Abstract

ABSTRACT GOES HERE ...

Contents

1 Introduction

1.1 TAU beginner's guide. Motivation and objective

The DLR TAU code is a very powerful tool for industrial aerodynamics. It is designed for high-end applications in research and industry. On the one hand, this requires both a profound *general knowledge in computational fluid dynamics* covering *(i)* numerical methods, *(ii)* turbulence and transition modelling, and *(iii)* grid design, which has to be achieved in university or from textbooks. On the other hand, *TAU code specific expertise* is necessary.
Due to the complexity of the DLR TAU code, a step-by-step tutorial seems beneficial written for "newbies", i.e., people who are new in the TAU business. This concerns *(a)* people who have already experience with another CFD code and now are starting with TAU, or *(b)* people who only have theoretical knowledge in CFD. The DLR TAU code is used Europe-wide as unstructured CFD solver. Therefore CFD-engineers who have been working so far with their former code now have to be aquainted with TAU. Moreover, the DLR TAU code is used more and more as a platform for national and european research projects in aerodynamic. This additionally gives rise to the need of a TAU tutorial for students and researchers.
The present TAU beginner's guide gives an introduction to the basic usage of the TAU code. It is not a stand alone document, but has to be used in combination with other documentation for the DLR TAU code available

- TAU user guide: Complete description of all parameters for DLR TAU code
- TAU technical documentation: Description of algorithms of DLR TAU code
- TAU best practice guide (work in progress): Recommendations for best usage of TAU code regarding transition and turbulence modelling, parameters for numerical method, and grid design

1.2 How to use this tutorial

The TAU beginner's guide is intended to be used as a tutorial. Together with this document, a collection of example test cases is provided. This gives a step-by-step introduction to the DLR TAU code. The TAU parameters are explained and suggestions for further reading in the technical documentation and/or in the user guide are given.
The example testcases can be found in the corresponding directories, which contain parameter file, grid and experimental data (if needed).

Table 1.1: Example test cases for TAU step-by-step tutorial

Testcase	Lessons learned
NACA0012	TAU preprocessing and basic solver
RAE2822	Transonic flow. numerical method (Flux solver, linear solver, multigrid) RANS turbulence modelling Comparison with exp. data
ONERA A-Airfoil	Subsonic flow Numerical method (Preconditioning, flux solver) Transition
NACA0012_unst1	Unsteady calculation 1: time-independent angle of attack, but unsteady flow due to large angle of attack
NACA0012_unst2	Unsteady calculation 2: forced harmonic pitch oscillation
NACA0021	Detached-eddy simulation
RAE2822_adap	grid adaptation

1.3 Steps for a CFD simulation

For a numerical simulation of 2D airfoil flows, the following steps have to be performed, see Figure 1.1.

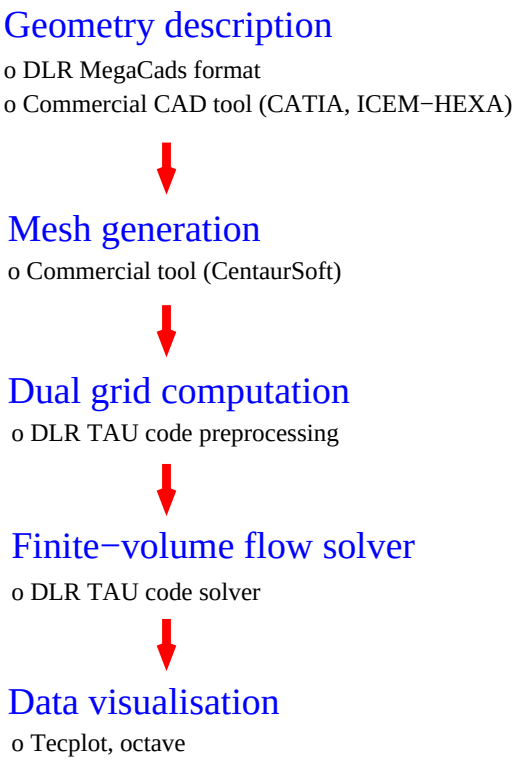


Figure 1.1: Schematic of process chain for CFD computations.

1.4 How to use this guide

Part I

Basic TAU usage

2 Installation of the TAU code

This user-guide is aimed at describing the basic installation procedure for the DLR TAU-Code, and to provide a short introduction to the handling of the code by way of some simple examples.

2.1 System Preparation

The following software should be installed on the target system in order to successfully compile the TAU-Code.
Required:

- MPICH (version 1.2.4 or higher)
- NetCDF (version 3.4 or higher, version 3.6.0 or higher required for large-file (>2GB) support)

Optional (required for TauPython):

- Python (version 2.3 or higher)
- SWIG (version 1.3.25)

2.1.1 Installation

2.1.2 .taudef-file

During the TAU-Code compilation the paths to the libraries and header-files of the software listed above need to be defined. The most common way to handle this is to create a file called *.taudef* in the home-directory, where all of the relevant paths are specified, as shown here:

```
NETCDF_INC      = /home/numerik/netcdf-3.5.0-64/include
NETCDF_LIB      = /home/numerik/netcdf-3.5.0-64/lib
PARALLEL_INC    = /home/numerik/mpich-64/include
PARALLEL_LIB    = /home/numerik/mpich-64/lib

PYTHON_INC_FILE = /usr/include/python2.4
PYTHON_BIN_FILE = /usr/bin/python
SWIG_BIN_FILE   = /usr/bin/swig
```

Note that the libraries must match whether you can create a 32 bit or a 64 bit executable.

2.1.3 Makefile

In the top-level Makefile the appropriate platform for which the code is to be compiled has to be selected by un-commenting the given line (shown here for a Linux Opteron system):

```
#PLATFORM = LINUX_486_MPI
#PLATFORM = LINUX_586_MPI
PLATFORM = LINUX_opteron_MPI
```

Once the platform has been selected the most straightforward way of creating the TAU-Code binaries is to execute the `make all` command; this creates executables for all of the standard TAU-Code modules in the `bin/` directory.
In order to create the TauPython bindings the command `make main` is used.

3 TAU Preprocessing

The TAU preprocessing has too tasks:

- compute the dual grid (control volumes for the FV-method)
- set the boundary conditions

A typical example for a hybrid primary mesh and the corresponding dual mesh is sketched in Fig. 3.1. The boundary conditions are specified in the so-called boundary mapping file (extension ".bmap").

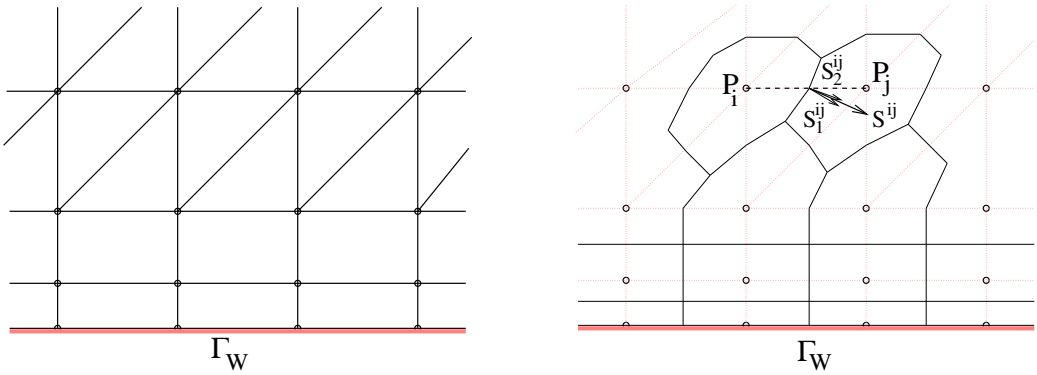


Figure 3.1: Hybrid primary grid (left) and corresponding dual grid (right).

3.1 Boundary mapping file

The boundary mapping file specifies the for each boundary marker the corresponding boundary condition.

- Farfield boundary condition
- Wall boundary condition
  - Euler wall (inviscid calculation)
  - viscous wall
- symmetry plane
- periodic

Each boundary part (or boundary segment) has a certain marker ID. This marker ID is given by the Centaur grid generator. The boundary markers can be checked by looking to the grid by using Tecplot. For this purpose, first we have to convert the TAU grid to tecplot format by typing:

```
~/taudir/bin/tau2plt  -g  naca0012_euler.grid
```

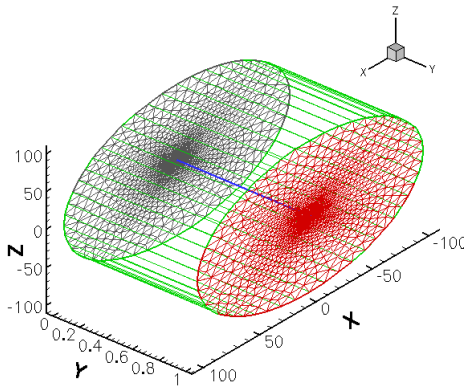


Figure 3.2: Boundary markers and pseudo 2D mesh.

This converts the TAU grid to tecplot format (with suffix .plt). In tecplot, the boundary markers can then be checked by clicking the buttom "Plot attributes..."
The boundary mapping file associates a boundary condition type (e.g., "Type: farfield") with each marker (e.g., "Markers: 1" for the first marker). An example ".bmap" file for a Euler calculation might read as follows.

```
 ------------------------------------------------------
 BOUNDARY MAPPING
 ------------------------------------------------------

            Markers: 1
               Type: farfield
               Name: Farfield
 block end
 --------------------------

            Markers: 2
               Type: euler wall
               Name: Inviscid
 block end
 --------------------------

            Markers: 3
               Type: symmetry plane
               Name: Side1
 block end
 --------------------------

            Markers: 4
               Type: symmetry plane
               Name: Side2
 block end
 --------------------------
```

For a viscous calculation

```
            Markers: 2
               Type: viscous wall
            Subtype: turbulent      // fully developed turbulent flow
#           Subtype: laminar        // laminar flow
```

```
             Name: Viscous
block end
--------------------------
```

More information on grid generation and boundary conditions can be found in Chapter 14.

3.2 Basic parameter file for TAU preprocessing

The preprocessing computes the set of control volumes for the Finite-Volume method. It is called by typing

```
~/taudir/bin/ptau3d.preprocessing  para_naca0012.euler
```

The TAU preprocessing has the following input/output files:

- Input files:
  - TAU parameter file "para_naca0012.euler"
  - Grid in TAU format "naca0012_euler.grid"
  - boundary mapping file "naca0012_euler.grid.bmap"
- Output files:
  - Dual grid "dualgrid_naca0012_euler"

A log-output is written to stdout, and may also be written to a log-file.
A (minimal) parameter file for TAU preprocessing "para_naca0012.euler" has to set the following parameters:

```
---------------------
BOUNDARY MAPPING
---------------------
       Boundary mapping filename: naca0012_euler.grid.bmap

---------------------
PREPROCESSING
---------------------

       Primary grid filename: naca0012_euler.grid

       Grid prefix: dualgrid_naca0012_euler

   Multigridding ---------------------------:-
       Number of multigrid levels: 1

   Runtime optimisation --------------------: -
       2D offset vector (0 / x=1,y=2,z=3): 2
```

- "Number of multigrid levels": Specifies the number of grid levels to be used in the multigrid scheme of the TAU solver. If a value larger than 1 is choosen, then dualgrids on coarser grid levels are generated by agglomeration.

4 TAU solver

4.1 Overview. Viscous and inviscid solver

The DLR TAU code is an unstructured finite-volume solver based on a dual grid approach.

- Inviscid solver:
  - Euler equation: `/taudir/bin/ptau3d.el`
- RANS + turbulence model equation solver:
  - RANS + Spalart-Allmaras one-equation model: eq `/taudir/bin/ptau3d.turb1eq`
  - RANS + k-ω type two-equation model: `/taudir/bin/ptau3d.turb2eq`

As an example the Euler solver is called by typing

```
~/taudir/bin/ptau3d.el para_naca0012.euler  log
```

Then the following files are read resp. written:

- Input files:
  - TAU parameter file "para_naca0012.euler"
  - TAU dual grid "dualgrid_naca0012_euler"
- Output files:
  - Solution at each field point (.pval-file, which is in netcdf format)
  - Log-file containing a test output of the parameters, a convergence history, integral force coefficients

4.1.1 Euler solver

The Euler solver is called by typing

```
~/taudir/bin/ptau3d.el para_naca0012.euler  log
```

Inviscid calculations use as default

```
Viscous calculation (0/1): 0
```

4.1.2 Navier-Stokes solver

There are different executables for Spalart-Allmaras type one-equation models and for k-ω type two equation models (for historical reasons regarding the code design). Viscous calculations use as default

```
Viscous calculation (0/1): 1
```

Turb1eq model

The RANS solver based on the Spalart-Allmaras type one-equation model is called by

```
~/taudir/bin/ptau3d.turb1eq  para.naca0012
```

More details on the turbulence modelling in TAU are given below.

Turb2eq model

The RANS solver based on the k-ω type two equation model is called by

```
~/taudir/bin/ptau3d.turb2eq  para.naca0012
```

4.2 Finite volume method

Fluid motion is governed by the fundamental conservation laws for mass, momentum and energy. The compressible Euler-equations can be written in conservative form as

$$\frac{\partial}{\partial t}\int_V \vec{W}\, dV \;=\; -\int_{\partial V} \mathbb{F}\cdot\vec{n}\, dS$$

where V is an arbitrary control volume with closed boundary surface ∂V, and $\vec{n}$ is the unit normal vector in outward direction. The vector of conservative variables $\vec{W}$, and the convective flux tensor $\mathbb{F}$ are given by

$$\vec{W} \;=\; \begin{pmatrix} \rho \\ \rho\vec{u} \\ \rho E \end{pmatrix}\;, \quad \mathbb{F} \;=\; \begin{pmatrix} \rho\vec{u} \\ \rho\vec{u}\otimes\vec{u} \;+\; p\mathbb{I} \\ \rho E\vec{u} + p\vec{u} \end{pmatrix}$$

where ρ is the density, $\vec{u}$ is the velocity, and E is the total energy. The pressure is given by virtue of the ideal gas law by $p = (\gamma-1)\rho(E - \frac{1}{2}\vec{u}^2)$.
The FVM formulation seeks a piecewise constant approximation of $\vec{W}$ w.r.t. each control volume V

$$\frac{\partial}{\partial t}\vec{W} \;= -\vec{\mathcal{R}}(\vec{W}) \;\equiv\; -\frac{\vec{\mathcal{Q}}(\vec{W})}{\int_V dV}\,, \qquad \text{with} \quad \vec{\mathcal{Q}}(\vec{W}) \;=\; \int_{\partial V}\mathbb{F}\cdot\vec{n}\, dS$$

The inviscid fluxes $\int_{\partial V}\mathbb{F}\cdot\vec{n}\, dS$ are discretised using a suitable numerical method as described below. In semi-discrete form the governing equations can be written for each control volume as

$$\frac{d}{dt}\vec{W} \;=\; -\vec{\mathcal{R}}(\vec{W})$$

For steady state problems $\frac{d\vec{W}}{dt} = 0$, the arising fixed-point problem $\vec{\mathcal{R}}(\vec{W}) = 0$ is solved by considering the corresponding time-dependent problem with fictious pseudo-time t^* and seeking its steady-state solution

$$\frac{d}{dt^*}\vec{W} + \vec{\mathcal{R}}(\vec{W}) = 0. \tag{4.4}$$

4.3 Discription of the flow conditions

Inviscid flow problems are fully specified by prescribing

- Onflow Mach number $Ma_\infty = u_\infty/a_\infty$
- Onflow temperature T_∞
- Angle of attack α

For viscid calculations, the Reynolds number $Re = u_\infty l \rho_\infty / \mu_\infty$ also has to be specified, where the characteristic lenght l is usually the chord length of the airfoil c. Then μ can be infered from Re.

```
------------------------------
FLOW CONDITIONS
------------------------------
          Reference Mach number: 0.5
          Reference temperature: 293.0
                Reynolds number: 1.e6
                Reynolds length: 1
           Angle alpha (degree): 3.0      // Angle of attack
                     Grid scale: 1.0
```

The parameters "Reynolds length" and "Grid scale" require some explaination

- Grid scale: Scaling factor of the mesh which discretises the CAD geometry. For example, if the CAD geometry is given in [mm], then grid scale is 0.001. It is strongly recommended scale the grid to SI unit [m] (see Example below)
- Reynolds length: Length scale used for the Reynolds number.

Scaling of the TAU grid can be done with the TAU utility program `setup_taugrid`

```
~/taudir/bin/setup_taugrid    naca0012.grid
```

- Choose option 4: Scale grid
- Enter scale factor
- Choose option 99: Write grid (NetCDF format)
- Enter new grid name

Example. Given a wing with chord length 0.36m at the wing root, which is taken as length scale for Reynolds number in the wind-tunnel experiment. Assume the wing root in the CAD geometry and the corresponding CFD mesh has length 1000 in grid units. Then call `setup_taugrid` with scale factor 0.00036 to scale the grid to SI unit [m]. Then use "Reynolds length: 0.36" and "Grid scale: 1.0" in the TAU parameter file.

4.4 Discretisation schemes for the inviscid fluxes

The DLR TAU code provides a variaty of discretisation schemes for the inviscid fluxes. Here we restrict ourselves to the following methods.

- Upwind scheme
  - van Leer
  - AUSM
  - AUSMDV
  - Roe
  - MAPS+
- Central scheme with artificial diffusion
  - Scalar dissipation being a central scheme with artificial dissipation based on $||\Delta p||$, where $\Delta p = \vec{\nabla} \cdot \vec{\nabla} p$ denotes the pressure Laplacian.
  - Matrix-type (anisotropic) artificial diffusion

4.4.1 Upwind schemes

Some properties of the above mentionned upwind schemes are resumed in the following table Due to

Upwind scheme	TAU-Parameter	Properties
van Leer	Van_Leer	very diffusive (smears shocks but very stable)
AUSM	AUSM_Van_Leer	
AUSMDV	AUSMDV	default
Roe	Roe	
MAPS+	MAPS+	Recommended for low Mach number flows

relatively large numerical errors for subsonic and transonic flows, central schemes are strongly recommended.
An example code fragment for the upwind method using the AUSMDV scheme may read

```
----------------------------
Inviscid fluxes
----------------------------
  Solver type upwind/central (0/1): 0
        Second order upwind solver: AUSMDV
```

4.4.2 Central schemes with artificial dissipation

The following table gives some properties for the central schemes:

Central scheme + artificial diss.	TAU-Parameter	Properties
Scalar dissipation	Classic_dissipation	Recommended for all subsonic and transonic flows; If unstable during starting period for subsonic flows, then perform precursor calc. with MAPS+ upwinding
Matrix dissipation		Highest numerical accuracy; Recommended for DES, LES

An example code fragment for the upwind method using the central scheme with scalar dissipation may read

```
----------------------------
Inviscid fluxes
----------------------------
  Solver type upwind/central (0/1): 1
                    Central solver: Classic_dissipation
   Central classic dissipation 2nd: 0.5
```

4.5 Linear solver

For steady state problems, the nonlinear equation from the FV discretisation is solved by iteration in pseudo time. Two approaches are supported:

- **Explicit Runge-Kutta scheme.** The steady state Euler equations are iterated in ficticious pseudo time using an explicit Runge-Kutta scheme.
- **Backward Euler.** The steady state Euler equations are iterated in ficticious pseudo time using a backward Euler implicit scheme. Within each pseudo time step, the implicit problem is solved with LU-SGS or SGS iterations. Thus this scheme is not really implicit, as matrix times vector operations are performed "on the flight" and no linear system is solved by a linear solver.

The convergence properties and the stability of each method strongly depends on the parameter "CFL number". The parameter "Maximal time step number" specifies the number of iterations of the non-linear solver.

4.5.1 Explicit Runge-Kutta scheme

For the Runge-Kutta scheme, the maximal CFL number for which stability can be ensured theoretically, is 1.4. In practice, a common choice is 1.0 to 1.3.
An example code fragment may look like

```
---------------------------
Nonlinear solver
---------------------------
                  CFL number: 1.3
           Relaxation solver: Runge_Kutta
    Number Runge-Kutta steps: 3
    Maximal time step number: 5000              // Number of pseudo-time
                                                // iteration steps
```

4.5.2 LU-SGS scheme

Regarding the Backward-Euler with LUSGS, theoretically there is no upper limit for the CFL number; practically best convergence can be obtained for CFL number around 10.

```
---------------------------
Nonlinear solver
---------------------------
                  CFL number: 10.0
           Relaxation solver: Backward_Euler
    Maximal time step number: 5000              // Number of pseudo-time
            Minimum residual: 1e-7

 Compute lusgs mapping (0/1): 1                 // used in preprocessing
```

4.6 Multigrid

Multigrid is a fundamental technique for accelerating convergence of the solver.

4.6.1 Single grid mode

In single grid mode, no multigrid is used, which is the default option

```
Multigrid ----------------------: -
          MG description filename: sg
```

4.6.2 Multigrid

If multigrid is used, then the number of multigrid levels has to be specified.

```
         Number of multigrid levels: 3
```

Then the preprocessing `ptau3d.preprocessing` has to be called to generate the coarse level dual grids.
There are a variaty of multigrid cycles available e.g., 2v, 3v, 4v, 5v, 2w, 3w, 4w, 5w. As an example

```
  Multigrid ----------------------: -
            MG description filename: 3v
```

4.7 Output results

The solver output consists of solution-files and of integral coefficients.

- **Field output**: Output of solution for each node of the primary grid (i.e., the cell average for each dual grid cell)
- **Surface output**: Output of solution for each node on a specified surface
- **Cutplane output**: Output of solution for each node on a curve, which is the intersection of the airfoil surface and a specified cutting plane.

Moreover, integral force coefficients (drag, lift, moment) are calculated and written during computation (for checking convergence) and for the final result.
The following solution files are written The **NetCDF-format** is a convenient data-format for scientific

Type of output	File extension	Format	Content of the file
Field output	.pval	netcdf	Solution at each field point
Surface output	.surface.pval	netcdf	Solution at each surface point
Cut-plane output	.cut_plane	ASCII	Solution along curve (intersection

computation. NetCDF (network Common Data Form) is an interface for array-oriented data access and a library that provides an implementation of the interface. The netCDF library also defines a machine-independent format for representing scientific data, see also for details *http://www.unidata.ucar.edu/software/netcdf/*.

4.7.1 Specification of output solution filenames

The solution is written to the file specified by the parameter "Output files prefix"

```
 ------------------------------
 IO
 ------------------------------
                Restart-data prefix: (none)
    Automatic parameter update (0/1): 1             // Append restart file
                                                       to the parameter file
 Write pointdata dimensionless (0/1): 0
                Output files prefix: naca0012_el    // Prefix for output files
```

4.7.2 Restart

If the option `Automatic parameter update (0/1): 1` is chosen, then the name of the pval-solution file is appended to the parameter file. This solution is then used as starting solution for restart of TAU solver.

4.7.3 Output of solution files (pval, surface and cut-plane)

The output of the field-data, surface-data and cut-plane-data files is specified by the following parameter settings:

```
 ------------------------------------------------
 Cut-Plane Data Output
 ------------------------------------------------
```

```
       Plane output description file: (thisfile)
                 Plane output values: xyz_cp_cf
                 Plane output period: 999999
                    Number of planes: 1
       Boundary type (euler/viscous): viscous
                     Plane support x: 0
                     Plane support y: 0.5
                     Plane support z: 0
                      Plane normal x: 0
                      Plane normal y: 1
                      Plane normal z: 0
-----------------------------------------------------
Surface Data Output
-----------------------------------------------------
     Surface output description file: (thisfile)
                                Surface output values: x_y_z_cp_cf
                                Surface output period: 999999
-----------------------------------------------------
Field Data Output
-----------------------------------------------------
       Field output description file: (thisfile)
                 Field output values: cp_cf_mach
```

Remarks.

- The quantities supported for output `Plane output values`, `Surface output values`, `Field output values` are listed in the user guide
- The `output period` specifies the period for writing output files
- As remark for cut-plane output, the parameters `Plane support` and `Plane normal` have to define a plane whose intersection with the mesh defines a non-empty plane.

5 Postprocessing and visualisation of the results

5.1 Convergence history

The objective is to assess the level of convergence achived during the iterative solution of the non-linear steady-state Euler equations.
We recall the situation of the iterative solution of a linear system in numerical linear algebra. We want so solve $Ax = b$ with $A \in \mathbb{R}^{n\times n}$ and $x, b \in \mathbb{R}^n$. We write the problem in defect-correction form, i.e., we seek a solution of the form $x^{n+1} = x^n + \mathrm{cor}^{n+1}$

$$A(x^n + \mathrm{cor}^{n+1}) \;=\; b \quad \Leftrightarrow \quad A\mathrm{cor}^{n+1} \;=\; b - Ax^n$$

Then $||b - Ax^n||$ is a criterion for the convergence already achived, provided the problem is not ill-conditioned. The quantity $b - Ax^n$ is often called residual.
For an explicit finite volume method, the situation is a little more complicated. It can be seen from (4.4) that for each cell V the change dW/dt^* in pseudo-time is equal to the net flux across ∂V, i.e.,

$$\frac{d}{dt^*}\vec{W} \;=\; -\vec{\mathcal{R}}(\vec{W}) \tag{5.1}$$

Thus the so-called residual $\vec{\mathcal{R}}(\vec{W})$ is a measure for the convergence. For each cell V, the steady state flux balance says that the net inflow equals the net outflow of mass, momentum and energy. i.e.,

$$\text{Steady state conditions:} \qquad \vec{\mathcal{R}}(\vec{W}^n) \;=\; 0 \qquad \text{for each dual grid cell } V_i$$

Therefore as a criterion for the rate of convergence, for time step n we define the (average) residual Res^n by

$$\mathrm{Res}^n \;=\; \sqrt{\frac{1}{N_{\mathrm{cells}}}\sum_{j=1}^{N_{\mathrm{cells}}}\vec{\mathcal{R}}(\vec{W}^n)}$$

Figure 5.1 shows an example of the convergence history.

5.2 Visualisation of field data

For the visualisation of the results, we use the commercial tool Tecplot. The flow field is investigated by plotting

- streamtraces of the velocity field
- contour lines for density, pressure, Mach number.

5.2.1 Streamtraces

We can visualise the flow field by plotting the so-called streamtraces. These are the curves, which are at each point $\vec{x}$ tangential to the velocity field.

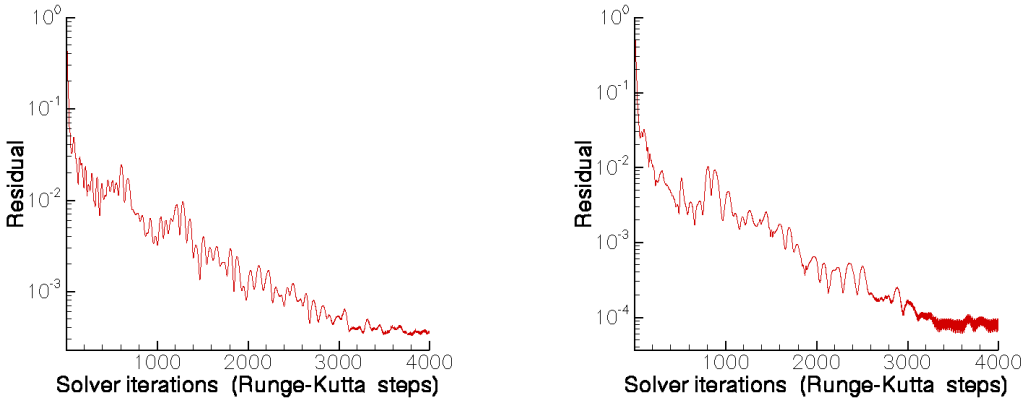


Figure 5.1: NACA0012: Convergence history of nonlinear solver for $Ma = 0.5$ (left) and $Ma = 0.7$ (right).

5.2.2 Contourlines for density and Mach number

The flowfield can be visualised by plotting the contour lines of density, pressure, and Mach number.

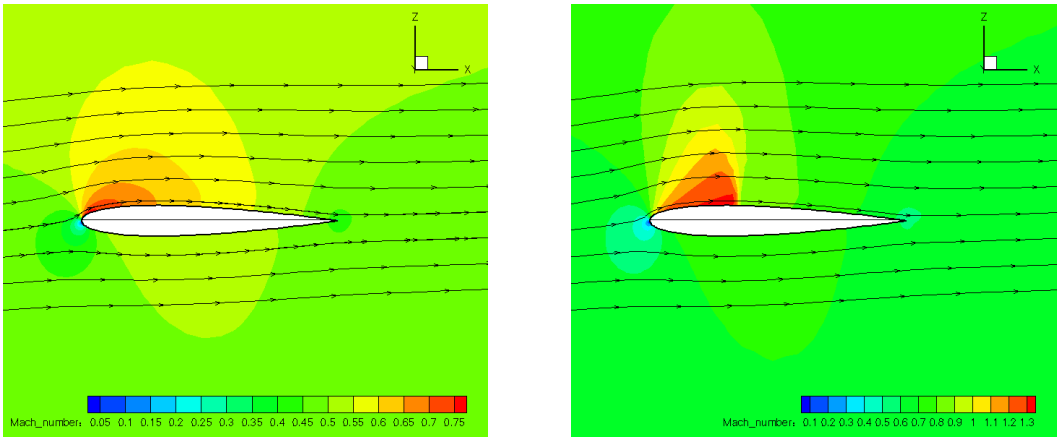


Figure 5.2: NACA0012 Euler calculation: Visualisation of the flow field using Mach number isolines and streamtraces of the velocity field $Ma = 0.5$ (left) and $Ma = 0.7$ (right).

5.3 Visualisation of surface data

5.4 Pressure coefficient and skin-friction coefficient

The flow solution on the surface is of prime importance for aerodynamic predictions. Instead of absolute quantities, it is more convenient to consider suitably normalised quantities

$$\text{Pressure coefficient} \qquad C_p = \frac{p - p_\infty}{q_\infty} \tag{5.2}$$

$$\text{Skin friction coefficient (viscous calculations only)} \qquad C_f = \frac{\tau_w}{q_\infty} \tag{5.3}$$

with

$$\tau_w = \frac{\mu}{\rho}\frac{\partial u_t}{\partial y}, \qquad q_\infty = \frac{1}{2}\rho_\infty u_\infty^2 \tag{5.4}$$

where u_t is the magnitude of wall-parallel velocity and y denotes the wall-normal direction. On an Airbus A340 in cruise flight, approximately one-half of the drag is skin friction drag. Thus the **skin-friction coefficient** determines the fuel required to maintain constant velocity flight.
We remark that the **lift coefficient** is the area between C_p at upper and lower side, i.e.

$$C_L \equiv \frac{F_{\text{lift}}}{\frac{1}{2}\rho_\infty u_\infty^2 c} = \oint C_p(x)\,\mathrm{d}x$$

where F_{lift} is the lift force, and c is the chord length.

5.5 Cut-plane output

Cut-plane output allows to write the solution at certain curves being the intersection of the surface and a specified plane. Cut plane output has to be sorted before visualised using Tecplot:

```
~/taudir/bin/sort_cut   naca0012_euler.grid   naca0012_euler.cut_plane.1.1000
```

Figure **??** shows C_p for two different Mach-numbers $Ma = 0.5$ and $Ma = 0.8$. For $Ma = 0.5$, the flow is (almost) subsonic. Therefore the increase in pressure on the lower side of the airfoil is continuous.
For $Ma = 0.8$ the flow is in the transonic regime. On the upper side, the flow has local Mach-number larger than one and is therefore supersonic. Therefore the increase in pressure is abrupt/discontinuous. This discontinuous change in pressure refers to a **shock**.

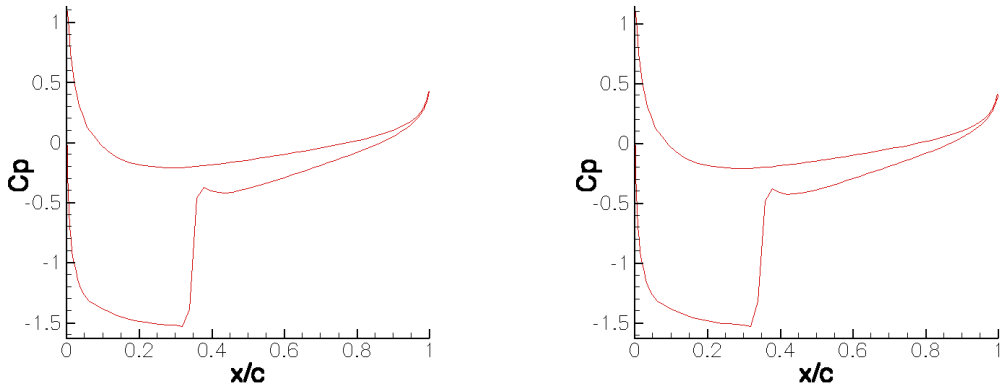


Figure 5.3: NACA0012 Euler calculation for $Ma = 0.7$: Pressure coefficient C_p for Roe solver (left) and van Leer scheme (right).

Part II

Advanced TAU usage

6 Unsteady calculations

For time-dependent problems, the time derivative is discretised using an implicit backward-difference formula (BDF) scheme. For example, the second order accurate BDF reads

$$\frac{3}{2\Delta t}\vec{W}^{n+1} - \frac{4}{2\Delta t}\vec{W}^{n} + \frac{1}{2\Delta t}\vec{W}^{n-1} = -\vec{\mathcal{R}}(\vec{W}^{n+1}) \tag{6.1}$$

SPT where $\vec{W}^{\nu}$ denotes the solution at time t_ν, $n+1$ denotes the current time level and the two previous time levels are denoted by n and $n-1$.
For each time step, the arising (nonlinear) steady-state problem (6.1) is solved using the so-called dual-time stepping scheme: Seek the steady-state solution $\vec{W}^{n+1}$ in a pseudo-time t^* of

$$\frac{d\vec{W}^{n+1}}{dt^*} = -\vec{\mathcal{R}}^{DTS}(\vec{W}^{n+1}) \tag{6.2}$$

with the modified residual

$$\vec{\mathcal{R}}^{DTS}(\vec{W}^{\nu}) \;\equiv\; \vec{\mathcal{R}}(\vec{W}^{\nu}) + \frac{3}{2\Delta t}\vec{W}^{\nu} - \frac{4}{2\Delta t}\vec{W}^{n} + \frac{1}{2\Delta t}\vec{W}^{n-1}$$

Associated TAU parameters

▷ "Number of time steps per period" resp.
▷ "Unsteady physical time step size"

7 Turbulence modelling

7.1 Statistical turbulence models (RANS)

The TAU code provides a variety of statistical turbulence models

- One-equation turbulence models based on the Spalart-Allmaras model
- Two-equation turbulence models based on the k-ω model

Several explicit-algebraic Reynolds stress models (EARSM) for the k-ω model are also implemented.
The list of available turbulence models can be found in the user guide. The list is also printed in the standard output of the solver. The models are described in the technical documentation.
A specific turbulence model can be choosen via the parameter

```
Turbulence model version: 1
```

7.2 Wall-functions

The DLR TAU code allows to use turbulence model specific wall functions. For a specific boundary marker at a viscous wall, usage of wall functions is activated using the parameter

```
                  Markers: 2
                     Name: wing
                     Type: viscous wall
Use wall function (0/1): 1
                  Subtype: turbulent
```

Notice that the usage of wall-functions is stored in the dual mesh and hence the `ptau3d.preprocessing` has to be called.
For k-ω type turbulence models, we strongly recommend to choose

```
Omega boundary condition type (0/1/2/3/4) : 2
```

7.3 Laminar-turbulent transition

The TAU code allows for describing the laminar-turbulent transition line. As an example for a 2D airfoil, where transition is tripped at $x/c = 0.03$ (with c being the chord length)

```
Boundary part namelist: wing
        Laminar height: 0.1
Number of plane points: 3
 TransitionCoordinates:
         0.03 0 0
         0.03 1 0
```

```
             0.03 1 1
transition end
 --------------------------
```

- `Boundary part namelist:` Referring to the boundary marker where transition is prescribed.
- `TransitionCoordinates:` Plane described by three points. Flow is laminar on the left and turbulent on the right of the plane.
- `Laminar height:` The flow is laminar in all field points which have a distance smaller than this value from a laminar wall part.

7.4 Detached Eddy Simulation

The most popular DES models are available in the TAU code. At the current stage of code design, they are treated as additional RANS models. E.g., the SA-DES model is `Turbulence model version: 1` for the executable `ptau3d.turb1eq`.

- Three-dimensionality
  - They require a 3D mesh, often a 2D mesh is multiplied in spanwise direction
  - Often periodic boundary conditions in spanwise-direction are imposed
- Unsteadyness
- LES-part to be specified, e.g., filtering technique and filter width
- Low-dissipation scheme for LES part

These issues will be discussed in the following subsections.

7.4.1 3D mesh generation

DES simulations must be 3D, as LES is always 3D. Given a suitable 2D grid, it is relatively easy to generate a 3D mesh by doubling the mesh in spanwise direction and subsequently scaling the grid. Given a 2D mesh `naca0012.grid` in the x-z plane, call

```
~/taudir/bin/setup_taugrid  naca0012.grid
```

Then choose the following options and settings

```
                          Option (1 = rewrite options): 8    // Double the grid
                  Translate or mirror the grid (t/m) ?: t
Which direction do you want to double the grid (x/y/z) ?: y
               Positive or negative y direction (p/n) ?: p
                        Epsilon for double point search: 1.0e-12
      Renumber surface markers of new grid part (y/n) ?: n
```

This doubles the mesh in spanwise direction (here: y-direction). When the desired number of grid nodes in spanwise direction is reached, type

```
                          Option (1 = rewrite options): 5    //  Stretch grid
```

and then enter desired scaling factor which determines the grid spacing in spanwise direction.
Finally write the grid in netcdf-format by choosing `Option (1 = rewrite options):  99`

7.4.2 3D airfoil flow with periodic boundary conditions

Consider a DES simulation for a 3D airfoil with periodic boundary conditions in spanwise direction (here y-direction).

```
-------------------------------------------------
 BOUNDARY MAPPING
 -------------------------------------------------
                            Markers: 3,4
                              Type : periodic plane
                               Name: Periodic
         Periodic translation vector: 0 1 0
              Periodic epsilon value: 1.0e-10
           Write surface data (0/1): 0
               Monitor forces (0/1): 1
         block end
 -------------------------------------------------
```

- `Periodic translation vector:` x-, y-, z- component of vector describing the translation between the two periodic planes

7.4.3 Low-dissipation scheme

For the spatial discretization, a scheme with a very low numerical dissipation should be used

```
Central solver: Matrix_dissipation
```

7.4.4 Turbulence model settings. LES Filtering

An implicit filtering with the classical DES filter width $\Delta = h_{\max}$ with $h_{\max} = \max\{\ h_x,\ h_y,\ h_z\ \}$. the default.

```
Turbulence ----------------------------------:
                LES/DES Filter type (0/1/2/3): 0
```

The classical LES filter width $\Delta = (Vol)^{1/3}$ is obtained for `LES/DES Filter type (0/1/2/3): 1`
For a fixed explicit filter width, choose

```
Turbulence ----------------------:
              Spatial filter width: 0.03125
     LES/DES Filter type (0/1/2/3): 3
```

The parameter `Spatial filter width` then specifies the fixed filter width.

7.4.5 Unsteady calculation

As an example, for a NACA0012 profile at 60° angle of attack we recommend

```
    Dual time ---------------------------------: -
                  Unsteady calculation (0/1/2): 1
              Unsteady physical time step size: 1.2577e-04
                  Unsteady physical time steps: 800
        Unsteady inner iterations per time step: 60
Minimum number of inner iterations per time step: 50
         Unsteady implicit scheme order (1/2/3): 2
                Compute flow statistics (0/1/2): 2
```

- `Unsteady calculation (0/1/2):` DES calculations are always unsteady
- `Unsteady physical time step size:` The time step is given by physical considerations.
- `Unsteady physical time steps:` Number of physical time steps
- `Unsteady inner iterations per time step:` Maximal number of inner iterations per time
- `Unsteady implicit scheme order (1/2/3):` order of BDF-scheme

8 Grid adaptation

9 Utility programs

10 Parallel computations

11 Runtime and cache optimisation

12 TAU usage in SUN Grid-engine environment

Part III

External tools

13 Geometry description

13.1 CAD geometry description

The starting point for a CFD application is a description of the geometry. There are a variaty of data formats for computed aided design (CAD). In industrial applications, the so-called IGES format is a popular, see the official webpage *http://de.wikipedia.org/wiki/IGES* and *http://www.nist.gov/iges/*
Geometry description files are generated by CAD tools. One widespread tool in aerospace and automotive industry is the CATIA software package (CATIA=Computer Aided Three-Dimensional Interactive Application), see *http://de.wikipedia.org/wiki/CATIA* and also the webpage *http://catia.cad.de/*.

13.2 The DLR MegaCads format

For 2D airfoil geometries, such an advanced CAD software is not necessary. We can use the CAD-format of the DLR MegaCads software which is relatively simple, see *http://www.megacads.dlr.de/*.
As an example for a 2D airfoil flow, the geometry may consist of 4 curves, viz., lower and upper part of the airfoil, and the so-called farfield boundary (e.g. a circle surrounding the airfoil with a sufficiently large radius). Each curve is discretised by a suitable number of points. For details see Appendix A.
The following list gives the arising curves and their orientation.

- Curve 1: Farfield boundary (lower part), counterclockwise orientation
- Curve 2: Farfield boundary (upper part), counterclockwise orientation
- Curve 3: Airfoil surface (lower side), clockwise orientation
- Curve 4: Airfoil surface (upper side), clockwise orientation

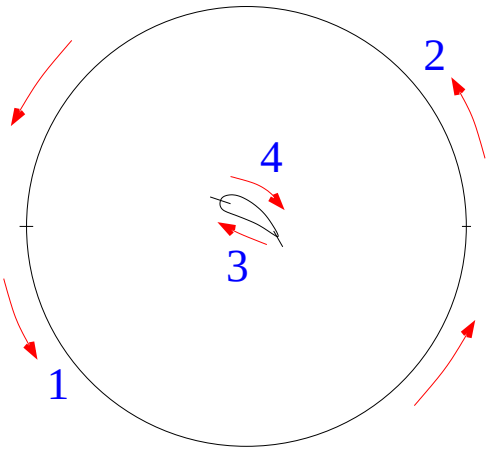


Figure 13.1: Illustration of Megacads geometry description. Numbers in blue designate the curve components. Arrows in red show orientation of the curve components.

The data are stored in a file with extension ".dat". Thus for an airfoil with name "naca0012", the geometry file may be named "naca0012.dat".

14 Grid generation

14.1 CentaurSoft grid generator

In the second step we have to generate the primary grid. We use the commercial mesh generation tool CentaurSoft *http://www.centaursoft.com/*.
For detailed information, see the online documentation *http://www.centaursoft.com/support/manual*.
Before starting the tool, we have to set the environment variables by typing

```
source /CENTAUR_PATH/Centaur/setup
```

where `CENTAUR_PATH` denotes the path where Centaur is installed.
The grid generation process with Centaur takes the following steps

(1) Set boundary markers: Associate a mathematical boundary condition with each boundary curve in the geometry file.

(2) Generate surface mesh

(3) Generate volume mesh

(4) Convert Centaur mesh to tecplot format for visualisation

(5) Convert Centaur mesh to TAU format for usage by TAU code

14.2 Euler meshes

First we describe the grid generation process for meshes suitable for a solution of the Euler equations, which are for brevity called Euler-meshes.

14.2.1 Set boundary markers

First we have to associate a mathematical boundary condition with each boundary curve in the geometry file. Moreover we have to specify the type of primary grid cells at this boundary (tetras, quads or hybrid cells), see Figure 14.1.
For the example testcase of a 2D airfoil flow, we impose a so-called farfield condition at the farfield boundary, which means (roughly speaking) that we have undisturbed onflow $\vec{u} = \vec{u}_\infty$ at the farfield boundary. At the airfoil surface, we impose in inviscid-wall (or no-penetration) condition $\vec{u} \cdot \vec{n} = 0$, where $\vec{n}$ is the surface normal vector. This means that the flow is parallel to the airfoil surface.

$$\vec{u} = \vec{u}_\infty \quad \text{on farfield boundary} , \qquad \vec{u} \cdot \vec{n} = 0 \quad \text{on airfoil surface}$$

This is done using the tool setupgrid, which is called by typing

```
setupgrid
```

Then the boundary conditions and type of primary grid cells can be specified as follows

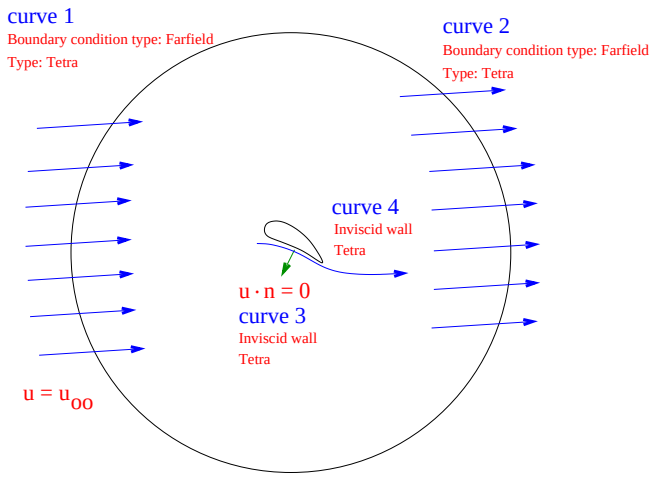


Figure 14.1: Mesh generation step 1: Association of boundary condition and type of primary grid cells for each boundary curve in the CAD file.

- Menu: File → Open dat-file: ”naca0012.dat”
- Set boundary condition at farfield
  - Create group: Boundary condition ”Farfield”, Type “Tetraedra”
  - Hotkey ’s’, then select (click on) curves. The selected curves then appear in ”Select group”-window. Then click on ”Move all”, which moves the selected curves to the current group.
- Set boundary condition at airfoil surface
  - Create group: Boundary condition ”Inviscid wall”, Type “Tetraedra”
  - Hotkey ’z’ (zoom), then zoom using left mouse klick
  - Hotkey ’s’ (select), then click on curves. They appear in ”Select group”-window. Then click on ”Move all”
- Menu: File → Write output files, Enter prefix, e.g., ”naca0012”

This writes the following files all with prefix ”naca0012”

.ors	restart file for setupgrid	complete information (most important file)
.sin	surface input file	surface discretisation
.tin	tetrahedral input file	discretisation in regions of tetrahedral elements
.lin	linear input file	point sources for surface discretisation

14.2.2 Parameter for surface mesh generation

ss:sm:eu The parameters for the surface mesh generation are specified in the file ”naca0012.sin”. We have to ensure

- a sufficient resolution of the leading edge and the trailing edge region
- a sufficient number of points on the airfoil surface

Figure 14.2 shows a typical pitfall: If the airfoil nose is not resolved fine enough, then the discretised surface is not sufficiently smooth. If the mesh is poor, then **any** numerical method will give poor results. Put in other words: If the mesh is not appropriate, then you are already lost before starting with the flow solver.
The surface mesh generation is controlled by the parameter file with extension ”.sin”.

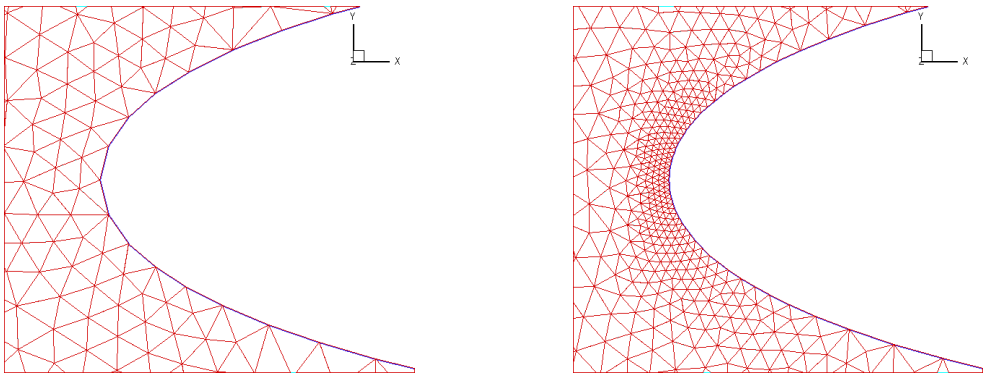


Figure 14.2: Different surface discretisations for NACA 0012 profile: Too coarse discretisation yielding a non-smooth surface (left) and sufficiently fine surface mesh (right).

```
1.8     !  Stretching ratio (1.5-2.1)
1.0     !  Scaling parameter (0.25-4.0)
0.02    !  Length Scale in absence of any features             // spacing h
                                                               // (reference length 1)
T       !  Activate interpanel curvature clustering
100. 8. !  Angle and factor for interpanel curvature clustering
60.     !  Factor for curvature clustering interior to panels  // #points for discret-
                                                               //  ising a full circle
2.      !  Factor for proximity clustering
2.      !  Factor for CAD clustering
```

14.2.3 Parameter for tetrahedral mesh generation

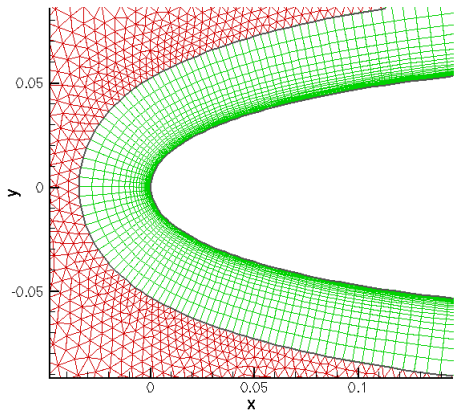
ss:tm:eu The parameters for the tetrahedral mesh generation are specified in the file ”naca0012.tin”. We have to ensure

- sufficient number of points in a circle whose diameter is the airfoil chord length
- regular tetrahedra (all angles $> 10°$)

```
1.9  !  Stretching ratio (1.5-2.1)                // streching ratio of tetras between
                                                  // near-wall and in farfield region
1.0  !  Scaling parameter (0.25-4.0)
5    !  Tet. Grid Quality (1-lowest -- 10-highest)
```

14.3 Navier-Stokes meshes

For viscous calculations, the boundary layer near the airfoil surface has to be resolved. This requires a suitable near-wall mesh.
For this purpose, the Centaur mesh generator builds a prismatic near-wall mesh. The resulting hybrid mesh consists of prismatic cells in boundary layers and unstructured cells remote from walls.



Inside the prismatic layer the grid points are located along rays starting at the point $\vec{x}_{\mathrm{wp}}$ on the airfoil surface

$$\vec{x}_{\mathrm{wp}} + y(n)\vec{r}$$

and end at the outer edge of the regular layer; the direction vector $\vec{r}$ may be non-constant. The point-

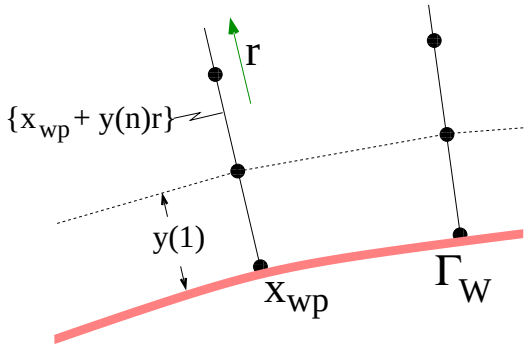


Figure 14.3: Prismatic mesh with point-distribution on wall-normal rays.

distribution is then determined by $y(n)$. In the prismatic near-wall mesh, the wall-normal point distribution is given by

$$\text{W:} \qquad y(n) \;=\; y(1) \sum_{k=0}^{n} q^k \tag{14.1}$$

where

- N_{pnts}: Number of nodes in wall-normal direction in prism layer
- $y(1)$: Distance of first mesh point above the wall
- q: Streching factor (growth rate) of the prismatic layers

$$q \;=\; \frac{y(n+1)}{y(n)} \;=\; \mathrm{const}$$

We may employ the formula for the geometrical series to obtain the thickness of the prismatic layer

$$y(N_{\mathrm{pnts}}) \;=\; y(1) \sum_{k=0}^{N_{\mathrm{pnts}}} q^k \;=\; y(1)\,\frac{1-q^{N_{\mathrm{pnts}}+1}}{1-q}$$

14.3.1 Set boundary markers

For the viscous case of the example 2D airfoil flow we impose the no-slip condition $\vec{u} = \vec{0}$ on the airfoil surface

$$\vec{u} = \vec{u}_\infty \quad \text{on farfield boundary} , \qquad \vec{u} = \vec{0} \quad \text{on airfoil surface}$$

This is done using the tool setupgrid. Recall the steps for setting the boundary markers for a Euler mesh (cf. Section 14.2.1). For the airfoil surface we then choose:

- Set boundary condition at airfoil surface to viscous wall
  - Create group: Boundary condition "Viscous wall", Type "Quads"
  - Hotkey 'z' (zoom), then zoom using left mouse klick
  - Hotkey 's' (select), then click on curves. They appear in "Select group"-window. Then click on "Move all"

14.3.2 Parameter for surface mesh generation

See section **??**.

14.3.3 Parameter for prismatic mesh generation

The parameters for the prismatic mesh generation are specified in the file "naca0012.pin". For turbulent flow problems it is essential to ensure

- the first node above the wall located at a wall-distance $y^+(1) \approx 1$ (in universal coordinates)

$$y^+(1) \; = \; \frac{y(1)\, u_\tau}{\nu}$$

  - $y(1)$: distance of the first node above the wall
  - u_τ: friction velocity
  - ν: kinematic viscosity

  As a first guess, a semi-empirical region for c_f a flat plate turbulent boundary layer may be used. Such a tool (called "arclen") can be provided by Stefan Melber (DLR BS).

- The structured layer should be thick enough to contain the entire boundary layer.

The relevant parameters in the prismatic input file are

```
\begin{verbatim}
1.0     !  Ramp angle(deg)-- growth rate on final layer (5-30)  # Set small value

27      !  No. of prismatic layers to be generated (5-30) # Number of prism. layers
6.0E-5  !  Initial marching step (case dependent)         # First spacing y(1)
1.23    !  Stretching factor (1.1-1.5)                   # stretching factor q

F       !  Chop prismatic layers (T/F)                   # Deactivate!
6.0E-5  !  Minimum nominal marching step (case dependent) # should be set to y(1)
```

14.3.4 Parameter for tetrahedral mesh generation

See section **??**.

14.4 Mesh generation

The Centaur mesh-generator is called by typing

```
makegrid  naca0012
```

This performs the following two steps

```
makegrid  naca0012  surface          // surface mesh generation
makegrid  naca0012  prism            // prismatic mesh generation
makegrid  naca0012  tetra            // tetrahedral mesh generation
```

The mesh is written to the file ”naca0012.hyb”.

14.5 Conversion of mesh to Tecplot format

Conversion of the mesh ”naca0012.hyb” to tecplot format for visualisation is done by typing

```
hybconvert  naca0012.hyb
```

which produces ”naca0012.hyb.tec”.

14.6 Conversion of mesh to TAU format

Finally convert the mesh from CentaurSoft format to TAU format by typing

```
~/taudir/bin/centaur2tau naca0012.hyb
```

This creates two files

- ”tau.grid”: the mesh in TAU format
- ”tau.grid.bmap”: the file containing the boundary mapping, i.e., the mapping of a boundary marker to the corresponding boundary condition.

14.7 Conversion of TAU mesh to tecplot format

A TAU mesh can be converted to tecplot format by typing

```
~/taudir/bin/tau2plt  -g  naca0012.grid
```

This produces ”naca0012.grid.plt” which can then be visualized by Tecplot.

Part IV

Appendix

A MegaCADs geometry description format

```
"DAT file from MegaCads V2.4"  // Header
4 1 1.000000e-05               // We have 4 curve components...
Curve Components:
1  1                           // Here begins curve component 1 ...
41                             // ... with 41 points whose
                                      x-, y-, z- coordinates
                                      are given in the next 41 lines ...
-9.940000000000e+01   -2.886000000000e-01    0.000000000000e+00
-9.660000000000e+01   -2.361000000000e+01    0.000000000000e+00
-8.840000000000e+01   -4.563000000000e+01    0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
 0.000000000000e+00   -9.966000000000e+01    0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
 8.587000000000e+01   -5.198000000000e+01    0.000000000000e+00
 9.675000000000e+01   -2.005000000000e+01    0.000000000000e+00
 1.004000000000e+02    2.857000000000e-01    0.000000000000e+00

2  1                           // Here begins curve component 2 ...
41                             // ... with 41 points
 1.000000000000e+02    2.857000000000e-01    0.000000000000e+00
 9.760000000000e+01    2.361000000000e+01    0.000000000000e+00
 9.273000000000e+01    3.851300000000e+01    0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
 0.000000000000e-01    9.994896766559e+01    0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
-8.868000000000e+01    4.511000000000e+01    0.000000000000e+00
-9.675000000000e+01    2.305225246045e+01    0.000000000000e+00
-9.945000000000e+01   -2.886414052776e-01    0.000000000000e+00

3  1                           // Here begins curve component 3 ...
201                            // ... with 201 points
0.999990130428069     -1.41276024470159e-06  0.000000000000e+00
0.999911176190404     -1.27138752672172e-05  0.000000000000e+00
0.999753280182866     -3.53109495457015e-05  0.000000000000e+00
0.999516467339062     -6.91936774121394e-05  0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
0.503141571982779     -0.0524520810423437    0.000000000000e+00
0.5                   -0.0526562766456404    0.000000000000e+00
```

```
 ..................    ..................    ..................
 ..................    ..................    ..................
0.000631521696991254  -0.00444836504229098   0.000000000000e+00
0.000157905358350007  -0.00223643794811647   0.000000000000e+00
3.94778980919385e-05  -0.00112125445274293   0.000000000000e+00
0.0                    0.0                   0.000000000000e+00

4  1                          // Here begins curve component 4 ...
201                           // ... with 201 points
0.0                    0.0                   0.000000000000e+00
9.86957193143507e-06   0.000561382934927348  0.000000000000e+00
3.94778980919399e-05   0.00112125445274294   0.000000000000e+00
8.88238095954951e-05   0.0016796083417195    0.000000000000e+00
0.000157905358350008   0.00223643794811648   0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
0.5                    0.0526562766456404    0.000000000000e+00
0.503141571982779      0.0524520810423437    0.000000000000e+00
 ..................    ..................    ..................
 ..................    ..................    ..................
0.999753280182866      3.53109495457015e-05  0.000000000000e+00
0.999960522101908      5.65087981540121e-06  0.000000000000e+00
1.0                    0.0                   0.000000000000e+00

Surface Components:             // 2D Plane in which the configuration
1 1                             // is embedded ...
 3 3                            // ... specified by 3x3 points
 1.000000000000e+02 -1.0000000000000+02  0.000000000000e+00
 1.000000000000e+02  0.0000000000000+00  0.000000000000e+00
 1.000000000000e+02  1.0000000000000+02  0.000000000000e+00
 0.000000000000e+00 -1.0000000000000+02  0.000000000000e+00
 0.000000000000e+00  0.0000000000000+00  0.000000000000e+00
 0.000000000000e+00  1.0000000000000+02  0.000000000000e+00
-1.000000000000e+02 -1.0000000000000+02  0.000000000000e+00
-1.000000000000e+02  0.0000000000000+00  0.000000000000e+00
-1.000000000000e+02  1.0000000000000+02  0.000000000000e+00
Mesh Generation:
4 1
Curve Segments:
1 1 1
2 2 1
3 3 1
4 4 1
Surface Regions:     // Specify which curve components form airfoil boundary
                     // and farfield boundary

1 1 1 2              // ... we have 2 closed loops (trimming loops)
4 1 3                // This line refers to the next line:
                     // We have 4 curve components. Each subsequent
                     // number gives the index where a new closed loop begins


1 2 3 4              // 1st trimming loop starts at index 1
^   ^                //   and contains curve 1 and curve 2
                     // 2nd trimming loop starts at index 3
                     //   and contains curve 3 and curve 4
```

B Complete parameter file for inviscid solver

Finally we give a complete parameter file for the inviscid solver:

```
 -------------------------------
 BOUNDARY MAPPING
 -------------------------------
          Boundary mapping filename: naca0012_euler.grid.bmap

 -------------------------------
 PREPROCESSING
 -------------------------------

          Primary grid filename: naca0012_euler.grid

                    Grid prefix: dualgrid_naca0012_euler

    ----------------------------
    Multigridding
    ----------------------------
      Number of multigrid levels: 1

    ----------------------------
    Runtime optimisation
    ----------------------------
2D offset vector (0 / x=1,y=2,z=3): 2


 -------------------------------
 IO
 -------------------------------
                  Restart-data prefix: (none)
     Automatic parameter update (0/1): 1
  Write pointdata dimensionless (0/1): 0
             Output files prefix: naca0012_euler_Roe

 -------------------------------
 FLOW CONDITIONS
 -------------------------------
       Viscous calculation (0/1): 0
           Reference Mach number: 0.5
           Reference temperature: 293.0
                 Reynolds number: 1.e6
                 Reynolds length: 1
            Angle alpha (degree): 3.0
                      Grid scale: 1.0
```

```
------------------------------
SOLVER
------------------------------

   ---------------------------
   Inviscid fluxes
   ---------------------------
     Solver type upwind/central (0/1): 0
                       Central solver: Classic_dissipation
      Central classic dissipation 2nd: 0.5

           Second order upwind solver: Roe
              Ausm scheme dissipation: 0.25

   ---------------------------
   Nonlinear solver
   ---------------------------
                     CFL number: 1.0
              Relaxation solver: Runge_Kutta
       Number Runge-Kutta steps: 3
       Runge-Kutta coefficients: 0.15 0.5 1 0 0 0
       Maximal time step number: 5000
               Minimum residual: 1e-7

------------------------------------------------------
Cut-Plane Data Output
------------------------------------------------------
       Plane output description file: (thisfile)
                 Plane output values: xyz_cp
                 Plane output period: 999999
                    Number of planes: 1
       Boundary type (euler/viscous): viscous
                     Plane support x: 0
                     Plane support y: 0.5
                     Plane support z: 0
                      Plane normal x: 0
                      Plane normal y: 1
                      Plane normal z: 0
------------------------------------------------------
Surface Data Output
------------------------------------------------------
     Surface output description file: (thisfile)
                                Surface output values: x_y_z_cp
                                Surface output period: 999999
------------------------------------------------------
Field Data Output
------------------------------------------------------
       Field output description file: (thisfile)
                 Field output values: cp_mach


------------------------------------------------------
UPDATE
------------------------------------------------------

   solver at Sun Jul  9 13:06:17 2006
             Restart-data prefix: naca0012_euler_Roe.pval.5000
```

Notice that the last line is appended automatically when solver completes successfully. Then the pval-file may be used for restarting the calculation.