_z^2)\left(\mu + \frac{\mu_T}{\sigma_k}\right)$.

The equation is approximately factored and the contribution of the S_D term to the left-hand side is added in the first factor only.

$$\begin{aligned}
&\left[I + \Delta t U\delta_\xi^{upwind} + \Delta t\left(\frac{Re}{M_\infty}\right)\beta'\omega - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\xi}(\chi_\xi\delta_\xi)\right] \\
&\left[I + \Delta t V\delta_\eta^{upwind} - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\eta}(\chi_\eta\delta_\eta)\right] \\
&\left[I + \Delta t W\delta_\zeta^{upwind} - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\zeta}(\chi_\zeta\delta_\zeta)\right]\Delta k = \Delta t RHS_k
\end{aligned} \tag{H-200}$$

This equation is solved in a series of three sweeps in each of the three coordinate directions:

$$\begin{aligned}
&\left[I + \Delta t U\delta_\xi^{upwind} + \Delta t\left(\frac{Re}{M_\infty}\right)\beta'\omega - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\xi}(\chi_\xi\delta_\xi)\right]\Delta k^* = \Delta t RHS_k \\
&\left[I + \Delta t V\delta_\eta^{upwind} - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\eta}(\chi_\eta\delta_\eta)\right]\Delta k^{**} = \Delta k^* \\
&\left[I + \Delta t W\delta_\zeta^{upwind} - \Delta t\left(\frac{M_\infty}{Re}\right)\frac{1}{\rho}\frac{\partial}{\partial\zeta}(\chi_\zeta\delta_\zeta)\right]\Delta k = \Delta k^{**} \\
&k^{(n+1)} = k^{(n)} + \Delta k
\end{aligned} \tag{H-201}$$

Each sweep requires the solution of a scalar tridiagonal matrix.

H.9 Limiting

As a precaution, many of the turbulence quantities are limited in practice. Whether all of these conditions are necessary or not for the models to work and/or converge well is unknown. Some are probably included only because they were tried at some point and never removed.

- In all models, the linearized S_D term is added to the left-hand-side matrix only if its contribution is positive.

- In all models, the terms D, $\hat{\nu}$, k, ω, and ε are not allowed to become negative. If the equation "wants" to yield a negative value, it is instead set to some very small positive number. (A counter also sums the number of times this happen and it is output in the `*.turres` output file.)

- In all the $k-\omega$ models as well as the $k-\varepsilon$ models for Abid and Gatski-Speziale, the production term in the k equation (i.e.

$$\frac{1}{\rho}\tau_{ij}^{T}\frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right)$$

  or an approximation thereof) is limited to be less than or equal to 20 times the destruction term in that equation.

- In all EASM models, the τ_{11}, τ_{22}, and τ_{33} terms are limited to be positive (realizability constraint).

- In all EASM models, the production term $\tau_{ij}^{T}\frac{\partial u_i}{\partial x_j}$ is limited to be positive.

- In the Girimaji EASM models, the production term

$$\frac{1}{\rho}\tau_{ij}^{T}\frac{\partial u_i}{\partial x_j}\left(\frac{M_\infty}{Re}\right)$$

  is limited (in both the k and ε equations) to be less than or equal to $40\varepsilon\left(\frac{Re}{M_\infty}\right)$.

- In the Girimaji EASM models, the term

$$\varepsilon\left(\frac{Re}{M_\infty}\right)-2\frac{\mu}{\rho}\left(\frac{\partial\sqrt{k}}{\partial x_j}\right)^2\left(\frac{M_\infty}{Re}\right)$$

  is limited to be positive.

- In all models, the nondimensional eddy viscosity μ_T is limited to be less than or equal to 100,000.

- In the Gatski-Speziale EASM models, η^2 and ζ^2 are each limited to be less than or equal to 10.

- In the Gatski-Speziale EASM models, C_μ is limited to be between 0.005 and 0.187.

- In the $k-\varepsilon$ models, f_μ is limited to be less than or equal to 1.

- In the Girimaji EASM models, η_1 and η_2 are each limited to be less than or equal to 1200.

- In the Girimaji EASM model, G_1 is limited to be between -0.005 and -0.2.

- In the Gatski-Speziale EASM $k-\omega$ models, "ω" in the denominator of the nonlinear terms in τ_{ij}^T is limited to be greater than or equal to the free-stream value of ω.

- In the EASM $k-\varepsilon$ models, "ε"in the denominator of the nonlinear terms in τ_{ij}^T is limited to be greater than or equal to the free-stream value of ε *only* in terms that appear in the Navier-Stokes equations. This limiting is not done for the τ_{ij} terms that appear in the S_P terms in the k and ε equations.

- In the one- and two-equation models, the diffusion term in one coordinate direction, for example, can be written as

$$\frac{\partial}{\partial\xi}\left[B\frac{\partial\chi}{\partial\xi}\right] = B_{i+\frac{1}{2}}\chi_{i+1} - \left(B_{i+\frac{1}{2}} + B_{i-\frac{1}{2}}\right)\chi_i + B_{i-\frac{1}{2}}\chi_{i-1}$$

In Baldwin-Barth and Spalart-Allmaras, the terms are limited as follows:

$$B_{i+\frac{1}{2}} = \max\left(B_{i+\frac{1}{2}}, 0\right)$$

$$B_{i-\frac{1}{2}} = \max\left(B_{i-\frac{1}{2}}, 0\right)$$

H.10 Wall Function

A simple approach for wall functions is taken from Abdol-Hamid, Lakshmanan, and Carlson[1]. This approach modifies the eddy-viscosity value $\mu_{T,\,w}$ at the wall only. The wall shear stress is estimated from law-of-the-wall values at the first cell center off the wall and is used to determine an "equivalent" eddy viscosity at the wall.

Wall functions can be employed when the grid spacing is too coarse in the direction normal to the wall; typically y^+ lies in the range $30 < y^+ < 200$. (The y^+ value of the first grid point off the wall *should* be $O(1)$ to ensure decent turbulent computations when no wall functions are used. See Section 2.2.) Wall functions are invoked in CFL3D by setting the value of **ivisc**(m) to be negative.

A caution: While wall functions can work very well, they can also sometimes cause problems. They are not strictly valid for separated flows, although many people use them anyway with reasonable success. It is recommended that wall functions *not* be used with the Baldwin-Lomax model in CFL3D. With other turbulence models, it is recommended that they be used sparingly, if at all. In our opinion, it is better to make a turbulent grid ($y^+ = O(1)$) whenever possible and *avoid the use of wall functions*.

The wall function replaces the eddy viscosity at the ghost cell center at walls as follows:

$$\mu_{T,\,ghost} = 2\mu_{T,\,w} - \mu_{T,\,1}$$

$$\mu_{T,\,w} = \mu_1\left[\frac{n^{+2}\mu_1}{\rho_1\left(\frac{\partial u}{\partial y}\right)_1 d^2}\left(\frac{M_\infty}{Re}\right) - 1\right] \qquad \text{(H-202)}$$

$$\left(\frac{\partial u}{\partial y}\right)_1 = \frac{\sqrt{(u_1 - u_w)^2 + (v_1 - v_w)^2 + (w_1 - w_w)^2}}{d_1}$$

where d_1 is the distance to the nearest wall from the first cell center off the wall and

$$\begin{aligned}
&\text{for} \quad R_c \le 20.24 \quad && n^+ = \sqrt{R_c} \\
&\text{for} \quad 20.24 < R_c \le 435 \quad && n^+ = a_0(1) + \sum_{n=2}^{7} a_0(n) R_c^{n-1} \\
&\text{for} \quad 435 < R_c \le 4000 \quad && n^+ = a_1(1) + \sum_{n=2}^{5} a_1(n) R_c^{n-1} \\
&\text{for} \quad R_c > 4000 \quad && n^+ = a_2(1) + \sum_{n=2}^{3} a_2(n) R_c^{n-1}
\end{aligned} \tag{H-203}$$

$$R_c = \frac{\rho_1 \sqrt{(u_1 - u_w)^2 + (v_1 - v_w)^2 + (w_1 - w_w)^2}\, d_1}{\mu_1} \left(\frac{Re}{M_\infty} \right) \tag{H-204}$$

$$\begin{aligned}
a_0(1) &= 2.354039 \\
a_0(2) &= 0.117984 \\
a_0(3) &= -4.2899192 \times 10^{-4} \\
a_0(4) &= 2.0404148 \times 10^{-6} \\
a_0(5) &= -5.1775775 \times 10^{-9} \\
a_0(6) &= 6.2687308 \times 10^{-12} \\
a_0(7) &= -2.916958 \times 10^{-15}
\end{aligned}$$

$$\begin{aligned}
a_1(1) &= 5.777191 \\
a_1(2) &= 6.8756983 \times 10^{-2} \\
a_1(3) &= -7.1582745 \times 10^{-6} \\
a_1(4) &= 1.5594904 \times 10^{-9} \\
a_1(5) &= -1.4865778 \times 10^{-13}
\end{aligned} \tag{H-205}$$

$$\begin{aligned}
a_2(1) &= 31.08654 \\
a_2(2) &= 5.0429072 \times 10^{-2} \\
a_2(3) &= -2.0072314 \times 10^{-8}
\end{aligned}$$

APPENDIX I

Angle Definitions

The definitions of α and β are the same as that given in reference 20 with the exception that the direction of positive β is reversed. The figures below illustrate the difference.

I.1 Standard-type Grid

(z is up and y is out the span)

This is the default for CFL3D-format grids (also applies when **ialph** = 0 for PLOT3D-format grids).

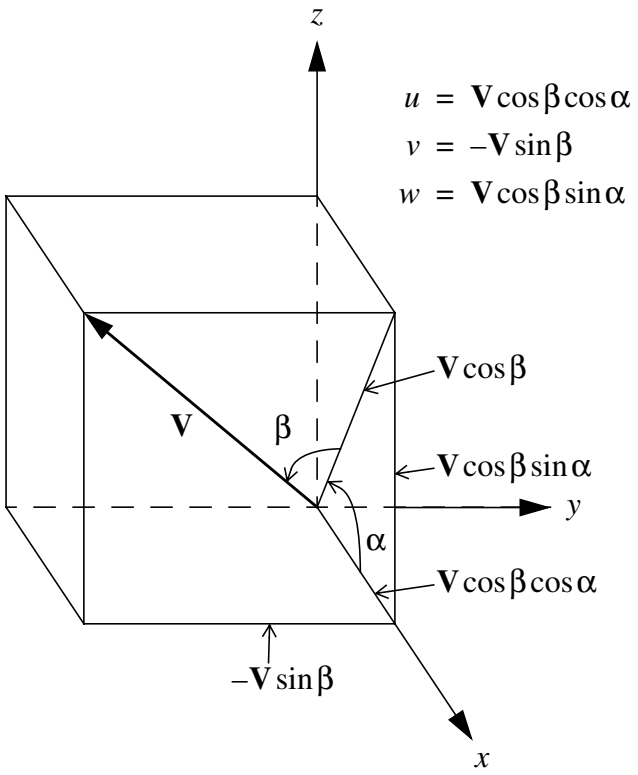


Figure I-1. Grid coordinate system for standard-type grids.

Drag = component of forces parallel to $\mathbf{V}$ direction

Lift = component of forces perpendicular to $\mathbf{V}\cos\beta$ direction in the $x-z$ plane

I.2 Non-standard-type Grid

(y is up and z is out the span)

(**ialph** > 0 for PLOT3D-format grids)

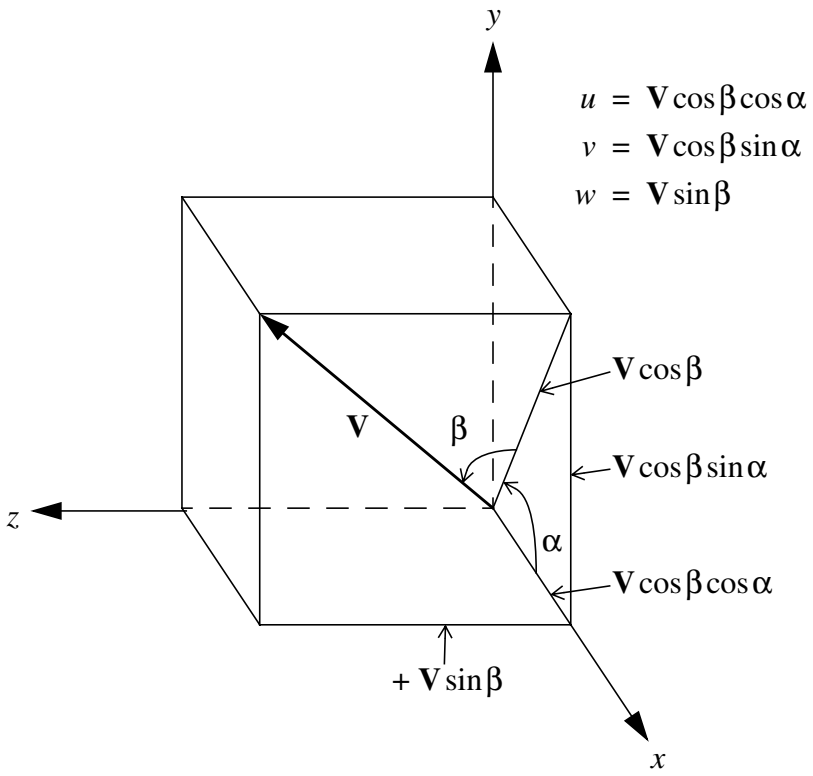
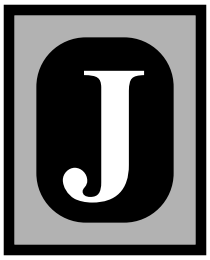


Figure I-2. Grid coordinate system for non-standard-type grids.

Drag = component of forces parallel to $\mathbf{V}$ direction

Lift = component of forces perpendicular to $\mathbf{V}\cos\beta$ direction in the $x-y$ plane



APPENDIX J

Subroutine Listing

CFL3D contains all the subroutines listed below:

CBSEM	RHS	AF3F	BC	LBCX	LBCX (continued)	DYNPTCH	TURBS
mgblk	resid	af3f	bc	tau	findmin_new	dynptch	blomax
setup	resp	amafj	bc1000	tau2x	bbdist	global2	barth3d
rp3d	fa	swafj	bc1001	colldat	bbdst1	patcher	spalart
setblk	i2x	amafk	bc1002	collx	calc_dist	loadgr	twoeqn
global	i2xs	swafk	bc1003	collv	collect_surf	collapse	triv
lead	fa2xj	amafi	bc1005	collq	distcc	rechk	
qinter	fa2xk	swafi	bc1008	coll2q	distcg	expand	
qout	fa2xi	diagj	bc1011	collqc0	initf	invert	
qface	gfluxr	diagk	bc1012	collxt	initi	shear	
plot3c	hfluxr	diagi	bc1013	collxtb	ifree	arc	
plot3d	ffluxr	tinvr	bc2002	addx	ffree	diagnos	
plot3t	xlim	tdq	bc2003	addxv	iialloc	direct	
update	fhat	dlutr	bc2004	add2x	ifalloc	project	
updateg	fill	dfbtr	bc2005	add2xv	makebb	extra	
resetg	fluxp	dlutrp	bc2006	init	calcbb	extrae	
xtbatb	fluxm	dfbtrp	bc2007	initvist	getvrt	topol	
grdmove	gfluxv	dfhat	bc2102	rrest	shells	topol2	
resetwk	hfluxv	dfluxpm	chksym	rrestg	spltbb	dsmin	
rpatch	ffluxv	gfluxl	chkrap	wrest	push	xe	
setqc0	dird	hfluxl	chkrot	wrestg	pop	xe2	
setdqc0	wmag	ffluxl	rie1d	metric	sort_x	newfit	
resadd	delv	abciz	rie1de	tmetric	move_real	trace	
readdat	prntcp	abcjz	blockk	cellvol	move_integer	transp	
getdhdr	trans	abckz	blockj	ctime	triang	rotatp	
	transmc	dabciz	blocki	vlutr	heap_sort		
	rotate	dabcjz	cblkk	vlutrp	cntsurf		
	rotatmc	dabckz	cblkj	bsub	celcen		
	hole		cblki	bsubp	yplusout		
	dthole		int2	q8sdot	calyplus		
	blkmax		int3	q8smax	forceout		
			rotateq	q8smin	histout		
			rotateq0	q8ssum			
			rotateqb	q8vrev			
			bcchk	l2norm			
			xupdt	l2norm2			
			getibk	force			
			intrbc	csurf			
			ccf	csout			
				rsmooth			
				xmukin			
				findmin			

J.1 CBSEM Routines

`mgblk`

Advances the solution in time using multigrid acceleration. At each level in the multigrid procedure, all the blocks are advanced before moving to the next level.

`setup`

Reads in the grid and restart data and calculates some preliminary information, such as the metrics, for subsequent use.

`rp3d`

Reads in the grids in PLOT3D format.

`setblk`

Initializes the `blank` array.

`global`

Reads in the case input file.

`lead`

Installs the attributes of a particular block into the common blocks.

`qinter`

Interpolates the solution from a coarser mesh to a finer mesh. The finer mesh can be either a global mesh or an embedded mesh. Also updates grid position of finer mesh if meshes are in motion.

`qout`

Outputs data for plotting or printing.

`qface`

Determines the primitive variables at the edges of the grid and installs them in the `qj0/qk0/qi0` arrays for use in output routines.

`plot3c`

Writes the output file for the cell-centered points in PLOT3D format. (Outputs the grid and/or solution in single precision for use with FAST and/or PLOT3D.) Also prints solution data to a printout file for a specified range of points.

`plot3d`

Writes the output file at the grid points in PLOT3D format. (Outputs the grid and/or solution in single precision for use with FAST and/or PLOT3D.) Also prints solution data to a printout file for a specified range of points.

`plot3t`

Writes turbulent information for the cell-centered points in PLOT3D format.

`update`

Updates the solution in time.

`updateg`

Updates the grid to a new position and obtains corresponding grid-boundary velocities for use in the boundary conditions. Also collocates new grid position to coarser levels and obtains grid-boundary velocities on coarser levels. Also updates moment center.

`resetg`

Checks to see if any blocks in the grid have been translated and/or rotated out of the bounds set in the input deck. If so, resets these blocks so that they are at or near the initial positions. If the rotational displacement of a block is reset, the solution must also be rotated to correspond to the reset position. (Resetting is allowed only for constant translational speed, **itrans** = 1, or constant rotational speed, **irotat** = 1.)

`xtbatb`

Stores grid speeds and accelerations on the boundaries for use in setting no-slip and wall pressure boundary conditions for dynamic meshes.

`grdmove`

Moves the grid from one position to another.

`resetwk`

Replaces all locations of the work array which were filled with integer values in subroutine `plot3d` with real values. (Otherwise, problems may arise later when a real array attempts to access the integer value located in memory.)

`rpatch`

Reads in the generalized-coordinate interpolation data for grid patching from a file.

`setqc0`

Stores conservative variables for use in second-order temporal differencing and sub-iteration.

`setdqc0`

Stores conservative variables ($\mathbf{Q}_n - \mathbf{Q}_{n-1}$) for use in second-order temporal differencing.

`resadd`

Adds additional terms to the right-hand side for sub-iteration and second-order temporal accuracy.

`readdat`

Reads in auxiliary data arrays for the "2000 series" of boundary conditions.

`getdhdr`

Sets the character data for the main output file headers when the "2000" series of boundary conditions are used.

J.2 RHS Routines

`resid`

Computes the residual contributions to the right-hand side.

`resp`

Computes and prints residuals and sums the forces.

`fa`

Accumulates fluxes to insure conservation with grid embedding.

`i2x`

Interpolates the primitive variables from coarser meshes onto twice finer meshes for grid embedding.

`i2xs`

Interpolates the primitive variables from coarser meshes onto twice finer meshes for grid embedding.

`fa2xj`

Accumulates fluxes in the j direction for use on a twice coarser mesh to insure conservation with grid embedding.

`fa2xk`

Accumulates fluxes in the k direction for use on a twice coarser mesh to insure conservation with grid embedding.

`fa2xi`

Accumulates fluxes in the i direction for use on a twice coarser mesh to insure conservation with grid embedding.

`gfluxr`

Computes residual contributions for the right-hand side in the j direction from the inviscid terms.

`hfluxr`

Computes residual contributions for the right-hand side in the k direction from the inviscid terms.

`ffluxr`

Computes residual contributions for the right-hand side in the i direction from the inviscid terms.

`xlim`

Performs monotone interpolations to the interfaces of the cells.

`fhat`

Computes Roe's[31] generalized flux at the interface given the left and right states at the interface.

`fill`

Fills the edges of the `q` array for safety using a multi-plane vectorization technique.

`fluxp`

Computes the "positive" parts of the fluxes using the flux-vector-splitting method of van Leer.[39]

`fluxm`

Computes the "negative" parts of the fluxes using the flux-vector-splitting method of van Leer.[39]

`gfluxv`

Calculates the right-hand-side residual contributions in the j direction due to the viscous terms.

`hfluxv`

Calculates the right-hand-side residual contributions in the k direction due to the viscous terms when `idf` = 0. Calculates the implicit matrix terms when `idf` > 0.

`ffluxv`

Calculates the right-hand-side residual contributions in the i direction due to the viscous terms.

`dird`

Evaluates directed distance from the $k = 0$ wall and the $i = 0/j = 0$ wall for use in evaluating the Baldwin-Lomax[10] turbulence model.

`wmag`

Evaluates the vorticity magnitude for use in determining the turbulent eddy viscosity.

`delv`

Evaluates the velocity derivatives at cell centers.

`prntcp`

Writes the pressures on the body (actually the cell centers closest to the body) to an output file for unsteady flow.

`trans`

Determines the increment to grid position due to a grid translation.

`transmc`

Determines the increment to moment center due to a grid translation.

`rotate`

Determines the increment to grid position due to a grid rotation.

`rotatmc`

Determines the increment to moment center due to a grid rotation.

`hole`

Zeroes out the right-hand-side residuals for the blanked points.

`dthole`

Updates the Δt values for the hole and fringe cells. The values will be replaced with $\Delta t_{\min}$.

`blkmax`

Determines the location of the maximum residual.

J.3 AF3F Routines

`af3f`

Advances the solution in time using a 3-factor spatially-split approximate factorization algorithm.

`amafj`

Formulates the implicit matrices in the j direction for the 3-factor algorithm.

`swafj`

Solves the block 5×5 tridiagonal equations for the 3-factor spatially-split algorithm in the j direction.

`amafk`

Formulates the implicit matrices in the k direction for the 3-factor algorithm.

`swafk`

Solves the block 5×5 tridiagonal equations for the 3-factor spatially-split algorithm in the k direction.

`amafi`

Formulates the implicit matrices in the i direction for the 3-factor algorithm.

`swafi`

Solves the block 5×5 tridiagonal equations for the 3-factor spatially-split algorithm in the i direction.

`diagj`

Solves the scalar tridiagonal equations to approximate the spatially-split factor in the j direction of the 3-D spatially-split algorithm.

`diagk`

Solves the scalar tridiagonal equations to approximate the spatially-split factor in the k direction of the 3-D spatially-split algorithm.

`diagi`

Solves the scalar tridiagonal equations to approximate the spatially-split factor in the i direction of the 3-D spatially-split algorithm.

`tinvr`

Multiplies the inverse of the diagonalizing matrix **T** times the residual contribution ($\mathbf{T}^{-1}R$).

`tdq`

Multiplies the inverse of the diagonalizing matrix **T** times the change in characteristic combination of variables ($\mathbf{T}^{-1}\Delta\mathbf{q}$).

`dlutr`

Performs the scalar tridiagonal (LU) decomposition.

`dfbtr`

Performs the back substitution for a scalar tridiagonal system of equation.

`dlutrp`

Performs the LU decomposition for a periodic scalar tridiagonal system of equations.

`dfbtrp`

Performs the back substitution for a periodic scalar tridiagonal system of equations.

`dfhat`

Computes a Jacobian matrix with respect to the primitive variables at the cell interface. The Jacobian evaluation is approximate, being taken as either A^+ or A^- ($T\Lambda\mathbf{T}^{-1}$), and is computed with metric terms from the interface and dependent variables fro the cell centers.

`dfluxpm`

Computes "positive" or "negative" parts of the flux Jacobians using the flux-vector-splitting method of van Leer.[39]

`gfluxl`

Computes the left-hand flux contributions due to the inviscid terms for the j direction.

`hfluxl`

Computes the left-hand flux contributions due to the inviscid terms for the k directions.

`ffluxl`

Computes the left-hand flux contributions due to the inviscid terms for the i direction.

`abciz`

Zeroes out the left-hand-side matrix element with the help of the blank array. For a point with `blank` = 0, all elements of matrices `a` and `c` become zero. Only diagonal elements matrix `b` is changed to 1.0 for i implicit; j sweep.

`abcjz`

Zeroes out the left-hand-side matrix element with the help of the blank array. For a point with `blank` = 0, all elements of matrices `a` and `c` become zero. Only diagonal elements matrix `b` is changed to 1.0 for j implicit; k sweep.

`abckz`

Zeroes out the left-hand-side matrix element with the help of the blank array. For a point with `blank` = 0, all elements of matrices `a` and `c` become zero. Only diagonal elements matrix `b` is changed to 1.0 for k implicit; j sweep.

`dabciz`

Uses the blank values to modify the coefficient matrices, `a`, `b`, `c`, for the diagonal inversion in the i direction.

`dabcjz`

Uses the blank values to modify the coefficient matrices, `a`, `b`, `c`, for the diagonal inversion in the j direction.

`dabckz`

Uses the blank values to modify the coefficient matrices, `a`, `b`, `c`, for the diagonal inversion in the k direction.

J.4 BC Routines

`bc`

Determines boundary data/conditions at edges of grids.

`bc1000`

Sets free-stream boundary conditions.

`bc1001`

Sets symmetry plane boundary conditions.

`bc1002`

Sets extrapolation boundary conditions.

`bc1003`

Sets characteristic inflow/outflow boundary conditions.

`bc1005`

Sets inviscid surface boundary conditions.

`bc1008`

Sets tunnel inflow boundary conditions.

`bc1011`

Sets singular axis (half plane) boundary conditions.

`bc1012`

Sets singular axis (full plane) boundary conditions.

`bc1013`

Sets extrapolation boundary conditions for a singular axis.

`bc2002`

Sets pressure ratio; extrapolates other quantities.

`bc2003`

Sets characteristic inlet boundary conditions at engine inlet given (estimated) inlet Mach number, total pressure ratio, total temperature ratio, and flow angle.

`bc2004`

Sets solid wall (viscous wall) boundary conditions.

`bc2005`

Sets periodic boundary conditions given angular rotation angle to the periodic face and its block number.

`bc2006`

Sets pressure via radial equilibrium condition; extrapolates other quantities.

`bc2007`

Sets all the primitive variables with standard CFL3D normalization; $\tilde{\rho}/\tilde{\rho}_\infty$, $\tilde{u}/\tilde{a}_\infty$, $\tilde{v}/\tilde{a}_\infty$, $\tilde{w}/\tilde{a}_\infty$, $\tilde{p}/(\tilde{\rho}_\infty\tilde{a}_\infty^2)$.

`bc2102`

Sets the pressure ratio as a function of time; extrapolates the other flow-field quantities.

`chksym`

Checks for symmetry boundary conditions in j, k, or i directions in order to apply boundary condition type 1011 (singular axis with half-plane symmetry).

`chkrap`

Checks for wrap-around in j, k, or i directions in order to apply boundary condition type 1012 (singular axis with full plane).

`chkrot`

Checks to make sure that the proper rotation angle for periodic boundary conditions has been set.

`rie1d`

Determines far-field boundary data using quasi-1-D characteristic relations for boundary condition type 1003.

`rie1de`

Determines far-field boundary data using quasi-1-D characteristic relations for boundary condition type 2003.

`blockk`

Transfers information from the block designated `ir` to the `qk0` array of the block designated `it`.

`blockj`

Transfers information from the block designated `ir` to the `qj0` array of the block designated `it`.

`blocki`

Transfers information from the block designated `ir` to the `qi0` array of the block designated `it`.

`cblkk`

Checks information transferred from the block designated `ir` to the `qk0` array of the block designated `it`.

`cblkj`

Checks information transferred from the block designated `ir` to the `qj0` array of the block designated `it`.

`cblki`

Checks information transferred from the block designated `ir` to the `qi0` array of the block designated `it`.

`int2`

Linearly interpolates `q` from one grid to ghost cells of another grid using generalized coordinates without using a limiter on the gradients.

`int3`

Linearly interpolates `q` from one grid to ghost cells of another grid using generalized coordinates using a limiter on the gradients.

`rotateq`

Rotates the solution contained in array `q` through a specified angle and stores the rotated solution in `qrot`.

`rotateq0`

Rotates the solution at ghost points contained in array `q0` (either `qi0`, `qj0`, or `qk0` through the angle $\Delta\theta_x/\Delta\theta_y/\Delta\theta_z$ and stores the rotated solution in `q0rot`.

`rotateqb`

Rotates the solution in the `qb` array through the angle $\Delta\theta_x/\Delta\theta_y/\Delta\theta_z$ for the chimera scheme with rotating grids.

`bcchk`

Determines if the boundary conditions have been set and fills the end-points for safety.

`xupdt`

Updates the fringe points of overlapped grids with boundary values which have been interpolated from other grids to provide the mechanism for coupling the various grids.

`getibk`

Reads the output from MaGGiE (but not the grids).

`intrbc`

Interpolates the corrections for boundary values for all grids overlapped in the current mesh.

`ccf`

Modifies u_f, w_f, a_f (velocities and speed of sound at the far field) based on point vortex correction (used when **i2d** = -1 and the far-field boundary condition type is 1003).

J.5 LBCX Routines

`tau`

Computes a residual correction and stores the values of `q` for later use in determining corrections to finer grids in the multigrid iteration scheme.

`tau2x`

Puts the restricted residual from a finer embedded mesh into a coarser mesh.

`colldat`

Restricts auxiliary boundary condition data arrays to coarser meshes.

`collx`

Restrict x, y, and z values to coarser meshes.

`collv`

Restricts volumes to coarser meshes.

`collq`

Restricts `q` (the primitive variables) with a volume-weighted interpolation and residuals to coarser meshes. Also restricts turbulent eddy viscosity in the case of turbulent flows to coarser meshes.

`coll2q`

Restricts `q` (the primitive variables) with a volume-weighted interpolation and residuals from finer embedded meshes to coarser meshes.

`collqc0`

Restricts conservative variables $\mathbf{Q}_n$ and $\mathbf{Q}_n - \mathbf{Q}_{n-1}$ to coarser meshes via summation over fine-grid cells for use in time-accurate multigrid.

`collxt`

Restricts `xt` (array containing grid speeds) to coarser meshes.

`collxtb`

Restricts `xtb` and `atb` (arrays containing grid boundary velocity and acceleration, respectively) to coarser meshes.

`addx`

Interpolates the solution or the correction from a coarser mesh to a finer mesh.

`addxv`

Interpolates the turbulence quantities from a coarser mesh to a finer mesh during mesh sequencing.

`add2x`

Interpolates the solution or the correction from a coarser mesh to a finer embedded mesh.

`add2xv`

Interpolates the turbulence quantities from a global mesh to an embedded mesh during mesh sequencing.

`init`

Sets the initial conditions on a mesh to be free stream.

`initvist`

Sets the turbulent initial conditions on a mesh.

`rrest`

Reads the restart file for a block.

`rrestg`

Reads the restart file to get the latest position for a dynamic mesh, along with corresponding metrics and grid-boundary speeds. Also reads `qc0` for a second-order accurate in time restart.

`wrest`

Writes the restart file for a block.

`wrestg`

Appends the latest position of a dynamic mesh to the end of the restart file. Also writes `qc0` for a second-order accurate in time restart.

`metric`

Calculates the cell-interface directed areas.

`tmetric`

Calculates the time-metric terms for a grid in motion.

`cellvol`

Calculates the cell volumes.

`ctime`

Calculates the time step for an input fixed Courant number or calculates the Courant number based on an input value of Δt.

`vlutr`

Performs the LU decomposition for a block 5×5 tridiagonal system of equations.

`vlutrp`

Performs the LU decomposition for a block 5×5 tridiagonal system of equations which is periodic.

`bsub`

Performs the back substitution for a block 5×5 tridiagonal matrix equation solution.

`bsubp`

Performs the back substitution for a block 5×5 tridiagonal matrix equation solution which is periodic.

`q8sdot (function)`

Computes the dot product between two vectors.

`q8smax (function)`

Finds the maximum value in an array.

`q8smin (function)`

Finds the minimum value in an array.

`q8ssum (function)`

Sums the elements of a vector

`q8vrev`

Reverses the elements in an array.

`l2norm`

Computes the L2-norm of the residuals or the change in primitive variables from one time to the next.

`l2norm2`

Computes the L2-norm of the residuals, after subtracting out the contribution of the unsteady terms that were added in subroutine `resadd`.

`force`

Integrates the forces on the body.

`csurf`

Integrates control surface mass flow and momentum/forces.

`csout`

Writes control surface mass flow and momentum/forces to an output file.

`rsmooth`

Performs implicit residual smoothing (constant coefficient).

`xmukin`

Computes Sutherland's formula with linear law at low temperatures.

`findmin`

Finds minimum distances from field points to viscous wall(s).

`findmin_new`

Serves as a driver routine for computing distances to the closest viscous surface.

`bbdist`

Serves as a driver routine for determining the nearest bounding box for each field point.

`bbdst1`

Identifies the nearest bounding box for each field point.

`calc_dist`

Finds the minimum distance to field points using the recursive-box algorithm.

`collect_surf`

Stores coordinates of surface points and identifies the neighbors of each surface point.

`distcc`

Averages the distance function based at grid points to one based at cell centers.

`distcg`

Collocates the fine-grid minimum distance function on a fine grid to a coarser grid.

`initf`

Initializes pointers to floating-point variables in recursive-box algorithm.

`initi`

Initializes pointers to integer variables in recursive-box algorithm.

`ifree`

"Frees up" pointers to integer variables in recursive-box algorithm.

`ffree`

"Frees up" pointers to floating-point variables in recursive-box algorithm.

`iialloc (function)`

Increments pointers to integer variables in recursive-box algorithm.

`ifalloc (function)`

Increments pointers to floating-point variables in recursive-box algorithm.

`makebb`

Serves as a driver routine for generating the bounding boxes for the recursive-box algorithm.

`calcbb`

Calculates bounding boxes.

`getvrt`

Searches for all points that fall within a bounding box.

`shells`

Performs a shell sort.

`spltbb`

Subdivides bounding boxes.

`push`

Places an item into the stack and adjusts the pointer accordingly.

`pop`

Removes an item from the stack and adjusts the pointer accordingly.

`sort_x`

Sorts surface points with respect to x-coordinate.

`move_real`

Rearranges items in the real array `x` based on the pointer `iperm`.

`move_integer`

Rearranges items in the integer array `x` based on the pointer `iperm`.

`triang`

Finds the closest distance from a field point to the actual surface (i.e. not simply the closest discrete surface point), using local triangulation of the surface.

`heap_sort`

Sorts a list of points.

`cntsurf`

Counts the number of viscous surface points.

`celcen`

Finds cell centers of field points.

`yplusout`

Calls the routines necessary for calculating yplus at the first grid point above solid walls.

`calyplus`

Calculates y^+ in turbulent flows at solid surfaces in a block and calculates statistics for the y^+ distribution (average y^+, maximum y^+ and its location, standard deviation).

`forceout`

Writes the forces and moments on individual blocks, as well as a global force/moment summary, to an output file.

`histout`

Writes the convergence history for mean-flow equations and turbulence equations to an output file.

J.6 DYNPTCH Routines

`dynptch`

Establishes zone-to-zone communication for block interfaces that move relative to one another, using a patched-grid technique (nonconservative).

`global2`

Reads the dynamic patch input parameters.

`patcher`

Calculates patched-grid interpolation coefficients.

`loadgr`

Loads the proper grid from a 1-D storage array to a 2-D work array.

`collapse`

Checks for collapsed points in the grid and expands any collapsed lines detected.

`rechk`

Checks for branch cuts.

`expand`

Expands the grid at boundaries.

`invert`

Determines generalized coordinates of cell centers of the “to” grid in terms of the generalized coordinate system(s) defined on the “from” grid(s).

`shear`

Determines the generalized coordinates of cell edge midpoints on $\xi = 0$ and $\eta = 0$ boundaries and determines the requisite shearing correction to the generalized coordinates near $\xi = 0$ and/or $\eta = 0$ boundaries.

`arc`

Performs arc-length correction to the generalized coordinates near a boundary if required when shearing correction has failed.

`diagnos`

Performs diagnostic checks on interpolation coefficients (generalized coordinates) found via search and inversion routines.

`direct`

Computes normalized directed area components, or equivalently, components of the unit normal to a cell face.

`project`

Projects a point onto a plane.

`extra`

Computes extra mid-cell points in ξ direction.

`extrae`

Computes extra mid-cell points in η direction.

`topol`

Searches appropriate "from" blocks for current "to" cell center. Driver routine for determining ξ and η of the cell center.

`topol2`

Searches appropriate "from" blocks in one direction only for current "to" cell center

`dsmin`

Finds the closest point in a grid to a specified point.

`xe`

Selects proper coordinates to use for inversion.

`xe2`

Sets up the coefficients for locally fitting a polynomial variation in the ξ and η directions.

`newfit`

Determines a new polynomial fit for cells with stubborn convergence.

`trace`

Writes the search routine history for the current "to" cell to unit 7.

`transp`

Translates the "from" block to provide complete coverage for interpolation for cases in which the complete physical domain is not modeled.

`rotatp`

Rotates "from" block to provide complete coverage for interpolation for cases in which the complete physical domain is not modeled.

J.7 TURBS Routines

`blomax`

Computes the turbulent viscosity distributions using the Baldwin-Lomax[10] two-layer eddy-viscosity model.

`barth3d`

Computes the turbulent viscosity distributions using the one-equation Baldwin-Barth[9] turbulence model.

`spalart`

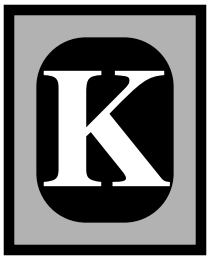
Computes the turbulent viscosity distributions using the one-equation Spalart[35] turbulence model.

`twoeqn`

Computes the turbulent viscosity distributions using the two-equation turbulence models.

`triv`

Solves a scalar tridiagonal system of equation



APPENDIX K

Version Differences

K.1 Significant Differences Between Versions 5.0 and 4.1

1. The most significant difference between the two versions is the generalization of moving grid capability and the added capability to handle sliding patched interfaces.
2. The viscous wall boundary condition now sets wall temperature locally rather than globally.
3. The capability to output control surface information and turbulence quantity information has been added.
4. There are slight differences in how the k – ω wall boundary conditions are set internal to the code (the new way is more "correct"). This change may result in very small differences in convergence, but the end result should be nearly identical to the old way.
5. The Baldwin-Lomax model can be applied on **jdim/kdim/idim** bodies now.
6. In 2-d, the "far-field point-vortex correction" boundary condition can now be applied. Set **i2d** = –1, in combination with 1003.
7. A few more boundary conditions have been added.
8. Several more turbulence models have been added. Also, the "SSTZ" version of the SST k – ω model has been removed.
9. A crude wall function capability has been added.
10. Some very minor differences, which may result in nearly insignificant differences between Version 4.1 and Version 5.0 results, are:
    (a) For the Wilcox $k - \omega$ model, Version 5.0 limits the production term in the k equation to be less than 20 times the destruction term. Version 4.1 does not.
    (b) In Version 4.1, the input alpha and beta angles (in degrees) are transformed to radians by dividing by 57.2958. In Version 5.0, the transformation (using parameter `radtodeg`) is more precise.
11. Both the input file and the restart file are different. See the following section for details on the differences in the input file. Any existing Version 4.1 input file or restart file can be automatically transformed to a Version 5.0 file by using `v4tov5_input.f` or `v4tov5_restart.f`, located in the **`Tools`** directory. (See "The Code and Supplementary Files" on page 9.)

K.2 Summary of Changes to the Input File

Version 5.0 of CFL3D originated when the generalization of moving grid capability (including sliding patched interfaces) was added to Version 4.1. As a consequence, the method for controlling the movement of any dynamic grids, patched or not, is now controlled from a new "section" at the end of the input file. Other than this, most changes to the input file, for either improved clarity or added capability, are relatively minor. After obtaining Version 5.0 of CFL3D, veteran users may notice some minor differences from Version 4.1. For those changes involving new input parameters, be sure to read about them in Chapter 3. Any existing Version 4.1 input file can be automatically transformed to a Version 5.0 input file by using `v4tov5_input.f`, located in the **Tools** directory. (See "The Code and Supplementary Files" on page 9.)

In the following, note that sample numerical values, which of course are case dependent, are included.

1. Line Type Three changed from

```
      XMACH      ALPHA       BETA  REUE,MIL   TINF,DR      ISND     C2SPE
     0.8750     00.000        0.0    02.660     460.0         0       0.0
```

   to

```
      XMACH      ALPHA       BETA  REUE,MIL   TINF,DR     IALPH     IHIST
     0.8750     00.000        0.0    02.660     460.0         0         0
```

   (Note: **isnd** and **c2spe** now handled by **Twtype** in the boundary condition section.)

2. Line Type Five changed from

```
           DT      IREST   IFLAGTS       FMAX      IUNST      RFREQ     ALPHAU
CLOC
      -02.000          0       000    05.0000          0    0.39600    0.22200
0.50000
```

   to

```
           DT      IREST   IFLAGTS       FMAX      IUNST     CFLTAU
      -02.000          0       000    05.0000          0       10.
```

   (Note: Unsteady grid motion is now handled in Line Types Thirty-Three through Fourth-Five)

3. Any boundary condition (Line Types Fourteen through Nineteen) with viscous wall (**bctype** 1004) changed from

```
          1         1      1004         0         0         0         0         0
```

   to

```
          1         1      2004         0         0         0         0         2
              TWTYPE         CQ
                 0.         0.
```

4. The 1-1 blocking section of the input file has been modified so that the parameter names are more pertinent and so that the bookkeeping of 1-1 interfaces is simplified. Line Types Twenty-Four through Twenty-Seven have been changed from

```
   1-1 BLOCKING DATA:
      NBLI
         2
     NBLON    NBLK(1)    NBLK(2)
         0          1          1
         0          3          4
  LIMBLK(1) LIMBLK(2) LIMBLK(3) LIMBLK(4) LIMBLK(5) LIMBLK(6) ISVA(1,1)
ISVA(1,2)
         1         1         1         2        41         1         1         2
```

```
          1         1         1         2        63         1         1         2
    LIMBLK(1) LIMBLK(2) LIMBLK(3) LIMBLK(4) LIMBLK(5) LIMBLK(6) ISVA(2,1)
ISVA(2,2)
          1       257         1         2       217         1         1         2
          1         1        51         2        63        51         2         3
```

to

```
   1-1 BLOCKING DATA:
      NBLI
        2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1             1      1      1      2     41      1      1      2
      2      3             1      1      1      2     63      1      1      2
 NUMBER   GRID     :    ISTA   JSTA   KSTA   IEND   JEND   KEND  ISVA1  ISVA2
      1      1             1    257      1      2    217      1      1      2
      2      4             1      1     51      2     63     51      2      3
```

The parameter **nblon** is no longer used (it is always 0). The values for **nblk**(1) and **nblk**(2) are now assigned under **grid** in Line Types Twenty-Five and Twenty-Six, respectively. The parameter **number** is intended to help the user keep track of interface numbers. This is particularly useful when there are a very large number of 1-1 interfaces. This parameter is not used internal to the code.

5. After **nprint** section, the following Line Types (Thirty-One and Thirty-Two) must be added:

```
CONTROL SURFACE:
NCS
  0
 GRID ISTART   IEND   JSTART   JEND   KSTART   KEND  IWALL  INORM
```

These lines (in combination with **ihstry**) control output of control surface information, such as mass flow. (Note: When using a control surface for print out, then **ncs** 0 and there will be at least one additional line under **grid**...)

6. The above items 1 - 5 are all that are *necessary* to transform an existing Version 4.1 input file (with non-moving grid) to a Version 5.0 input file. For a moving grid, Version 5.0 now has the additional section Line Types Thirty-Three through Forty-Five, required if and only if **iunst** > 0.

CFL3D Papers

General References

1. Biedron, R. and Thomas, J., "A Generalized Patched-Grid Algorithm with Application to the F-18 Forebody with Actuated Control Strake," *Computing Systems in Engineering*, Vol. 1, Nos. 2-4, pp. 563-576, 1990.

2. Compton, W., Thomas, J., Abeyounis, W., and Mason, M., "Transonic Navier-Stokes Solutions of Three-Dimensional Afterbody Flows," NASA TM 4111, July 1989.

3. Ghaffari, F., Luckring, J., Thomas, J., Bates, B., and Biedron, R., "Multiblock Navier-Stokes Solutions About the F/A-18 Wing-LEX-Fuselage Configuration," *Journal of Aircraft*, Vol. 30, No. 3, pp. 293-303, 1993.

4. Rumsey, C., Biedron, R., and Thomas, J., "CFL3D: Its History and Some Recent Applications," NASA TM-112861, May 1997; presented at the "Godunov's Method for Gas Dynamics" Symposium, Ann Arbor, MI, May 1-2, 1997.

5. Rumsey, C. and Vatsa, V., "Comparison of the Predictive Capabilities of Several Turbulence Models," *Journal of Aircraft*, Vol. 32, No. 3, pp. 510-514, 1995.

6. Rumsey, C., Sanetrik, M., Biedron, R., Melson, N., and Parlette, E., "Efficiency and Accuracy of Time-Accurate Turbulent Navier-Stokes Computations," *Computers & Fluids*, Vol. 25, No. 2, pp. 217-236, 1996.

7. Thomas, J., Krist, S., and Anderson, W., "Navier-Stokes Computations of Vortical Flows Over Low-Aspect-Ratio Wings," *AIAA Journal*, Vol. 28, No. 2, pp. 205-212, 1990.

8. Vatsa, V., Thomas, J., and Wedan, B., "Navier-Stokes Computations of a Prolate Spheroid at Angle of Attack," *Journal of Aircraft*, Vol. 26, No. 11, pp. 986-993, 1989.

Other References (Partial List)

9. Berry, J., Chaffin, M., and Duque, E., "Helicopter Fuselage Aerodynamic Predictions: Navier-Stokes and Panel Method Solutions and Comparison with Experiment," Amer-

ican Helicopter Society Aeromech Specialists Conference, pp. 3.5-1 - 3.5-20, January 1994.

10. Garriz, J., Vatsa, V., and Sanetrik, M., "Issues Involved in Coupling Navier-Stokes Mean-Flow and Linear Stability Codes," AIAA 94-0304, 1994.

11. Ghaffari, F., Bates, B., Luckring, J., and Thomas, J., "Transonic Navier-Stokes Solutions About a Complex High-Speed Accelerator Configuration," AIAA 90-0430, 1990.

12. Hsieh, T. and Wardlaw, A., "Unsteady Aerodynamics of a Transient Pitching Missile Body from 0 Degrees to 25 Degrees," AIAA 94-3500, 1994.

13. Jones, K., Biedron, R., and Whitlock, M., "Application of a Navier-Stokes Solver to the Analysis of Multielement Airfoils and Wings Using Multizonal Grid Techniques," AIAA 95-1855, 1995.

14. Krist, S. L., *A Grid-Overlapping Technique Applied to a Delta Wing in a Wind Tunnel*, Masters thesis, George Washington University, 1994.

15. Krist, S., Washburn, A., and Visser, K., "A Computational and Experimental Investigation of a Delta Wing with Vertical Tails," AIAA 93-3009, 1993.

16. Lessard, V., "Analysis of a High Speed Civil Transport Configuration at Subsonic Flow Conditions Using a Navier-Stokes Solver," NASA CR 4490, 1993.

17. Londenberg, W., "Transonic Navier-Stokes Calculations About a 65 degree Delta Wing," NASA CR 4635, 1994.

18. Londenberg, W., "Turbulence Model Evaluation for the Prediction of Flows Over a Supercritical Airfoil with Deflected Aileron at High Reynolds Number," AIAA 93-0191, 1993.

19. McMillin, S., Pittman, J., and Thomas, J., "Computational Study of Incipient Leading-Edge Separation on a Supersonic Delta Wing," *Journal of Aircraft*, Vol. 29, No. 2, pp. 203-209, 1992.

20. McMillin, S., Thomas, J., and Murman, E., "Navier-Stokes and Euler Solutions for Lee-Side Flows Over Supersonic Delta Wings - A Correlation With Experiment," NASA TP-3035, 1990.

21. Melnik, R., Siclari, M., Marconi, F., Barber, T., and Verhoff, A., "An Overview of a Recent Industry Effort at CFD Code Certification," AIAA 95-2229, 1995.

22. Milholen, W., Chokani, N., and Al-Saadi, J., "Performance of Three-Dimensional Compressible Navier-Stokes Codes at Low Mach Numbers," *AIAA Journal*, Vol. 34, No. 7, pp. 1356-1362, 1996.

23. Newsome, R., "Navier-Stokes Simulation of Wing-Tip and Wing-Juncture Interactions for a Pitching Wing," AIAA 94-2259, 1994.

24. Rivers, M. and Wahls, R., "Comparison of Computational and Experimental Results for a Supercritical Airfoil," NASA TM-4601, 1994.

25. Robinson, B. and Yeh, D., "Toward Certification for CFD Codes for Aft End/Nozzle Configurations," AIAA 94-2242, 1994.

26. Rudy, D., "Validation Studies for CFL3D, CFL3DE, and GASP Codes," Paper No. 115, 10th National Aero-Space Plane Technology Symposium, April 1991.

27. Rumsey, C., "Computation of Acoustic Waves Through Sliding-Zone Interfaces," *AIAA Journal*, Vol. 35, No. 2, pp. 263-268, 1997; also, "Computation of Acoustic Waves Through Sliding-Zone Interfaces Using an Euler/Navier-Stokes Code," AIAA 96-1752, 1996.

28. Stephens, M., Shih, T., and Civinskas, K., "Computation of Flow and Heat Transfer in a Rectangular Channel with Ribs," AIAA 95-0180, 1995.

29. Tai, T., "Simulation and Analysis of V-22 Tiltrotor Aircraft Forward Flight Flowfield," AIAA 95-0045, 1995.

30. Thomas, J., "An Implicit Multigrid Scheme for Hypersonic Strong-Interaction Flowfields," *Communications in Applied Numerical Methods*, Vol. 8, pp. 683-693, 1992.

31. Yagle, Patrick, *An Investigation of the Effect of Turbulence Model and Boundary Layer Transition on Predicting Vortical Flows on 65 degree Delta Wings at Mach 1.6*, Masters Thesis, George Washington University, September 1995.

Turbulence Model References

$k-\varepsilon$:

32. Abid, R., "Evaluation of Two-Equation Turbulence Models for Predicting Transitional Flows", *International Journal of Engineering Science*, Vol. 31, pp. 831-840, 1993.

Baldwin-Barth:

33. Baldwin, B. and Barth, T., "A One-Equation Turbulent Transport Model for High Reynolds Number Wall-Bounded Flows," NASA TM-102847, August 1990.

Baldwin-Lomax:

34. Baldwin, B. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flow," AIAA 78-257, 1978.

35. Degani, D. and Schiff, L., "Computation of Supersonic Viscous Flows Around Pointed Bodies at Large Incidence," AIAA 83-0034, 1983.

$k-\omega$ (SST):

36. Menter, F., "Improved Two-Equation $k-\omega$ Turbulence Models for Aerodynamic Flows," NASA TM 103975, 1992.

37. Menter, F., "Zonal Two Equation $k-\omega$ Turbulence Models for Aerodynamic Flows," AIAA 93-2906, 1993.

38. Menter, F. and Rumsey, C., "Assessment of Two-Equation Turbulence Models for Transonic Flows," AIAA 94-2343, 1994. Some corrections to this paper are:

Equation (1) should be:

$$\sigma\mu_t = \mu + \sigma\mu_t$$
$$\sigma^*\mu_t = \mu + \sigma^*\mu_t$$

Equation (17) should be: $\Gamma = \max(2\Gamma_3, \Gamma_1)$

Spalart-Allmaras

39. Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aerospatiale*, No. 1, pp. 5-21, 1994.

40. Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," AIAA 92-0439, 1992.

Wilcox $k-\omega$:

41. Wilcox, D., "Reassessment of the Scale Determining Equation for Advanced Turbulence Models", *AIAA Journal*, Vol. 26, No. 11, pp. 1299-1310, 1988.

EASM $k-\omega$ and $k-\varepsilon$ (as eddy-viscosity or as non-equilibrium):

42. Gatski, T. and Speziale, C., "On Explicit Algebraic Stress Models for Complex Turbulent Flows," *Journal of Fluid Mechanics*, Vol. 254, pp. 59-78, 1993.

43. Abid, R., Rumsey, C., and Gatski, T., "Prediction of Nonequilibrium Turbulent Flows with Explicit Algebraic Stress Models," *AIAA Journal*, Vol. 33, No. 11, pp. 2026-2031, November 1995.

44. Gatski, T., "Prediction of Airfoil Characteristics with Higher Order Turbulence Models," NASA TM 110246, April 1996.

45. Girimaji, S., "Fully-Explicit and Self-Consistent Algebraic Reynolds Stress Model," NASA CR 198243, December 1995; also ICASE Report No. 95-82, 1995; also *Theoretical and Computational Fluid Dynamics*, Vol. 8, pp. 387-402, 1996.

References

1. Abdol-Hamid, K., Lakshmanan, B., and Carlson, J., "Application of Navier-Stokes Code PAB3D with k – ε Turbulence Model to Attached and Separated Flows," NASA TP 3480, 1995.

2. Abid, R., "Evaluation of Two-Equation Turbulence Models for Predicting Transitional Flows," *International Journal of Engineering Science*, Vol. 31, pp. 831-840, 1993.

3. Abid, R., Rumsey, C., and Gatski, T., "Prediction of Nonequilibrium Turbulent Flows with Explicit Algebraic Stress Models," *AIAA Journal*, Vol. 33, No. 11, pp.2026-2031, November 1995.

4. Amtec Engineering, Inc., *Tecplot Version 6 User's Manual*, 1993.

5. Anderson, W. K., *Implicit Multigrid Algorithms for the Three-Dimensional Flux Split Euler Equations,* PhD thesis, Mississippi State University, 1986.

6. Anderson, W. K., Thomas, J. L., and Whitfield, D. L., "Multigrid Acceleration of the Flux Split Euler Equations," AIAA 86-0274, 1986.

7. Anderson, W. K., Thomas, J. L., and Whitfield, D. L., "Three-Dimensional Multigrid Algorithms for the Flux-Split Euler Equations," NASA TP 2829, November 1988.

8. Bachalo, W. and Johnson, D., *An Investigation of Transonic Turbulent Boundary Layer Separation Generated on an Axisymmetric Flow Model*, AIAA 79-1479, 1979.

9. Baldwin, B. and Barth, T., "A One-Equation Turbulent Transport Model for High Reynolds Number Wall-Bounded Flows," NASA TM 102847, August 1990.

10. Baldwin, B. and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flow," AIAA 78-257, 1978.

11. Baysal, O., Fouladi, K., and Lessard, V. R., "A Multigrid and Upwind Viscous Flow Solver on 3-D Overlapped and Embedded Grids," *AIAA Journal*, Vol. 29, No. 6, pp. 903-910, 1990.

12. Benek, J. A., Steger, J. L., Dougherty, F. C., and Buning, P. G., "Chimera: A Grid-Embedding Technique," AEDC-TR-85-64, 1986.

13. Brandt, A., "Multi-Level Adaptive Solutions to Boundary-Value Problems," *Mathematics of Computation*, Vol. 31, No. 138, pp. 333-390, 1977.

14. Coakley, T., "Implicit Upwind Methods for the Compressible Navier-Stokes Equations," AIAA 83-1958, July 1983.

15. Cook, P., McDonald, M., and Firmin, M., "Airfoil RAE 2822 – Pressure Distributions, and Boundary Layer Wake Measurements," AGARD AR-138, p. A6, May 1979.

16. Degani, D. and Schiff, L., "Computation of Supersonic Viscous Flows Around Pointed Bodies at Large Incidence," AIAA 83-0034,1983.

17. Dorney, D. J., Davis, R. L., and Sharma, O. P., "Unsteady Multi-Stage Analysis Using a Loosely-Coupled Blade Row Approach," AIAA 95-0179, January 1995.

18. Dring, R. P., Joslyn, H. D., Hardin, L. W., and Wagner, J. H., "Turbine Rotor-Stator Interaction," *Journal of Propulsion and Power*, Vol. 104, pp. 729-742, October 1982.

19. Favre, A., "Equations des Gaz Turbulents Compressibles," *J. Mecanique*, Vol.4, No. 3, pp. 361-390, 1965.

20. Gainer, T. and Hoffman, S., "Summary of Transformation Equations and Equations of Motion Used in Free-Flight and Wind-Tunnel Data Reduction and Analysis," NASA SP-3070, 1972.

21. Girimaji, S., "Fully-Explicit and Self-Consistent Algebraic Reynolds Stress Model," NASA CR 198243, December 1995; also ICASE Report No. 95-82, 1995 and *Theoretical and Computational Fluid Dynamics*, Vol. 8, pp. 387-402, 1996.

22. Hall, E., Topp, D., and Delaney, R., *ADPAC User's Manual*, NASA CR 195472, April 1996.

23. Janus, J., Horstman, H., and Whitfield, D., "Unsteady Flowfield Simulation of Ducted Prop-Fan Configurations," AIAA 92-0521, 1992.

24. Krist, S. L., *A Grid-Overlapping Technique Applied to a Delta Wing in a Wind Tunnel*, Masters thesis, George Washington University, 1994.

25. Krist, S. L., Thomas, J. L., Sellers, III., W. L., and Kjelgaard, S. O., "Embedded Grid Formulation Applied to a Delta Wing," AIAA 90-0429, 1990.

26. Menter, F.,"Improved Two-Equation $k - \omega$ Turbulence Models for Aerodynamic Flows," NASA TM 103975, 1992.

27. Menter, F. and Rumsey, C., "Assessment of Two-Equation Turbulence Models for Transonic Flows," AIAA 94-2343, 1994.

28. Pulliam, T., "Time Accuracy and the Use of Implicit Methods," AIAA 93-3360-CP, 1993.

29. Rai, M., "Three-Dimensional Navier-Stokes Simulations of Turbine Rotor-Stator Interaction; Parts I and II," *Journal of Propulsion and Power*, Vol. 5, No. 3, pp. 307-319, 1989.

30. Ristorcelli, R., "Diagnostic Statistics for the Assessment and Characterization of Complex Turbulent Flows," ICASE Report No. 95-67; also, NASA CR 198221, December 1995.

31. Roe, P., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, 1981.

32. Rumsey, C., Biedron, R., and Thomas, J., "CFL3D: Its History and Some Recent Applications," NASA TM-112861, May 1997; presented at the "Godunov's Method for Gas Dynamics" Symposium, Ann Arbor, MI, May 1-2, 1997.

33. Rumsey, C., Sanetrik, M., Biedron, R., Melson, N. D., and Parlette, E., "Efficiency and Accuracy of Time-Accurate Turbulent Navier-Stokes Computations," *Computers & Fluids*, Vol. 25, No. 2, pp. 217-236, 1996.

34. Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aerospatiale*, No. 1, pp. 5-21, 1994.

35. Spalart, P. and Allmaras, S., "A One-Equation Turbulence Model for Aerodynamic Flows," AIAA 92-0439, 1992.

36. Steinbrenner, J. P., Chawner, J. R., and Fouts, C. L., "The GRIDGEN 3-D Multiple Block Grid Generation System," WRDC-TR-90-3022.

37. Thomas, J. L., Krist, S. L., and Anderson, W. K., "Navier-Stokes Computations of Vortical Flows Over Low-Aspect-Ratio Wings," *AIAA Journal*, Vol. 28, No. 2, pp. 205-212, 1990.

38. Thomas, J. and Salas, M., "Far-Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations," *AIAA Journal*, Vol. 24, No. 7, pp. 1074-1080, 1986.

39. van Leer, B., "Flux Vector Splitting for Euler Equations," *Lecture Notes in Physics*, Vol. 170, pp. 501-512, 1982.

40. van Leer, B., "Upwind-Difference Methods for Aerodynamic Problems Governed by the Euler Equations," *Lectures in Applied Mathematics* 22, pp. 327-336, 1985.

41. van Leer, B., Thomas, J., Roe, P., and Newsome, R., "A Comparison of Numerical Flux Formulas for the Euler and Navier-Stokes Equations," AIAA 87-1104-CP, June 1987.

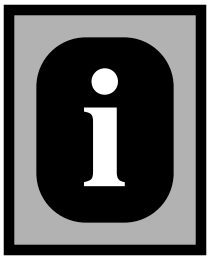
42. Vatsa, V. and Wedan, B., "Development of a Multigrid Code for 3-D Navier-Stokes Equations and Its Application to a Grid-Refinement Study," *Computers & Fluids* 18, 391, 1990.

43. Walatka, P. P., Buning, P. G., Pierce, L., and Elson, P. A., *PLOT3D User's Manual*, NASA TM 101067, March 1990.

44. Walatka, P. P., Clucas, J., McCabe, R. K., and Plessel, T., *FAST User Guide*, RND-92013, 1992.

45. White, F. M., *Viscous Fluid Flow*. McGraw-Hill Book Company, 1974.

46. Wilcox, D., *Turbulence Modeling for CFD*, DCW Industries, Inc., La Canada, California, 1993.



index

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 07704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1998	Technical Memorandum

4. TITLE AND SUBTITLE
CFL3D User's Manual (Version 5.0)

5. FUNDING NUMBERS
WU 522-31-61-02

6. AUTHOR(S)
Sherrie L. Krist
Robert T. Biedron
Christopher L. Rumsey

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER
L-17702

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA/TM-1998-208444

11. SUPPLEMENTARY NOTES
Krist: BaNANePOS, Inc., Hampton, VA; Biedron and Rumsey: Langley Research Center, Hampton, VA
Also available electronically via the world wide web: http://fmad-www.larc.nasa.gov/~rumsey/CFL3D/cfl3d.html

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified–Unlimited
Subject Category 02 Distribution: Standard
Availability: NASA CASI (301) 621-0390

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***
This document is the User's Manual for the CFL3D computer code, a thin-layer Reynolds-averaged Navier-Stokes flow solver for structured multiple-zone grids. Descriptions of the code's input parameters, non-dimensionalizations, file formats, boundary conditions, and equations are included. Sample 2-D and 3-D test cases are also described, and many helpful hints for using the code are provided.

14. SUBJECT TERMS
Computational fluid dynamics; Navier-Stokes; Multiblock; Turbulence models

15. NUMBER OF PAGES
369

16. PRICE CODE
A16

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	