



Edge 3.1

Installation and User Guide
October 2003

FOI, Division of Aeronautics, FFA
Department of Computational Physics
SE-172 90, Stockholm
Sweden

1	<i>Introduction</i>	5
2	<i>Users guide</i>	6
2-1	Installation	6
2-1.1	Introduction	6
2-1.2	Directory structure	6
2-1.3	Compiling	7
2-1.3.1	Setting up the configuration file	7
2-1.3.2	Compiling all programs	7
2-1.3.3	Compiling separate parts	7
2-1.3.4	Compiling with external linking	8
2-1.3.5	How to access executables	8
2-2	Running Edge	9
2-2.1	Steps to take to run Edge	9
2-2.1.1	Setting up an input file	9
2-2.1.2	Specifying boundary conditions	10
2-2.1.3	Running the preprocessor	10
2-2.1.4	Running the flow solver	12
2-2.1.5	Postprocessing	12
2-2.1.6	Running Edge in parallel	12
2-2.1.7	Running the grid adaption	12
2-2.2	List of all programs in Edge	13
2-3	The FFA-format	14
2-4	Generated/required files by Edge	16
2-4.1	The input file Edge.ainp	17
2-4.2	The grid file Edge.bmsh	17
2-4.3	The boundary condition file Edge.aboc	17
2-4.4	The edge file Edge.bedg	18
2-4.5	The solution file Edge.bout (Edge.bini)	19
2-4.6	The solution file for post processing Post.bout	19
2-4.7	The residual file Edge.bres	19
2-5	Hints and recommendations	20
2-5.1	Specifying the free stream viscosity and Reynolds number	20
2-5.2	Turbulent calculations	20
2-5.2.1	Specification of transition	20
2-5.3	Convergence problems	21
3	<i>Theoretical Formulation</i>	22
3-1	Overview	22

3-2	Geometrical considerations	22
3-3	Governing Equations	25
3-3.1	Governing equations for systemrotation	27
3-4	Fluid Modelling	28
3-4.1	Calorically perfect gas.....	28
3-4.2	Thermally perfect gas.....	29
3-5	Turbulence Models	29
3-5.1	Eddy viscosity two-equation models	29
3-5.1.1	The Wilcox standard k-w turbulence model	29
3-5.2	Explicit algebraic Reynolds stress model	30
3-5.3	Numerical treatment.....	30
3-5.3.1	Restriction of the time step	30
3-5.3.2	Free stream and initial conditions	31
3-6	Spatial Discretization	31
3-6.1	Inviscid fluxes	31
3-6.1.1	Central scheme with artificial dissipation.....	31
3-6.1.2	Upwind schemes	34
3-6.2	Viscous fluxes	36
3-7	Time Integration and Multigrid	37
3-7.1	Multigrid strategy.....	37
3-7.2	Multistage Runge-Kutta	39
3-7.3	Implicit residual smoothing	40
3-8	Local Low-Speed Preconditioning.....	41
3-8.1	Preconditioning matrix.....	41
3-9	Time Accurate Calculations	42
3-9.1	Explicit time accurate.....	42
3-9.2	Implicit time accurate.....	43
3-10	Boundary Conditions.....	44
3-10.1	Euler wall and symmetry plane	45
3-10.2	Viscous wall	45
3-10.3	Farfield boundary using characteristics.....	46
3-10.4	Internal inlet conditions.....	47
3-10.5	Internal outlet conditions.....	47
4	<i>Appendix</i>	49
4-1	Input variables	49

4-1.1	General parameters for several programs	49
4-1.1.1	File names	49
4-1.1.2	Parallel calculation	50
4-1.1.3	Integration of forces and moments	50
4-1.2	Preprocessor	50
4-1.3	Flow solver	51
4-1.3.1	Gas related options	51
4-1.3.2	Viscosity related data	52
4-1.3.3	Computation in a relative frame of reference	52
4-1.3.4	Turbulence model parameters	53
4-1.3.5	Free stream/initial solution data/post processing options	54
4-1.3.6	Spatial discretization	54
4-1.3.7	Multigrid options	55
4-1.3.8	Steady state time stepping	56
4-1.3.9	Residual smoothing	56
4-1.3.10	Unsteady time marching	57
4-1.3.11	Low speed preconditioning	57
4-1.3.12	Miscellaneous	57
4-1.4	Grid adaption	58
4-2	Data Sets in FFA-format	58

1 Introduction

Edge is a flow solver for unstructured grids of arbitrary elements. The solver is based on an edge-based formulation and uses a node-centered finite-volume technique to solve the governing equations. The control volumes are non-overlapping and are formed by a dual grid obtained from the control surfaces for each edge, an example is given in Figure 1. All elements are connected through matching faces and no hanging nodes are accepted. The governing equations are integrated explicitly towards steady state with Runge-Kutta time integration. The convergence is accelerated with agglomeration multigrid and implicit residual smoothing.

Edge is divided into several smaller parts where the main parts are

- Preprocessor - converts element information to edge-based information
- Edge flow solver - performs the flow calculations
- Help programs - for specifying boundary conditions, export to different formats, listing files etc.

This guide is divided into a users guide of how to install and use Edge and a theoretical description of the ingoing parts with emphasis on the flow solver. The users guide describes the different parts of the system and explains how to go from a given grid to the preparation of the flow solution to be post processed.

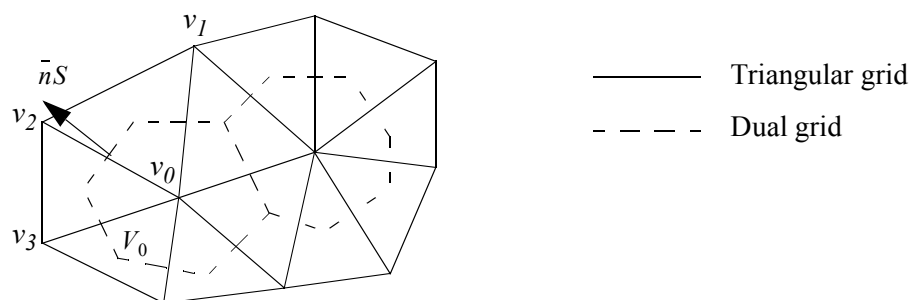


FIGURE 1 An input triangular grid and its dual grid forming the control volume.

2 *Users guide*

2-1 Installation

2-1.1 Introduction

Edge solves the 2D/3D Reynolds averaged Navier-Stokes equations on an unstructured grid. The code is written in FORTRAN 90 and is designed for high performance on modern computer architecture. The code is in its structure similar to the code for structured multiblock meshes EURANUS although the data management is more modern.

The code has been produced by former FFA and by FOI/FFA.

2-1.2 Directory structure

The Edge system is usually obtained as a compressed tar file. This file should be uncompressed and untared at any place found appropriate. In the following the directory where Edge is installed is denoted EDGE_HOME

The following files and directories should be part of the Edge system and are found at EDGE_HOME:

```
bin developers_help doc examples lib Makefile preprocessor
programs README solver util
```

where

- adaption - directory for source code for the adaption
- bin - directory in which scripts and executable files are installed
- developers_help - directory with documents/scripts for those who take part in the development
- doc - directory with relevant documentation like this document
- examples - directory under which some example cases with grids and README files can be found
- INSTALL - a file with the installation guide lines
- lib - directory where libraries with compiled object files and FORTRAN 90 module files are stored

- Makefile - the global make file from which all compiling and installation is done
- preprocessor - directory with the source code for the preprocessor
- programs - directory where the source code for all help programs can be found
- README - a file with the most relevant information to get started, reproduced here
- solver - directory with the source code for flow solver
- util - directory with data handling routines in common for almost all programs
- util_src - directory with common source code routine for some programs

2-1.3 Compiling

The installation of the Edge system is shortly described in the file INSTALL and is repeated here.

The main makefile 'Makefile' normally needs no change. Two variables need to be defined before compiling. These variables are

```
PWD, MACHINE
```

where $\$(PWD)$ is a variable to detect the current directory where the installation will be carried out and is usually defined to the current directory on most systems. It may be defined in the global makefile by an absolute path if not defined. $\$(MACHINE)$ is a variable used to make it possible to install and execute Edge on different platforms by assigning different names to $\$(MACHINE)$. If any of these variables have no value they may be set at the top of the Makefile or in the shell where the compiling will be carried out.

The variable $\$(EDGE_HOME)$ is used to define the installation directory of Edge,

```
EDGE_HOME =  $\$(PWD)$ 
```

2-1.3.1 Setting up the configuration file

On each platform a unique file must be set up containing the variables unique for the specific platform like e.g. name of the compiler, compiler flags, double precision flags, libraries to link with etc. This file is called

```
config/config. $\$(MACHINE)$ 
```

and should contain the definition of all variables needed. These variables are listed in the INSTALL file and are therefore not listed here. This configuration file is included in the main Makefile.

A number of sample configuration files can be found in the config directory. Some of the variables have also default values in the main Makefile.

2-1.3.2 Compiling all programs

All programs including single and double precision sequential versions of the flow solver, the preprocessor and help programs are generated by the command

```
make all
```

which is recommended.

2-1.3.3 Compiling separate parts

It is also possible to compile separate parts of the source. E.g.

```
make edge
```

creates a single and double precision version of the flow solver for sequential computations.

```
make preprocessor
```

creates the preprocessor,

```
make programs
```

creates all help programs.

It is possible to create a single program, e.g.

```
make dir DIR=programs/uvol
```

creates the help program uvol located at `programs/uvol`.

2-1.3.4 Compiling with external linking

2-1.3.4.1 Edge linked to MPI

To run Edge in parallel with the grid blocked into smaller partitions the message passing library MPI is used for the communication between the processors. The MPI specific flags in the configuration file must be properly defined. The parallel MPI versions of Edge are then created by executing

```
make solver_mpi
```

2-1.3.4.2 Conversion programs from/to Tau

There exists two programs to convert from/to the data format used for the German CFD code for unstructured grids, the Tau code. Their format relies on the NetCDF format which is free ware and must be installed on the platform in use. The NetCDF specific flags in the configuration file must be properly defined.

The conversion programs are compiled and installed with the command

```
make programs_netcdf
```

2-1.3.4.3 Conversion programs from/to CGNS

There exists two programs to convert from/to the common data format CGNS. CGNS must be installed on the platform in use in order to be able to install these programs. The CGNS specific flags in the configuration file must be properly defined.

The conversion programs are compiled and installed with the command

```
make programs_cgns
```

2-1.3.5 How to access executables

Usually the directory with the executable files are not in the path of the user. This may be obtained by copying all files in the directory `$(EDGE_HOME)/bin/$(MACHINE)` to a directory publicly available. This can be done from the installation directory `$(EDGE_HOME)` by defining the variables

```
COMDIR=
COMBIN=
```

and the default input file `default_edge.ainp` (discussed below) will be copied to `$(COMDIR)/lib` and all files at `$(EDGE_HOME)/bin/$(MACHINE)` copied to `$(COMBIN)` with the command

```
make install
```

Alternatively the user may modify his path by making the following commands at the `$(EDGE_HOME)` directory

```
make path
```

or

```
make bashpath
```

depending on the shell used. The resulting commands shall be executed in the window where Edge is intended to be used.

2-2 Running Edge

2-2.1 Steps to take to run Edge

To start a computation with Edge all that is necessary is a grid file. The format for that grid file is described below. The steps to take to make a computation are the following:

1. Set up an input file with relevant parameters for your computation
2. Set proper boundary conditions
3. Run the preprocessor
4. Run the flow solver
5. Post processing

Most files, input and output, are written in the ffa-format described in Section 2-3 on page 14. The different files are explained in Section 2-4 on page 16.

2-2.1.1 Setting up an input file

An input file with all global parameters is used to control the execution of all programs available in Edge. The input file contains necessary parameters and controls not only the flow solver but also the preprocessor and many of the help programs. The description below relies on editing the input file with a text editor. The editing may also be done with a GUI if available.

An example input file with default setup can be found at

```
$(EDGE_HOME)/lib/default_edge.ainp
```

and the user should copy this file to his working directory and give it a relevant name and keep the suffix '.ainp'. The most relevant parameters that usually need to be modified are

- File names assigned to the variables CFIMSH, CFIEDG, CFIOUT, CFI INI, CFIBOC, CFIRES
- Free stream values of the primitive variables PFREE,TFEE, UFREE, VFREE, WFREE
- Parameter for an Euler/Navier-Stokes calculation INSEUL
- For a viscous calculation the free stream viscosity RMU and specification for laminar/turbulent calculation ITURB
- Sequential or parallel computation by specifying $NPART \geq 1$

There is a large number of parameters (>100) for different options in the preprocessor, numerical schemes in the flow solver, turbulence models, etc. More parameters than the above mentioned may have to be changed. A complete list of all options for all programs is given in Appendix. The values of the different parameters are given default values that should for most cases work well and need therefore not be adjusted.

It is recommended to use different input files and different file names for computations with different conditions.

2-2.1.2 Specifying boundary conditions

The boundary conditions must be specified for each boundary that is given in the grid. To set the boundary conditions, execute

```
bound
```

and the program will ask for the input file and a specification of a boundary type for each boundary. The available boundary conditions are

```
1->Euler wall
2->Adiabatic wall
3->Isothermal wall
4->Symmetry
5->Farfield
6->Farfield with vortex correction in 2D
7->Total states inlet
8->Pressure outlet
9->Extrapolation
```

where the details of these boundary conditions are given in the theoretical description, Section 3-10 on page 44. For some of the external (not walls) boundary condition a question is asked whether free stream values should be supplied. It is only necessary to supply these values if they deviate from the free stream values give in the input file. If no values are supplied, the values from the input file are used.

The output from this program is written to a file with the name given to the variable CFIBOC.

2-2.1.3 Running the preprocessor

The preprocessor is started with the command

```
preprocessor inputfile
```

In the first step of the preprocessor, the elements are discretized to control volumes surrounding each vertex. The volume elements accepted so far are triangle, quadrilateral, tetrahedron, hexahedron and prismatic element. The enumeration of these elements is depicted in figure Figure 2. The boundary elements are bar, triangle and quadrilateral. The orientation of boundary elements is automatically adjusted by the preprocessor to point outward from the computational domain.

In the second step, the preprocessor fuses control volumes to coarser grid cells, to be used for multigrid. The input parameter NLEVEL determines how many grid levels that are created.

If the input parameter NPART is greater than 1 then a partitioning of the grid is performed, enabling parallel computations on NPART processors.

The input parameter COLTYPE specifies type of edge colouring. The colour property is used to improve the efficiency of the solver in terms of computation speed. Vector colouring or Cache colouring can be selected. Vector colouring is optimal for vector processors. Cache colouring uses the advantages of the fast cache memory on processors with hierarchical memory arrangement. When Cache colouring is selected, the input parameter COLMAX is used to adapt the edge colouring to the size of the cache memory of the current computing platform. Recommended values are COLMAX=100 for a processor with 256 kB cache memory, COLMAX=1000 for 2Mb cache memory per processor and COLMAX=2000 for 4Mb cache memory.

If the input parameter PDUAL is set to 1, insight files for each grid level with the extension.'dgrd' will be created. For 2D grids the dual grid will be plotted and for 3D grids the dual boundary grid will be plotted.

The output filename is given by the variable CFIEDG. For a parallel computation (NPART>1) NPART files are output with extension '_p1', ... '_pNPART' to the file name for each partitioning. Each of the processors in the flow solver will read a separate file, one partition is

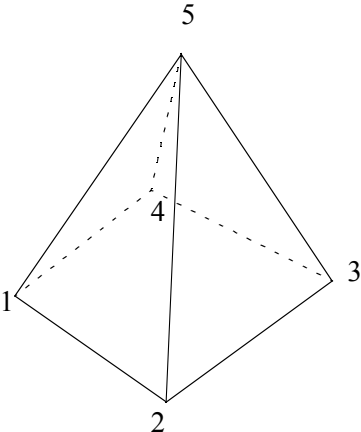
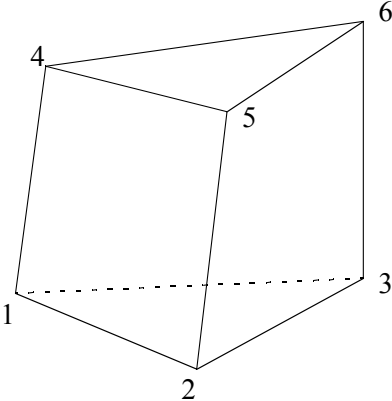
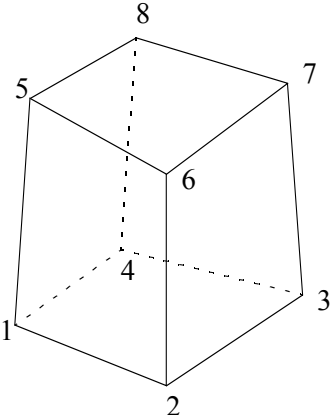
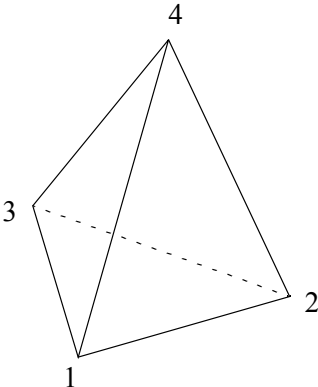
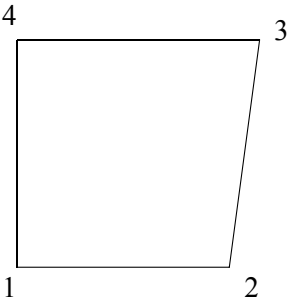
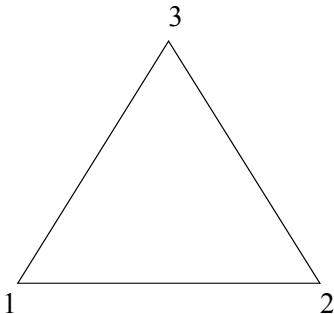


FIGURE 2: Enumeration of corner points for the volume elements: triangle, quadrilateral, tetrahedron, hexahedron prismatic element and pyramid.

2-2.1.4 Running the flow solver

There are four different versions of the flow solver to chose from:

```
edge_run - sequential computation in single precision
edge_dble_run - sequential computation in double precision
edge_mpi_run - parallel computation in single precision
edge_mpi_dble_run - parallel computation in double precision
```

Each of these scripts requires the input file as parameter, the parallel options also require the number of processors/partitions.

2-2.1.5 Postprocessing

There is no postprocessing program contained in the Edge system. There are two programs to export to the commercial flow solver Enight

```
ubody - outputs grid/solution boundary data on Enight 5 format
uvol - outputs grid/solution volume data on Enight 5 format
```

There is a program called

```
pres
```

to output the solution on a boundary in ASCII format on a data file.

The program

```
plotres
```

can be used to monitor the convergence and plots the residual and integrated forces and moment. It requires the open source software XMGR.

2-2.1.6 Running Edge in parallel

A parallel computation can be carried out by splitting the input grid into a desired number of partitions $N_{PART} > 1$. The intention is that a parallel calculation shall produce the same output and be handled in the same way as a sequential calculation, i.e. a user should not notice any difference except for the specification of $N_{PART} > 1$, choice of flow solver script (`edge_mpi_run`, `edge_mpi_dble_run` in stead of `edge_run`, `edge_dble_run`) and the computational speed up.

The preprocessor outputs a file for each partition/processor to be read by each of the slow solver processes. During the iterations, each of the flow solver processes outputs separate solution files that, at the end of the calculation, are merged into a single solution file. The resulting flow solution is therefore the same as the flow solution from a sequential computation.

2-2.1.7 Running the grid adaption

When a solution is obtained the grid can be adapted to the solution with the command

```
hadaption inputfile adapted_mesh.bmsh
```

The original grid is then h-refined to a new output grid. The old cells are divided into smaller cells. The original boundary geometry is only approximately preserved, since the boundary is assumed to be given by the triangular surface grid. In each adaptation run the edges can only be divided once. A tetrahedra is divided into either 2, 4 or 8 new tetrahedron. The edges are divided in two equally pieces if the difference in the flow variables is too large between the two end nodes. The grid adaptation is controlled by four parameters in the input file:

CELMAX - Maximum cell size (dimensional). Edges longer than this value will always be divided into two edges with a new node in the middle.

CELMIN - Minimum cell size (dimensional). Edges shorter than this value will never be divided.

CELFLO - Maximum indicator change in one cell. Normal value is 0.02-0.04. This means that the edges will be divided if the difference in the flow variables, normalized with the free stream values, between the two end nodes is larger than 2-4%.

RMAGNI - Maximum change number of nodes maximum by this factor. The grid adaptation will create a grid with more nodes than the previous grid. This factor sets the maximum allowed number of nodes compared to the previous grid. If the adaptation gives more nodes than this value no new grid will be created. The user can then either increase this value, or increase CELFLO in order to reduce the adaptation sensitivity. If every tetrahedra are refined into 8 new ones RMAGNI must be larger than 8.0.

In order to run the flow solver with the adapted grid the preprocessor must first be executed on the new grid. Update the grid file name, CIFIMSH, in the inputfile and then run the preprocessor.

2-2.2 List of all programs in Edge

Below is a list with all programs available in the Edge distribution. Most of these programs require arguments, in many cases the input file. By executing the program without arguments the required arguments are displayed.

Name	Function
bound	Interactive program to set the boundary conditions
cgns2ffa	Converts grid/solution from CGNS format to ffa format
edge_mpi_dble_run	Starts flow solver in parallel with mpi and double precision
edge_run	Starts flow solver in single precision
edge_dble_run	Starts flow solver in double precision
edge_mpi_run	Starts flow solver in parallel with mpi
ensa2bout	Converts a solution from ensight format to ffa format
ensa2ffa	Converts a grid from ensight format to ffa format
ffa23	Converts a 2d grid and/or solution to 3d by adding a parallel plane in z
ffa2cgns	Converts grid/solution from ffa format to CGNS format
ffa2tau	Converts a 3d grid/solution to the tau format
ffa2tgrid	Converts a grid to the fluent tgrid format
ffa32	Converts a 3d grid with two planes to a 2d grid
ffaa2b	Converts any ASCII ffa-format file to binary

TABLE 1. List of all available programs/scripts in Edge

Name	Function
ffab2a	Converts any binary ffa-format file to ASCII
ffalist	Lists the contents of any file in ffa-format
ffamirror	Mirrors a grid file in its plane(s) of symmetry thus removed
ffanode	Interactive program to print out the coordinates for a given node
ffauct	Cuts a boundary with a plane in a 3D problem
ffauinterp	Interpolates the solution based on the output from ffauct, ffauline
ffauline	Interactive program to extract data along a line
ffaumax	Extracts extreme values of the unknowns and their location
force	Computes forces and moments
merge_input	Updates the input file from the default file. Adds new variable, removes variables not found.
merge_partitions	Merge solution files from a parallel run. This is called from edge_mpi_run
merge_partitions_double	Merge solution files (double precision) from a parallel run. This is called from edge_mpi_dble_run
nastran	Converts a ffa format grid file to Nastran format
quad2tri	Converts a 2d grid with quads to triangles
plotres	XMGR program to plot convergence history
preprocessor	produces dual grid, agglomeration and partitioning for parallel computations.
pres	Extracts data on a boundary column by column
scale	Scales a given grid
split_partitions	Split a solution file into its partitions for a parallel computation. Called from edge_mpi_run
split_partitions_double	Split a solution file (in double precision) into its partitions for a parallel computation. Called from edge_mpi_dble_run
struct2unstruct	Converts a euranus mesh file with boundary conditions to an Edge mesh file
tau2ffa	Converts grid/solution from tau to ffa format
tecbody	Converts a 3d grid and solution to Tecplot format
vrml	Converts a variable to vrml format
ubody	Exports the boundaries of a 3d grid and solution to Enight 5 format
uvol	Exports the interior and boundaries of a grid and solution to Enight 5 format

TABLE 1. List of all available programs/scripts in Edge

2-3 The FFA-format

All data files use a common data format, the FFA (Flexible Format Architecture) format. There are a few exceptions for smaller text files in ASCII format.

The FFA format is hierarchic and is self explained, meaning that the data has a name, type and size specification from which data can be identified.

The basic component in the FFA format is the *data set*. A data set is divided into an identifier and the data. The data set can have one or several sub data sets, where a sub data set is a valid data set. in this way a tree structure is built.

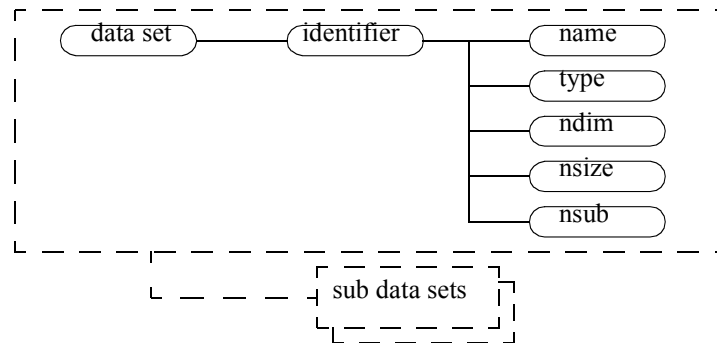


FIGURE 3 Graphical representation of a data set with sub data sets.

The 5 descriptors are:

1. name (character*16) is the name of the data set
2. type (character*4) is the type of the data set given with four characters. The first character defines the type of data like integer or real, the second character defines the type of data set and the third character for field variables will specify where the data is given. The different types are given in Table 2 on page 15 - Table 4 on page 16
3. ndim (integer*4) gives the dimension of data
4. nsize (integer*4) gives the number of data. Ex. a vector field has ndim=3 and nsize=npoints.
5. nsub (integer*4) gives the number of sub data sets.

B	binary*1
I	integer*4
R	real*4
D	real*8
C	complex(4+4)
Z	double complex (8+8)
A	character*1
S	character*16
L	character*72
N	no data

TABLE 2. The different data types specified in type(1:1)

I	descriptor
F	field
B	boundary information
C	constants
T	data tables
L	data list

TABLE 3. The different data types specified in type(2:2)

N	node values
F	face values
C	cell centre values

TABLE 4. The location of a field specified in type(3:3)

The data set identifier on a binary file is written as

```
WRITE (UNIT) CNAME , CTYPE , NDIM , NSIZE , NSUB
```

In binary the data is written in one fortran record:

```
WRITE (UNIT) ( DATA ( JSIZE , JDIM ) , JSIZE=1 , NSIZE ) , JDIM=1 , NDIM )
```

Note that the binary files are not platform independent. For files in ASCII a line starting with * is interpreted as a comment. Note also that one file contains one data set.

The convention for file names is that the suffix is denoted '.b*' for binary files and '.a*' for ASCII files. Any file in ffa-format may be changed from/to ASCII/binary with the conversion programs ffaa2b, ffab2a, see Section 2-2.2 on page 13. Any file in ffa-format may be listed with the program ffafile.

A number of routines to create these data sets, add them as subsets, remove them, copy them, read them from file, write them to file etc. are available in the directory \$(EDGE_HOME)/util and are used by all programs in the Edge system. Due to the structure of the format, many of the routines are recursive.

2-4 Generated/required files by Edge

To start Edge it is sufficient with a grid file. A number of example grid files can be found in the directory \$(EDGE_HOME)/examples. In the table below the required input and generated files are displayed to run the most common programs. The file names are the default names given in the default input file \$(EDGE_MOE)/lib/default_edge.ainp

Files required	Application	Files generated
Edge.ainp Edge.bmsh	bound	Edge.aboc
Edge.ainp Edge.bmsh Edge.aboc	preprocessor	Edge.bedg
Edge.ainp Edge.aboc Edge.bedg (Edge.bini)	Flow solver	Edge.bres Edge.bout Edge.log Post.bout

TABLE 5. Required/generated files from common Edge applications

The files and their format and contents are explained below. The file names may be changed in the input file.

Below each of the ingoing files are explained and examples are given of their contents.

2-4.1 The input file Edge.ainp

The input file Edge.ainp should initially be a copy of the default file from $\$(EDGE_HOME)/lib/default_edge.ainp$ containing default values for all variables. The file is an ASCII file (suffix '.ainp') in ffa-format. The complete list with explanation of all variables is given in Appendix, many variables are also found in the theoretical description.

2-4.2 The grid file Edge.bmsh

A grid file is required to start any application of Edge. The grid file is written in ffa-format and is usually written in binary format (suffix '.bmsh') due to its size. The grid file (in fact any file in ffa-format) may be written in ASCII though and the suffix changed to '.amsh'. An example grid file is listed below.

```

N -      0 x 0  unstr_grid_data - 9
L -      1 x 1  title = "12.2_adapt_1"
RF - 88703 x 2  coordinates = 1.14088404 1.14053142
I -      1 x 1  n_regions = 1
I -      1 x 1  n_boundaries = 4
N -      0 x 0  region - 4
L -      1 x 1  region_name = "volume"
I -      1 x 1  n_element_groups = 2
N -      0 x 0  element_group - 2
L -      1 x 1  element_type = "tria3"
IF - 87235 x 3  element_nodes = 36736
N -      0 x 0  element_group - 2
L -      1 x 1  element_type = "quad4"
IF - 44457 x 4  element_nodes = 1 2 3
N -      0 x 0  boundary - 3
L -      1 x 1  boundary_name = "slat:p"
I -      1 x 1  n_belem_groups = 1
N -      0 x 0  belem_group - 2
L -      1 x 1  bound_elem_type = "bar2"
IF - 371 x 2  bound_elem_nodes = 279 280
N -      0 x 0  boundary - 3
L -      1 x 1  boundary_name = "wing:p"
I -      1 x 1  n_belem_groups = 1
N -      0 x 0  belem_group - 2
L -      1 x 1  bound_elem_type = "bar2"
IF - 591 x 2  bound_elem_nodes = 650
N -      0 x 0  boundary - 3
L -      1 x 1  boundary_name = "flap:p"
I -      1 x 1  n_belem_groups = 1
N -      0 x 0  belem_group - 2
L -      1 x 1  bound_elem_type = "bar2"
IF - 278 x 2  bound_elem_nodes = 1 2 3
N -      0 x 0  boundary - 3
L -      1 x 1  boundary_name = "outer"
I -      1 x 1  n_belem_groups = 1
N -      0 x 0  belem_group - 2
L -      1 x 1  bound_elem_type = "bar2"
IF - 21 x 2  bound_elem_nodes = 46104

```

The list above is obtained from the program fflist and contains the *TYPE* of the data set to the left, the size and dimension $NSIZE \times NDIM$ in the middle and the *NAME* and either the number of subsets *NSUB* or the first data to the right.

The grid file contains a list of coordinates, a list for each element type (triangles tria3 and quadrilaterals quad4) with node numbers referring to the list of coordinates. Each of the boundaries, 4 in the example, should be given a name (boundary_name), number of element types (n_belem_groups), and a type (bound_elem_type) and list of nodes (bound_elem_nodes) for each type.

2-4.3 The boundary condition file Edge.aboc

This file is generated by the program bound and contains, for each boundary, a complete description of the boundary condition to be used. This is a small file and is therefore usually an ASCII file. An example of it is given below where the output from fflist is listed.

```

N -      0 x 0  boundary_data - 4
NB -      0 x 0  boundary - 3
L -      1 x 1  b_name = "slat:p"

```

```

L - 1 x 1 b_class = "wall"
L - 1 x 1 b_type = "weak adiabatic"
NB - 0 x 0 boundary - 3
L - 1 x 1 b_name = "wing:p"
L - 1 x 1 b_class = "wall"
L - 1 x 1 b_type = "weak adiabatic"
NB - 0 x 0 boundary - 3
L - 1 x 1 b_name = "flap:p"
L - 1 x 1 b_class = "wall"
L - 1 x 1 b_type = "weak adiabatic"
NB - 0 x 0 boundary - 3
L - 1 x 1 b_name = "outer"
L - 1 x 1 b_class = "external"
L - 1 x 1 b_type = "weak characteristic"

```

2-4.4 The edge file Edge.bedg

The edge file is produced by the preprocessor. The element information from the grid file has been processed and replaced with edge and node information. For each edge the two nodes are given (edge_nodes) as well as the control surface (edge_surfaces) building up the control volumes. For each boundary the boundary nodes (b_nodes), the most orthogonal inner node (b_inner_nodes) and a control surface is computed. The preprocessor also agglomerates the control volumes to coarser grids so the same type of information (node_f2c) is repeated for each grid level (grid) together with connectivity information between the grids.

In case when a parallel computation is to carried out (NPART>1) then NPART files are produced for each partition/processor with names Edge.bedg_p1,..., Edge_pNPART.

This file is usually given in binary due to its size being larger than the grid file, an example file is listed below. Due to the amount of information the output is truncated and only the fine grid information is given.

```

N - 0 x 0 Preprocessed_grid - 12
L - 1 x 1 title = "12.2_adapt_1"
I - 1 x 1 n_dim = 2
NB - 0 x 0 grid - 15
I - 1 x 1 n_nodes = 88703
DF - 88703 x 2 coordinates = 1.14088404 .
DF - 88703 x 1 volumes = 2.59401833E-10
I - 1 x 1 n_edges = 220397
IF - 220397 x 2 edge_nodes = 1 3 5
DF - 220397 x 2 edge_surfaces = -7.45058060E-08
I - 1 x 1 n_edge_colors = 13
IF - 13 x 2 color_indices = 1 630 1259
I - 1 x 1 n_b_colors = 3
I - 1 x 1 n_bound = 4
NB - 0 x 0 boundary - 5
LB - 1 x 1 b_name = "slat:p"
I - 1 x 1 n_b_nodes = 371
IF - 371 x 1 b_nodes = 279 280 281
DF - 371 x 2 b_surfaces = -1.01195648E-04
IF - 371 x 1 b_inner_nodes = 1519 1520
NB - 0 x 0 boundary - 5
LB - 1 x 1 b_name = "wing:p"
I - 1 x 1 n_b_nodes = 591
IF - 591 x 1 b_nodes = 650 651 652
DF - 591 x 2 b_surfaces = -1.328219664
IF - 591 x 1 b_inner_nodes = 1890 1891
NB - 0 x 0 boundary - 5
LB - 1 x 1 b_name = "flap:p"
I - 1 x 1 n_b_nodes = 278
IF - 278 x 1 b_nodes = 1 2 3
DF - 278 x 2 b_surfaces = -6.41189516
IF - 278 x 1 b_inner_nodes = 1241 1242
NB - 0 x 0 boundary - 5
LB - 1 x 1 b_name = "outer"
I - 1 x 1 n_b_nodes = 21
IF - 21 x 1 b_nodes = 46090 46091
DF - 21 x 2 b_surfaces = 28.4852600
IF - 21 x 1 b_inner_nodes = 88690
IF - 88703 x 1 node_f2c = 1 2 3
.....

```

2-4.5 The solution file Edge.bout (Edge.bini)

The file Edge.bout is generated from the flow solver and contains the nodal values of all primitive variables in the flow solution. The nodal ordering is the same as in the grid file. This file may also be used as an initial solution file (INPRES=1) with the suffix '.bini'. This file is usually a binary file. The contents of the file may vary with the physical model, e.g. Euler/Navier-Stokes, laminar turbulent, steady/unsteady etc.

Below is an example output from a turbulent steady state solution

```

N -      0 x 0  solution - 7
LI -     1 x 1  title = "Edge, Re=9m, alpha=12"
LI -     1 x 1  date  = " 010816 - 11:14:39"
LI -     1 x 1  author = ""
LI -     1 x 1  program = "Edge"
LI -     1 x 1  grid_ref = ""
N -      0 x 0  free_stream_data - 7
DC -     1 x 1  pressure = 100000.000
DC -     1 x 1  temperature = 300.000000
DC -     1 x 3  velocity = 67.8771973
DC -     1 x 1  gamma = 1.39999998
DC -     1 x 1  rgas = 287.000000
DC -     1 x 1  mu_reference = 8.96086385E-06
DC -     1 x 1  prantl_number = 0.720000029
N -      0 x 0  block - 5
DF -    88703 x 1  density = 1.14204168
DF -    88703 x 3  velocity = 0.00000000E+00
DF -    88703 x 1  pressure = 99098.6484
DF -    88703 x 1  turb_kin_energy = 0.0
DF -    88703 x 1  turb_omega = 3.79523712E+09

```

2-4.6 The solution file for post processing Post.bout

The file Post.bout is generated whenever additional data on a separate file is desired (IPOST>0). The format is exactly the same as for the solution file Edge.bout, the contents is determined from the variable IPOST.

2-4.7 The residual file Edge.bres

The residual file Edge.bres is output from the flow solver, it is written together with the flow solution file Edge.bout. This file is intended to monitor the rate of convergence (e.g. with the program plotres) and contents of this file contains residuals, integrated forces and moments plus a few additional quantities. An example file is given below.

```

N -      0 x 0  time_history - 22
IT -     764 x 1  iteration_number = 0 10 20
DT -     764 x 1  Cl = 2.71546450E-02 3.79440784
DT -     764 x 1  Cd = 0.587789536 4.42009068
DT -     764 x 1  Cm = 6.28631338E-02 3.38216138
DT -     764 x 1  cpu_time = 1.63100004 4.12300014
DT -     764 x 1  work_units = 0.00000000E+00
DT -     764 x 1  max_heattransfer = 3.86982224E-09
DT -     764 x 1  max_temperature = 300.000000
DT -     764 x 1  rho_rmsresidual = 6.68303108
DT -     764 x 1  rho_maxresidual = 8.20240974
DT -     764 x 1  rhou_rmsresidual = 8.49785137
DT -     764 x 1  rhou_maxresidual = 10.2165756
DT -     764 x 1  rhov_rmsresidual = 7.83720016
DT -     764 x 1  rhov_maxresidual = 9.57186031
DT -     764 x 1  rhow_rmsresidual = -20.0000000
DT -     764 x 1  rhow_maxresidual = -20.0000000
DT -     764 x 1  rhoe_rmsresidual = 12.1643066
DT -     764 x 1  rhoe_maxresidual = 13.6774025
DT -     764 x 1  rhoq_01_rmsresid = 6.37393808
DT -     764 x 1  rhoq_01_maxresid = 8.08717918
DT -     764 x 1  rhoq_02_rmsresid = 16.0840683
DT -     764 x 1  rhoq_02_maxresid = 18.2894745

```

2-5 Hints and recommendations

2-5.1 Specifying the free stream viscosity and Reynolds number

The free stream viscosity should be specified such that the desired Reynolds number is obtained for a viscous calculation (INSEUL=1). This can be done two ways. If the free stream temperature, pressure (and hence density) and velocity are specified such that the desired free stream Mach number is obtained, the free stream viscosity μ_∞ , (RMU) is computed from

$$Re = \frac{|\bar{U}_\infty| L \rho_\infty}{\mu_\infty} \quad (\text{EQ 1})$$

such that the desired Reynolds number is obtained. Note that L represents a length in the **dimension of the grid** on which the Reynolds number is based. Alternatively the free stream viscosity can be computed from the Sutherland law

$$\mu_\infty = S_1 \frac{T_\infty^{3/2}}{T_\infty + S_2}, S_1 = 1,458 \times 10^{-6}, S_2 = 110 \text{ K} \quad (\text{EQ 2})$$

commonly used for situations where no Reynolds number is given, e.g. internal flows.

The viscosity can vary with the temperature (ICOVIC=0) as

$$\mu = \mu_\infty \left(\frac{T}{T_\infty} \right)^{3/2} \frac{T_\infty + 110}{T + 110} \quad (\text{EQ 3})$$

or it can be constant (ICOVIC=1).

2-5.2 Turbulent calculations

Turbulent calculations may be obtained by setting the input variable ITURB=2, see Appendix. There are several differential two-equation models available. Each of these models has a number of variables not listed in the appendix.

For low speed calculations it is recommended to have a negative value of the variable TURFIX for an upwind discretization of the turbulent equations, TURFIX=-1.1 is recommended. A positive value may add too much dissipation and hence destroy the accuracy.

2-5.2.1 Specification of transition

It is possible to specify transition from version 3.1 onward Edge and onward. The user should then specify the variable NLAMREG to the number of laminar regions in the computational domain. A laminar region is a region of a wall boundary that can be specified by a number of limiting planes (LAMINAR_REGION) defined by a coordinate and a direction into the laminar region for each plane. The laminar region is then defined by the convex region built up by the planes.

It is possible to limit the laminar region close to the wall by the parameter MAXLAMDIST so that turbulent flow is obtained at nodes at larger distances.

Specification of laminar regions is error prone and it is therefore recommended that the user uses the post processing option IPOST=4 in the flow solver to output the array specifying laminar/turbulent flow (0/1 respectively) as well as the distances to the walls.

2-5.3 Convergence problems

The default setting for the numerical parameters usually provides good convergence rates and stable solutions. It may happen though, that the convergence is poor or there might even be cases that diverge.

There may be many reasons behind convergence problems. One of the first tests to find the reason behind it is to lower the CFL numbers (CFL, CFLVIS, CFLREF) to see if this stabilizes and improves the convergence. Another parameter may be to reduce the number of multigrid levels (NGRID) to see if the coarser grids has an overall influence.

It may be that the numerical dissipation is too small so the parameters VIS4, VIS2, VIS0, VIS4KZ, VIS2KZ should be increased. For upwind discretization the parameters LIMITE, ENTRFX, TURFIX are concerned.

The residual smoothing has occasionally caused problems and it may be worthwhile to alter the value of RSMPAR to see the effect. This is rare though.

Another reason behind poor convergence is that the flow is actually unsteady in nature and hence there is no steady state solution. In this cases the unsteady option (ITIMAQ \geq 1) may be considered.

3 *Theoretical Formulation*

3-1 Overview

Edge is a flow solver for unstructured grids of arbitrary elements. Edge solves the Reynolds Averaged Navier-Stokes compressible equations in either a steady frame of reference or in a frame with system rotation. Turbulence can be modelled with differential eddy viscosity models or explicit algebraic Reynolds stress models.

The solver is based on an edge-based formulation and uses a node-centered finite-volume technique to solve the governing equations. The control volumes are non-overlapping and are formed by a dual grid obtained from the control surfaces for each edge. All elements are connected through matching faces. The governing equations are integrated explicitly towards steady state with Runge-Kutta time integration. The convergence is accelerated with agglomeration multigrid and implicit residual smoothing.

Edge contains different spatial discretizations for the mean flow as well as the turbulence, different gas models, steady state and time accurate time integration, low speed preconditioning etc.

3-2 Geometrical considerations

The finite-volume technique requires control volumes surrounding the unknowns in the nodes. The control volumes are non-overlapping and are formed by the dual grid obtained from the control surfaces at the edges. The dual grid is supplied by the preprocessor, Berglind (2000), as an input to the flow solver. The grid with its dual grid is depicted in the example in two dimensions in Figure 4.

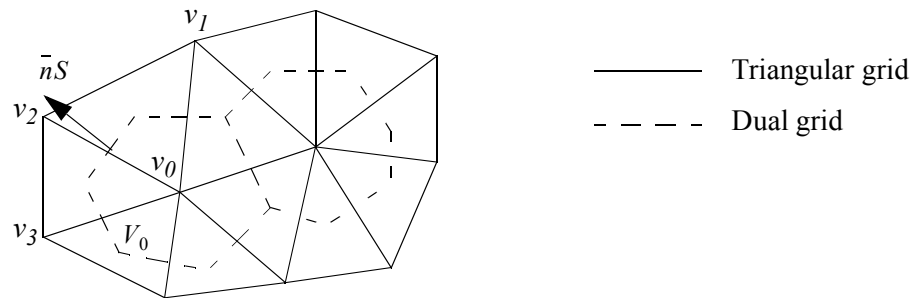


FIGURE 4: The input grid (triangular grid) and its dual grid forming the control volumes.

The coordinates of the input grid are provided for each node and the connectivity is supplied in an edge based manner where an edge connects two nodes. In addition to the node numbers a control surface $\bar{n}S$ is supplied for each edge. The control surfaces of all edges emerging from a node enclose the control volume of a node. The surface vector for each edge is oriented from the node with the first index to the node with the second index. The sum of the surface vectors $\Delta\vec{S}_i$ over a closed volume is equal to the null vector,

$$\sum_i \Delta\vec{S}_i = \vec{0}. \quad (\text{EQ 4})$$

This check is performed in the preprocessor for all control volumes of the dual grid

If each edge surface is considered to lie in a plane, the volume of the cone formed by the edge surface and one of the nodes 0 and k is

$$\frac{1}{2 \text{ndim}} \bar{n}_{0k} S_{0k} \cdot (\bar{x}_k - \bar{x}_0) \quad (\text{EQ 5})$$

where ndim is the space dimension. Quantities with two sub indices denote edge quantities where the indices denote the two nodes connecting the edge.

All edge volumes of a node sum up to the control volume, i.e.

$$V_0 = \frac{1}{2 \text{ndim}} \sum_{k=1}^{n_0} \bar{n}_{0k} S_{0k} \cdot (\bar{x}_k - \bar{x}_0). \quad (\text{EQ 6})$$

n_0 is the number of neighbor nodes to node v_0 or, equivalently, the number of edges connected to node v_0 .

To close the control volumes at boundaries, control surfaces are supplied at the boundary. In Figure 5 the control surfaces to node v_3 are given at all edges connected to v_3 , whereas the edges to the boundary node v_1 do not close the control volume. To close it, a control surface is given separately for the boundary node v_1 .

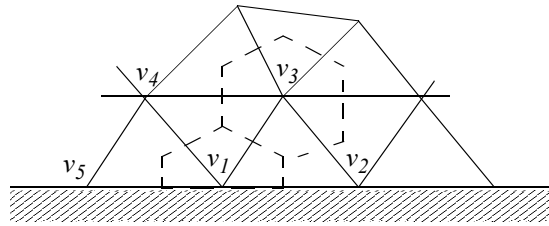


FIGURE 5: Control volumes at an inner node and a boundary node.

At a corner point where two or more boundaries meet the boundary control surface is split into control surfaces for each boundary condition separately. In that case, the boundary node may occur in several boundary conditions.

In addition to the control surface supplied for each boundary node, an inner point is also supplied at all boundaries. The inner nodes is used in some of the boundary conditions described in Section 3-10 on page 44 and the inner node is chosen as the end node of the contiguous edge most orthogonal to the boundary surface. In the example above, node v_3 is the inner node to node v_1 .

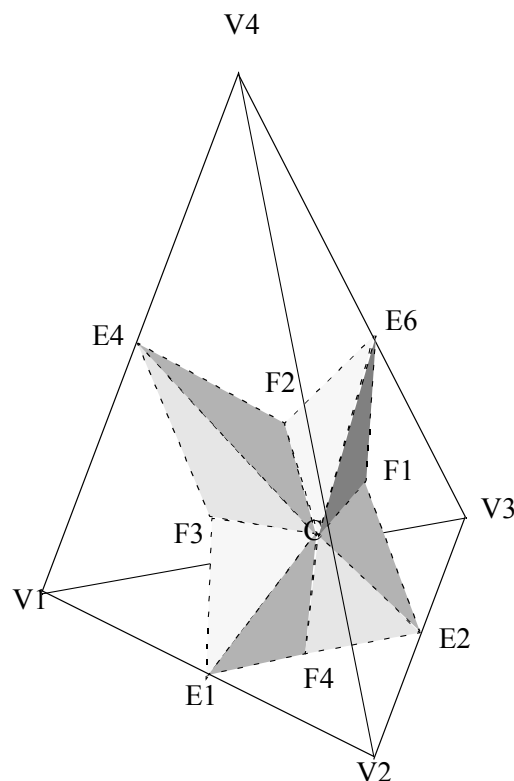


FIGURE 6: Facets for the edges E1, E2, E4 and E6 of a tetrahedron.

In 3D a similar discretization is done. The centroid dual consists of triangular facets between the centroids of the cells, the faces and the edges. The control volume of a grid point consists of faces intersecting the midpoint of each edge. The faces for all edges contiguous to an internal grid point

forms a closed surface. For a boundary grid point additional boundary faces have to be added to the control volume.

Each face associated with an edge consists of several triangular facets, see Figure 6. The facets are defined by the points: the centre of the grid cell C , the centre of a face F and the midpoint of the edge E . In figure Figure 6, the facets associated to the edge $E1$ are $E1-C-F3$ and $E1-C-F4$. If the surface vectors for two adjacent triangles with one common edge are added, the resulting surface vector will be one half of the cross product of the diagonals. The surface vector associated with an edge is computed by adding the surface vectors of the corresponding facets.

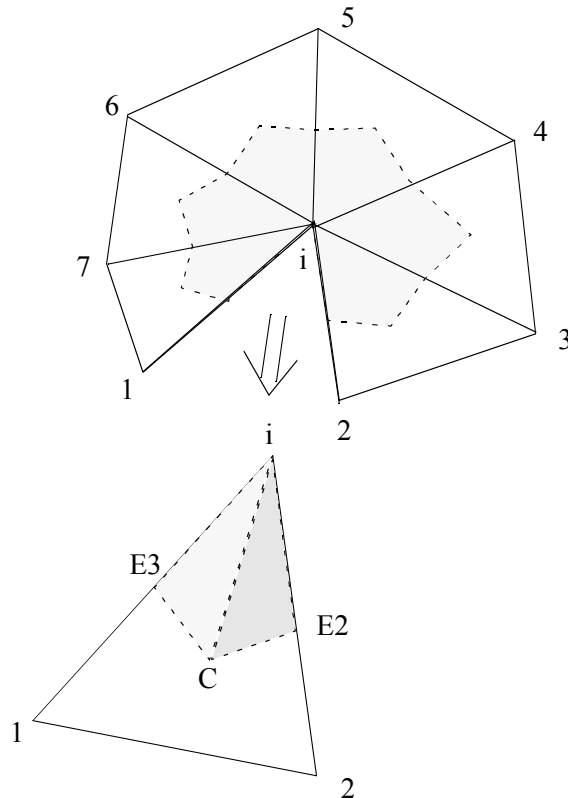


FIGURE 7: Node surface for a boundary control volume.

At the boundaries, the surfaces that close the boundary control volumes have to be computed. In a loop over the boundary surface elements the surfaces for each node is accumulated, see Figure 7. The contribution to node i from the triangle 1-2- i in Figure 7 is the surface vectors for the triangles $C-E2-i$ and $C-i-E3$.

3-3 Governing Equations

The Reynolds-Averaged Navier-Stokes equations written in a Cartesian frame of reference rotating with an angular velocity Ω , can be expressed as

$$\frac{\partial U}{\partial t} + \nabla \vec{F}_I + \nabla \vec{F}_V = Q \quad (\text{EQ 7})$$

F_I and F_V are resp. the inviscid and viscous flux vectors and Q is the vector of source terms.

These equations are obtained by time averaging the Navier-Stokes system, with first-order closure, based on Boussinesq's assumption:

$$-\overline{\rho w_i'' w_j''} = \mu_t \left[\frac{\partial \tilde{w}_i}{\partial x_i} + \frac{\partial \tilde{w}_j}{\partial x_i} - \frac{2}{3} (\nabla \tilde{w}) \delta_{ij} \right] - \frac{2}{3} \bar{\rho} k \delta_{ij} \quad (\text{EQ 8})$$

with w_i the relative velocity component in direction x_i .

The flux vectors resolve into Cartesian components

$$\begin{aligned} F_I &= f_{I1} \vec{1}_x + f_{I2} \vec{1}_y + f_{I3} \vec{1}_z \\ F_V &= f_{V1} \vec{1}_x + f_{V2} \vec{1}_y + f_{V3} \vec{1}_z \end{aligned} \quad (\text{EQ 9})$$

and

$$U = \begin{bmatrix} \bar{\rho} \\ \bar{\rho} \tilde{w}_1 \\ \bar{\rho} \tilde{w}_2 \\ \bar{\rho} \tilde{w}_3 \\ \bar{\rho} \tilde{E} \end{bmatrix} \quad f_{Ii} = \begin{bmatrix} \bar{\rho} \tilde{w}_i \\ \bar{p}^* \delta_{1i} + \bar{\rho} \tilde{w}_i \tilde{w}_1 \\ \bar{p}^* \delta_{2i} + \bar{\rho} \tilde{w}_i \tilde{w}_2 \\ \bar{p}^* \delta_{3i} + \bar{\rho} \tilde{w}_i \tilde{w}_3 \\ (\bar{\rho} \tilde{E} + \bar{p}^*) \tilde{w}_i \end{bmatrix} \quad -f_{Vi} = \begin{bmatrix} 0 \\ \tau_{i1} \\ \tau_{i2} \\ \tau_{i3} \\ q_i + \tilde{w}_j \tau_{ij} + \left(\mu + \frac{\mu_t}{\sigma_k} \right) k_i \end{bmatrix} \quad (\text{EQ 10})$$

where the shorthand notation, i is used to denote derivatives with respect to x_i .

The density and the pressure in (EQ 10) are time averaged values related to the instantaneous value through:

$$q = \bar{q} + q' \quad (\text{EQ 11})$$

where \bar{q} is the time averaged value and q' the fluctuating part and

$$\bar{q}' = 0 \quad (\text{EQ 12})$$

The energy, velocity components and temperature are density weighted averages defined as

$$\tilde{q} = \frac{\overline{\rho q}}{\bar{\rho}} \quad (\text{EQ 13})$$

k is the turbulent kinetic energy and is defined as

$$k = \frac{1}{2} \overline{w_i w_i} \quad (\text{EQ 14})$$

Below all superscripts that denote an averaging are removed for clarity. All variables are supposed to be averaged.

In contrast to the laminar case, both the static pressure and the total energy contain contributions from the turbulence kinetic energy k and are defined as:

$$p^* = p + \frac{2}{3}\rho k \quad (\text{EQ 15})$$

$$E = e + \frac{1}{2}w_i w_i + k \quad (\text{EQ 16})$$

The stresses and the heat fluxes are given by

$$\tau_{ij} = (\mu + \mu_t) \left[\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i} - \frac{2}{3}(\nabla \cdot \vec{w})\delta_{ij} \right] \quad (\text{EQ 17})$$

$$q_i = (\kappa + \kappa_t) \frac{\partial T}{\partial x_i} \quad (\text{EQ 18})$$

The viscosity may either be constant (input variable ICOVIC=1) or may be derived from Sutherland's law (ICOVIC=0). The specification of the free stream viscosity and Reynolds number is described in Section 2-5.1 on page 20.

The laminar thermal conductivity is found from the viscosity and a user specified Prandtl number Pr (variable PRREF of the input file):

$$\kappa = \frac{\mu C_p}{Pr} \quad (\text{EQ 19})$$

The turbulent viscosity follows from the turbulence model. Turbulence models are discussed in Section 3-4.2 on page 29. The turbulent conductivity is obtained from the turbulent viscosity through a turbulent Prandtl number:

$$\kappa_t = \frac{\mu_t C_p}{Pr_t} \quad (\text{EQ 20})$$

3-3.1 Governing equations for systemrotation

By setting the input option IROT = 1 configurations with at constant system rotation can be calculated in a steady state manner. The system rotation is entered through the input option OMGRO which is a the rotation vector Ω giving the magnitude and direction of the rotation. The relation between the relative velocity and the absolute velocity then becomes

$$w = u - \Omega \times r \quad (\text{EQ 21})$$

In order to achieve high accuracy also in the farfield of the computational domain the equations are not solved in the form displayed in (EQ 10). Rather (EQ 21) is substituted into (EQ 10). After some algebraic manipulations the fluxes and source terms take on the following form:

$$U = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho E_r \end{bmatrix} \quad f_{li} = \begin{bmatrix} \rho w_i \\ p^* \delta_{1i} + \rho w_i u_1 \\ p^* \delta_{2i} + \rho w_i u_2 \\ p^* \delta_{3i} + \rho w_i u_3 \\ (\rho E_r + p^*) w_i \end{bmatrix} \quad -f_{vi} = \begin{bmatrix} 0 \\ \tau_{i1} \\ \tau_{i2} \\ \tau_{i3} \\ q_i + w_j \tau_{ij} + \left(\mu + \frac{\mu_i}{\sigma_k} \right) k_i \end{bmatrix} \quad (\text{EQ 22})$$

$$Q = \begin{bmatrix} 0 \\ -\rho \Omega \times u \\ 0 \end{bmatrix} \quad (\text{EQ 23})$$

where $E_r = E - \frac{1}{2}(\Omega \times r)^2$ has a contribution from the centrifugal force which is regarded as a potential. Note that the viscous flux has not changed and that the source term has become simpler. The main advantage of this formulation is that in computing the fluxes the contribution from the system rotation is always obtained at the cell face and never averaged from cell centres. Since the stored variables are the absolute velocities several of the numerical routines must be adopted such that they add the missing part of the relative flow.

3-4 Fluid Modelling

Different modelizations can be chosen (input variable ITYGAS), as explained below.

3-4.1 Calorically perfect gas

The perfect gas law is used as constitutive equation:

$$p = \rho r T \quad (\text{EQ 24})$$

with r the gas constant for the perfect gas under consideration:

$$r = \frac{R}{M} \quad (\text{EQ 25})$$

with R the universal gas constant and M the molecular weight of the perfect gas.

γ is the ratio of specific heats at constant pressure, c_p , and at constant volume, c_v :

$$\gamma = \frac{c_p}{c_v} \quad (\text{EQ 26})$$

When the gas temperature is so low that the vibrational and electronic modes are not excited the internal energy of the gas will be proportional to the temperature and thus the specific heats and γ are constants. The gas is called calorically perfect. This is the usual assumption of moderate speed aerodynamics.

The relation between r and c_p is given by

$$r = \frac{\gamma - 1}{\gamma} c_p \quad (\text{EQ 27})$$

The values of γ and c_p are user input, resp. variables GAMMA and CP.

The static pressure is obtained from the conservative variables through the following relation:

$$p = (\gamma - 1) \left[\rho E - \frac{(\rho w)^2}{2\rho} \right] \quad (\text{EQ 28})$$

3-4.2 Thermally perfect gas

At higher temperatures, the variation of the internal energy with temperature is given by the excitation of the translational, rotational, vibrational and electronic modes of the gas molecules.

Under conditions where no chemical reactions or ionization occur the internal energy of the gas is now a function of temperature only: the gas is thermally perfect.

The perfect gas law is still used as constitutive equation, but the specific heats are now functions of the temperature. These functions are user input, cf. input variables IGTAB,IGTTAB.

The static pressure is now obtained as

$$\pi = \rho r T(e) \quad (\text{EQ 29})$$

with T obtained implicitly from the relation of the internal energy to temperature.

3-5 Turbulence Models

For viscous calculation there are two differential two-equation turbulence models available by switching on the parameter ITURB=2. The choice between the differential models is made with the variable TURB_MOD_NAME.

The available turbulence models are shortly discussed below.

3-5.1 Eddy viscosity two-equation models

In these turbulence models two additional transport equations for the turbulent kinetic energy, k , and some auxiliary quantity (ϵ , ω or τ) is solved. Most of the models below are integrated all the way to the wall which requires a good resolution normal to the wall.

3-5.1.1 The Wilcox standard $k-\omega$ turbulence model

The standard $k-\omega$ turbulence model by Wilcox (1988) is used if TURB_MOD_NAME='std Wilcox k-omega'. The additional equations can be put in the form of (EQ 7) with

$$\begin{aligned}
 U &= \begin{bmatrix} \rho k \\ \rho \omega \end{bmatrix}, & f_{Ii} &= \begin{bmatrix} \rho k w_i \\ \rho \omega w_i \end{bmatrix}, & -f_{Vi} &= \begin{bmatrix} (\mu + \mu_t \sigma^*) k_i \\ (\mu + \mu_t \sigma) \omega_i \end{bmatrix} \\
 Q &= \begin{bmatrix} P_k - \beta^* \rho k \omega \\ \gamma \frac{\omega}{k} P_k - \beta \rho \omega^2 \end{bmatrix}
 \end{aligned} \tag{EQ 30}$$

where P_k , the production of turbulent kinetic energy is given by

$$P_k = \frac{\partial w_i}{\partial x_j} \left(\mu \left[\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i} - \frac{2}{3} (\nabla w) \delta_{ij} \right] - \frac{2}{3} \rho k \delta_{ij} \right) \tag{EQ 31}$$

Eddy Viscosity

$$\mu_T = \rho \frac{k}{\omega} \tag{EQ 32}$$

Auxiliary Relation

$$\varepsilon = \beta^* k \omega \tag{EQ 33}$$

Finally, the closure coefficients are as follow:

$$\gamma = 0,556, \quad \beta = 0,075, \quad \beta^* = 0,09, \quad \sigma = 0,5, \quad \sigma^* = 0,5. \tag{EQ 34}$$

3-5.2 Explicit algebraic Reynolds stress model

The available model is the one by Wallin & Johansson (2000) coupled to the $k-\omega$ turbulence model by Wilcox and is obtained by setting `TURB_MOD_NAME='W&J EARSM + std k-omega'`.

The EARSM may be written in a common way where the anisotropy tensor \mathbf{a} is written in terms of the strain- and rotation rate tensors \mathbf{S} and $\mathbf{\Omega}$ as

$$\mathbf{a} = \sum_{\lambda=1}^{10} \beta_{\lambda} \mathbf{T}^{(\lambda)} \tag{EQ 35}$$

where the β coefficients are functions of the five invariants of \mathbf{S} and $\mathbf{\Omega}$. The \mathbf{T} 's are:

$$\begin{aligned}
 \mathbf{T}^{(1)} &= \mathbf{S}, \\
 \mathbf{T}^{(2)} &= \mathbf{S}^2 - \frac{1}{3} I_S \mathbf{I}, & \mathbf{T}^{(3)} &= \mathbf{\Omega}^2 - \frac{1}{3} I_{\Omega} \mathbf{I}, & \mathbf{T}^{(4)} &= \mathbf{S}\mathbf{\Omega} - \mathbf{\Omega}\mathbf{S}, \\
 \mathbf{T}^{(5)} &= \mathbf{S}^2\mathbf{\Omega} - \mathbf{\Omega}\mathbf{S}^2, & \mathbf{T}^{(6)} &= \mathbf{S}\mathbf{\Omega}^2 + \mathbf{\Omega}^2\mathbf{S} - \frac{2}{3} I_{\Omega} \mathbf{I}, \\
 \mathbf{T}^{(7)} &= \mathbf{S}^2\mathbf{\Omega}^2 + \mathbf{\Omega}^2\mathbf{S}^2 - \frac{2}{3} I_S I_{\Omega} \mathbf{I}, & \mathbf{T}^{(8)} &= \mathbf{S}\mathbf{\Omega}\mathbf{S}^2 - \mathbf{S}^2\mathbf{\Omega}\mathbf{S}, & \mathbf{T}^{(9)} &= \mathbf{\Omega}\mathbf{S}\mathbf{\Omega}^2 - \mathbf{\Omega}^2\mathbf{S}\mathbf{\Omega}, \\
 \mathbf{T}^{(10)} &= \mathbf{\Omega}\mathbf{S}^2\mathbf{\Omega}^2 - \mathbf{\Omega}^2\mathbf{S}^2\mathbf{\Omega}
 \end{aligned} \tag{EQ 36}$$

3-5.3 Numerical treatment

3-5.3.1 Restriction of the time step

Depending on the input variable IUPWKZ either a central scheme (IUPWKZ=0) or an upwind scheme (IUPWKZ=1) is used to discretize the convective terms of the turbulent equations, cf. also Section 3-6 on page 31. The order of the upwind method may be specified. IORDKZ=1 for first order and IORDKZ=2 for second order accuracy.

To control the stability of the calculation, especially in the beginning of the computation, the local time step is restricted by the residual divided by the solution:

$$\Delta t^* = \frac{\Delta t}{1 - \frac{\Delta t}{CFL} \min\left(2\frac{R(\rho k)}{\rho k}, 2\frac{R(\rho \epsilon)}{\rho \epsilon}, \rho^-\right)} \quad (\text{EQ 37})$$

where R is the residual and ρ^- is the spectral radius of the negative source terms corresponding to a point implicit treatment of these source terms. This restriction prevents negative values of the turbulent quantities. A similar restriction is applied in the multigrid cycles when updating the solution from coarse grid corrections, for more information see Eliasson *et al.* (2000).

3-5.3.2 Free stream and initial conditions

Initially, freestream values for k and ω are imposed. A good estimation of these freestream values can be derived from the turbulence level TUFREE (typically 1% or less) and the ratio of freestream turbulent and laminar viscosity VRFREE:

$$k_\infty = \frac{3}{2}(TUFREE \times u_\infty)^2 \quad \mu_{T\infty} = VRFREE \mu_\infty \quad (\text{EQ 38})$$

3-6 Spatial Discretization

The finite volume discretization is obtained by applying the integral formulation of the governing equations in (EQ 7) to the control volume surrounding the unknown U_0 at node v_0 :

$$\frac{\partial}{\partial t}(U_0 V_0) + \sum_{k=1}^{n_0} f_{0k} S_{0k} + \sum_{k=1}^{n_0} f_{V0k} S_{0k} = Q_0 V_0 \quad (\text{EQ 39})$$

where n_0 is the number of neighbors to node v_0 and where $f_{0k} = \vec{F}_0 \vec{n}$, $f_{V0k} = \vec{F}_V \vec{n}$ are the convective and viscous fluxes. The surfaces S_{0k} enclose the control volume for node v_0 and form the dual grid illustrated in Figure 4 in 2D for a given triangulation. The flux vectors f_{0k} and f_{V0k} are computed on the edge consisting of nodes v_0 and v_k where S_{0k} is given, the source term Q_0 is computed directly at the node.

3-6.1 Inviscid fluxes

The schemes for the convective flux f_{0k} considered here are based on a central discretization with dissipation terms of either artificial dissipation type or upwind flux difference splitting type.

3-6.1.1 Central scheme with artificial dissipation

The convective flux across the cell face between nodes v_0 and v_1 is computed as

$$f_{01} = f\left(\frac{U_0 + U_1}{2}\right) - d_{01} \quad (\text{EQ 40})$$

where d_{01} denotes the artificial dissipation. A blend of second and fourth differences are chosen as artificial dissipation, this corresponds to a blend of first and third differences for the fluxes. The following form has been shown to be suitable:

$$d_{01} = (\epsilon_{01}^{(2)}(U_1 - U_0) - \epsilon_{01}^{(4)}(\nabla^2 U_1 - \nabla^2 U_0))\phi_{01}\lambda_{01} \quad (\text{EQ 41})$$

∇^2 denotes the undivided Laplacian operator

$$\nabla^2 U_0 = \sum_{k=1}^{n_0} (U_k - U_0) = -n_0 U_0 + \sum_{k=1}^{n_0} U_k \quad (\text{EQ 42})$$

λ_{01} denotes the local spectral radius

$$\lambda_{01} = (|\bar{u}_{01} \cdot \bar{n}_{01}| + c_{01})S_{01} \quad (\text{EQ 43})$$

where $\bar{u}_{01} = (\bar{u}_0 + \bar{u}_1)/2$ and $c_{01} = (c_0 + c_1)/2$ denote the cell face speed and cell face sound of speed respectively. \bar{n}_{01} denotes the normal direction of the control surface to the edge between nodes v_0, v_1 and S_{01} its size.

ϕ_{01} is a factor introduced to account for the stretching in the grid. It is defined as

$$\phi_{01} = 4 \frac{\phi_0 \phi_1}{\phi_0 + \phi_1} \quad (\text{EQ 44})$$

where ϕ_0 is

$$\phi_0 = \left\{ \frac{\lambda_0}{4\lambda_{01}} \right\}^p \quad (\text{EQ 45})$$

the ratio between the integrated spectral radius where

$$\lambda_0 = \sum_{k=1}^{n_0} (|\bar{u}_{0k} \cdot \bar{n}_k| + c_{0k})S_k = \sum_{k=1}^{n_0} \lambda_{0k} \quad (\text{EQ 46})$$

and where λ_{0k} is given in (EQ 43). $p = 1/2$ is a factor that was chosen to have a close resemblance with the Martinelli eigenvalue scaling for structured grids, Martinelli (1987). This gives a

dissipation proportional to the local spectral radius λ_{01} in the direction of the stretching. In the other directions a value slightly larger than the local spectral radius is obtained.

$\varepsilon_{01}^{(2)}$ is chosen to be active in the neighborhood of shocks and small in smooth regions of the flow:

$$\varepsilon_{01}^{(2)} = \kappa^{(2)} \left(\left| \sum_{k=1}^{n_0} (p_k - p_0) \right| / \sum_{k=1}^{n_0} (p_k + p_0) \right) sc_2 \quad (\text{EQ 47})$$

where $\kappa^{(2)}$ is a constant coefficient provided as input (VIS2) and sc_2 is a scaling factor to reduce the dependency on the number of neighbors. The size of sc_2 is chosen such that for six neighbors the dissipation equals the dissipation in a structured scheme and a unit value is obtained,

$$sc_2 = \frac{12}{n_0 + n_1}. \quad (\text{EQ 48})$$

The fourth difference dissipation is switched off in the vicinity of shocks:

$$\varepsilon_{01}^{(4)} = \max(0, \kappa^{(4)} - \varepsilon_{01}^{(2)}) sc_4 \quad (\text{EQ 49})$$

where $\kappa^{(4)}$ is another user defined constant (VIS4) and sc_4 is a scaling factor chosen in accordance with sc_2 :

$$sc_4 = \frac{36}{(n_0 + n_1)^2} \quad (\text{EQ 50})$$

The turbulent equations have VIS2KZ, VIS4KZ corresponding to VIS2, VIS4.

On coarser grids a simplified form of the artificial dissipation operator based on second differences only is used to save computational time but also to increase the amount of dissipation. The coarse grid operator looks like

$$d_{01} = \varepsilon_{01}^{(0)} (U_1 - U_0) \Phi_{01} \lambda_{01} \quad (\text{EQ 51})$$

where

$$\varepsilon_{01}^{(0)} = \kappa^{(0)} sc_2 \frac{ndim}{3} \quad (\text{EQ 52})$$

and where $\kappa^{(0)}$ is another user defined constant (VIS0).

At a boundary the artificial dissipation should not contribute, i.e. the flux on the boundary is set to zero. In addition, a boundary condition on the second derivatives in the undivided Laplacian (EQ 42) is required. Following Mavriplis the conservative variables are extrapolated linearly which corresponds to a normal second derivative,

$$\frac{\partial^2 U}{\partial n^2} = 0, \quad (\text{EQ 53})$$

Requiring no normal derivative of the variables in the computation of the Laplacian is equivalent to only account for the contributions along the boundary. With the notation in Figure 5 the undivided Laplacian becomes

$$\nabla^2 U_1 = \sum_{k=2,5} (U_k - U_1) = U_2 - 2U_1 + U_5. \quad (\text{EQ 54})$$

The pressure sensor in (EQ 47) is not modified at a boundary, nor are the scaling factors sc_2 and sc_4 . In Figure 5 there are four flux contributions to the residual in node v_1 , d_{12} , d_{13} , d_{14} and d_{15} , the number of legs to node v_1 is $n_1 = 4$.

3-6.1.2 Upwind schemes

In addition to the central scheme, upwind schemes are available of second order accuracy. The upwind scheme is of Roe flux difference splitting type as opposed to the more commonly used MUSCL type upwind schemes. The main reason for this is to have a scheme as similar as possible to what is available in *EURANUS*, the structured counterpart.

As for the central scheme, the convective term is computed as a central part with additional dissipation. The central part is computed as an average of the fluxes through:

$$f_{01} = \frac{1}{2}(f(U_0) + f(U_1)) - d_{01}, \quad (\text{EQ 55})$$

compare (EQ 40).

The upwind dissipation d_{01} is here computed as

$$d_{01} = \frac{1}{2}R\tilde{\Lambda}R^{-1}(U_1 - U_0) = \frac{1}{2}R\tilde{\Lambda}L^{-1}(V_1 - V_0) = \frac{1}{2}R\tilde{\Lambda}dW_{01} \quad (\text{EQ 56})$$

where U, V denote the conservative and the primitive variables respectively, the primitive variables being the ones used in the computer code. $dW_{01} = L^{-1}(V_1 - V_0) = R^{-1}(U_1 - U_0)$ denotes the characteristic variables. The tensor R is the right eigenvector matrix to the flux Jacobian,

$$\frac{\partial f}{\partial q} = R\Lambda R^{-1} \quad (\text{EQ 57})$$

where the diagonal tensor Λ contains the eigenvalues. A similar expression can be obtained for the tensor L belonging to the primitive variables.

The diagonal matrix $\tilde{\Lambda}$ is obtained as

$$\tilde{\Lambda} = |\Lambda^*|(I - \Phi) \quad (\text{EQ 58})$$

where Φ is a diagonal matrix with limiters for second order accuracy. Note that $\Phi = 0$ for a first order scheme.

For a Roe flux difference splitting scheme the components of R, L, Λ must be computed from the Roe averaged variables (IROEAV=1)

$$\begin{aligned}
\rho_{01} &= \sqrt{\rho_0} \sqrt{\rho_1} \\
\bar{u}_{01} &= \frac{\bar{u}_0 \sqrt{\rho_0} + \bar{u}_1 \sqrt{\rho_1}}{\sqrt{\rho_0} + \sqrt{\rho_1}} \\
H_{01} &= \frac{H_0 \sqrt{\rho_0} + H_1 \sqrt{\rho_1}}{\sqrt{\rho_0} + \sqrt{\rho_1}} \\
c_{01}^2 &= (\gamma - 1) \left[H_{01} - |\bar{u}_{01}|^2 \right]
\end{aligned} \tag{EQ 59}$$

However, arithmetic averages usually provide good solutions and are computationally less expensive and may be used as an option (IROEAV=0).

The diagonal matrix Λ^* in (EQ 58) contains the eigenvalues adjusted with an entropy fix to prevent the eigenvalues to become zero and produce unphysical solutions. The following entropy fix is used for each of the eigenvalues:

$$|\lambda_i|^* = \begin{cases} \frac{\lambda_i^2 + \delta^2}{2\delta}, & |\lambda_i| \leq \delta \\ |\lambda_i|, & |\lambda_i| > \delta \end{cases} \tag{EQ 60}$$

where δ is a small fraction of the spectral radius, usually around 5%. The variable ENTRFX determines the size of the entropy fix. Two values for ENTRFX are to be specified. The first one applies to the linear field (the characteristic variables that propagate with speed $\bar{u} \cdot \bar{n}$), the second value to the non-linear field (the characteristic variables that propagate with speed $\bar{u} \cdot \bar{n} \pm c$). For the turbulent equations the corresponding variable is TURFIX, see Appendix.

To achieve second order accuracy (IORDER=2) the limiter Φ in (EQ 58) has to be computed from divided differences of the solution. Here we chose differences of the characteristics to avoid oscillations in the pressure, although it is computationally more expensive.

Gradients of all primitive variables are needed to compute the characteristics in the nodes. The gradient in a node is computed by evaluating the surface integral of the gradient theorem

$$\nabla v_0 = \frac{1}{V_0} \oint_{\partial V_0} v \bar{n} dS \cong \frac{1}{V_0} \sum_{k=1}^{n_0} \frac{1}{2} (v_k + v_0) \bar{n}_{0k} S_{0k} \tag{EQ 61}$$

where v denotes a component of the primitive variables. The node valued characteristics dW_0, dW_1 may be obtained as

$$dW_0 = L_0(\nabla V_0 \cdot (\bar{x}_1 - \bar{x}_0)), \quad dW_1 = L_1(\nabla V_1 \cdot (\bar{x}_1 - \bar{x}_0)) \tag{EQ 62}$$

in addition to the face value dW_{01} in (EQ 56).

However, on a structured Cartesian grid these characteristics correspond to a broad stencil, $dW_i = (W_{i+1} + W_{i-1})/2$ using the expression in (EQ 56). To have a scheme that provides identical results compared to a structured scheme on a regular grid the following expression is used to compute the node valued characteristics:

$$\begin{aligned} dW_0 &= 2L_0(\nabla V_0 \cdot (\bar{x}_1 - \bar{x}_0)) - dW_{01} \\ dW_1 &= 2L_1(\nabla V_1 \cdot (\bar{x}_1 - \bar{x}_0)) - dW_{01} \end{aligned} \quad (\text{EQ 63})$$

which, on a regular Cartesian structured grid corresponds to

$$dW_{01} = dW_{i+\frac{1}{2}} = W_{i+1} - W_i, \quad dW_0 = W_i - W_{i-1}, \quad dW_1 = W_{i+2} - W_{i+1} \quad (\text{EQ 64})$$

i.e. dW_0, dW_1 correspond to the left and right compact differences the for the cell face $i + \frac{1}{2}$ flux.

There are three different limiters that can be used, the minmod limiter, the van Leer limiter and the superbee limiter. The variable LIMITE controls the type of limiter, the minmod limiter is the default and recommend limiter. The minmod function chooses the argument with the smallest amplitude provided all arguments have the same sign. If the signs are different the limiter is zero and the scheme reduces locally to a first order accurate one.

Two different ways of limiting is available. In the first way the minmod limiter is computed as

$$\phi dw_{01} = \text{minmod}(dw_0, dw_{01}, dw_1) \quad (\text{EQ 65})$$

where ϕ, dw denote a component of Φ, dW respectively. This way of limiting is similar to the new family of symmetric TVD schemes used in *EURANUS*, the parameter for this is IARDIS=2.

The second way of limiting is more consistent with the flux difference splitting technique, IARDIS=1. The limiter is then computed depending on the sign of the eigenvalues to the flux Jacobian.

$$\phi dw_{01} = \begin{cases} \text{minmod}(dw_0, dw_{01}), \lambda_i \geq 0 \\ \text{minmod}(dw_{01}, dw_1), \lambda_i < 0 \end{cases} \quad (\text{EQ 66})$$

At boundaries no particular modification to the scheme is made. The numerical flux due to the upwind dissipation is zero.

3-6.2 Viscous fluxes

A compact discretization of the thin-layer approximation (IFULNS=0) or the fully viscous terms (IFULNS=1) is used. A thin-layer approximation only contains the normal derivatives of the viscous terms.

The viscous flux for the momentum equations can be divided as

$$\tau_{ij} n_j = (\tau_{ij} n_j)_{il} + (\tau_{ij} n_j)_{\text{tan}} \quad (\text{EQ 67})$$

where $(\tau_{ij} n_j)_{il}$ contains only normal derivatives and leads to a thin-layer discretization if only this term is considered. A fully viscous approximation is obtained if also the remaining part of the viscous terms $(\tau_{ij} n_j)_{\text{tan}}$ is added. $\bar{n} = (n_x, n_y, n_z) = (n_1, n_2, n_3)$ are the components of the normal.

The thin-layer part can then be formulated as, see Gnoffo (1990)

$$(\tau_{ij} n_j)_{il} = \mu \left(\frac{\partial u_i}{\partial n} + \frac{1}{3} \left(\frac{\partial u_j}{\partial n} n_j \right) n_i \right) \quad (\text{EQ 68})$$

The normal derivatives in (EQ 68) can be approximated on the edges as, Haaslbacher (1999)

$$\frac{\partial \phi_{01}}{\partial n} = \frac{\phi_1 - \phi_0}{|\bar{x}_1 - \bar{x}_0|} \quad (\text{EQ 69})$$

with the notation from Figure 4 and where the normal is directed from node v_0 to node v_1 . With this formulation only two points are involved in computing the normal gradients at the edges and hence automatically leads to a compact second derivative.

By recalling the identity of the Laplace's equation

$$\int_{\Omega} \Delta \phi dV = \oint_{\partial \Omega} \frac{\partial \phi}{\partial n} dS \quad (\text{EQ 70})$$

the following approximation of the Laplace's equation at node v_0 is obtained:

$$\Delta \phi_0 \approx \frac{1}{V_0} \sum_{k=1}^{n_0} \frac{S_{0k}}{|\bar{x}_1 - \bar{x}_0|} (\phi_k - \phi_0) \quad (\text{EQ 71})$$

The remaining parts of the viscous terms $(\tau_{ij} n_j)_{\text{tan}}$ contain gradients which may be added using the Green-Gauss formulation in (EQ 61) and thus a fully viscous approach can be obtained.

3-7 Time Integration and Multigrid

3-7.1 Multigrid strategy

Multigrid is used to speed up the rate of convergence. The preprocessor agglomerates the coarser grids that are read in by the flow solver. The input variable NGRID denotes the number of grid levels to be used in the flow solver.

The multigrid method is based on the FAS approach. V (input parameter MGRSTR=1), W (MGRSTR=2) and F-cycle (MGRSTR=3) can be chosen by the user, or if preferred their saw tooth variant (MGRSTR=4,5,6 for resp. V,W,F saw tooth). In the saw tooth variant no smoothing (i.e. Runge-Kutta solver) will be applied in the coarse-to-fine part of the cycle. As a result, the solution is only prolonged when passing from coarse to fine.

Consider a set of meshes denoted with an index $l = 1, \dots, L$ with L the finest level. The Navier-Stokes problem on the finest level can be written as:

$$\frac{\partial U^L}{\partial t} + N_L(U^L) = 0 \quad (\text{EQ 72})$$

where $N_L(U^L)$ is the spatial discretization of the Navier-Stokes operator on the finest mesh L . The problem is then approximated on coarser levels l as:

$$\frac{\partial U^l}{\partial t} + N_l(U^l) = F_l \quad (\text{EQ 73})$$

with F_l the forcing function, defined recursively as:

$$F_l = N_l(I_{l+1}^l U^{l+1}) + \hat{I}_{l+1}^l [F_{l+1} - N_{l+1}(U^{l+1})] \quad (\text{EQ 74})$$

where I_{l+1}^l and \hat{I}_{l+1}^l represent restriction operators of resp. the unknowns and the residuals. They are defined as:

$$\hat{I}_{l+1}^l R^{l+1} = \sum R^{l+1} \quad (\text{EQ 75})$$

$$I_{l+1}^l U^{l+1} = \frac{\sum \Omega^{l+1} U^{l+1}}{\sum \Omega^{l+1}} \quad (\text{EQ 76})$$

where R^{l+1} is defined as:

$$R^{l+1} = F_{l+1} - N_{l+1}(U^{l+1}) \quad (\text{EQ 77})$$

and Ω represents the cell volume. The summation in (EQ 75) and (EQ 76) is over the fine cells contained within a coarse cell.

After temporal discretization, (EQ 73) becomes:

$$S\Delta U^l + N_l(U^{l(0)}) = F_l \quad (\text{EQ 78})$$

$U^{l(0)}$ is the current solution on mesh l which has to be smoothed. One has:

$$U^{l(0)} = I_{l+1}^l U^{l+1} \quad (\text{EQ 79})$$

ΔU^l is an update of $U^{l(0)}$ and is to be calculated.

S is the smoother. It is the operator that corresponds to the chosen time integration method, in this case an explicit Runge-Kutta time stepping.

Note that the number of smoothing sweeps (i.e. the number of times the Runge-Kutta operator) can be chosen by the user for each grid level (NSWPCF, cf. also below).

Once the solution on the coarsest mesh is smoothed, the coarse-to-fine sweep of the multigrid cycle is initiated. The current solutions on finer grids are updated with the solution on the next coarser level:

$$U^l = U^l + I_{l-1}^l (U^{l-1} - I_{l-1}^{l-1} U^l) \quad (\text{EQ 80})$$

The operator I_{l-1}^l is a prolongation operator, currently of injection type. A similar procedure as in (EQ 37) is used to guarantee positivity for concerned quantities.

In the basic V,W,F cycle (MGRSTR = 1,2,3 resp.) the new solution on the finer mesh is smoothed before proceeding to the next finer level, by solving (EQ 75), with $U^{l(0)} = U^l$. The number of sweeps of the smoothing operator is user input, NSWPCF.

The user has also the possibility to smooth the corrections after prolongation (2nd term in the right-hand-side of (EQ 80)) before adding them to the current finer grid solution, by applying the residual smoothing operator to the corrections. This is governed by the input parameter SMCOR which has the same definition as RSMPAR, cf. Section 3-7.3 on page 40.

The computing cost of a multigrid cycle is reduced by using simplifying assumptions on coarser (i. e. all levels but the finest) grids (input parameter MGSIMP). If MGSIMP=1, a first-order accurate upwind scheme is used on coarser levels (if IUPWIN=1), else a more diffusive central scheme is used on coarser levels (IUPWIN=0).

In order to create a good initial solution full multigrid is also available; it is used if the input parameter IFULMG=1 which is default. The calculations start then on the coarsest mesh. After a residual drop of RESFMG orders of magnitude, the solution is prolonged to the next grid level. Two-grid cycles are now applied until the residual dropped once more RESFMG orders, after which the third grid level is included and three-grid cycles are used. The procedure is repeated until the finest grid is reached. The variable RESFMG is user input. The input variable NFMGCY gives the maximum number of cycles spent in each stage of the Full Multigrid.

3-7.2 Multistage Runge-Kutta

An explicit q-stage Runge-Kutta scheme for the equation

$$\frac{dU}{dt} = F(U) \quad (\text{EQ 81})$$

can be written

$$\begin{aligned} u^1 &= u^n + \alpha_1 \Delta t F(u^n) \\ u^2 &= u^n + \alpha_2 \Delta t F(u^1) \\ &\dots \\ u^q &= u^n + \Delta t F(u^{q-1}) \\ u^{n+1} &= u^q \end{aligned} \quad (\text{EQ 82})$$

The coefficients α_i determine the stability area and the order of accuracy of the Runge-Kutta scheme. They can be chosen in such a way that they suit the problem to be solved.

The local time step is computed for each node v_0 according to

$$\Delta t_0 = \min\left(CFL \frac{V_0}{\lambda_0}, CFLVIS \frac{V_0}{\lambda_{V0}}\right) \quad (\text{EQ 83})$$

where V_0 is the volume and λ_0 is the integrated convective spectral radius according to (EQ 46). λ_{V0} is the corresponding viscous spectral radius

$$\lambda_{V0} = \sum_{k=1}^{n_0} \tilde{\mu}_{0k} 2^{ndim} \frac{S_k^2}{\rho_{0k}} \quad (\text{EQ 84})$$

where $\tilde{\mu}_{0k}$ is the sum of the dynamic and turbulent viscosity in a turbulent calculation on the edge between nodes v_0 and v_k , i.e. $\tilde{\mu}_{0k} = 1/2(\tilde{\mu}_0 + \tilde{\mu}_k)$. CFL and $CFLVIS$ are user input and chosen according the stability region of the Runge-Kutta scheme used.

If the input variable $CFLVIS < 0$ the viscous CFL number, $CFLVIS$, is replaced by the inviscid CFL number, CFL . The actual local time step is then by weighting the inviscid and viscous time step according to a harmonic mean formula:

$$\Delta t_0 = CFL V_0 \frac{\frac{1}{\lambda_0} \frac{1}{\lambda_{r0}}}{\frac{1}{\lambda_0} + \frac{1}{\lambda_{r0}}} \quad (\text{EQ 85})$$

For explicit time accurate calculations (ITIMAQ=2), global time stepping is used, i.e. in all cells the same time step is used, which is, for stability reasons, the minimum of the local time steps.

For steady state calculations a 3-stage, first order accurate scheme which provides good smoothing for both central and upwind schemes is recommended and is default,

$$\alpha_1 = 0,66667, \alpha_2 = 0,66667, \alpha_3 = 1,0. \quad (\text{EQ 86})$$

This scheme allows good smoothing properties at $CFL = 1,5$ with one computation of the artificial viscosity in the first stage. The parameters for the Rung-Kutta scheme the number of stages (NSTAGE) and the coefficients (IRKCO). The first element of IRKCO corresponds to the first coefficient, the 2nd element to the second, and so forth. A maximum of 10 coefficients (i.e. a 10-stage scheme) is foreseen.

The default setting is such that the numerical dissipation is calculated only once in stead of in all Runge-Kutta stages. It is computed in the first stage. This approach reduces the computational cost substantially. The disadvantage is that the dissipative terms need to be stored separately.

The recalculation of the dissipative residuals is governed by the variables ISWV and IBOTH. For a m-stage Runge-Kutta scheme the m first elements of this vector are used, the first element corresponding to the 1st stage, the 2nd element to the 2nd stage and so on. If the element of ISWV is zero it means that the dissipative residual should not be calculated at the corresponding stage, and the latest available dissipative residuals will be used.

Note that for consistency the first value of ISWV should always be one.

3-7.3 Implicit residual smoothing

To increase the maximum time step, implicit residual smoothing is employed. The smoothing is employed for each residual to all equations independently. The smoothing for node v_0 can for the residual of each unknown be written

$$\hat{R}_0 = R_0 + \varepsilon \nabla^2 \hat{R}_0 \quad (\text{EQ 87})$$

where ε is a constant, ∇^2 is the undivided Laplacian defined in Equation (EQ 42). Here \hat{R}_0 indicates a new smoothed residual. This may be written

$$(1 + n_0 \varepsilon) \hat{R}_0 - \varepsilon \sum_{k=1}^{n_0} \hat{R}_k = R_0 \quad (\text{EQ 88})$$

For a structured solver a tridiagonal system of equations is obtained. For the unstructured solver a sparse, diagonally dominant matrix is obtained. The sparse matrix is not inverted exactly, instead a few Jacobi iterations of (EQ 88) are done. This results in the following iterative scheme for the residual smoothing:

$$R_0^{n+1} = \frac{R_0^0 + \varepsilon \sum_{k=1}^{n_0} R_k^n}{1 + \varepsilon n_0}; \quad n \geq 0 \quad (\text{EQ 89})$$

Usually 2 iterations gives sufficient smoothing, the parameter for the number of iterations is NSM-CYC.

Theoretically for structured grids where implicit residual smoothing is combined with the direct solution of tridiagonal systems, the time step can be increased according to the value of ε ,

$$\alpha = \frac{CFL^*}{CFL} \leq \sqrt{4\varepsilon + 1} \quad (\text{EQ 90})$$

where CFL , CFL^* are the CFL numbers of the unsmoothed and smoothed scheme. Typical values using a structured grid is $\alpha = 2$ corresponding a doubling of the CFL number and to $\varepsilon = 0,75$.

Here, where a few Jacobi iterations are carried out, usually

$$\alpha = \frac{CFL^*}{CFL} \leq 1,3. \quad (\text{EQ 91})$$

for good efficiency. This correspond to a value of approximately $\varepsilon \leq 0,2$. The input parameter RSMPAR= α , default RSMPAR=1.3.

On stretched grids only smoothing in the direction of the stretching should be used. The smoothing in the other directions must be reduced or removed. With structured grids this is achieved by letting the coefficient ε be a function of geometric quantities. A similar procedure is used here.

The stretching is accounted for as follows:

$$R_0^{n+1} = \frac{R_0^0 + \varepsilon n_0 \sum_{k=1}^{n_0} R_k^n \Psi_k / \sum_{k=1}^{n_0} \Psi_k}{1 + \varepsilon n_0}; \quad n \geq 0 \quad (\text{EQ 92})$$

where Ψ_k contains the geometric information

$$\Psi_k = \frac{S_{0k}^2}{|x_k - x_0|} \quad (\text{EQ 93})$$

This type of smoothing increases slightly the smoothing in the direction of the stretching and it is removed in the other directions. On a regular grid it reduces to Equation (EQ 89). This smoothing is also applied to the corrections in the multigrid procedure. To account for the stretching the parameter IRESMO=1 should be used.

3-8 Local Low-Speed Preconditioning

3-8.1 Preconditioning matrix

In order to overcome problems with stiffness of Euler and Navier-Stokes equations in low speed flows the low speed preconditioning has been added to the code. It is based on the Turkel's type of preconditioner and Navier-Stokes equations are modified as

$$M\Gamma^{-1}M^{-1}\int_{\Omega}\frac{\partial U}{\partial t}d\Omega + \sum_{faces}(\vec{F}i\vec{n})^*\Delta S + \sum_{faces}(F_V\vec{n})\Delta S = \int_{\Omega}Qd\Omega \quad (\text{EQ 94})$$

where Γ^{-1} is preconditioning matrix for the primitive variables $V = (\rho, u, v, w, p)$

$$\Gamma^{-1} = \begin{bmatrix} -\delta + 1 & 0 & 0 & 0 & \frac{c^2 + \delta\beta^2 - \beta^2}{c^2\beta^2} \\ 0 & 1 & 0 & 0 & \frac{\alpha u}{\rho\beta^2} \\ 0 & 0 & 1 & 0 & \frac{\alpha v}{\rho\beta^2} \\ 0 & 0 & 0 & 1 & \frac{\alpha w}{\rho\beta^2} \\ -\delta c^2 & 0 & 0 & 0 & \frac{c^2 + \delta\beta^2}{\beta^2} \end{bmatrix} \quad (\text{EQ 95})$$

and $M = (\partial U)/(\partial V)$ is the transformation matrix between conservative and primitive variables. The coefficient α varies between -1 and 1, δ is in subsonic flow-field equal to 1 and in supersonic flow field equal to 0. The coefficient β is calculated using

$$\beta^2 = \min(\max(K_1 u_{loc}^2, K_2 u_{inf}^2), c^2) \quad (\text{EQ 96})$$

where u_{loc} is local velocity in the mesh cell. There are, however, the problems with stability especially in the regions of rapid changes of velocity - typical examples could be stagnation point or shock wave - boundary layer interaction. Stability problems in the stagnation point could be removed by increasing K_2 constant resulting in covering this problematic region with constant β .

Low speed preconditioning is turned on by the parameter IPREPA=1. The parameters K_1 and K_2 are set with the parameters RKBET1 and RKBET2 respectively. There is also a global Mach number RM0 above which no preconditioning is used. Usually RM0=1.

Recommended values of constants are

$$\alpha = \text{ALPHA} = 0, K_1 = \text{RKBET1} = 1, K_2 = \text{RKBET2} = 4, \text{RM0} = 1, K_3 = \text{RKBET3} = 0.33$$

3-9 Time Accurate Calculations

There are two ways to perform time accurate calculations, either explicit Runge-Kutta time marching with a global time step or implicit time marching with explicit subiterations.

3-9.1 Explicit time accurate

Explicit time accurate calculation is available by setting the parameter ITIMAQ=2. In that case a global time steps is automatically computed from the minimum local time step in all domains. The calculation of the local time step can be found in (EQ 83) - (EQ 85). A global time step is computed provided the parameter ISGLTS=0 which is default.

Optionally, the parameter ISGLTS=1 and the time step can be set by the user using the parameter DELTAT.

A Runge-Kutta scheme of at least second order accuracy in time is recommended, a good choice is the fourth order accurate, four stage Runge-Kutta with coefficients

$$\alpha_1 = 0,25, \alpha_2 = \frac{1}{3}, \alpha_3 = 0,5, \alpha_4 = 1 \quad (\text{EQ 97})$$

The explicit time accurate option is not compatible with multigrid (NGRID=1) or implicit residual smoothing (RSMPAR<=0).

3-9.2 Implicit time accurate

The implicit solver is written as

$$\frac{\beta_1(UV)^{n+1} + \beta_0(UV)^n + \beta_{-1}(UV)^{n-1}}{\Delta t} + \gamma_1 R(U^{n+1}) + \gamma_0 R(U^n) + \gamma_{-1} R(U^{n-1}) = 0 \quad (\text{EQ 98})$$

where UV denote the unknowns times the volume. The coefficients $\beta_1, \beta_0, \beta_{-1}$ and $\gamma_1, \gamma_0, \gamma_{-1}$ can be chosen to yield desired accuracy and stability. The implicit solver is available as ITIMAQ=1, the parameters $\beta_1, \beta_0, \beta_{-1}$ are available with names BETNP1, BETN, BETNM1 and $\gamma_1, \gamma_0, \gamma_{-1}$ with names GAMNP1, GAMN, GAMNM1 respectively. GAMNM1=0 is no longer an variable.

Introduce the pseudo time τ , denote the dependent variables U^{n+1} by $U^*(\tau)$ and consider the steady state problem

$$V^{n+1} \frac{d}{d\tau} U^* + R^*(U^*) = 0 \quad (\text{EQ 99})$$

where

$$R^*(U^*) = \frac{\beta_1 V^{n+1}}{\Delta t} U^* + \gamma_1 R(U^*) + Q \quad (\text{EQ 100})$$

and

$$Q = \frac{\beta_0(UV)^n + \beta_{-1}(UV)^{n-1}}{\Delta t} + \gamma_0 R(U^n) + \gamma_{-1} R(U^{n-1}) \quad (\text{EQ 101})$$

is a constant source term. As steady state in pseudo time is approached

$$\frac{d}{d\tau}U^* \rightarrow 0 \Rightarrow U^* \rightarrow U^{n+1}. \quad (\text{EQ 102})$$

Within each real time step, the set of ordinary differential equations (EQ 99) is solved using an explicit Runge-Kutta method. To accelerate the convergence, we can adopt the same multigrid strategy combined with local time stepping and implicit residual smoothing as for a steady state calculation.

The local time step will be influenced by the scaling of the residual and the additional term of the dependent variables in (EQ 99). The local time step is obtained as:

$$\Delta\tau_0 = \min\left(\frac{\Delta t_0}{\gamma_1}, CFLVIS\frac{\Delta t}{\beta_1}\right). \quad (\text{EQ 103})$$

where Δt_0 is the original steady state local time step in (EQ 83) or (EQ 85), Δt is the implicit time step (parameter DELTA) specified by the user.

The user can specify any three-level implicit scheme, the best scheme (which is default) is the second order accurate backward difference scheme

$$\beta_1 = \frac{3}{2}, \beta_0 = -2, \beta_{-1} = \frac{1}{2}, \gamma_1 = 1, \gamma_0 = 0, \gamma_{-1} = 0 \quad (\text{EQ 104})$$

which is A-stable and provides good smoothing also for large implicit time steps.

The user must supply a convergence criteria using the coefficient RESTAQ where the inner iterations are interrupted and the solution is advance to the next time level when

$$\log(\max(|R(\rho)|)) < RESTAQ \quad (\text{EQ 105})$$

Other residuals than the density can be specified with the variable NRRES. Note that convergence must be obtained in each time step before the solution can be advance to the next time level. The user is recommended to put a value of RESTAQ such that the integrated forces don't change when the convergence criteria is satisfied. If the convergence fails the time step (DELTA) is most likely too large.

The user can also specify the minimum and maximum number of inner iterations (parameters ITMNAQ and ITMXAQ respectively.)

The different numerical and physical models combine as for a steady state solution, more details of the implicit time accurate method is found in Eliasson *et al.* (1996).

3-10 Boundary Conditions

Implemented boundary conditions in the *Edge* code are:

- Euler wall,
- Symmetry plane,
- Viscous wall, adiabatic or isothermal,
- Farfield boundary using characteristic boundary conditions.
- Internal inlet conditions
- Internal outlet conditions

The Euler wall, symmetry plane, farfield and internal boundary conditions use a weak formulation, i.e. the boundary conditions are imposed through the flux and all unknowns on these boundaries are updated like any interior unknown.

On viscous walls, however, the velocity is imposed strongly through a no-slip condition. In addition, the density is imposed strongly for an isothermal wall. A strong formulation implies that values of a strongly imposed variable on the boundary are explicitly fixed, i.e. they remain at their imposed values and are not considered as unknowns.

The notations from Figure 5 are used in the following. Note that only the central, inviscid terms and the viscous terms that contribute to the fluxes on the boundaries, the dissipative fluxes are set to zero.

3-10.1 Euler wall and symmetry plane

At an Euler wall the normal component of the velocity is zero

$$\bar{u}_1 \cdot \bar{n} = 0 \quad (\text{EQ 106})$$

and hence the inviscid wall flux becomes

$$\dot{f}_1 = \begin{pmatrix} 0 \\ p_1 n_x S \\ p_1 n_y S \\ p_1 n_z S \\ 0 \end{pmatrix}. \quad (\text{EQ 107})$$

The fluxes are added to the residuals for node v_1 . The same boundary condition is used for a symmetry plane. Note that condition (EQ 106) is only implied through the flux, the unknown velocity itself will not necessarily satisfy this condition exactly. This is a consequence of using a weak boundary condition.

3-10.2 Viscous wall

On a viscous wall the velocity is imposed strongly through a no-slip boundary condition

$$\bar{u}_1 = 0 \quad (\text{EQ 108})$$

Both a weak and a strong formulation have been tested and evaluated. The strong formulation has shown better rates of convergence and has therefore been the chosen one. With a strong formulation, the residual of the velocity on the boundary does not need to be solved for since the velocity will be kept constant according to Equation (EQ 108). This also implies that the fluxes for the velocity need not to be computed. In addition to the velocity, for a turbulent calculation, also the turbulent quantities are imposed strongly through

$$k_1 = 0 \quad (\text{EQ 109})$$

$$\omega_1 = \frac{9\mu_1}{\beta\rho_1|\bar{x}_1 - \bar{x}_3|^2} \quad (\text{EQ 110})$$

where the value ω_1 on the boundary is obtained as for structured grids recommended by Hellsten (1998), β is a constant $\beta = 0,075$ and $|\bar{x}_1 - \bar{x}_3|$ is the distance from the wall node v_1 to the closest internal node (v_3) for which $|\bar{x}_1 - \bar{x}_k|$ forms an angle to the boundary that is closest to orthogonal.

At an iso-thermal wall there is a contribution from the viscous terms to the energy equation. The remaining boundary flux becomes

$$f_{E_1} = -\kappa \frac{\partial T}{\partial n_1} \quad (\text{EQ 111})$$

where the gradient on the boundary node is computed as the difference of the temperature in the interior node and the wall node

$$\frac{\partial T}{\partial n_1} = \frac{T_3 - T_1}{|\bar{x}_1 - \bar{x}_3|} \quad (\text{EQ 112})$$

For turbulent calculations there is an additional contribution from the turbulence and the corresponding boundary flux becomes

$$f_{E_1} = -(\kappa + \kappa_T) \frac{\partial T}{\partial n_1} \quad (\text{EQ 113})$$

The wall density is imposed strongly at an isothermal wall through the relation

$$\rho_1 = \frac{p_1}{RT_w} = \frac{(\gamma - 1)E_1}{RT_w} \quad (\text{EQ 114})$$

where T_w is the specified wall temperature.

At an adiabatic wall there is no contribution from the viscous terms to the energy equation at a wall since the temperature gradient is zero,

$$\frac{\partial T}{\partial n_1} = 0 \quad (\text{EQ 115})$$

and hence the boundary flux is zero for both the density and energy equation.

3-10.3 Farfield boundary using characteristics

Characteristic boundary conditions are used at the farfield. These boundary conditions can be used for both subsonic and supersonic in- and outflow where the characteristics are either set from free stream quantities for ingoing characteristics or extrapolated from the interior for outgoing characteristics.

Primitive variables are used and stored in the program since they lead to sparse and computationally less expensive expressions.

Given a set of local primitive variables $V_1 = (\rho, u, v, w, p)^T$ at the boundary a new set of primitive variables \bar{v}_1' have to be computed to be used in the flux evaluation for node v_1 . The characteristic variables are denoted \bar{w} and the relation between the primitive and characteristic variables is

$$V = L W \quad (\text{EQ 116})$$

where L is given in appendix A with its inverse.

By computing characteristics based on both local and free stream primitive variables,

$$W_L = L^{-1} V_L, \quad W_\infty = L^{-1} V_\infty \quad (\text{EQ 117})$$

either local or the free stream characteristics are used depending on the sign of the eigenvalue. The local variables V_L are computed by local extrapolation,

$$V_L = V_1. \quad (\text{EQ 118})$$

and the components of the transformation matrices L, L^{-1} are computed using free stream values and the local surface normal vector,

$$L = L(V_\infty, \bar{n}), \quad L^{-1} = L^{-1}(V_\infty, \bar{n}) \quad (\text{EQ 119})$$

Depending on the sign of the eigenvalue the components of characteristics W_1 can be obtained

$$w_{1i} = \begin{cases} w_{Li} & , \lambda_i > 0 \\ w_{\infty i} & , \lambda_i \leq 0 \end{cases} \quad (\text{EQ 120})$$

where λ_i denotes the i th eigenvalue belonging to the i th characteristic. The variables V_1 can then be obtained from (EQ 116) from which the flux on the boundary can be computed.

3-10.4 Internal inlet conditions

At a subsonic inlet the recommended boundary condition is to specify the total temperature and total pressure as well as the flow angle. A weak formulation is used, only constant values are allowed so far.

3-10.5 Internal outlet conditions

At a subsonic outlet the recommended boundary condition is to specify the static pressure. A weak formulation is used, only a constant values are allowed so far.

References

- Berglind, T. "An Agglomeration Algorithm for Navier-Stokes Grids", AIAA-2000-2254.
- Eliasson P. and Nordström J. (1996) "Computations and Measurements of Unsteady Pressure on a Delta Wing with an Oscillating Flap", Wiley, Proc. to ECCOMAS 1996, Paris, pp. 478-484.
- Eliasson, P. (2001) "Edge, a Navier-Stokes Solver for Unstructured Grids", FOI-R--0298--SE.
- Eliasson, P. and Wallin, S. (2000) "A Positive Multigrid Scheme with Two-Equation Turbulence Models", ECCOMAS 2000, Barcelona.
- Gatski T.B. & Speziale C.G. (1993) 'On explicit algebraic stress models for complex turbulent flows', *J. Fluid Mech.*, **254**.
- Gnoffo, P.A. "An Upwind-Biased, Point-Implicit Relaxation Algorithm for Viscous, Compressible Perfect-Gas Flows", NASA TP-2953, 1990. Hirsch Ch. (1990) 'Numerical Computation of Internal and External Flows. Volume 2', John Wiley & Sons.
- Haaselbacher, A., McGuirk, J., and Page, G. (1999) "Finite Volume Discretization Aspects on Mixed Unstructured Grids", *AIAA Journal*, vol. 37, No. 2.
- Hellsten, A. (1998) "On the solid-wall boundary condition for ω in the $k-\omega$ - type turbulence modes", Helsinki University of Technology, Report No B-50, Series B.
- Jameson A., Schmidt W., Turkel E. (1981) 'Numerical Solutions of the Euler Equations by Finite Volume Methods using Runge-Kutta Time-Stepping Schemes', AIAA-81-1259.
- Jameson A. (1985) 'Transonic Flow Calculation for Aircraft', Lecture Notes in Mathematics, Vol. 1127, Numerical methods in Fluid Dynamics, pp. 156-242, Springer Verlag.
- Lacor C., Zhu Z.W., Hirsch Ch. (1993). 'A new family of limiters within the Multigrid/Multiblock Navier-Stokes Code EURANUS', AIAA-93-5023, AIAA/DGLR 5th Int. Aerospace Planes and Hypersonics Technology Conf., Nov. 30-Dec. 3, München.
- Martinelli (1987) 'Calculation of Viscous Flows with Multigrid Methods', PhD Thesis, MAE Department, Princeton University.
- Rizzi, A., Eliasson, P., Lindblad, I., Hirsch, C., Lacor, C. and Haeuser, J. (1993) "The Engineering of Multiblock \ Multigrid Software for Navier-Stokes Flows on Structured Meshes", *Computers and Fluids*, Vol. 22, pp. 341-367.
- Roe P.L. (1981) 'Approximate Riemann solvers, parameter vectors and difference schemes', *J. Comp. Physics*, Vol. 43, pp. 357-382.
- Wallin, S., and Johansson, A. (2000) "A complete explicit Algebraic Reynolds Stress Model for Incompressible and Compressible Turbulent Flows", *Journal of Fluid Mechanics*, 403, pp 89-132.
- Wilcox D.C. (1988) 'Reassessment of the Scale Determining Equation for Advanced Turbulence Models', *AIAA J.*, Vol. 26, No. 11.
- Yee H.C. (1987) 'Construction of Explicit and Implicit Symmetric TVD Schemes and their Applications', *Journal of Computational Physics*, Vol. 68, pp. 151-179.
- Zhang H.S., So R.M.C., Speziale C.G., Lai Y.G. (1991) 'A Near-Wall Model for Compressible Turbulent Flow', ICASE Report 91-82.

4 *Appendix*

4-1 Input variables

4-1.1 General parameters for several programs

4-1.1.1 File names

The file names are used in several programs and should be specified as desired.

Name	Description	Value	Default
CFIMSH	Mesh file name	*.bmsh, *.amsh	Edge.bmsh
CFIEDG	Edge file name from the preprocessor	*.bedg, *.aedg	Edge.bedg
CFIEDGEPER	Edge file for perturbation mesh	*.bedg, *.aedg	Edgeper.bedg
CFIBOC	Boundary condition file	*.aboc, *.bboc	Edge.aboc
CFIRES	Residual file	*.bres, *.ares	Edge.bres
CFIOUT	Solution output file also used as initial solution for restart (INPRES=2)t	*.bout *.aout	Edge.bout
CFIINI	Initial solution file, used whenINPRES=1	*.bini, *.aini	Edge.bini
CFIPOST	Post processing file name	.bout	Post.bout

4-1.1.2 Parallel calculation

The parameter NPART applies to the preprocessor, the flow solver and the programs that split and

Name	Description	Value	Default
NPART	Number of partitions. NPART>1 specifies a parallel calculation. Used by the preprocessor and flow solver.	≥1	1

merge solution files for the parallel computations.

4-1.1.3 Integration of forces and moments

The first 6 of these variables are used by the flow solver to output lift, drag and moment during the

Name	Description	Value	Default
IDCLP	Direction in which the force is projected to obtain the lift		0 0 1
IDCDP	Direction in which the force is projected to obtain the drag		1 0 0
IDCMP	Direction in which the pitching moment is projected		0 1 0
IXMP	Point on the axis for all moments		0 0 0
SREF	Reference area to non-dimensionalize forces and moments		1
CREF	Reference length for the moment		1
IDCCP	Direction in which the side force is projected		0 1 0
IDCNP	Direction in which the yaw moment is projected		0 0 1
IDCRP	Direction in which the roll moment is projected		1 0 0
BREF	Reference width for the moment		1

iterations. The program 'force' uses all of the parameters to compute a complete set of forces and moments.

4-1.2 Preprocessor

Name	Description	Value	Default
CRDIM	Coarsening ratio for multigrid in each dimension, 2 is recommended.	>1	2
NLEVEL	Maximum number of coarse grids produced by the preprocessor	≥1	5
RMSING	Removes singletons in the agglomeration (1)	0,1	1
NPLAYER	Number of prismatic layers. Used for semicoarsening in the prismatic region and, if semicoarsening is desired, should be put to the maximum number of prismatic layers in the grid	≥1	1
CRATIOB	Coarsening ratio in the prismatic layer when NPLAYER>1. CRATIOB=4 is recommended. If NPLAYER=1 CRATIO apply.	≥1	2
PDUAL	=1 to output the dual grid in Enight 5 format to be plotted	0,1	0

Name	Description	Value	Default
COLTYPE	Type of edge colouring, 0=no colouring, 1=Cache colouring, 2=Vector colouring	0,1,2	1
COLMAX	Number of nodes in each colour (COLTYPE=1)	>0	1000
NLAMREG	Number of laminar regions	≥ 0	0
LAMINAR_SETTING	Data set containing data for all regions	No data	
LAMINAR_REGION	Data set for each laminar region	No data	
PLANE_COO	Planes with coordinates for the laminar region		
PLANE_NORMALS	Normals of the planes with for the laminar region		
MAXLAM-DIST	Maximum distance to the nearest wall for all laminar regions	>0	0.003

4-1.3 Flow solver

4-1.3.1 Gas related options

Name	Description	Value	Default
ITYGAS	Type of gas, 1=calorically perfect gas 2=thermally perfect gas	≥ 1	1
GAMMA	Gamma	>1	1.4
CP	Cp		1004.5
NSPEC	Number of species (ITYGAS=2)	≥ 1	1
SCHMID	Schmidt number for NSPEC>1	>0	0.5
SCHMTU	Turbulent Schmidt number for NSPEC>1	>0	0.9
ITHEDA	Thermally perfect gas option, only one option currently	3	3
RGAS	Gas constant		287
NGTAB	Input table size for thermally perfect gas (ITYGAS=2)	≥ 1	2
IGTTAB	Table for variable temperature (ITYGAS=2)	>0t	0,2000
IGTAB	Table for variable gamma (ITYGAS=2). One dimension per specie required	>1	1.4 1.4
IRTAB	Table for variable gas constant RGAS (ITYGAS=2). One dimension per specie required		287 287

4-1.3.2 Viscosity related data

Name	Description	Value	Default
INSEUL	Viscous calculation (1) or Euler calculation (0)	0,1	0
	Parameters below apply to viscous calculations only		
ICOVIC	Constant viscosity (1), $\mu = \mu_{\infty}$	0,1	0
	or variable viscosity (0) $\mu = \mu_{\infty} \left(\frac{T}{T_{\infty}} \right)^{\frac{3}{2}} \frac{T_{\infty} + 110}{T + 110}$		
RMU	Free stream viscosity μ_{∞}	>0	1.7894e-5
PRREF	Prandtl number		0.72
IFULNS	Full Navier-Stokes (1) or thin layer approximation (0)	0,1	0

4-1.3.3 Computation in a relative frame of reference

Name	Description	Value	Default
IROT	=1 for a computation in a relative frame of reference	0,1	0
OMGROT	Angular velocity and direction for the rotation		0,0,0
OMGCOO	Coordinates for a point on the axis		0,0,0

4-1.3.4 Turbulence model parameters

In addition to the different turbulence models available according to the table below there are spe-

Name	Description	Value	Default
ITURB	=0 for a laminar calculation, =2 for a calculation with a differential turbulence model	0,2	0
TURB_MODEL_NAME (ITURB=2)	Name of the turbulence model	'std Wilcox k-omega' 'W&J EARSM + std k-omega' 'Kok_TNT k-omega' 'W&J EARSM + Kok_TNT k-omega' 'Menter BSL k-omega' 'W&J EARSM + Menter BSL k-omega' 'Menter SST k-omega'	'std Wilcox k-omega'
IUPWKZ	=1 for an upwind discretization of the turbulence, =0 for a central + artificial dissipation discretization	0,1	1
VIS2KZ	Amount of second order artificial dissipation (IUPWKZ=1)	≥0	1
VIS4KZ	Amount of fourth order artificial dissipation (IUPWKZ=1)	≥0	0.05
TURFIX	Entropy fix for the turbulent equations (IUPWKZ=1). Scaled with spectral radius (>1) or the magnitude of the velocity (<1). 1.25 means a lowest value of 25% of spectral radius of the eigenvalue, -1.25 means 25% of velocity	≥1, ≤-1	1.1
IORDKZ	Order of accuracy for the upwind scheme (IUPWKZ=1)	1,2	2

cific parameters for each turbulence model that are not given here. These parameters may be found in the default input file. Note that laminar regions are specified as part of the preprocessing.

4-1.3.5 Free stream/initial solution data/post processing options

Name	Description	Value	Default
INPRES	0=start from free stream. 1=initial solution available, specified in CFIINI, 2=restart from previous solution specified in CFIOUT	0,1,2	0
IPOST	Post processing option, output in file name specified by CFIPOST. Several options can be combined by summation. 0-No output, 1- skin friction, heat transfer 2- Mach number, Cp and energy 4- Distance to the wall and laminar/turbulent field 8- Residual to all unknowns 16- Reynolds stresses	≥ 0	0
PFREE	Free stream pressure	> 0	100000
TFREE	Free stream temperature T_∞	> 0	300
UFREE	Free stream velocity in x-direction		277.75
VFREE	Free stream velocity in y-direction		0.
WFREE	Free stream velocity in z-direction		0.
RMU	Free stream viscosity μ_∞	> 0	1.7894e-5
TUFREE	Free stream turbulence level $k_\infty = \frac{3}{2}(TUFREE \times u_\infty)^2$	> 0	0.001
VRFREE	Viscosity ratio $\mu_{T_\infty} = VRFREE \mu_\infty$	> 0	10
SPMFREE	Free stream values for the species (ITYGAS=2, NSPEC>1)	$\geq 0, \leq 1$	1/NSPEC

4-1.3.6 Spatial discretization

Name	Description	Value	Default
IUPWIN	Central (0) or upwind discretization of mean flow	0,1	0
VIS2	Amount of second order artificial dissipation $\kappa^{(2)}$ for central scheme	≥ 0	0.5
VIS4	Amount of fourth order artificial dissipation $\kappa^{(4)}$ for central scheme	≥ 0	0.05
VIS0	Coarse grid 2nd order dissipation $\kappa^{(0)}$ when simplification are used (MGSIMP=1)	≥ 0	0.1

Name	Description	Value	Default
IARDIS	Type of upwind scheme for the mean flow, 1=flux difference splitting, 2=symmetric TVD	1,2	2.
IORDER	Order of accuracy for upwind scheme for the mean flow	1,2	2.
LIMITE	Type of limiter for the mean flow, 2 values for the linear and non-linear fields respectively. 1=minmod, 2=van Leer, 3=superbee	2*1,2,3	2*1
IROEAV	Roe (1) or arithmetic (0) averages for the mean flow	0,1	1
ENTRFX	Entropy fix for the mean flow, 2 values for the linear and non-linear fields respectively. Scaled with spectral radius (>1) or the magnitude of the velocity (<1). 1.25 means a lowest value of 25% of spectral radius of the eigenvalue, -1.25 means 25% of velocity	2* ≥ 1 , 2* ≤ -1	1.2, 1.05

4-1.3.7 Multigrid options

Name	Description	Value	Default
NGRID	Number of grid levels	≥ 1	4
MGRSTR	Multigrid strategy, 1=V cycles, 2 W, 3 F, 4 V saw tooth cycles, 5 W saw tooth, 6 F saw tooth	≥ 1 , <7	2
IFULMG	Full multigrid (1) or not (0). Can not be used with restart (INPRES=2)	0,1	1
NFMGCY	Maximum number of full multigrid cycles on coarser levels	≥ 1	500.
NSWPFC	Number of sweeps on the different grid levels in coarse to fine sweep. One number for each level starting with the finest	10* ≥ 0	10*1
NSWPFC	Number of sweeps on the different grid levels in fine to coarse sweep. One number for each level starting with the finest	10* ≥ 0	10*1
MGSIMP	Multigrid simplification on coarser grids (1). Uses a first order upwind scheme on coarser grids or a simplified numerical dissipation algorithm	0,1	1
LEVTOP	Top level on which to compute, 1=finestl	≥ 1	1
NINIGR	Level on which the initial solution (INPRES ≥ 1) is given. 1 is the finest level	≥ 1	1
IPROLO	Type of prolongation. 0=injection, 1=account for gradients on coarser grids, 2=first order with limiters	0,1,2	0
IRESTR	Type of restriction, currently only linear	0	0
RESFMG	Convergence criterion for the full multigrid cycling. Iterations stopped when max and rms residual of the logarithm of the current residual (NRRES). Default the density residual.	<0	-4.5

4-1.3.8 Steady state time stepping

Name	Description	Value	Default
ITMAX	Maximum number of iterations	≥ 0	1000
NSTAGE	Number of Runge-Kutta stages	≥ 1	3
IRKCO	Runge-Kutta coefficients		0.66667 0.66667 1.
ISWV	One number for each stage when to compute viscous fluxes (1) or not (0)	10*0,1	1,0,0,...
ISWS	One number for each stage when to compute source terms (1) or not (0)	10*0,1	1,0,0,...
IBOTH	1=ISWV applies to both numerical dissipation and the physical viscous dissipation, 0=ISWV applies only to physical dissipation, Numerical dissipation recalculated in all stages.	0,1	1.
CFL	CFL number	≥ 0	1.5
CFLVIS	Viscous CFL number. Minimum convective/viscous time step chosen. For negative values only CFL used and harmonic mean of the time steps is used	≥ 0	-1
CFLREF	CFL reduction factor on coarser grids. CFL and CLFVIS are reduced with $CFLREF^{(IGRID-1)}$ where IGRID ≥ 1 is the grid level, 1 being the finest level	≥ 0	0.8
RESRED	Convergence criterion. Iterations stopped when max and rms residual of the logarithm of the current residual (NRRES). Default the density residual.	<0	-5.5

4-1.3.9 Residual smoothing

Name	Description	Value	Default
RSMPAR	Implicit residual smoothing parameter, $\alpha = CFL^*/CFL > 1$ to be active		1.3
IRESMO	Type of residual smoothing. 0=standard smoothing with constant coefficients, 1=account for grid stretching	0,1	1
NSMCYC	Number of Jacobi smoothing cycles	≥ 1	2.
SMCOR	Smoothing of corrections in multigrid. Defined as RSM-PAR and IRESMO=0 is applied for this smoothing		1.5.

4-1.3.10 Unsteady time marching

Name	Description	Value	Default
ITIMAQ	Time stepping option. 0=steady state marching, 1=time accurate dual time stepping, 2=explicit time accurate	0,1,2	0
ITMAX	Number of time steps	≥ 0	1000
ISGLTS	0=use global time step DELTAT when ITIMAQ=2, 1=compute it when ITIMAQ=2	0,1	0
DELTAT	Time step when ITIMAQ ≥ 1		0.005.
RESTAQ	Convergence criterion for the inner iteration in dual time stepping (ITIMAQ=1). The logarithm of the maximum norm of the chosen residual (NRRES) is used, default is the density residual.		1
ITMXAQ	Maximum number of inner iterations	≥ 1	100
ITMNAQ	Minimum number of inner iterations	≥ 1 , \leq ITMXAQ	3
BETNP1, BETN, BETNM1	The three components for the unknown in the implicit dual time stepping. Default 2nd order backward difference.		1.5, -2, 0.5
GAMNP1, GAMN	The two components for the residual in the implicit dual time stepping. Default 2nd order backward difference.		1,0

4-1.3.11 Low speed preconditioning

Name	Description	Value	Default
IPREPA	Turns on the low speed preconditioning (1)	0,1	0
RKBET2	The parameter β^2	≥ 0	4
RM0	The parameter M_0	> 0	1.

4-1.3.12 Miscellaneous

Name	Description	Value	Default
IWRSOL	Output solution and residual file every IWRSOL iteration	≥ 1	500
IMULOU	Outputs a unique solution (1) every IWRSOL iteration by adding the iteration number to the suffix of the output file specified in CFIOUT	0,1	0
EPS	Small number to avoid division with zero		1.e-28

Name	Description	Value	Default
ISTOUT	Output convergence data to standard output every ISTOUT iteration	≥ 1	1
NRRES	Residual for which convergence is monitored in RESRED, RESFMG, RESTAQ	$\geq 1, \leq$ number of equations	1 (density)

4-1.4 Grid adaption

Name	Description	Value	Default
CELMAX	Maximum cell size		0.7
CELMIN	Minimum cell size		0.1
CELFLO	Maximum indicator change in one cell		0.05
RMAGNI	Change number of cells maximum by this factor		4.0

4-2 Data Sets in FFA-format

Below follows a list of some of the data sets used to define the input and output data to most of the files in both Edge and Euranus.

Data set	Type	Dim	Size	nsub	Explanation
grid	N	0	0	?	grid data set, contains the mesh
boundary_data	N	0	0	?	data set with boundary data
solution	N	0	0	?	contains the solution
time_history	N	0	0	?	time history, the data set that contains the residuals
time	N	0	0	?	grid solution etc. at a specific time
grid_level	N	0	0	?	Different grid levels (multi grid)
block	N	0	0	?	A block (structured (=domain) or unstructured)
free_stream_data	N	0	0	?	free stream data

TABLE 6. Directory data sets

Data set	Type	Dim	Size	nsub	Explanation
boundary_data	N	0	0	?	data set with boundary data
connectivity	IB	1	8	?	Connectivity between two boundaries
rot_symmetry	IB	1	8	?	Symmetry from rotation around an axis

TABLE 7. Boundary condition data sets

Data set	Type	Dim	Size	nsub	Explanation
trans_symmetry	IB	1	8	?	Symmetry from translation
mirror	IB	1	6	?	mirror boundary, no flow through
pol_sing_line1	IB	1	6	?	Polar singular line, a collapsed boundary
wall	IB	1	6	?	Wall boundary
external	IB	1	6	?	External boundary
boundary_type	SI	1	1	0	type of boundary given as data, given as sub data set to the boundary conditions above

TABLE 7. Boundary condition data sets

Data set	Type	Dim	Size	nsub	Explanation
density	RFC	1	?	0	Density
velocity	RFC	3	?	0	Velocity
pressure	RFC	1	?	0	Static pressure
temperature	RFC	1	?	0	Static temperature
energy	RFC	1	?	0	Total energy
turb_kin_energy	RFC	1	?	0	Turbulent kinetic energy (k)
turb_dissipation	RFC	1	?	0	Turbulent rate of dissipation (ϵ)
turb_time_scale	RFC	1	?	0	Turbulent time scale (τ)
density_species	RFC	NSP	?	0	Density for the NSP species
vibr_energy	RFC	NVI	?	0	The NVI vibrational energies
vibr_temperature	RFC	1	?	0	The vibrational temperature

TABLE 8. Field data sets that might occur in the solution file

The real field data sets above could also be given in double precision (8 bytes/real) with type D**.

Data set	Type	Dim	Size	nsub	Explanation
iteration_number	IT	1	?	0	Iteration number
Cl	RT	1	?	0	Non-dimensional lift
Cd	RT	1	?	0	Non-dimensional drag
Cm	RT	1	?	0	Non-dimensional moment
cpu_time	RT	1	?	0	CPU time
work_units	RT	1	?	0	Work units
max_heattransfer	RT	1	?	0	Maximum heat transfer
max_temperature	RT	1	?	0	Maximum temperature
total_time	RT	1	?	0	Time elapsed for time accurate calculation
inner_iterations	IT	1	?	0	# of inner iteration when time accurate (ITIMAQ=1)
rho_rmsresidual	RT	1	?	0	Density rms residual
rho_maxresidual	RT	1	?	0	Density max residual
rho_rmsresidual	RT	1	?	0	x-momentum rms residual

TABLE 9. Data sets that may occur in the residual file

Data set	Type	Dim	Size	nsub	Explanation
rhou_maxresidual	RT	1	?	0	x-momentum max residual
rhov_rmsresidual	RT	1	?	0	y--momentum rms residual
rhov_maxresidual	RT	1	?	0	y-momentum max residual
rhov_rmsresidual	RT	1	?	0	z-momentum rms residual
rhov_maxresidual	RT	1	?	0	z-momentum max residual
rhoe_rmsresidual	RT	1	?	0	energy rms residual
rhoe_maxresidual	RT	1	?	0	energy max residual
rhok_rmsresidual	RT	1	?	0	turbulent energy rms residual
rhok_maxresidual	RT	1	?	0	turbulent energy max residual
rhoep_rmsresidual	RT	1	?	0	turbulent dissipation rms residual
rhoep_maxresidual	RT	1	?	0	turbulent dissipation max residual
rhoet_rmsresidual	RT	1	?	0	turbulent time scale rms residual
rhoet_maxresidual	RT	1	?	0	turbulent time scale max residual
rhos_rmsresidual	RT	NSP	?	0	species rms residual
rhos_maxresidual	RT	NSP	?	0	species max residual
evs_rmsresidual	RT	NVI	?	0	electronic vibration rms residual
evs_maxresidual	RT	NVI	?	0	electronic vibration max residual
eel_rmsresidual	RT	1	?	0	electronic energy rms residual
eel_maxresidual	RT	1	?	0	electronic energy max residual

TABLE 9. Data sets that may occur in the residual file