Chapter 12
Numerical Solutions of Viscous Laminar Flows

OBJECTIVES AND GUIDELINES

This chapter will guide you into the programming of CFD codes for viscous laminar flows, modeled by the Navier–Stokes equations, where the main additions to the inviscid model are the viscous internal friction stresses and the thermal conduction, both being diffusion type terms, represented by second order derivatives. The basic equations are summarized in Section 12.1.

This is a major change from inviscid flows, as it affects the solid wall boundary conditions, where all the velocity components have to vanish, following the non-slip boundary condition for viscous flows. The main consequence of it is the presence of viscous and thermal boundary layers near solid walls, where the velocity and temperature profiles can vary extremely rapidly over a short distance, requiring hereby high density grids in these near-wall regions.

We will consider essentially low speed flows and this will provide us with the opportunity to introduce an alternative numerical approach for the solution of the Navier–Stokes equations, namely the pressure correction method. The method is widely used, also in several commercial codes, and was initiated many years ago, for low speed industrial flows.

This particular numerical approach is to be put in relation with the method presented in the previous chapter for the Euler equations, which has been developed within the aeronautical community and oriented at high speed flows where the variations of density can become dominant. Hence, they are considered as density-based methods. Today, both families of methods have been extended to handle flows at all speed regimes, although the pressure correction methods might display some difficulties for high supersonic flow conditions.

Very few viscous test cases are known having analytical solutions. Among the best known is the Couette flow, generated by the uniform motion of a flat plate at a certain distance from a fixed plate, leading to a linear velocity distribution as a result of the viscous effects. When a temperature difference is imposed between the plates, the exact parabolic solution for the temperature profile allows the verification of thermal effects.

Another fundamental test case is the laminar boundary layer flow over a flat plate, with the well-known Blasius profile, considered as an exact solution. Despite its simple geometry, it displays most of the flow features encountered in viscous external or internal flows, such as the flow over a wing or a turbomachinery blade row, with the presence of leading edge, although of zero thickness, a boundary layer flow and a wake.

Section 12.2 will first guide you to the extension toward viscous flows of the program developed for the Euler equations in Chapter 11. This is not so straightforward

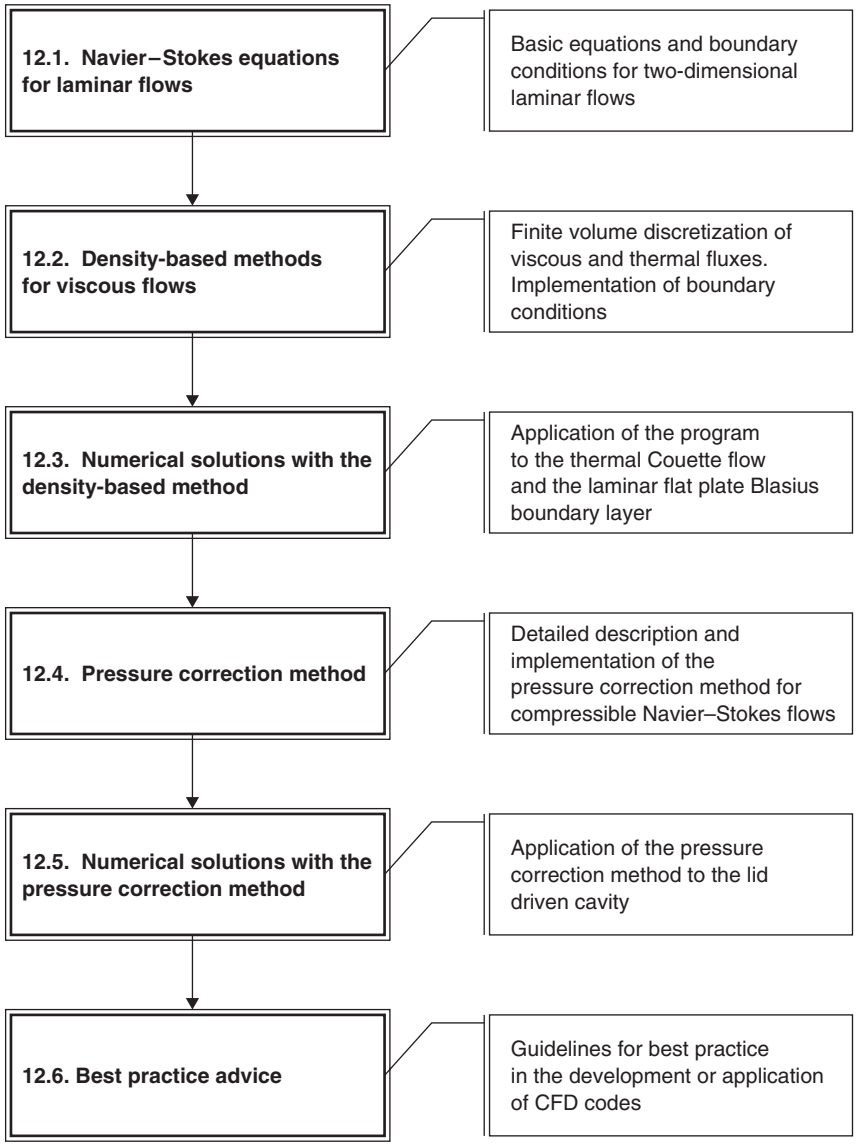

Figure 12.0.1 *Content and guide to this Chapter.*

extension, particularly on a general finite volume structured grid, as it requires the addition of the flux terms associated to the central discretized viscous and heat conduction fluxes, and a modification of the wall boundary conditions. You will be able to apply your program to the thermal Couette flow and to the laminar flat plate boundary layer flow, following Section 12.3.

Section 12.4 will introduce you to the pressure correction method and will guide you to its programming, with an application to the well-known test case of the lid driven

cavity flow in Section 12.5. The lid driven cavity case has no analytical solution, but is representative of a complex internal flow structure and is considered as a standard test case.

Finally the last section 12.6 will provide some Best Practice Advice for either the development of a CFD code, or the use of existing, commercial or research, CFD codes.

Figure 12.0.1 summarizes the roadmap of this chapter.

Sections 12.2 and 12.3 have been written with the active participation of Dr. Benoit Tartinville, from Numeca International, who produced also the results of the two viscous cases of Section 12.3. Dr. Sergey Smirnov, from the Vrije Universiteit Brussel, contributed significantly to Section 12.4 on the pressure correction method, and produced the program and the results for the lid driven cavity of Section 12.5. Their contribution is gratefully acknowledged.

12.1 NAVIER–STOKES EQUATIONS FOR LAMINAR FLOWS

The system of Navier–Stokes equations can be written in various ways based on the derivations in Chapter 1. We refer you to Table 1 of Chapter 1 for a summary of the conservation laws in different forms. A general differential form is provided by the following equations, in absence of external forces and heat fluxes:

$$\begin{aligned}
&\frac{\partial \rho}{\partial t} + \vec{\nabla}\cdot(\rho\vec{v}) = 0 \\
&\frac{\partial \rho\vec{v}}{\partial t} + \vec{\nabla}\cdot(\rho\vec{v}\otimes\vec{v} + p\bar{\bar{I}} - \bar{\bar{\tau}}) = 0 \\
&\frac{\partial \rho E}{\partial t} + \vec{\nabla}\cdot(\rho\vec{v}H - k\vec{\nabla}T - \bar{\bar{\tau}}\cdot\vec{v}) = 0
\end{aligned} \tag{12.1.1}$$

Assuming Newtonian fluids, the shear stress tensor has the following Cartesian components, based on equation (1.3.6):

$$\tau_{ij} = \mu\left[\left(\frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j}\right) - \frac{2}{3}(\vec{\nabla}\cdot\vec{v})\delta_{ij}\right] \tag{12.1.2}$$

In two-dimensions and Cartesian coordinates, we obtain the system:

$$\begin{aligned}
&\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} + \frac{\partial \rho v}{\partial y} = 0 \\
&\frac{\partial \rho u}{\partial t} + \frac{\partial(\rho u^2 + p)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} = \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} \\
&\frac{\partial \rho v}{\partial t} + \frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2 + p)}{\partial y} = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} \\
&\frac{\partial \rho E}{\partial t} + \frac{\partial(\rho uH)}{\partial x} + \frac{\partial(\rho vH)}{\partial y} = \frac{\partial(\tau_{xx}u + \tau_{xy}v)}{\partial x} + \frac{\partial(\tau_{yx}u + \tau_{yy}v)}{\partial y} \\
&\qquad + \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)
\end{aligned} \tag{12.1.3}$$

The right-hand side collects the terms to be added to the inviscid fluxes.

The energy equation can be written in many equivalent forms, for instance equation (1.4.15) for the internal energy:

$$\rho \frac{de}{dt} \equiv \rho \left[\frac{\partial e}{\partial t} + (\vec{v} \cdot \vec{\nabla}) e \right] = -p(\vec{\nabla} \cdot \vec{v}) + \vec{\nabla} \cdot (k \vec{\nabla} T) + (\bar{\bar{\tau}} \cdot \vec{\nabla}) \cdot \vec{v} \tag{12.1.4}$$

For incompressible flows, the above equations simplify further, since the continuity equation reduces to the condition of divergence free velocity:

$$\vec{\nabla} \cdot \vec{v} = 0 \tag{12.1.5}$$

and the shear stress definition becomes

$$\tau_{ij} = \mu \left(\frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j} \right) \tag{12.1.6}$$

leading to the following non-conservative momentum equation (1.4.40)

$$\frac{\partial \vec{v}}{\partial t} + (\vec{v} \cdot \vec{\nabla}) \vec{v} = -\frac{1}{\rho} \vec{\nabla} p + \nu \Delta \vec{v} \tag{12.1.7}$$

and the energy equation for the temperature, writing $e = c_{\mathrm{v}} T$

$$\rho c_{\mathrm{v}} \left[\frac{\partial T}{\partial t} + (\vec{v} \cdot \vec{\nabla}) T \right] = \vec{\nabla} \cdot (k \vec{\nabla} T) + (\bar{\bar{\tau}} \cdot \vec{\nabla}) \cdot \vec{v} \tag{12.1.8}$$

Note that for liquids the specific heat coefficients at constant volume and constant pressure are equal, i.e. $c_{\mathrm{v}} = c_{\mathrm{p}}$ and either values can be used in equation (12.1.8), when applied to incompressible liquids.

You can choose to discretize directly this system by applying finite differences, if you can generate a Cartesian grid for your problem. For a general curvilinear grid, we recommend that you apply the finite volume method, by extending directly the program you have developed for the Euler equations.

With reference to equation (11.4.1), we have to add the contributions from the viscous and thermal fluxes to the momentum and energy conservation equations.

This gives the following finite volume discretized equation:

$$\frac{\mathrm{d}}{\mathrm{d}t} [\overline{U}_{i,j} \Omega_{i,j}] = -\sum_{\text{faces}} [\vec{F}^* \cdot \Delta \vec{S} - \vec{F}_{\mathrm{v}} \cdot \Delta \vec{S}] \equiv -R_{i,j} \tag{12.1.9}$$

where $\vec{F}_{\mathrm{V}}$ represents the viscous and thermal fluxes, with components f_v and g_v:

$$f_v = \begin{vmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xx} u + \tau_{xy} v + k \dfrac{\partial T}{\partial x} \end{vmatrix} \quad g_v = \begin{vmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yx} u + \tau_{yy} v + k \dfrac{\partial T}{\partial y} \end{vmatrix} \tag{12.1.10}$$

The estimation of the viscous and thermal fluxes requires the calculation of the velocity and temperature gradients. ***Since these terms describe diffusion effects, they have to be discretized by central formulas***.

12.1.1 Boundary Conditions for Viscous Flows

The presence of the viscous and thermal fluxes makes the time-dependent Navier–Stokes equations of the mixed parabolic–hyperbolic type, as seen in Chapter 3, and this has an impact on the number and nature of boundary conditions to be imposed, in particular at solid walls.

We consider here high Reynolds number flows, which is the most often occurring situation in both industrial as environmental systems. If you recall the significance of the Reynolds number as the ratio of convective to viscous effects, a high Reynolds number means that the flow system will be dominated by the convective terms, that is will be dominated by its inviscid properties. This can be clearly seen when analyzing viscous flows, with the important exception of the near-wall regions, where the viscous effects, leading to a boundary layer configuration, dominate the flow behavior.

The main consequence is that at inlet and outlet boundaries, we can keep the same boundary conditions as for the inviscid computations, but the wall boundary conditions will change drastically.

At solid walls, the velocity relative to the wall has to vanish. This is the ***no-slip boundary condition***, whereby the three velocity components are zero at the body surface.

We have to add a boundary condition for the temperature status of the solid wall. You can encounter several situations; the most current being:

- ***Adiabatic walls***: Whereby we express that there is no heat flux through the solid surface. This is expressed by the Neumann condition

$$\frac{\partial T}{\partial n} = 0 \quad \text{at the solid wall} \tag{12.1.11}$$

- ***Constant temperature wall***: Here we assume that the solid surface is kept at a fixed temperature T_w, leading to a Dirichlet type boundary condition

$$T = T_w \quad \text{at the solid wall} \tag{12.1.12}$$

- ***Imposed heat flux***: In this case, the solid wall is the source of a fixed heat flux q_e to or from the fluid flow, for instance when the solid surface is part of a heat exchanger system. This flux will be positive for a heated wall or negative for a cooled wall. The boundary condition generalizes the adiabatic condition (12.1.11) to

$$\frac{\partial T}{\partial n} = q_e \quad \text{at the solid wall} \tag{12.1.13}$$

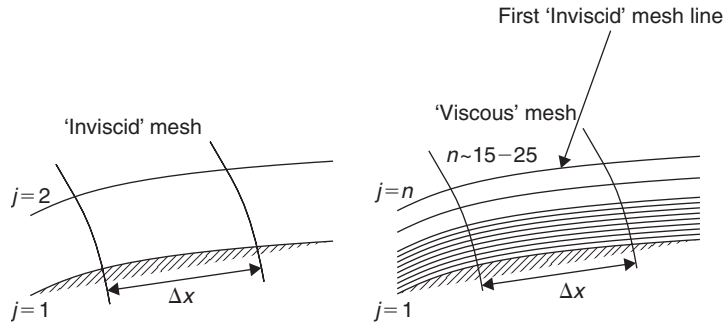

Figure 12.1.1 *Difference between an inviscid grid and a boundary layer oriented grid.*

12.1.2 Grids for Boundary Layer Flows

The presence of boundary layers has dramatic consequences on the grid requirements.

If you consider a flat plate, the developing boundary layer, as already discussed in Section 4.3, when we introduced the discretization for non-uniform grids, has a thickness to length ratio of the order of $1/\sqrt{\text{Re}}$. Hence, for a Reynolds number of 10^6 and a plate with a length of 1 m, the boundary layer thickness will be of the order of 1 mm at the end of the plate. For an incoming moderate velocity of say 30 m/s, the velocity will vary from zero at the wall to 30 m/s over a distance of 1mm, which is a very strong variation. As a consequence you need to arrange for a minimum number of grid points, of the order of 20–25 mesh points over this short distance, as illustrated on Figure 12.1.1. You can consider that the boundary layer mesh is inserted between the wall and the fist grid line of the inviscid mesh ($j = 2$). Moreover, since the velocity gradients take their highest values near the wall and decrease progressively toward the edge of the boundary layer, it is recommended to cluster the grid points close to the solid surface with a progressive coarsening when moving away from the wall. Refer to Section 4.3 for a discussion of this important issue.

Hence, viscous grids must be highly concentrated near solid walls.

12.2 DENSITY-BASED METHODS FOR VISCOUS FLOWS

To extend your inviscid finite volume program developed in Chapter 11, proceed as follows:

- Add an option in the main program to distinguish between inviscid or viscous simulations.
- Add subroutines to calculate the velocity and temperature gradients.
- Add subroutines to calculate the viscous stresses and thermal fluxes.
- Add subroutines for the viscous solid wall boundary conditions.

Referring to the cell-centered configuration of Figure 11.4.1, the viscous and thermal fluxes are calculated directly for each face, in a finite volume formulation, as described hereafter.

12.2.1 Discretization of Viscous and Thermal Fluxes

All the points and grid coordinates used in the following equations refer to Figure 11.4.1. Independently of the choice made for the inviscid part of the solver – either using an upwind or a central scheme – ***the viscous and thermal fluxes should always be discretized using a central scheme***.

Following equations (12.1.9) and (12.1.10), the viscous and thermal fluxes at the cell face $(\mathrm{i}+1/2, \mathrm{j})$ can be expressed as

$$(\vec{F}_v \cdot \Delta\vec{S})_{i+1/2,j} = (\vec{F}_v \cdot \Delta\vec{S})_{\mathrm{AB}} = f_{v,\mathrm{AB}}(y_\mathrm{B} - y_\mathrm{A}) - g_{v,\mathrm{AB}}(x_\mathrm{B} - x_\mathrm{A}) \quad (12.2.1)$$

The main difficulty resides in the dependence of the flux components f_v and g_v on the velocity and temperature gradients, and we have to decide how to estimate these gradients at the cell faces. Note that all quantities appearing in the flux terms are to be considered as ***face-averaged values***.

Again we are confronted with a 'local' decision, as several options can be selected to evaluate the gradients.

They can be computed directly at a mid-point of the cell face, or evaluated at the cell corners and averaged to obtain an average cell-face value.

Though the first approach is more direct, it is more expensive than the second one. Indeed for a two-dimensional calculation, the total number of cell faces is about two times the number of corners. Nonetheless, the first approach will be retained here, as it is more robust.

Thus, we can proceed as follows:

- Calculate the face velocity and temperature gradients by *programming a loop over all the cell faces*.
- Compute the face-averaged gradients by using the Gauss divergence theorem over a selected control volume, following equation (5.3.16), where the overbar indicates the cell average value:

$$\int_\Omega (\vec{\nabla}U)\mathrm{d}\Omega \triangleq \overline{\vec{\nabla}U}\Omega = \oint_S U\mathrm{d}\vec{S} \quad (12.2.2)$$

This equation is interpreted here in a two-dimensional space, where Ω is the face area and the contour integral is performed over the face boundaries.

- In order to obtain the gradients on face AB of Figure 12.2.1, we consider the control volume 1234 around AB, indicated as the shaded area. As faces 1 and 3 pass through the cell centers of cells (i, j) and $(i+1, j)$, the values at these points can be used directly. However, for the faces 2 and 4, an arithmetic average over four cells is required in order to compute the variables on these faces. Thus, the computation of the gradients at a cell face requires an access to the quantities at six different points. For instance, the gradient on face AB, labeled $(i+1/2, j)$, requires the quantities at points (i, j), $(i, j-1)$, $(i, j+1)$, $(i+1, j)$, $(i+1, j-1)$,

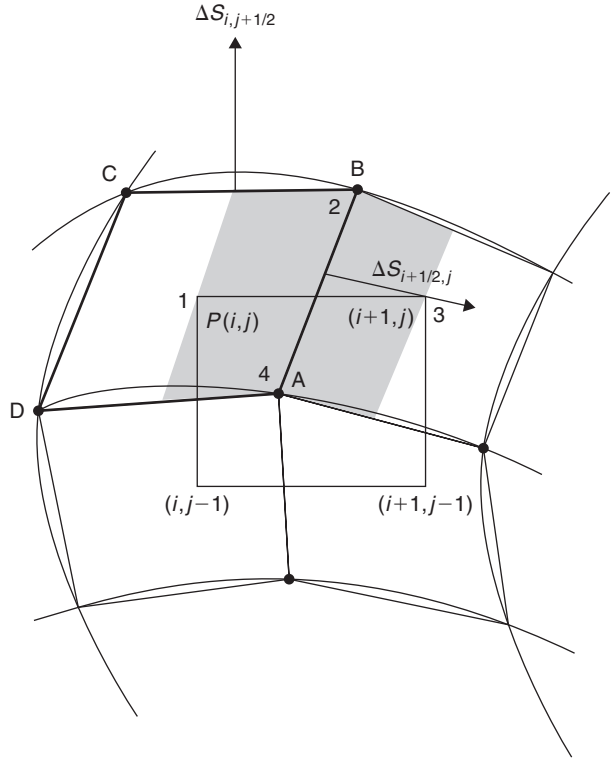

Figure 12.2.1 *Control volume used for computing the velocity and temperature gradients at face i + 1/2, j (dashed area). The faces around this volume are numbered 1234.*

and $(i+1, j+1)$ following:

$$\begin{cases} U_{\text{face1}} = U_{i,j} \\ U_{\text{face2}} = \dfrac{1}{4}(U_{i,j} + U_{i+1,j} + U_{i,j+1} + U_{i+1,j+1}) \\ U_{\text{face3}} = U_{i+1,j} \\ U_{\text{face4}} = \dfrac{1}{4}(U_{i,j} + U_{i+1,j} + U_{i,j-1} + U_{i+1,j-1}) \end{cases} \tag{12.2.3}$$

$$\overline{(\vec{\nabla}U)}_{i+1/2,j} = \frac{1}{\Omega_{1234}}(U_{\text{face1}}\Delta\vec{S}_{\text{face1}} + U_{\text{face2}}\Delta\vec{S}_{\text{face2}} + U_{\text{face3}}\Delta\vec{S}_{\text{face3}} + U_{\text{face4}}\Delta\vec{S}_{\text{face4}})$$

The x- and y-derivatives are obtained by projecting the average gradients in the corresponding direction. For instance, for the temperature gradients:

$$\begin{aligned} \left.\frac{\partial T}{\partial x}\right|_{i+1/2,j} &= (\vec{\nabla}T)_{i+1/2,j} \cdot \vec{e}_x \\ \left.\frac{\partial T}{\partial y}\right|_{i+1/2,j} &= (\vec{\nabla}T)_{i+1/2,j} \cdot \vec{e}_y \end{aligned} \tag{12.2.4}$$

- In case the viscosity is not constant, for instance when the temperature is variable, the dynamic viscosity at cell faces is estimated by using arithmetic averaging of cell-center values.
- Compute the shear stresses at cell faces by using equation (12.1.2), following:

$$(\tau_{xx})_{i+1/2,j} = \mu_{i+1/2,j}\left[2\left.\frac{\partial u}{\partial x}\right|_{i+1/2,j} - \frac{2}{3}\left(\left.\frac{\partial u}{\partial x}\right|_{i+1/2,j} + \left.\frac{\partial v}{\partial y}\right|_{i+1/2,j}\right)\right]$$

$$(\tau_{xy})_{i+1/2,j} = \mu_{i+1/2,j}\left[\left.\frac{\partial u}{\partial y}\right|_{i+1/2,j} = \left.\frac{\partial v}{\partial x}\right|_{i+1/2,j}\right] \tag{12.2.5}$$

- For faces located on boundaries, take into account the particular boundary conditions, as defined hereafter.
- Compute the fluxes using equation (12.2.1) and send the contribution to the right cell and its negative value to the left cell, in a counter-clockwise sense.

12.2.2 Boundary Conditions

As you certainly have experienced by now, if you have followed the code the development in Chapter 11, the numerical translation of the boundary conditions is one of the most critical issues in CFD.

12.2.2.1 *Physical boundary conditions*

As already mentioned above the boundary treatment described in Chapter 11 can still be applied for inlet and outlet boundaries. But the boundary conditions on walls have to be adapted to the viscous flow solver. Therefore, it is recommended to make two independent subroutines for inviscid and viscous wall boundary conditions.

As mentioned above, the major change compared to the inviscid code is the solid wall no-slip boundary condition, which imposes that all the velocity components vanish at the solid body surface. This is expressed by the Dirichlet type condition:

$$\vec{v} = 0 \quad \text{at the solid walls.} \tag{12.2.6}$$

The computation of the cell-face gradients, following equation (12.2.3), requires an access to all the neighboring cell values, which are not available for faces on the solid boundaries. Hence, a special treatment has to be implemented for these cells, by selecting control volumes entirely inside the computational domain. The control volume 1234, defined to compute the gradients at wall faces, is modified according to Figure 12.2.2, and the formulas (12.2.3) are replaced by

$$\begin{cases} U_{\text{face1}} = U_{i,j} \\ U_{\text{face2}} = \dfrac{1}{4}(U_{i,j} + U_{i+1/2,j} + U_{i,j+1} + U_{i+1/2,j+1}) \\ U_{\text{face3}} = U_{i+1/2,j} \\ U_{\text{face4}} = \dfrac{1}{4}(U_{i,j} + U_{i+1/2,j} + U_{i,j-1} + U_{i+1/2,j-1}) \end{cases}$$

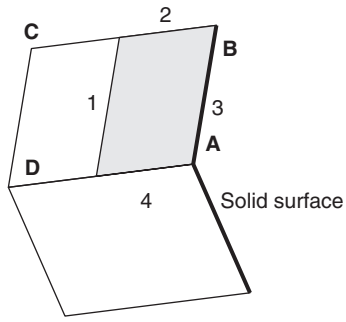

Figure 12.2.2 *Control volume used for computing the velocity and temperature gradients at face $i+1/2, j$ which corresponds to a viscous wall (dashed area). The faces around this volume are numbered 1234.*

$$(\overline{\vec{\nabla} U})_{i+1/2,j} = \frac{1}{\Omega_{1234}}(U_{\text{face1}}\Delta\vec{S}_{\text{face1}} + U_{\text{face2}}\Delta\vec{S}_{\text{face2}} + U_{\text{face3}}\Delta\vec{S}_{\text{face3}} + U_{\text{face4}}\Delta\vec{S}_{\text{face4}}) \tag{12.2.7}$$

where the values at locations $i+1/2$ are the values imposed on the solid boundaries.

This requires an additional loop on all the cells close to the solid body.

Another method, which does not require a special loop over the boundary cells, consists in adding a row of additional cells outside of the computational domain. The flow variables are imposed in these 'ghost' cells in order to match the required boundary condition at the corresponding cell faces. The external loop on all the faces of the computational domain can be directly used. Though this method is less time consuming it requires to store variables in 'ghost' cells and therefore requires more memory.

You are here confronted with a representative choice between time and memory consumption.

For the energy equations three boundary conditions can be used: adiabatic, isothermal and with imposed heat flux. Depending on the type of boundary condition specified, i.e. Dirichlet or Neumann, either the temperature or its gradient has to be specified on the wall cell face.

If a Neumann type condition is applied the temperature gradient is directly defined and does not need to be calculated. The wall temperature can be inferred by writing the normal temperature gradient as a function of the temperature derivatives along grid lines:

$$(\vec{n}\cdot\vec{\nabla}T)_{i+1/2,j} = \frac{\Delta\vec{S}_{i+1/2,j}}{|\Delta\vec{S}_{i+1/2,j}|\Omega_{i,j}} \cdot \left[\Delta\vec{S}_{i+1/2,j}\frac{\partial T}{\partial\xi}\bigg|_{\text{wall}} + \Delta\vec{S}_j\frac{\partial T}{\partial\eta}\bigg|_{\text{wall}}\right] \tag{12.2.8}$$

where ξ, η represent the coordinate in the i,j directions, and assuming i as being the direction away from the wall. The surface $\Delta\vec{S}_j$ is defined as in equation (11.4.12).

This allows to estimate the gradients along the mesh line direction:

$$\left.\frac{\partial T}{\partial \xi}\right|_{\text{wall}} = -\frac{1}{|\Delta\vec{S}_{i+1/2,j}|^2}\left[\Delta\vec{S}_{i+1/2,j}\cdot\Delta\vec{S}_j\left.\frac{\partial T}{\partial \eta}\right|_{\text{wall}}\right] + \frac{\Omega_{i,j}}{|\Delta\vec{S}_{i+1/2,j}|}(\vec{n}\cdot\vec{\nabla}T)_{i+1/2,j} \tag{12.2.9}$$

The temperature difference along the j-direction can be deduced following:

$$\left.\frac{\partial T}{\partial \eta}\right|_{\text{wall}} = \frac{1}{2}(T_{i,j+1} - T_{1,j-1}) \tag{12.2.10}$$

and the temperature at the cell face can be computed as

$$T_{i+1/2,j} = T_{i,j} + \frac{1}{2}\left.\frac{\partial T}{\partial \xi}\right|_{\text{wall}} \tag{12.2.11}$$

If a Dirichlet type condition is imposed, the same procedure used for the momentum equations should be used. Either the subroutines that compute the gradients are adapted to this boundary condition, or the values inside the 'ghost' cells have to be adapted to the imposed condition.

12.2.2.2 *Numerical boundary conditions*

As for the inviscid flow solver, other variables have to be imposed on the solid wall boundaries. Since we have already imposed the condition on the velocity vector and on the temperature, only the pressure remains. A common assumption is that the pressure gradient normal to the wall vanishes. Since this gradient can be expressed as function of the pressure derivative along grid lines, following equation (12.2.8), we can write

$$(\vec{n}\cdot\vec{\nabla}p)_{i+1/2,j} = \frac{\Delta\vec{S}_{i+1/2,j}}{\left|\Delta\vec{S}_{i+1/2,j}\right|\Omega_{i,j}}\cdot\left[\Delta\vec{S}_{i+1/2,j}\left.\frac{\partial p}{\partial \xi}\right|_{\text{wall}} + \Delta\vec{S}_j\left.\frac{\partial p}{\partial \eta}\right|_{\text{wall}}\right] \tag{12.2.12}$$

Therefore, following the same procedure as for the temperature, the pressure difference along the i-direction can be obtained on the boundary cell faces, and the pressure at the wall can be computed.

The density follows from the knowledge of pressure and temperature at the wall surface.

12.2.2.3 *Periodic boundary conditions*

Periodic boundary conditions are often used when a periodicity is observed in the geometry of the domain to be meshed or in the flow motion to be represented. This is often the case in problems where the flow does not vary in a given direction, such

as axisymmetric problems, or for turbomachinery applications, where the geometry has a periodicity, allowing only one blade passage to be meshed.

Another application of a periodic boundary condition occurs for simple one-dimensional problems, for which the streamwise direction is infinite without any flow variation, and we need to handle this problem with a two-dimensional code. In this case, we define a two-dimensional mesh with a few cells in the streamwise direction, selecting 2 or 3 cells in that direction. The constancy of all variables in that direction will be modeled by imposing a periodicity condition to connect the upstream and downstream boundaries in the streamwise direction, expressing equality of all the flow variables between these two boundaries.

Such a boundary condition is used hereafter for the simulation of the one-dimensional Couette flow problem with your 2D code.

12.2.3 Estimation of Viscous Time Step and CFL Conditions

As for the inviscid flow solver, since we are not interested in the transient behavior, a local time step could be used. Therefore, each cell will progress at its maximum time step. Following equation (E.9.3.1) the local viscous time step is defined as

$$\Delta t_{i,j} \leq \text{VNN} \frac{\Omega_{i,j}^2}{8\mu(|\Delta\vec{S}_i|^2 + |\Delta\vec{S}_j|^2 + 2|\Delta\vec{S}_i \Delta\vec{S}_j|)}$$
$$\Delta\vec{S}_i = \frac{1}{2}(\Delta\vec{S}_{i+1/2,j} + \Delta\vec{S}_{i-1/2,j}) \quad \Delta\vec{S}_j = \frac{1}{2}(\Delta\vec{S}_{i,j+1/2} + \Delta\vec{S}_{i,j-1/2}) \tag{12.2.13}$$

where VNN is the von Neumann number, defined by $\alpha\Delta t/\Delta x^2$, for the one-dimensional diffusion equation. Here the diffusion coefficient is replaced by the kinematic viscosity for the momentum equation and by the thermal diffusivity for the energy equation.

For the complete Navier–Stokes equations the local time step to be applied is the minimum between the inviscid and viscous time steps, along the lines of equation (8.2.15).

12.3 NUMERICAL SOLUTIONS WITH THE DENSITY-BASED METHOD

We will guide you now to the application of the general finite volume code, developed along the guidelines just described in Section 12.2, to two problems with a simple geometry, where in fact Cartesian grids can be used. However, once you have acquired the first experience in solving viscous and thermal problems, you will also be able to apply your code to more general cases, such as the cylinder flow, the internal bump case of Chapter 11, and to many other cases you might be interested in, requiring more general curvilinear grids.

The first test case of the Couette flow is rather simple, as it does not have a streamwise variation and is essentially one-dimensional. Nevertheless, the incorporation of thermal effects makes the case of interest, as it will help us illustrate some of the requirements associated with thermal problems.

The second test case is the laminar flat plate boundary layer, with the Blasius profile as reference solution.

The exact solutions of these two test cases are for incompressible conditions, but will be treated here in the low Mach number compressible mode.

12.3.1 Couette Thermal Flow

One of the simplest cases to verify the discretization of viscous effects is the laminar flow between two parallel walls, a fixed wall and a moving one at a distance L with velocity U. In order to include the computation of the thermal fluxes both walls are considered as isothermal, at different temperatures. The moving upper wall has a higher fixed temperature than the static wall (see Figure 12.3.1).

The analytical solution for incompressible flow conditions is easily derived. Since the plates have infinite lengths, there is no physically relevant length scale in the streamwise direction and therefore all x-derivatives have to vanish. In addition, since the flow has only a streamwise velocity component, the normal velocity component is zero everywhere, i.e. $v = 0$, reducing the momentum equation to its streamwise component. The shear stress tensor (12.1.6) is also reduced to a single component τ_{12}.

$$\tau_{12} = \mu \frac{\partial u}{\partial y} \quad \text{all other } \tau_{ij} = 0 \tag{12.3.1}$$

Consequently, equations (12.1.7) and (12.1.8) simplify considerably to the one-dimensional system:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nu \frac{\partial u^2}{\partial y^2} \\ \rho c_v \frac{\partial T}{\partial t} &= k \frac{\partial^2 T}{\partial y^2} + \mu \left(\frac{\partial u}{\partial y} \right)^2 \end{aligned} \tag{12.3.2}$$

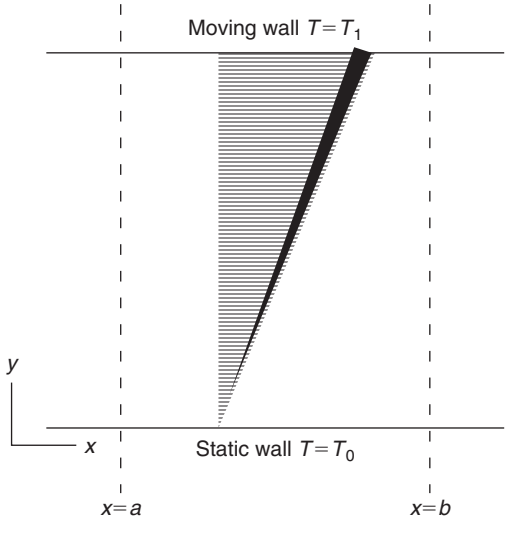

Figure 12.3.1 *Representation of the Couette flow test case.*

Note that these equations are parabolic in space and time, since all the convection terms have vanished.

They are to be solved with the Dirichlet boundary conditions for velocity and temperature:

$$\begin{array}{lll} y = 0 & u = 0 & T = T_0 \\ y = L & u = U & T = T_1 \end{array} \tag{12.3.3}$$

The steady state solution is now easily obtained, by setting the time derivatives to zero. The momentum equation reduces to

$$\frac{\partial^2 u}{\partial y^2} = 0 \tag{12.3.4}$$

leading to a linear velocity profile

$$u(y) = \frac{y}{L} U \tag{12.3.5}$$

where L denotes the distance between the two plates and U is the velocity of the moving wall.

Introducing this solution in the steady temperature equation, leads to

$$\frac{\partial^2 T}{\partial y^2} = -\frac{\mu}{k} \left(\frac{\partial u}{\partial y} \right)^2 = -\frac{\mu U^2}{k L^2} \tag{12.3.6}$$

This generates a parabolic temperature profile, depending on the parameter in the right-hand side.

The velocity scale is fixed by the upper wall velocity U; the temperature scale is determined by the temperature difference $\Delta T = T_1 - T_0$ and the length scale by the distance L between the plates. Hence, we define the non-dimensional variables

$$\tilde{U} = u/U \quad \tilde{T} \triangleq \frac{T(y) - T_0}{T_1 - T_0} \quad Y = y/L \tag{12.3.7}$$

and the above temperature equation becomes

$$\frac{\partial^2 \tilde{T}}{\partial Y^2} = -\frac{\mu U^2}{k \Delta T} \tag{12.3.8}$$

The right-hand side coefficient can be written as the product of the Prandtl and Eckert non-dimension numbers:

$$P_r = \frac{\mu c_\mathrm{p}}{k} \quad E_c = \frac{U^2}{c_\mathrm{p} \Delta T} \quad P_r E_c = \frac{\mu U^2}{k \Delta T} \tag{12.3.9}$$

The Eckert number is the ratio of the dynamic temperature induced by fluid motion to the characteristics temperature difference in the fluid.

The analytical velocity and temperature profiles are then easily obtained as

$$\tilde{U}(Y) = Y$$
$$\tilde{T}(Y) = \frac{T(y) - T_0}{T_1 - T_0} = \left[1 + \frac{1}{2} P_r E_c (1 - Y)\right] Y \tag{12.3.10}$$

The wall heat transfer coefficient is an important quantity in engineering applications and is generally expressed by the non-dimensional Nusselt number, defined here as a measure of the intensity of the heat flux via

$$\mathrm{Nu} \triangleq \frac{\partial \tilde{T}}{\partial Y} = \frac{L}{\Delta T} \frac{\partial T}{\partial y} \tag{12.3.11}$$

For the Couette flow, it takes the following values at the wall

$$\mathrm{Nu} = 1 + \frac{P_r E_c}{2} \quad \text{at } y = 0$$
$$\mathrm{Nu} = 1 - \frac{P_r E_c}{2} \quad \text{at } y = \mathrm{L} \tag{12.3.12}$$

An interesting property of this solution is that the fluid maximum temperature is greater than the upper wall temperature, when the product $P_r E_c$ is greater than two.

12.3.1.1 *Numerical simulation conditions*

If you exercise your critical judgment, referring to the scheme properties developed in Chapter 9, you might recognize that we have here a purely parabolic problem and wonder as to the adequacy of the application of the explicit Runge–Kutta time integration method to this diffusion dominated test case. It is true indeed that this option is not optimal for the Couette flow test case, but we wish to guide you here in the practice of a general finite volume code, valid for low and high speeds, so that you can verify for yourself the range of applications you can cover. As seen in Chapter 9, the domain of stability of the Runge–Kutta method includes a part of the negative real axis of the eigenvalue spectrum, and therefore it remains valid for pure diffusion problems.

A first issue is the treatment of a one-dimensional flow case, where nothing is happening in the x-direction. This can be treated by generating a two-dimensional mesh with a limited number of mesh points in the x-direction, and applying periodic boundary conditions at the two ends of the domain, between $x = a$ and $x = b$, in Figure 12.3.1. The periodic boundary conditions express that all quantities at $x = b$ are equal to their corresponding values at the same ordinate at $x = a$. In principle two or three mesh points should be sufficient in the x-direction.

Another issue is connected to the numerical values of the flow variables. Although the non-dimensional solution (12.3.10) is independent of the levels of temperature differences and physical distances between the two walls, your code is written for the physical variables and consequently their numerical values can influence the overall accuracy of the computed results. Moreover, we apply here a density-based code, for which we consider low compressible conditions, with a Mach number around 0.1.

As seen in Chapter 11, this is totally acceptable for the analysis of incompressible flows, assuming perfect gas relations for the considered fluid. Because of the thermal effects, we have to ensure that numerically the perfect gas relations still remain close to constant density conditions and therefore we have to limit the absolute values of the temperature difference between the two endplates.

Since the non-dimensional solution only depends on the product $P_r E_c$, this parameter defines completely the numerical solution, when solved in the non-dimensional form. As this is not the case here, we have to select all the physical quantities of the fluid and the physical set-up, in order to fully define the dimensional form of the solution.

We select here the following values:

- The fluid is a perfect gas with the following properties:
  Specific Heat: $c_p = 1006$ J/kg/K
  Gamma: $\gamma = 1.4$
  Kinematic viscosity: $\nu = 1.57 \times 10^{-5}$ m^2/s
  Prandtl number: $P_r = 0.708$.
- The Reynolds numbers based on the velocity of the moving wall is 4000.
- The physical conditions of this Couette flow are chosen as $P_r E_c = 4$, with the following variables set according to:
  $L = 0.83$ mm
  $T_0 = 293$°K
  $T_1 = 294$°K
  $U = 75.4$ m/s
- Note the very small temperature difference selected of 1°K, which requires double precision arithmetic.

Since this problem is slightly more complex than the simple Couette flow without any thermal effect, it is recommended that you verify first the implementation of the viscous effects, without taking into account the thermal fluxes, to check the obtained linear velocity distribution.

12.3.1.2 *Grid definition*

A regular grid has been set-up for this case with 65 points in-between walls and 3 points in the axial direction, over a length L.

The two lateral sides of the computational domain are connected assuming a periodic repetition of the channel.

In order to investigate the sensitivity of the numerical results to the grid density, coarser meshes are constructed by simply removing each second point in the vertical the wall-to-wall direction. Hereby we generate four different grids: (65×3), (33×3), (17×3) and (9×3).

12.3.1.3 *Results*

The Navier–Stokes equations are solved using the cell-centered approach until a steady state is reached, with CFL and Von Neumann numbers put to 1.8 and with the dissipation coefficient $\kappa^{(4)} = 1/100$.

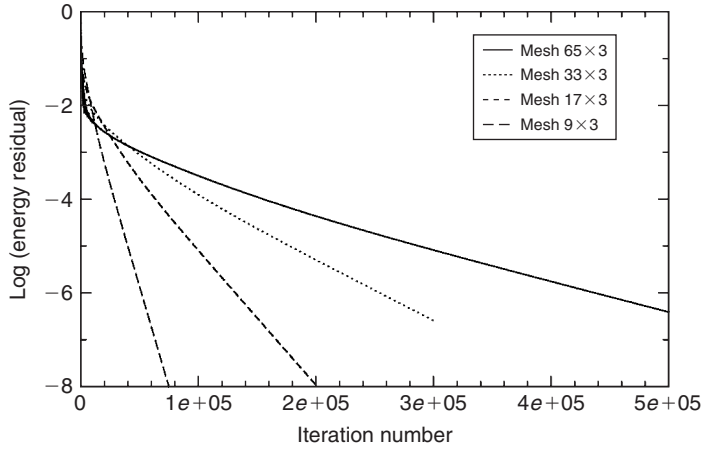

Figure 12.3.2 *Convergence history of the normalized L_2-norm of the energy residual on four different meshes.*

A Runge–Kutta time integration method is also applied with the coefficients provided by equation (11.4.11). The computations are stopped when the residuals have decreased by more than 6 orders of magnitude.

The convergence history of the energy equation is provided in Figure 12.3.2. The energy residual is decreased by about 8 orders of magnitude. Such a convergence level can only be obtained by using a double precision version of your code. It is seen that the rate of convergence strongly depends on the mesh used. A coarser mesh is associated to a more rapid convergence, whereas the convergence rate decreases if the grid density is refined.

The very slow convergence results largely from the application of the general code you have developed, which is oriented at 2D convection dominated inviscid or viscous flows. The Couette flow, on the other hand, is a pure one-dimensional parabolic problem since convection does not play any role in this particular case, and the applied algorithms are therefore not optimal.

Note also that in a more advanced code, multigrid acceleration will normally be available, reducing considerably the required number of iterations.

In practice, 3–4 orders of magnitude might be sufficient for an 'engineering solution', but is it important that you verify that your code can reach machine accuracy, to ensure that the algorithm is correctly programmed, in all its details.

The accuracy of the solution is also strongly influenced by the grid density used. On the one hand, all the meshes used are able to reproduce the linear distribution of axial velocity in-between walls (see Figure 12.3.3) demonstrating the second order accuracy of the scheme, ***for which a linear variation has to be exactly reproduced***. On the other hand, the quadratic distribution of temperature is not accurately captured when using too coarse meshes (see Figure 12.3.4). The meshes with 33 and 65 points from wall to wall are able to accurately predict the analytical solution. This finding illustrates that a sufficient refinement is necessary in order to capture the flow features present in laminar boundary layers.

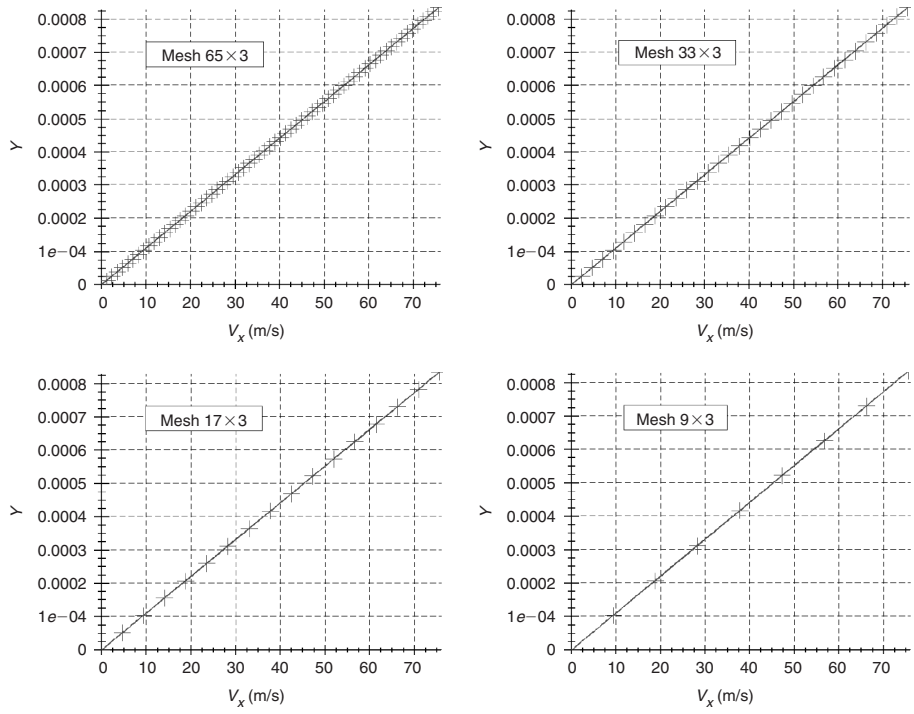

Figure 12.3.3 *Wall-to-wall distribution of axial velocity as deduced from analytical result (continuous line) and from the numerical result (plus signs) (for color image refer Plate 12.3.3).*

An important aspect of many engineering flow problems is the ability to predict local extrema of important flow variables. Since thermal stresses have an impact on solid structure lifetime, it is of importance to accurately predict the maximum temperature inside a flow. Therefore, we will perform a grid convergence study based on the prediction of the maximum temperature between the solid walls. According to equation (12.3.10) the maximum temperature inside the flow is obtained at a position $y/L = 3/4$ from the static wall and its value is 294.125 K. The computed maximum temperatures are reported in Table 12.3.1, including the relative error in % of the wall temperature difference. As expected, the maximum temperature error is reduced if the mesh is refined, and an error lower than 1% of the temperature variation can be obtained with a mesh having at least 33 grid cells in the wall normal direction. This number has to be doubled if we need a precision of less than 0.1%.

Another critical quantity in presence of thermal effects is the heat flux through a boundary, as expressed by the Nusselt number, defined here by equation (12.3.11). This is generally highly grid dependent, although in the present case, as the temperature gradient is linear, it will be less sensitive to the grid, as seen from Table 12.3.2.

12.3.1.4 *Other options for solving the Couette flow*

The method applied here, based on a more general finite volume formulation, is of course not optimal for this simple Couette flow. Since the problem is actually mono-dimensional, you can solve equations (12.3.2) and (12.3.3) much easier and

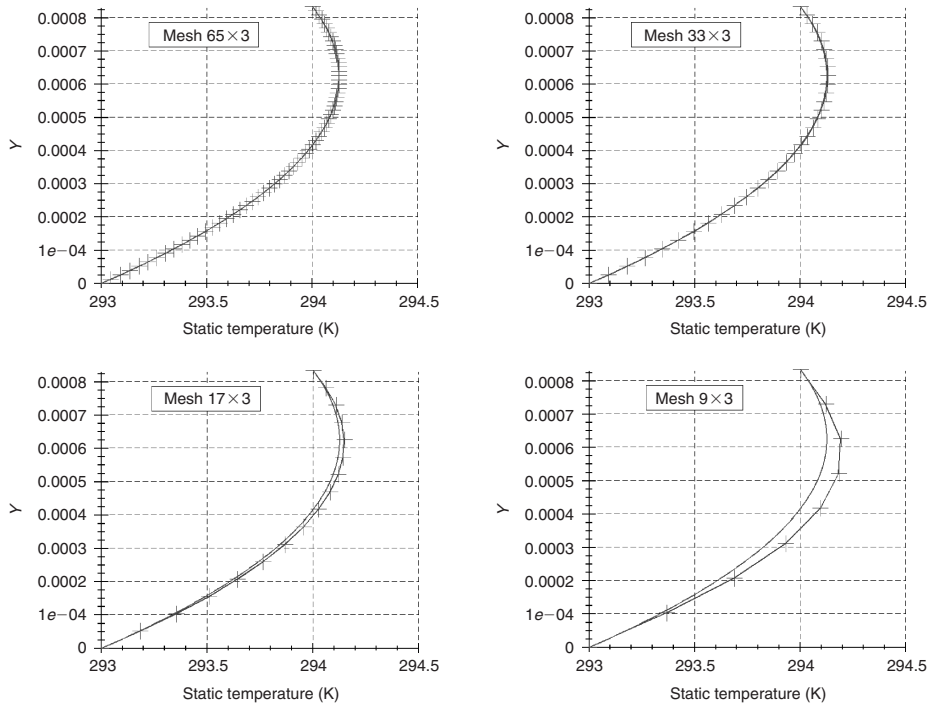

Figure 12.3.4 *Wall-to-wall distribution of static temperature (line with plus signs), compared to the analytical solution (continuous line), for the four different grids (for color image refer Plate 12.3.4).*

Table 12.3.1 *Maximum temperature computed using the different meshes.*

	Temperature (K)	*Error in temperature (K)*	*Relative error in temperature (%)*
Analytical	294.1250	–	–
65×3	294.1255	5×10^{-4}	0.05
33×3	294.1284	3.4×10^{-3}	0.34
17×3	294.1462	2.12×10^{-2}	2.12
9×3	294.1898	6.48×10^{-2}	6.48

Table 12.3.2 *Estimated values of the Nusselt number on several grids.*

	Nusselt number at lower wall	*Nusselt number at upper wall*	*Relative error at lower wall (%)*	*Relative error at upper wall (%)*
Analytical	3	−1	–	–
65×3	3.0072	−1.0034	0.24	0.34
33×3	3.0075	−1.0030	0.25	0.30
17×3	3.0070	−1.0025	0.23	0.25
9×3	3.0008	−0.9993	0.17	−0.07

faster, by applying a dedicated algorithm for parabolic problems. For instance an implicit Crank–Nicholson scheme to the centrally discretized diffusion terms, will lead to a much faster code. We recommend that you program equations (9.3.16), referring also to Problem P.9.10.

12.3.2 Flat Plate

One of the most popular applications of laminar viscous flows is the boundary layer development along a flat plate. The main advantages of this case are its relevance for a number of practical flow problems and the availability of an exact solution, obtained by solving the Blasius equation.

As for the Couette flow problem, we consider here weak compressible conditions, with a free stream Mach number around 0.2, which is still acceptable for the analysis of incompressible flows, assuming perfect gas relations for the considered fluid.

The fluid is a perfect gas fluid with the following properties:

Specific Heat: $c_p = 1006$ J/kg/K
Gamma: $\gamma = 1.4$
Kinematic viscosity: $\nu = 1.57 \times 10^{-5}$ m^2/s
Prandtl number: $P_r = 0.708$

We select the length of the plate equal to 0.2 m and the free stream velocity is fixed to match a Mach number of 0.2, leading to $U = 68.3$ m/s.

The Reynolds number based on the free stream velocity and the plate length is 8.7×10^5.

12.3.2.1 *Exact solution*

The exact solution of the development of a laminar boundary layer over a flat plate with an incoming uniform velocity U in the axial plat direction has been solved by Blasius nearly a century ago. It is based on the self-similarity of the velocity profiles along the plate for an incompressible fluid. A detailed description of the Blasius solution can be found in Schlichting (1979).

After a few simplifications of the Navier–Stokes equations, and assuming the boundary layer approximations, we obtain to the Blasius equation:

$$f\frac{\mathrm{d}^2 f}{\mathrm{d}\eta^2} + 2\frac{\mathrm{d}^3 f}{\mathrm{d}\eta^3} = 0 \tag{12.3.13}$$

with the following boundary conditions

$$\begin{aligned} f(0) &= 0 \\ f'(0) &= 0 \\ f'(\eta) &\to 1 \quad \text{if } \eta \to \infty \end{aligned} \tag{12.3.14}$$

where f is the function solution of the Blasius equation. η is a non-dimensional coordinate normal to the plate

$$\eta = \frac{y}{\left(\frac{\nu x}{U}\right)^{1/2}} \tag{12.3.15}$$

Equation (12.3.13) is an ordinary differential equation and can be solved numerically with an arbitrary accuracy. The two components of the velocity vector can be inferred from the function f

$$\begin{aligned} u &= U\frac{\mathrm{d}f}{\mathrm{d}\eta} \\ v &= \frac{1}{2}\left(\frac{U\nu}{x}\right)^{1/2}\left(\eta\frac{\mathrm{d}f}{\mathrm{d}\eta} - f\right) \end{aligned} \tag{12.3.16}$$

where U is the free stream velocity.

From this solution we can deduced the distribution of the friction coefficient along the single sided plate

$$C_f = \frac{0.664}{\sqrt{Re_x}} \quad \mathrm{Re}_x = \frac{Ux}{\nu} \tag{12.3.17}$$

The Blasius solution is tabulated in Table 12.3.3.

12.3.2.2 *Grid definition*

A first important decision is the selection of the computational domain. We have actually two options, each with its specific problems:

Option 1: Locate the computational domain boundaries upstream and downstream of the leading and trailing edges of the plate. This is the most realistic choice, with the advantage of allowing the simulation of the approach of the flow toward the leading edge and the downstream wake. However, it requires a very dense mesh around the leading and trailing edges, and the associated range of boundary conditions.

Option 2: Select the computational domain between the leading edge and the trailing edge of the plate. This avoids the grid concentrations of option 1, but creates a non-realistic flow at the leading edge, since a uniform flow is assumed at the leading edge.

Due to its simplicity we choose here the second option.

To close the computational domain, an outlet boundary has to be defined at a certain distance parallel to the plate. This boundary may not influence the development of the boundary layer and it should be placed sufficiently far from the plate. According to the Blasius solution, the thickness of the boundary layer at a distance x from its leading edge is of the order:

$$\delta_\infty(x) \cong 5\frac{x}{\sqrt{\mathrm{Re}_x}} \tag{12.3.18}$$

For the above-mentioned conditions, the boundary layer thickness at the end of the plate $x = 0.2$ m, is of the order of 1 mm. Therefore, the outlet boundary parallel to the

Table 12.3.3 *Blasius solution for the flat plate boundary layer.*

η	$\mathrm{d}f/\mathrm{d}\eta$	$\eta\,\mathrm{d}f/\mathrm{d}\eta - f$
0.00000E+00	0.00000E+00	0.00000E+00
0.14142E+00	0.46960E−01	0.33177E−02
0.28284E+00	0.93910E−01	0.13282E−01
0.42426E+00	0.14081E+00	0.29858E−01
0.56569E+00	0.18761E+00	0.53025E−01
0.70711E+00	0.23423E+00	0.82696E−01
0.84853E+00	0.28058E+00	0.11873E+00
0.98995E+00	0.32653E+00	0.16098E+00
0.11314E+01	0.37196E+00	0.20916E+00
0.12728E+01	0.41672E+00	0.26296E+00
0.14142E+01	0.46063E+00	0.32193E+00
0.15556E+01	0.50354E+00	0.38563E+00
0.16971E+01	0.54525E+00	0.45345E+00
0.18385E+01	0.58559E+00	0.52475E+00
0.19799E+01	0.62439E+00	0.59881E+00
0.21213E+01	0.66147E+00	0.67483E+00
0.22627E+01	0.69670E+00	0.75202E+00
0.24042E+01	0.72993E+00	0.82955E+00
0.25456E+01	0.76106E+00	0.90656E+00
0.26870E+01	0.79000E+00	0.98226E+00
0.28284E+01	0.81669E+00	0.10558E+01
0.31113E+01	0.86330E+00	0.11940E+01
0.33941E+01	0.90107E+00	0.13167E+01
0.36770E+01	0.93060E+00	0.14209E+01
0.39598E+01	0.95288E+00	0.15051E+01
0.42426E+01	0.96905E+00	0.15720E+01
0.45255E+01	0.98037E+00	0.16215E+01
0.48083E+01	0.98797E+00	0.16569E+01
0.50912E+01	0.99289E+00	0.16812E+01
0.53740E+01	0.99594E+00	0.16972E+01
0.56569E+01	0.99777E+00	0.17072E+01
0.59397E+01	0.99882E+00	0.17133E+01
0.62225E+01	0.99940E+00	0.17168E+01
0.65054E+01	0.99970E+00	0.17187E+01
0.67882E+01	0.99986E+00	0.17198E+01
0.70711E+01	0.99994E+00	0.17203E+01
0.73539E+01	0.99997E+00	0.17206E+01
0.76368E+01	0.99999E+00	0.17207E+01
0.79196E+01	0.99999E+00	0.17207E+01
0.82024E+01	0.10000E+01	0.17208E+01
0.84853E+01	0.10000E+01	0.17208E+01

wall is fixed at a distance of about 0.02 m from the plate, i.e. 20 times the boundary layer thickness.

This decision results from a compromise between accuracy and computational cost. Indeed, the location of the lateral boundary can be a source of errors, but putting this limit too far from the plate will result in a higher number of points in the free stream region where the flow is almost uniform.

Based on the above definition of the computational domain, a rectangular grid is set-up, with an adequate refinement in the regions of strong flow variations, i.e. in the leading edge region and close to the plate. Far from the plate the flow is almost uniform and allows coarser grid cells.

An exponential stretching technique is applied for which the ratio of two subsequent grid cells is constant:

$$\frac{\Delta y_i}{\Delta y_{i-1}} = a \tag{12.3.19}$$

and the coordinates of the grid points are given by

$$y_i = y_1 + \frac{a^{i-1} - 1}{a - 1}\Delta y_1 \tag{12.3.20}$$

Since in our application the plate is located at $y_1 = 0$, the grid distribution normal to the plate depends on the cell width at the first inner cell and on the number of cells. Since we want enough points in the laminar boundary layer, the cell spacing at the first inner cell will be fixed to 10^{-5} m, and we select 65 points in the wall normal direction. In order to have the outer point located at a distance 0.02 m from the plate, the factor a should be equal to $a = 1.083317311$, leading to a distribution of 28 cells located in the 1 mm thick boundary layer at $x = 0.2$ m.

Since the thickness of the boundary layer evolves as the square root of the distance to the leading edge, the mesh should also be refined in the streamwise direction. We select a streamwise distribution of grid points following also an exponential stretching as described in equation (12.3.20), with 65 points along the 0.2 long plate and the thickness of the first grid cell in the streamwise direction is fixed to 0.1 mm. Therefore, the parameter a is fixed to $a = 1.083317311$.

With these distributions in the streamwise and wall normal directions we can construct the regular mesh displayed in Figure 12.3.5.

The boundary in the far field ***must be treated as an outlet boundary*** through which mass flow can escape, since due to the boundary layer growth, the flow field has a non-zero vertical component at any distance from the wall. If this boundary is considered as an inviscid boundary on which the free stream velocity is imposed, this will impose a pressure force on the flat plate in contradiction with the physics of the problem as it will not allow the vertical velocity component to expand as it should from the theory.

In order to investigate the sensitivity of the numerical results to the grid density, coarser meshes are constructed by simply removing every second point in the wall normal direction, defining five different grids: (65×65), (65×33), (65×17), (65×9) and (65×5).

This will also allow you to evaluate the influence of the number of boundary layer grid points on the numerical accuracy. This is an important issue, as in practical

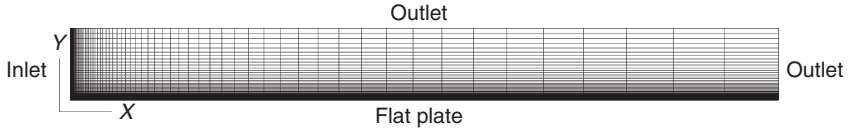

Figure 12.3.5 *Mesh defined for the laminar flat plate test case.*

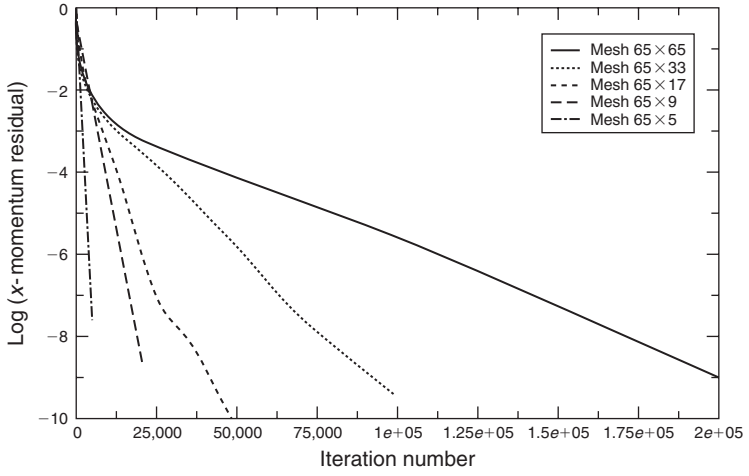

Figure 12.3.6 *Convergence history of the normalized L_2-norm of the x-momentum residual on five different meshes.*

industrial case, it is quite difficult and costly to ensure 25–30 points in boundary layers of complex 3D geometries.

Boundary conditions are specified at the inlet and outlet of the computational domain. Assuming atmospheric condition for the static temperature and pressure and an inlet Mach number of 0.2, the inlet total pressure and total temperature are fixed to 104165 Pa and 290.304 K, respectively (following equations (11.1.9) and (11.1.12)). The inlet velocity vector is imposed in the x-direction. At the outlet boundaries, as shown on Figure 12.3.5, the static pressure is fixed to the atmospheric value, i.e. 101300 Pa.

12.3.2.3 *Results*

The Navier–Stokes equations are solved using the cell-centered approach until a steady state is reached, with CFL and Von Neumann numbers put to 1.5 and the dissipation coefficient $\kappa^{(4)} = 1/100$.

A Runge–Kutta time integration method is also used with the coefficients provided by equation (11.4.11). The computations are stopped when the residuals have decreased by more than 6 orders of magnitude.

The convergence history of the x-momentum equation is provided in Figure 12.3.6. This residual is decreased by about 8 orders of magnitude. Such a convergence level

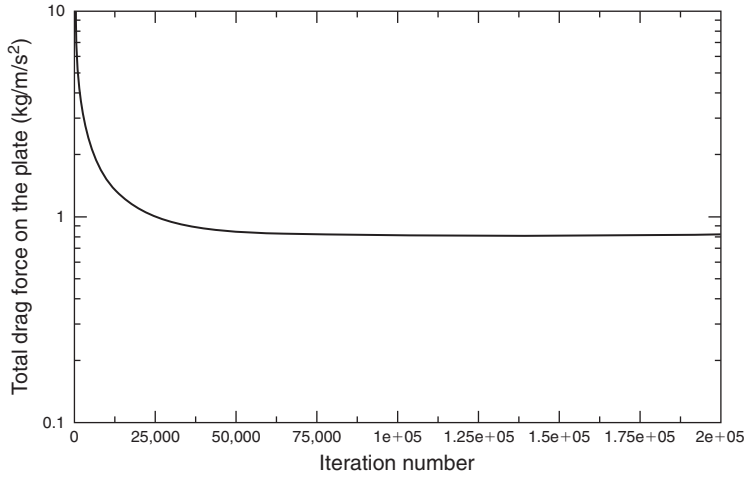

Figure 12.3.7 *Convergence history of the drag force along the plate for the finest mesh.*

can only be obtained by using a double precision version of your code. As for the Couette flow problem, it is seen that the rate of convergence strongly depends on the mesh used. A coarser mesh is associated to a more rapid convergence, whereas the convergence rate decreases if the grid density is refined.

You can see that after about 25,000 iterations the x-momentum residual on the finest mesh calculation has decreased by more than 3 orders of magnitude. From this observation, we could be led to judge that the simulation has reached a sufficient level of convergence and that it can be stopped. However, by monitoring the total drag force acting on the plate, it can be seen on Figure 12.3.7 that it we have to wait for more than 50,000 iterations, for a fully converged solution. Indeed, small cells located all along the plate require more iterations to reach a fully converged solution. Therefore, it is suggested to monitor not only the global residuals but also some key parameters in order to identify if a converged solution has been reached. This is particularly important for viscous calculations where numerous small cells should be located in the boundary layer.

Note that this high number of iterations would be considerably reduced by the addition of the multigrid technique.

The solution obtained on the finest mesh is displayed on Figure 12.3.8. It appears that, with a sufficient number of points in the boundary layer the solver is able to reproduce the analytical distribution of friction coefficient along the plate and the mainstream flow component.

Note however that in the region of the largest curvature of the axial velocity profile, the numerical results show a lower velocity compared to the exact solution, indicating the influence of numerical dissipation. This can be improved by more sophisticated dissipation terms, as obtained from formulations based on matrix dissipation, where the coefficients of the artificial dissipation terms (11.4.7) are replaced by terms containing the full Jacobian matrix, as opposed to its spectral radius (see for instance Jameson, 1995a, b). Alternatively, the second order upwind schemes, as will be seen

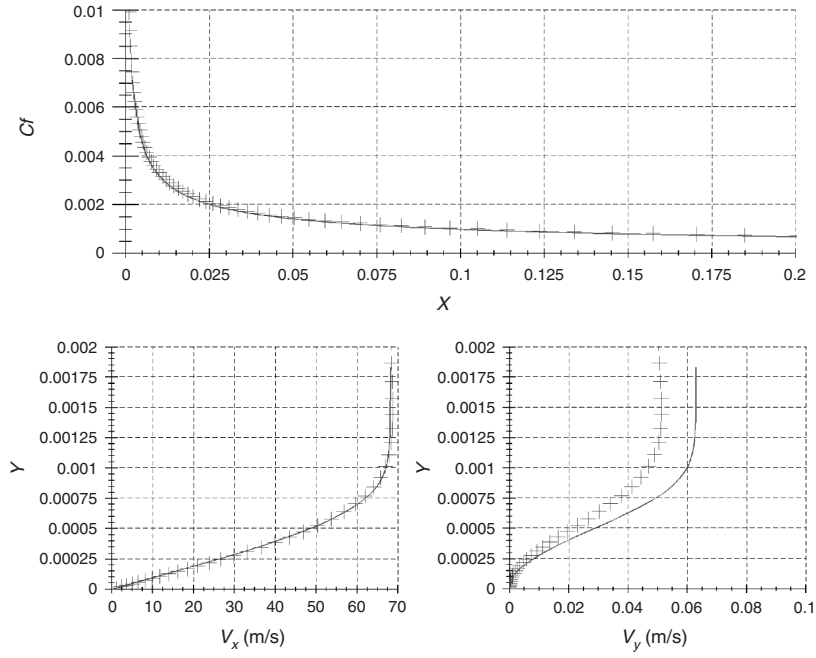

Figure 12.3.8 *Distribution of friction coefficient (upper panel), axial and wall normal velocities at $x = 0.2$ m (lower left and right panels) as obtained from analytical result (continuous line) and from the numerical result on the finest mesh (plus signs). The vertical axis is the distance in meters (for color image refer Plate 12.3.8).*

in Volume II, have also a reduced dependency to the number of boundary layer grid points.

The velocity in the wall normal direction differs significantly from the Blasius profiles, since relatively small errors in the axial velocity have a great impact on the wall normal velocity, as the latter is about three orders of magnitude lower than the former. This is also related to the large dissipation associated to the selected formulation of the central schemes.

An important outcome of viscous flow computations is the prediction of the friction coefficient along the solid surfaces. According to equation (12.3.17) the analytical drag coefficient at the end of the plate ($x = 0.2$ m) is equal to 0.00713. The computed drag coefficients are reported in Table 12.3.4 together with the relative error. As can be seen, the error in the friction coefficient is reduced if the mesh is refined, but is still at high values with the meshes used in the present calculations. The second column displays an interesting information, namely the number of mesh points in the inner part of the boundary layer, over the first one millimeter.

Accurate predictions of drag coefficients form a challenging and difficult issue in CFD, and represent one of the most sensitive criteria for accuracy assessment.

The interested reader might consult with interest the summary papers of recent workshops held on drag prediction evaluations of aeronautical relevant configurations,

Table 12.3.4 *Friction coefficient computed using the different meshes.*

	Number of cells in the boundary layer (first one millimeter)	*Cf*	*Relative error in Cf (%)*
Analytical		0.000713	–
65×65	28	0.00075	5.2
65×33	14	0.000803	12.7
65×17	7	0.000917	28.7
65×9	4	0.001371	92.3
65×5	2	0.002196	208.0

in Hemsch and Morrison (2004), also referred to in the general introduction to this book.

The errors on the friction coefficient are a consequence of the reduced accuracy on the velocity profiles, when the mesh is coarsened. This can be seen from Figure 12.3.9, where the distributions of skin friction and velocity are displayed on the two successive coarser grids, namely 65×33 and 65×17, having respectively, 14 and 7 points in the boundary layer. You will notice that 14 points might still be acceptable, for 'engineering' accuracy, but the error on the skin friction is of 12%, as seen from Table 12.3.4.

12.4 PRESSURE CORRECTION METHOD

The methods known as ***pressure correction*** are among the first developed for the numerical solutions of the full Navier–Stokes equations for incompressible flows. The method was originally applied by Harlow and Welch (1965) in the MAC, Marker-and-Cell, method for the computation of free surface incompressible flows. It is closely related to the ***fractional step method***, also called ***projection method***, developed independently by Chorin (1967), (1968) and Temam (1969); see also Temam (1977).

It has been adapted to industrial flow simulations by Patankar and Spalding (1972) and described in details in several books (Patankar (1980); Anderson (1995); Ferziger and Peric (1997); Wesseling (2001)).

The methods falling in this class can be applied to the stationary as well as to the time-dependent incompressible flow equations. They consist of a basic iterative procedure between the velocity and the pressure fields. For an initial approximation of the pressure, the momentum equation can be solved to determine the velocity field. The obtained velocity field does not satisfy the divergence free, continuity equation and has therefore to be corrected. Since this correction has an impact on the pressure field, a related pressure correction is defined, obtained by expressing that the corrected velocity satisfies the continuity equation. This leads to a Poisson equation for the pressure correction.

The pressure correction methods have been extended to compressible flows, and various approaches can be defined. Weak compressibility can be handled through a simplified system of equations obtained by developing a low Mach number expansion of the Navier–Stokes equations, omitting then the terms that are of higher order in

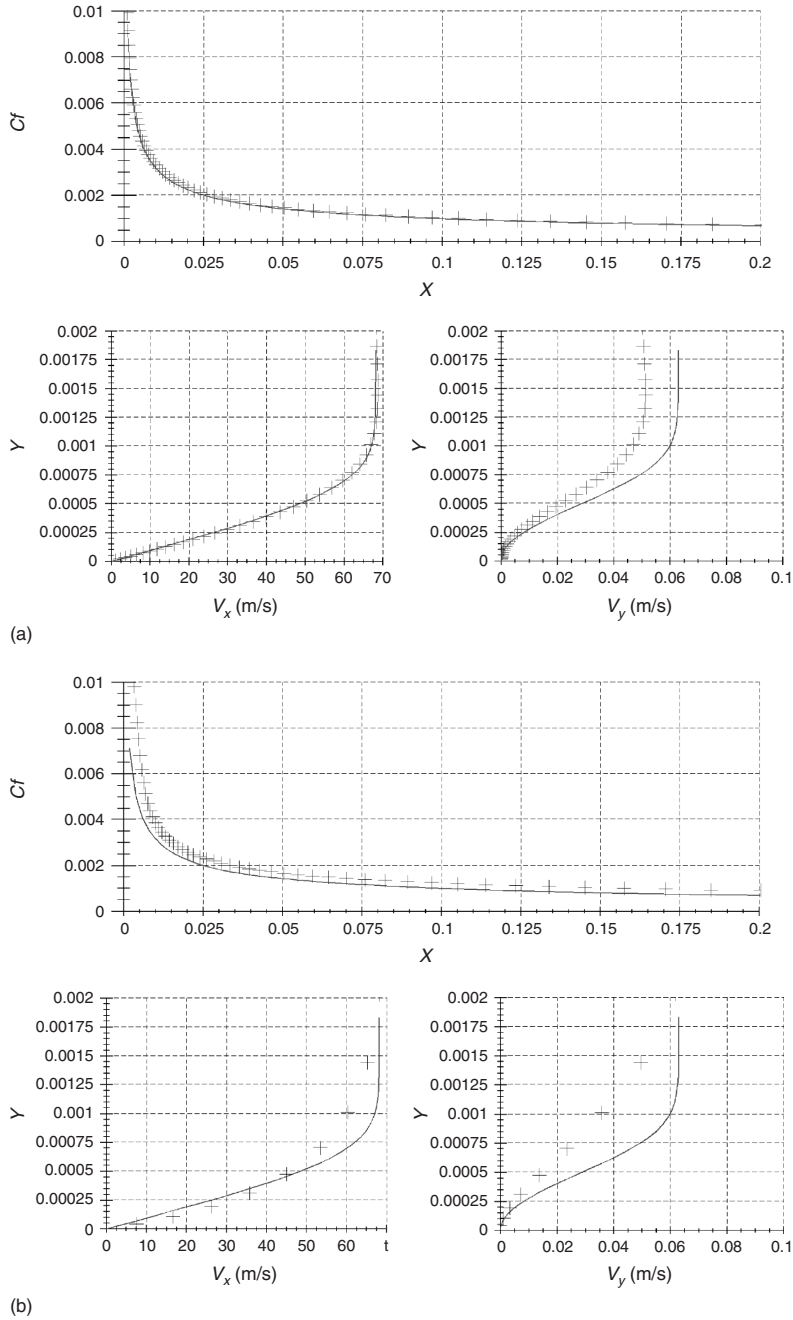

Figure 12.3.9 *Distributions of skin friction and velocity on the two successive coarser grids of 65 × 33 and 65 × 17, having respectively, 14 and 7 points in the boundary layer. (a) Numerical solution on 65 × 33 mesh with 14 points in the boundary layer and (b) numerical solution on 65 × 17 mesh with 7 points in the boundary layer.*

the Mach number (see, for instance, Majda and Sethian (1985)). A full compressible extension can be found in the book of Ferziger and Peric (1997).

12.4.1 Basic Approach of Pressure Correction Methods

We restrict the presentation here to strictly incompressible, isothermal flows, defined in Section 1.4.4 by the system formed by the continuity and momentum equations, written here in non-conservative form.

The mass conservation equation reduces in the case of incompressible flows to

$$\vec{\nabla} \cdot \vec{v} = 0 \tag{12.4.1}$$

which appears as a constraint to the general time-dependent equation of motion, written here in absence of external volume forces

$$\frac{\partial \vec{v}}{\partial t} + (\vec{v} \cdot \vec{\nabla})\vec{v} = -\frac{1}{\rho}\vec{\nabla} p + \nu \Delta \vec{v} \tag{12.4.2}$$

The only unknowns are velocity and pressure.

An equation for the pressure can be obtained by taking the divergence of the momentum equation (12.4.2), and introducing the divergence free velocity condition (12.4.1), leading to

$$\frac{1}{\rho}\Delta p = -\vec{\nabla} \cdot (\vec{v} \cdot \vec{\nabla})\vec{v} \tag{12.4.3}$$

which can be considered as a Poisson equation for the pressure for a given velocity field. Note that the right-hand side contains only products of first order velocity derivatives, because of the incompressibility condition (12.4.1). Indeed, in tensor notations, the velocity term in the right-hand side term is equal to $(\partial_j v_i) \cdot (\partial_i v_j)$.

Before describing the pressure correction method we have to select a time integration scheme for the momentum equations, considering the pressure gradient as known. For reasons of simplicity and in order to point out the essential properties of the pressure correction approach, we will select an explicit method of first order accuracy in time, although it is not recommended in practice. Even for time-dependent problems the time step restriction imposed by stability conditions for the parabolic, convection–diffusion momentum equations is generally smaller than the physical time constant of the flow. Hence, the time steps allowed by the requirements of physical accuracy are large enough to allow the larger numerical time steps of implicit schemes. Typically, semi-implicit time integration schemes are recommended. For instance, the viscous fluxes can be treated implicitly by means of the Crank–Nicholson formulation, while convective fluxes can be handled with an Adams–Bashworth second order method. This provides a higher accuracy for time-dependent simulations and allows for large time steps leading to more efficient calculations in terms of computational costs.

The fundamental approach of pressure correction methods is the decoupling of the pressure field from the velocity field. This is expressed by solving the momentum

equation with a known pressure field, for instance with the pressure obtained at the previous iteration.

A variety of methods have been developed and applied in practice, based on various decoupling approaches. In its simplest form, with an explicit time discretization, we solve for an intermediate velocity field $\vec{v}^*$, solution of

$$\frac{\vec{v}^* - \vec{v}^n}{\Delta t} = -\vec{\nabla} \cdot (\vec{v} \otimes \vec{v})^n - \frac{1}{\rho}\vec{\nabla} p^n + \nu \Delta \vec{v}^n \tag{12.4.4}$$

The solution $\vec{v}^*$ of this equation does not satisfy the continuity equation. Hence the final values are defined by adding corrections to the intermediate values

$$\vec{v}^{n+1} = \vec{v}^* + \vec{v}' \quad p^{n+1} = p^n + p' \tag{12.4.5}$$

where the final values with superscript $n+1$ have to be solutions of

$$\begin{aligned} \frac{\vec{v}^{n+1} - \vec{v}^n}{\Delta t} &= -\vec{\nabla} \cdot (\vec{v} \otimes \vec{v})^n - \frac{1}{\rho}\vec{\nabla} p^{n+1} + \nu \Delta \vec{v}^n \\ \vec{\nabla} \cdot \vec{v}^{n+1} &= 0 \end{aligned} \tag{12.4.6}$$

Introducing (12.4.5) in the above equation and subtracting (12.4.4), leads to the following relation between the pressure and velocity corrections:

$$\vec{v}' = -\frac{\Delta t}{\rho}\vec{\nabla} p' \tag{12.4.7}$$

Note that expressing the velocity correction as a gradient of a scalar function conserves the vorticity of the intermediate velocity field. That is, the correction field is a potential flow.

Taking the divergence of the first of the equations (12.4.6) gives the Poisson equation for the pressure correction:

$$\Delta p' = \frac{\rho}{\Delta t}\vec{\nabla} \cdot \vec{v}^* \tag{12.4.8}$$

Equation (12.4.3) assumes that the solution at time level n satisfies exactly the divergence-free condition. In the numerical process, the velocity at level n might not satisfy exactly this condition. In this case, the non-zero value of $D^n \stackrel{\Delta}{=} \vec{\nabla} \cdot \vec{v}^n$ should be introduced in the pressure Poisson equation. This situation is more likely to occur in stationary computations where n represents an iteration count. With time-dependent calculations, it is recommended to satisfy accurately mass conservation at each time step, in particular by discretizing the integral form of the mass conservation law on a finite volume mesh.

The Poisson equation for the pressure is solved with Neumann boundary conditions, on the normal pressure gradient, obtained by taking the normal component of equations (12.4.6). The details of the implementation depend on the selected space discretization and on the mesh.

An alternative approach is the ***fractional step, or projection***, method based on a slightly different definition of the intermediate pressure, whereby the pressure term is simply omitted, leading to the complete decoupling of the intermediate velocity field

$$\frac{\vec{v}^* - \vec{v}^n}{\Delta t} = -\vec{\nabla} \cdot (\vec{v} \otimes \vec{v})^n + \nu \Delta \vec{v}^n \tag{12.4.9}$$

followed by the pressure equation

$$\frac{\vec{v}^{n+1} - \vec{v}^*}{\Delta t} = -\frac{1}{\rho} \vec{\nabla} p^{n+1} \tag{12.4.10}$$

The final value of velocity field is obtained from equation (12.4.10). The pressure is calculated in such a way that the velocity field at level $(n+1)$ satisfies the divergence free condition:

$$\frac{\rho}{\Delta t} [\vec{\nabla} \cdot \vec{v}^{n+1} - \vec{\nabla} \cdot \vec{v}^*] = -\Delta p^{n+1} \tag{12.4.11}$$

leading to the an equation similar to (12.4.8)

$$\Delta p^{n+1} = \frac{\rho}{\Delta t} \vec{\nabla} \cdot \vec{v}^* \tag{12.4.12}$$

The final value of the velocity field $\vec{v}^{n+1}$ velocities are updated from (12.4.10).

The numerical resolution of the pressure Poisson equation is a crucial step of the whole approach, since the overall efficiency of the code will depend on its performance. Hence all possible convergence optimization and acceleration techniques should be applied. In particular preconditioning and multigrid techniques are strongly recommended for this step of the computation, and eventually for other steps.

12.4.2 The Issue of Staggered Versus Collocated Grids

The choice of a space discretization is, as for compressible flows, between centered or upwind methods, at least for the convection terms, since the diffusive contributions are always centrally discretized.

The most current choice is the central discretization of the convection terms, which raises a particular problem with the pressure correction approach.

The central discretization for the convection terms requires the addition of higher order artificial dissipation terms to create the required damping of high frequency errors, as introduced in the previous examples. However the absence of the time derivative of the density in the continuity equation creates an additional decoupling in the centrally discretized equations, with pressure correction methods. This is best illustrated on the one-dimensional system of incompressible Navier–Stokes equations. The conservation equations take the form

$$\begin{aligned} &\frac{\partial u}{\partial x} = 0 \\ &\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \frac{\partial^2 u}{\partial x^2} \end{aligned} \tag{12.4.13}$$

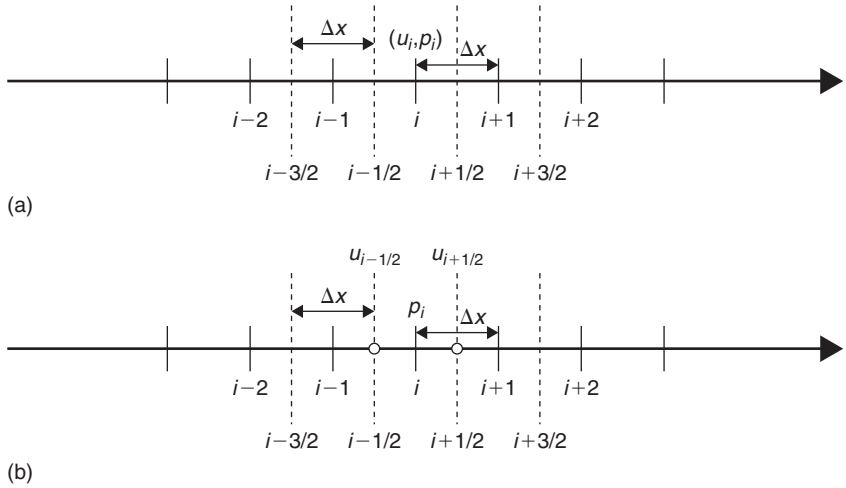

Figure 12.4.1 *Standard 'collocated' and staggered grids: (a) standard collocated grid and (b) staggered grid.*

We consider a central finite difference discretization of the above equations on a standard uniform grid, where all the variables are defined on the same mesh points. This is called a ***collocated mesh***, in the context of pressure correction methods (see Figure 12.4.1a). The centrally discretized equations become

$$\frac{u_{i+1}^{n+1} - u_{i-1}^{n+1}}{2\Delta x} = 0 \tag{12.4.14}$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + \frac{(u_{i+1}^n)^2 - (u_{i-1}^n)^2}{2\Delta x} = -\frac{1}{\rho}\frac{p_{i+1} - p_{i-1}}{2\Delta x} + \nu\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} \tag{12.4.15}$$

In the framework of the pressure correction method, the momentum equation is split into two parts by introducing the intermediate velocity u^*, based on the fractional step method, solution of

$$\frac{u_i^* - u_i^n}{\Delta t} + \frac{(u_{i+1}^n)^2 - (u_{i-1}^n)^2}{2\Delta x} = \nu\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2} \tag{12.4.16}$$

$$\frac{u_i^{n+1} - u_i^*}{\Delta t} = -\frac{1}{\rho}\frac{p_{i+1} - p_{i-1}}{2\Delta x} \tag{12.4.17}$$

The expression for u_i^{n+1}, obtained from equation (12.4.17), can be substituted into (12.4.14), resulting in the following 1D equation:

$$\frac{p_{i+2} - 2p_i + p_{i-2}}{4\Delta x^2} = \frac{\rho}{\Delta t}\frac{u_{i+1}^* - u_{i-1}^*}{2\Delta x} \tag{12.4.18}$$

This is a Poisson equation for the pressure, which ensures that the continuity equation (12.4.14) is satisfied for the newly updated velocity u_i^{n+1}. The stencil on which the Laplace operator is discretized $(i+2, i, i-2)$ contains, however, only odd or even indices, which leads to a decoupling of the discrete pressure field and often results in high frequency oscillations of pressure.

As can be seen indeed, the pressure at point i is not influenced by the velocity component u_i^n of the same point and in return u_i^n is not affected by p_i. Hence velocity and pressure are decoupled on even and odd points; see also Section 4.2 and the discussion around the Lax–Friedrichs scheme in Chapter 7, for an illustration of analog cases. This decoupling is not present with compressible flows due to the density–velocity coupling in the continuity equation. It will generate additional high frequency oscillations, requesting the introduction of artificial dissipation terms.

A solution to the odd–even decoupling problem, has been introduced by Harlow and Welch (1965), by defining a ***staggered mesh***, where the velocity and pressure are not defined in the same mesh points. As seen in Figure 12.4.1b, the velocity is directly defined at the half mesh points, while the pressure remains defined at the central mesh point. The central discretization of the continuity equation of (12.4.13) now becomes

$$\frac{u_{i+1/2}^{n+1} - u_{i-1/2}^{n+1}}{\Delta x} = 0 \tag{12.4.19}$$

With the fractional step method, equation (12.4.17) becomes on the staggered mesh

$$\frac{u_{i+1/2}^{n+1} - u_{i+1/2}^{*}}{\Delta t} = -\frac{1}{\rho}\frac{p_{i+1} - p_i}{\Delta x} \tag{12.4.20}$$

By substituting expressions for $u_{i+1/2}^{n+1}$ and $u_{i-1/2}^{n+1}$ derived from (12.4.20) into this equation, we obtain

$$\frac{p_{i+1} - 2p_i + p_{i-1}}{\Delta x^2} = \frac{\rho}{\Delta t}\frac{u_{i+1/2}^{*} - u_{i-1/2}^{*}}{\Delta x} \tag{12.4.21}$$

In this discrete Poisson equation the pressure and velocity in all nodes are fully coupled, which completely eliminates the problem of odd–even decoupling.

Staggered meshes are currently applied with central discretization and the most popular two-dimensional arrangement is shown in Figure 12.4.2, where the u and v velocity components are located on different cell faces. The equations are discretized in conservation form, the control volumes depending on the considered equations. The mass equation is discretized on the volume centered on the point (i, j), while the x-momentum conservation is expressed on the volume centered for the location of u, i.e. $(i+1/2, j)$. Similarly, the y-momentum conservation is expressed on the volume centered on the location of v, i.e. $(i, j+1/2)$.

The Poisson equation for the pressure is obtained from the divergence of the discretized momentum equation. *This step should be performed by exactly the same discrete operations as applied to express mass conservation.* This is required for global consistency and conservation. It is fairly straightforward on a Cartesian mesh, but becomes essential on arbitrary meshes.

The Poisson equation for the pressure is solved with Neumann boundary conditions on the normal pressure gradient on the walls and at the inlet, while at the outlet

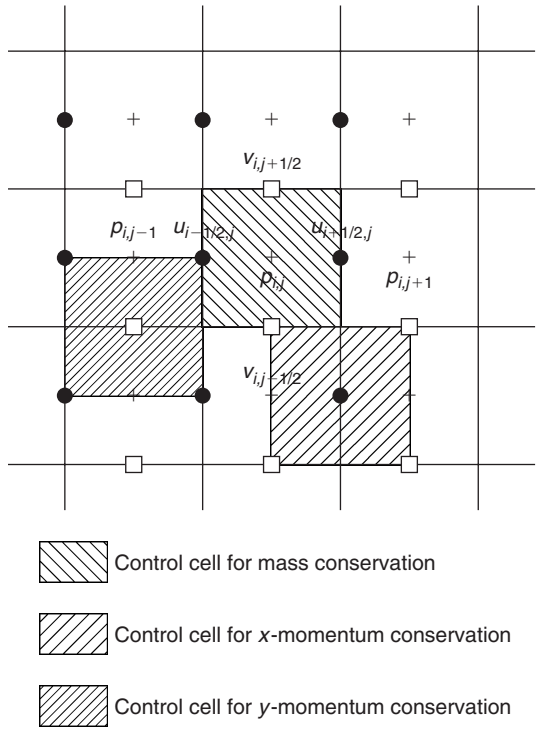

Figure 12.4.2 *Staggered, two-dimensional finite difference mesh for centrally discretized pressure correction methods.*

the pressure is set to a certain value (e.g. atmospheric pressure). The details of the implementation depend on the selected space discretization and on the mesh.

An additional condition is essential for the numerical accuracy of the resolution of the pressure equation, namely that the compatibility condition, obtained from Green's theorem applied to the Poisson equation, should be identically satisfied by the space discretization. Applied to equation (12.4.12), we should have identically, for the integral of the normal pressure gradient on boundary $\vec{S}$ of the computational domain Ω:

$$\int_{\Omega} \Delta p^{n+1} \mathrm{d}\Omega = \oint_{S} \vec{\nabla} p \cdot \mathrm{d}\vec{S} = \oint_{S} \frac{\partial p}{\partial n} \mathrm{d}S = \frac{\rho}{\Delta t} \int_{\Omega} \nabla \cdot \vec{v}^{*} \mathrm{d}\Omega = \frac{\rho}{\Delta t} \oint_{S} \vec{v}^{*} \cdot \mathrm{d}\vec{S} \tag{12.4.22}$$

12.4.3 Implementation of a Pressure Correction Method

We consider the application of the fractional step method to the simulation of incompressible flows. The method is described for a cell-centered finite volume method on a Cartesian mesh, with all flow variables defined in the same points at the centers of the computational cells, although the staggered grid approach is applied to connect values at the cell centers with face defined values, as seen on Figure 12.4.3, where different control volumes are shown. Boundaries of the computational domain are

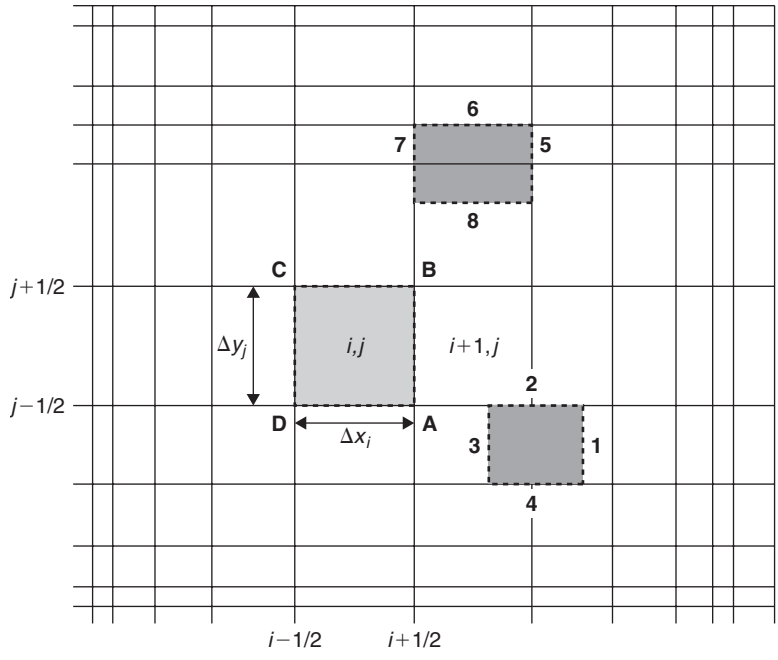

Figure 12.4.3 *Cell-centered finite volume mesh for pressure correction method on non-uniform Cartesian grid.*

located on cell faces and you can follow the treatment of the boundary conditions as described in Section 12.2.

Another way to impose boundary conditions in such a configuration can be considered through introduction of ghost or dummy cells, i.e. rows of cells neighboring the computational domain and having the same size as the first row of cells inside the domain.

You can extend the generality of your code by allowing for non-uniform Cartesian grids to account for clustered grids near solid walls. We consider a rectangular domain with side lengths L_x and L_y with N_x and N_y mesh points in both directions and variable mesh sizes. The Cartesian mesh has $N_x \times N_y$ nodes, dividing the domain into $(N_x - 1)(N_y - 1)$ rectangular cells, the sides of varying sizes: $\Delta x_i = x_{i+1} - x_i$, $\Delta y_i = y_{i+1} - y_i$.

The two-dimensional incompressible Navier–Stokes equations are discretized on the non-uniform Cartesian grid of Figure 12.4.3. A cell-centered second order finite volume discretization is selected, with an explicit first order time integration.

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$$

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \frac{1}{\rho}\frac{\partial \tau_{xx}}{\partial x} + \frac{1}{\rho}\frac{\partial \tau_{xy}}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\frac{\partial^2 u}{\partial x^2} + \nu\frac{\partial^2 u}{\partial y^2}$$

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \frac{1}{\rho}\frac{\partial \tau_{xy}}{\partial x} + \frac{1}{\rho}\frac{\partial \tau_{yy}}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\frac{\partial^2 v}{\partial x^2} + \nu\frac{\partial^2 v}{\partial y^2} \tag{12.4.23}$$

12.4.3.1 *Numerical discretization*

In the pressure correction method, the velocities are updated from the momentum equations, while pressure is obtained by solving the Poisson equation derived by taking a divergence of the momentum equations and taking conservation of mass into account.

We refer to Figure 12.4.3 and apply a finite volume formulation on the contour ABCD for the velocity components u and v at the cell centers.

Applying the fractional step formulation of equations (12.4.9) to (12.4.12), we write the scheme as follows, for the intermediate velocity components u^* and v^*

$$\frac{u^*_{i,j} - u^n_{i,j}}{\Delta t} + \frac{(u^2)^n_{i+1/2,j} - (u^2)^n_{i-1/2,j}}{\Delta x_i} + \frac{(uv)^n_{i,j+1/2} - (uv)^n_{i,j-1/2}}{\Delta y_j}$$
$$= \frac{1}{\rho}\frac{(\tau_{xx})^n_{i+1/2,j} - (\tau_{xx})^n_{i-1/2,j}}{\Delta x_i} + \frac{1}{\rho}\frac{(\tau_{xy})^n_{i,j+1/2} - (\tau_{xy})^n_{i,j-1/2}}{\Delta y_j} \tag{12.4.24}$$

$$\frac{v^*_{i,j} - v^n_{i,j}}{\Delta t} + \frac{(uv)^n_{i+1/2,j} - (uv)^n_{i-1/2,j}}{\Delta x_i} + \frac{(v^2)^n_{i,j+1/2} - (v^2)^n_{i,j-1/2}}{\Delta y_j}$$
$$= \frac{1}{\rho}\frac{(\tau_{yx})^n_{i+1/2,j} - (\tau_{yx})^n_{i-1/2,j}}{\Delta x_i} + \frac{1}{\rho}\frac{(\tau_{yy})^n_{i,j+1/2} - (\tau_{yy})^n_{i,j-1/2}}{\Delta y_j} \tag{12.4.25}$$

where the interface values are obtained by the weighted averages

$$u_{i+1/2,j} = \frac{\Delta x_i u_{i+1,j} + \Delta x_{i+1} u_{i,j}}{\Delta x_i + \Delta x_{i+1}}$$
$$v_{i,j+1/2} = \frac{\Delta y_j v_{i,j+1} + \Delta y_{j+1} v_{i,j}}{\Delta y_j + \Delta y_{j+1}} \tag{12.4.26}$$

and similarly for the other variables.

You can calculate the shear stress components as follows:

$$(\tau_{xx})^n_{i+1/2,j} = 2\mu^n_{i+1/2,j}(u_x)^n_{i+1/2,j}$$
$$(\tau_{xy})^n_{i+1/2,j} = \mu^n_{i+1/2,j}((u_y)^n_{i+1/2,j} + (v_x)^n_{i+1/2,j})$$
$$(u_x)^n_{i+1/2,j} = \frac{u^n_{i+1,j} - u^n_{i,j}}{(\Delta x_{i+1} + \Delta x_i)/2} \quad (v_x)^n_{i+1/2,j} = \frac{v^n_{i+1,j} - v^n_{i,j}}{(\Delta x_{i+1} + \Delta x_i)/2}$$
$$(u_y)^n_{i+1/2,j} = \frac{1}{2}\left[\frac{u^n_{i+1,j+1} - u^n_{i+1,j-1}}{(\Delta y_{j+1} + 2\Delta y_j + \Delta y_{j-1})/2} + \frac{u^n_{i,j+1} - u^n_{i,j-1}}{(\Delta y_{j+1} + 2\Delta y_j + \Delta y_{j-1})/2}\right]$$
$$(v_y)^n_{i+1/2,j} = \frac{1}{2}\left[\frac{v^n_{i+1,j+1} - v^n_{i+1,j-1}}{(\Delta y_{j+1} + 2\Delta y_j + \Delta y_{j-1})/2} + \frac{v^n_{i,j+1} - v^n_{i,j-1}}{(\Delta y_{j+1} + 2\Delta y_j + \Delta y_{j-1})/2}\right] \tag{12.4.27}$$

$$
\begin{aligned}
(\tau_{yy})^n_{i,j+1/2} &= 2\mu^n_{i,j+1/2}(v_y)^n_{i+1/2,j} \\
(\tau_{yx})^n_{i,j+1/2} &= \mu^n_{i,j+1/2}((u_y)^n_{i,j+1/2} + (v_x)^n_{i,j+1/2}) \\
(u_y)^n_{i,j+1/2} &= \frac{u^n_{i,j+1} - u^n_{i,j}}{(\Delta y_{j+1} + \Delta y_j)/2} \quad (v_y)^n_{i,j+1/2} = \frac{v^n_{i,j+1} - v^n_{i,j}}{(\Delta y_{j+1} + \Delta y_j)/2} \\
(u_x)^n_{i,j+1/2} &= \frac{1}{2}\left[\frac{u^n_{i+1,j+1} - u^n_{i-1,j+1}}{(\Delta x_{i+1} + 2\Delta x_i + \Delta x_{i-1})/2} + \frac{u^n_{i+1,j} - u^n_{i-1,j}}{(\Delta x_{i+1} + 2\Delta x_i + \Delta x_{i-1})/2}\right] \\
(v_x)^n_{i,j+1/2} &= \frac{1}{2}\left[\frac{v^n_{i+1,j+1} - v^n_{i-1,j+1}}{(\Delta x_{i+1} + 2\Delta x_i + \Delta x_{i-1})/2} + \frac{v^n_{i+1,j} - v^n_{i-1,j}}{(\Delta x_{i+1} + 2\Delta x_i + \Delta x_{i-1})/2}\right]
\end{aligned}
\tag{12.4.28}
$$

Once the intermediate velocity components are estimated, you can obtain the values of the updated velocity and pressure variables, satisfying the continuity equation, following (12.4.10). Applying a finite volume formulation on the contour ABCD to this equation and projecting in the x- and y-directions, we obtain

$$
\begin{aligned}
\frac{u^{n+1}_{i,j} - u^*_{i,j}}{\Delta t} &= -\frac{1}{\rho}\frac{p_{i+1/2,j} - p_{i-1/2,j}}{\Delta x_i} \\
\frac{v^{n+1}_{i,j} - v^*_{i,j}}{\Delta t} &= -\frac{1}{\rho}\frac{p_{i,j+1/2} - p_{i,j+1/2}}{\Delta y_j}
\end{aligned}
\tag{12.4.29}
$$

The pressure is obtained by expressing that the velocity components at level $(n+1)$ satisfy the divergence free continuity equation, as in (12.4.6). Discretized on the finite volume mesh ABCD we obtain

$$
\frac{u^{n+1}_{i+1/2,j} - u^{n+1}_{i-1/2,j}}{\Delta x_i} + \frac{v^{n+1}_{i,j+1/2} - v^{n+1}_{i,j-1/2}}{\Delta y_j} = 0 \tag{12.4.30}
$$

The interface velocities at the $(n+1)$ level are obtained by applying once again the finite volume formulation of equation (12.4.10), but this time on a staggered control volume such as 1234, Figure 12.4.3, centered on face AB, for the x component equation

$$
\rho\frac{u^{n+1}_{i+1/2,j} - u^*_{i+1/2,j}}{\Delta t}\Delta y_j\frac{\Delta x_{i+1} + \Delta x_i}{2} = (p^{n+1}_{i+1,j} - p^{n+1}_{i,j})\Delta y_j \tag{12.4.31}
$$

and on the control volume 5678, centered around BC, for the vertical component, leading to

$$
\rho\frac{v^{n+1}_{i,j+1/2} - v^*_{i,j+1/2}}{\Delta t}\Delta x_i\frac{\Delta y_{j+1} + \Delta y_j}{2} = (p^{n+1}_{i,j+1} - p^{n+1}_{i,j})\Delta x_i \tag{12.4.32}
$$

The values on faces CD and DA are obtained similarly.

The intermediate values u^* and v^* on the cell faces are obtained from the relations (12.4.26).

Substituting these relations in equation (12.4.30), leads to the pressure Poisson equation:

$$\frac{1}{\Delta x_i}\left[\frac{p^{n+1}_{i+1,j}-p^{n+1}_{i,j}}{(\Delta x_{i+1}+\Delta x_i)/2}-\frac{p^{n+1}_{i,j}-p^{n+1}_{i-1,j}}{(\Delta x_i+\Delta x_{i-1})/2}\right]$$
$$+\frac{1}{\Delta y_j}\left[\frac{p^{n+1}_{i,j+1}-p^{n+1}_{i,j}}{(\Delta y_{j+1}+\Delta y_j)/2}-\frac{p^{n+1}_{i,j}-p^{n+1}_{i,j-1}}{(\Delta y_j+\Delta y_{j-1})/2}\right]$$
$$=\frac{\rho}{\Delta t}\left[\frac{u^*_{i+1/2,j}-u^*_{i-1/2,j}}{\Delta x_i}+\frac{v^*_{i,j+1/2}-v^*_{i,j-1/2}}{\Delta y_j}\right] \tag{12.4.33}$$

The whole procedure of updating the solution is the following:

- Calculate the intermediate velocity field $u^*_{i,j}, v^*_{i,j}$ from (12.4.24) and (12.4.25).
- Obtain the pressure by solving the Poisson equation (12.4.33).
- Obtain the solution at the next time step $u^{n+1}_{i,j}, v^{n+1}_{i,j}$ from (12.4.29), where the pressure at the cell faces is obtained by applying relations (12.4.26).

We have now to focus on the most critical issue of pressure correction methods, namely the efficient resolution of the pressure Poisson equation. This is a crucial step of the whole approach, since the overall efficiency of your code will depend on its performance.

12.4.3.2 *Algorithm for the pressure Poisson equation*

The pressure Poisson equation (12.4.33) is a standard elliptic equation and you can call upon the various methods introduced in Chapter 10. In advanced codes, various convergence optimization and acceleration techniques are applied, in particular preconditioning and multigrid techniques are strongly recommended for this step of the computation, and many of these techniques are described in the literature on pressure correction methods.

Here, we suggest you to choose a simple line Gauss–Seidel method along a vertical line. As a first approximation of the pressure, its discrete values obtained on the previous time level are used. Given an approximation of the pressure p^k, the next one is obtained from the following relation:

$$\frac{1}{\Delta x_i}\left[\frac{p^{k}_{i+1,j}-p^{k+1}_{i,j}}{(\Delta x_{i+1}+\Delta x_i)/2}-\frac{p^{k+1}_{i,j}-p^{k+1}_{i-1,j}}{(\Delta x_i+\Delta x_{i-1})/2}\right]$$
$$+\frac{1}{\Delta y_j}\left[\frac{p^{k+1}_{i,j+1}-p^{k+1}_{i,j}}{(\Delta y_{j+1}+\Delta y_j)/2}-\frac{p^{k+1}_{i,j}-p^{k+1}_{i,j-1}}{(\Delta y_j+\Delta y_{j-1})/2}\right]=Q_{i,j} \tag{12.4.34}$$

where $Q_{i,j}$ is the right-hand side of (12.4.33). Not that the k index denotes an iteration number and not a time level (n in the previous section). Equation (12.4.34) can be

rewritten as follows:

$$a_{i,j}p_{i,j-1}^{k+1} + b_{i,j}p_{i,j}^{k+1} + c_{i,j}p_{i,j+1}^{k+1} = Q_{i,j} - \frac{1}{\Delta x_i}\left[\frac{p_{i+1,j}^{k}}{(\Delta x_{i+1} + \Delta x_i)/2} + \frac{p_{i-1,j}^{k+1}}{(\Delta x_i + \Delta x_{i-1})/2}\right] \tag{12.4.35}$$

where

$$\begin{aligned} a_{i,j} &= \frac{2}{\Delta y_j}\frac{1}{\Delta y_j + \Delta y_{j-1}} \\ c_{i,j} &= \frac{2}{\Delta y_j}\frac{1}{\Delta y_{j+1} + \Delta y_j} \\ b_{i,j} &= -\frac{2}{\Delta x_i}\left[\frac{1}{\Delta x_{i+1} + \Delta x_i} + \frac{1}{\Delta x_i + \Delta x_{i-1}}\right] - (a_{i,j} + c_{i,j}) \end{aligned} \tag{12.4.36}$$

We have to add the boundary conditions, for instance for $i = 1$ and a Neumann boundary condition (12.4.35) can be rewritten taking into account the boundary condition at $i = 1/2$:

$$a_{1,j}p_{1,j-1}^{k+1} + \left(b_{1,j} + \frac{1}{\Delta x_1}\frac{2}{\Delta x_1 + \Delta x_0}\right)p_{1,j}^{k+1} + c_{1,j}p_{1,j+1}^{k+1} = Q_{1,j} - \frac{1}{\Delta x_1}\frac{p_{2,j}^{k}}{(\Delta x_2 + \Delta x_1)/2} \tag{12.4.37}$$

where the Neumann boundary condition is expressed as $p_0^{k+1} = p_1^{k+1}$.

You can apply this similarly for the other boundary conditions. If the pressure is imposed at certain boundaries, as a Dirichlet condition, then you can introduce this value directly in the corresponding equation.

The algebraic system can be efficiently solved with the Thomas Algorithm (see Appendix A in Chapter 10).

The iterations are to be repeated until a prescribed convergence criterion is satisfied (e.g. $\max_{i,j} |p_{i,j}^{k+1} - p_{i,j}^{k}| < \varepsilon$).

For reasons of accuracy, it is recommended to alternate this algorithm with a line Gauss–Seidel method in the horizontal direction (at $j =$ constant), applying the Thomas algorithm in two different mesh directions. Equation (12.4.34) is replaced by

$$\frac{1}{\Delta x_i}\left[\frac{p_{i+1,j}^{k+1} - p_{i,j}^{k+1}}{(\Delta x_{i+1} + \Delta x_i)/2} - \frac{p_{i,j}^{k+1} - p_{i-1,j}^{k+1}}{(\Delta x_i + \Delta x_{i-1})/2}\right] + \frac{1}{\Delta y_j}\left[\frac{p_{i,j+1}^{k} - p_{i,j}^{k+1}}{(\Delta y_{j+1} + \Delta y_j)/2} - \frac{p_{i,j}^{k+1} - p_{i,j-1}^{k+1}}{(\Delta y_j + \Delta y_{j-1})/2}\right] = Q_{i,j} \tag{12.4.38}$$

We leave it to you as an exercise to work out the details of its implementation.

12.5 NUMERICAL SOLUTIONS WITH THE PRESSURE CORRECTION METHOD

We apply now the developed method to the incompressible lid driven cavity flow. This well-known flow configuration results from the uniform motion of the upper wall of a square box, wherein the flow is induced by the viscous stresses, similarly to the Couette flow. The flow within the lid driven rectangular two-dimensional cavity is maintained by the continuous diffusion of kinetic energy injected by the moving wall. This energy is initially confined to a thin viscous layer of fluid next to the moving boundary. After a period of time, which depends on the Reynolds number, the redistribution of energy reaches an equilibrium leading to a steady state laminar flow. In case of high Reynolds number flows, this steady state solution is never reached, due to instabilities leading to transition to turbulence.

It can actually be considered as the two-dimensional extension of the Couette flow.

12.5.1 Lid Driven Cavity

We consider the domain included in a square of unit length, with $0 \leq x, y \leq 1$, where the upper boundary at $y = 1$, moves with a constant velocity $U = 1$.

The Reynolds number based on the size of the domain, the velocity of the moving wall, density $\rho = 1$ and viscosity $\mu = 0.01$ is $\mathrm{Re} = 100$.

The boundary conditions are set as follows

$$\begin{array}{ll} u(x,0) = 0 & v(x,0) = 0 \\ u(x,1) = 1 & v(x,1) = 0 \\ u(0,y) = 0 & v(0,y) = 0 \\ u(1,y) = 0 & v(1,y) = 0 \end{array} \tag{12.5.1}$$

The calculation is performed on a 41×41 Cartesian uniform mesh, which divides the computational domain in 1600 equidistant cells, with $\Delta x = \Delta y = 0.025$. In the solution of the pressure Poisson equation a Neumann boundary condition is imposed on the boundaries. The time step is taken as $\Delta t = 0.01$. A line Gauss–Seidel method is used to solve the pressure equation iteratively, iterations repeated till the maximum absolute value of the residual is smaller that 10^{-3}. In the selected time-dependent approach, the Poisson equation is converged for each time step. This option is selected to enable you also to handle unsteady flows or to detect spontaneous unsteadiness when they occur.

Figure 12.5.1 shows the convergence history of the pressure equation plotted for the first 10 time steps. As can be seen from the graph, the residual value for the first iteration of the Poisson solver at each time step decreases as the calculation proceeds, which means that less and less iterations are needed to reach the convergence criterion. This is typical for pressure correction methods when applied to unsteady flows with a steady state limit, at which the Poisson equation is satisfied automatically. Each jump in this figure represents the passage at the next time step, while the residual reduction in between represents the convergence behavior of the Gauss–Seidel relaxation method.

The diagrams on Figure 12.5.2 display the streamlines of the solution at different transient stages, at $t = 0.5$, 1, 10 and 30. The flow undergoes a recirculation motion

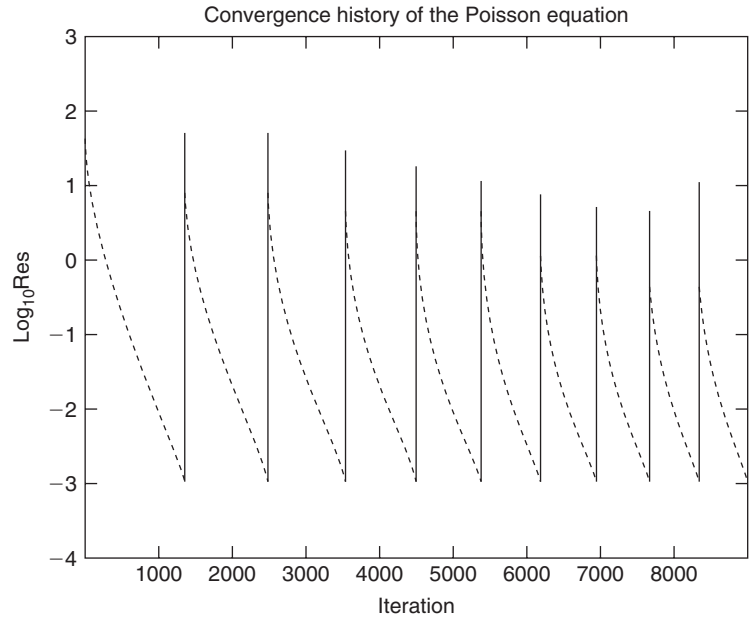

Figure 12.5.1 *Convergence history of the Poisson equation for the pressure.*

imposed by the viscous effects. The solution at $t = 0.5$ and $t = 1$ represent a transient stage of the solution, where the circular motion of the fluid is still developing. At $t = 10$ and $t = 30$ the flow in the cavity is fully developed and has reached its steady state.

On a quantitative basis, Figure 12.5.3 compares the velocity distributions along the centerlines $x = 0.5$ and $y = 0.5$, with a reference solution obtained by Dr. Sergey Smirnov, at the Vrije Universiteit Brussel, Belgium, on a fine grid of 161×161 with a fourth order accurate compact scheme for the space discretization.

The maximum error on the velocity distribution is of the order of 10% on this 41×41 mesh, indicating that a finer resolution is required.

12.5.2 Additional Suggestions

You can now run your code on many other cases, such as:

- The lid driven cavity by increasing the Reynolds number of your simulation, until you start detecting the initial process toward transition. This will require you to increase the grid resolution.
- The flat plate problem.
- Other cases with a Cartesian grid, such as the backward facing step.
- You could also extend now your code to more general grids and run the cylinder case in laminar mode.

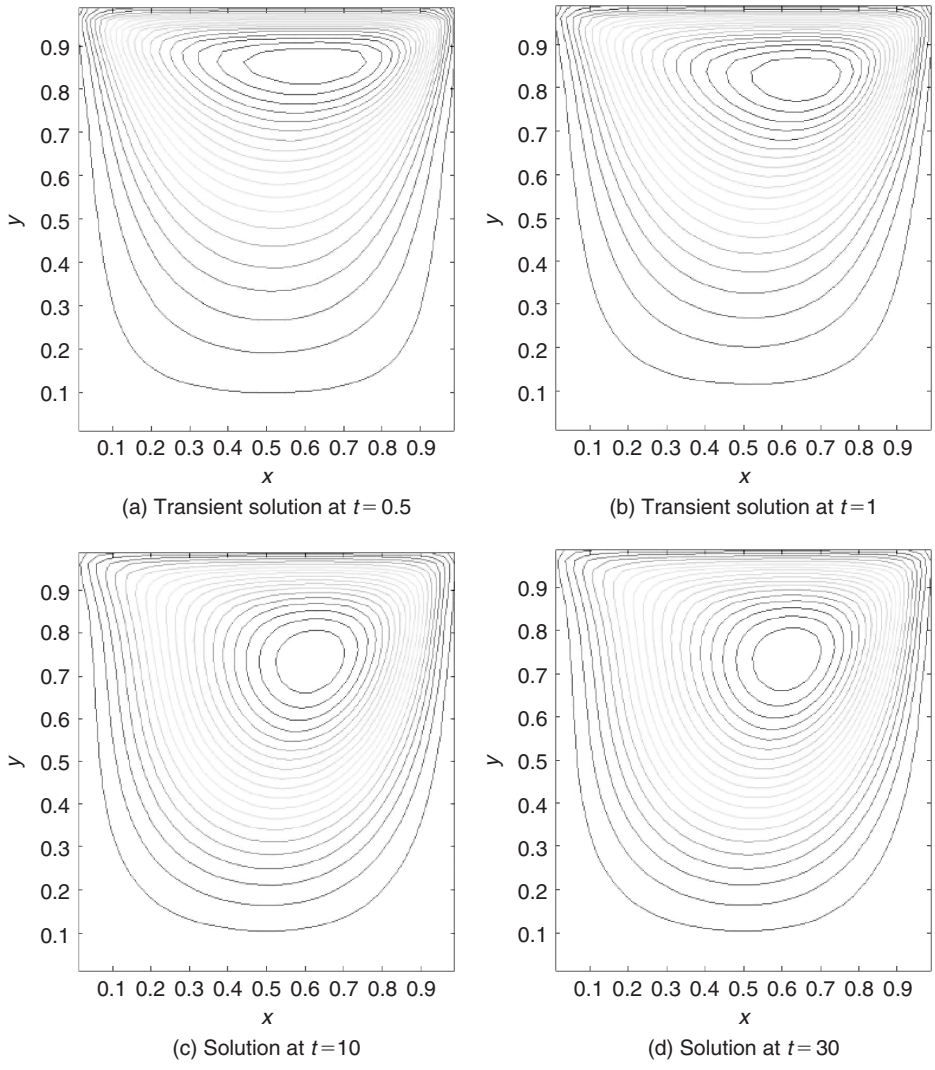

Figure 12.5.2 *Streamlines of the flow field at different transient stages, for t = 0.5, 1, 10 and 30 (for color image refer Plate 12.5.2).*

12.6 BEST PRACTICE ADVICE

CFD software systems form today an essential part of the world of Computer Aided Engineering (CAE), supporting the design and analysis of industrial products involving fluid flows. Many design decisions of systems, whose performance depends on their internal or external flow behavior, are based on the results of CFD simulations, either with in-house or commercial CFD codes. This raises the question of

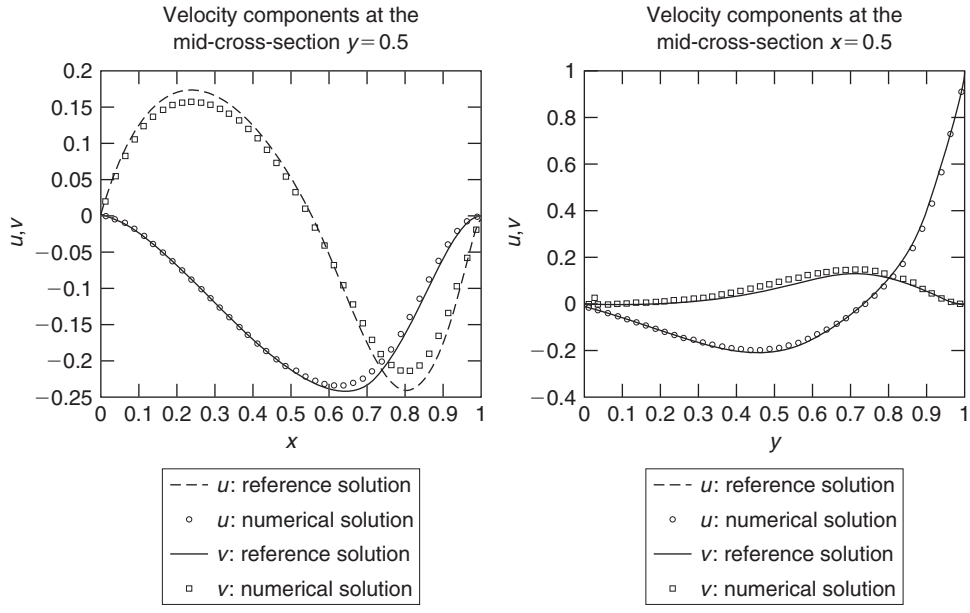

Figure 12.5.3 *Comparison of the computed velocity distributions along the centerlines $x = 0.5$ and $y = 0.5$, with a reference solution.*

the reliability and the confidence we can attach to the results of CFD, in presence of numerous sources of errors and of uncertainties.

The overwhelming majority of industrial and environmental fluid flow systems are turbulent and the modeling of turbulence remains a dominant factor of uncertainty, as none of the available models today are fully satisfactory in their prediction capability of the complex phenomena of turbulence.

We have not dealt with turbulence, in this introductory text on the basics of CFD, and left this important issue to the Volume II, although it has a considerable effect on the level of uncertainty of CFD results.

In order to respond to the needs of the increasing number of CFD users in industry, a demand has arisen for recommendations of best practices in the application of CFD codes, in regard of the complexity of industrial flow systems.

One of the first efforts toward the establishment of best practice guidelines (BPG) for CFD has been generated by the ERCOFTAC (European Research Community on Flow, Turbulence And Combustion) association; http://www.ercoftac.org. This effort has led to a document, Casey and Wintergerste (2000), providing an extensive set of recommendations, and is available from this organization.

As we have considered here only the numerical issues and since we attempted to develop your awareness of the various pitfalls and error sources, particularly in the last two chapters, we will summarize here some of the basic guidelines and recommendations in applying CFD codes, based essentially on the content of this ERCOFTAC document.

12.6.1 List of Possible Error Sources

The first issue is to attempt to summarize all the possible error sources. We can distribute them as follows:

- ***Discretization or numerical error***: These errors are due to the approximations resulting from the space and time numerical discretization, and have been analyzed in details in Parts III and IV of this book.
- ***Iteration or convergence error***: These errors occur due to the difference between a fully converged solution on a finite number of grid points and a solution that is not fully converged. Ideally, each calculation should be run up to the reduction of the residuals to machine accuracy. Although this can be performed on simple cases as illustrated in the last two chapters, it is hardly ever possible on industrial simulations with million of points.
- ***Round-off errors***: These errors are due to the fact that the difference between two values of a parameter is below the machine accuracy of the computer. This is caused by the limited number of computer digits available for storage of a given physical value. It might require to shift to double precision arithmetic, when dealing with very small variations between flow variables, or when very short distances between mesh points are introduced.
- ***Application uncertainties***: Many variables defining the flow conditions, such as operational and/or geometrical data are often not precisely defined or not well known. Examples of this are uncertainties in the precise geometry due to manufacturing tolerances, uncertain inflow data or models, such as turbulence properties or fluid properties.
- ***User errors***: Errors can arise from mistakes introduced by the user. They can cover various aspects, such as inadequate or poor grid generation; incorrect boundary condition; incorrect choice of numerical parameters, such as time step or relaxation coefficients; post-processing errors. Experience and great care are required to minimize their risk of occurrence.
- ***Code errors***: Errors due to bugs in the software cannot be excluded, despite all verification efforts, as it is humanly impossible to cover all possible combinations of code parameters with a finite number of verification tests.
- ***Model uncertainties***: It refers to the physical models that have to be introduced to describe complex flow properties, such as turbulence, multiphase flows, combustion, real Newtonian or non-Newtonian fluids.

To minimize the effects of these error sources, a series of recommendations can be collected, that we recommend to your attention, as a kind of checklist when running a CFD code.

12.6.2 Best Practice Recommendations

As seen in the previous chapters, an essential component of a CFD simulation and a major potential source of errors is the choice of the grid and the resulting grid quality.

RECOMMENDATIONS ON GRIDS

The key recommendation is to ensure smooth grids, avoiding abrupt changes in grid size or shape, as this can lead to a significant loss of accuracy. Hence take good care to:

- Define the computational domain, in order to minimize the influence and interactions between the flow and the far-field conditions. In particular,
  - Place inlet and outlet boundaries as far away as possible from the region of interest. In particular, if uniform far-field conditions are imposed, you should ensure that the boundary is not in a region where the flow may still vary significantly.
  - Avoid inlet or outlet boundaries in regions of strong geometrical changes or in regions of recirculation.
- Avoid jumps in grid density or in grid size.
- Avoid highly distorted cells or small grid angles.
- Ensure that the grid stretching is continuous.
- Avoid unstructured tetrahedral meshes in boundary layer regions.
- Refine the grids in regions with high gradients, such as boundary layers, leading edges of airfoils and any region where large changes in flow properties might occur.
- Make sure that the number of points in the boundary layers is sufficient for the expected accuracy. Avoid less than 10 points over the inner part of the boundary layer thickness.
- Monitor the grid quality by adequate mesh parameters, available in most of the grid generators, such as aspect ratio, internal angle, concavity, skewness, negative volume.

RECOMMENDATIONS ON SOLUTION ASSESSMENT

Once you run your code, the following recommendations will be useful to enhance your confidence in the results obtained:

- Check very carefully the selected boundary conditions for correctness and compatibility with the physics of the flow you are modeling.
- Verify all the numerical settings and parameters, before launching the CFD run.
- Verify that your initial solution is acceptable for the problem to be solved.
- Monitor the convergence to ensure that you reach machine accuracy. It is recommended to monitor, in addition to the residuals, the convergence of representative quantities of your problem, such as a drag force or coefficient, a velocity, temperature or pressure at selected points in the flow domain.
- Look carefully at the behavior of the residual convergence curve in function of number of iterations. If the behavior is oscillatory, or if the residual does not converge to machine accuracy by showing a limit cycle at a certain level of residual reduction, it tells you that some inaccuracy affects your solution process.

- Apply internal consistency and accuracy criteria, by verifying:
  - Conservation of global quantities such as total enthalpy and mass flow in steady flow calculations.
  - The entropy production and drag coefficients with inviscid flows, which are strong indicators of the influence of numerical dissipation, as they should be zero.
- Check, whenever possible, the grid dependence of the solution by comparing the results obtained on different grid sizes.
- Some quantities are more sensitive than others to error sources. Pressure curves are less sensitive than shear stresses, which in turn are less sensitive than temperature gradients or heat fluxes, which require finer grids for a given accuracy level.
- If your calculation appears difficult to converge, you can
  - Look at the residual distribution and associated flow field for possible hints, e.g. regions with large residuals or unrealistic levels of the relevant flow parameters.
  - Reduce the values of parameters controlling convergence, such as the CFL number or some under-relaxation parameter, when available.
  - Consider the effects of different initial flow conditions.
  - Check the effect of the grid quality on the convergence rate.
  - Use a more robust numerical scheme, such as a first order scheme, during the initial steps of the convergence and switch to more accurate numerical schemes as the convergence improves.

RECOMMENDATIONS ON EVALUATION OF UNCERTAINTIES

This is a very difficult issue, as the application uncertainties are generally not well defined and require a sound judgment about the physics of the considered flow problem. Some recommendations can be offered:

- Attempt to list the most important uncertainties, such as
  - Geometrical simplifications and manufacturing tolerances around the CAD definition.
  - Operational conditions, such as inlet velocity or inlet flow angle.
  - Physical approximations, such as handling an incompressible flow as a low Mach number compressible flow. This type of uncertainty is manageable, as it can more easily be quantified.
  - Uncertainties related to turbulence or other physical models.
- Perform a sensitivity analysis of the relevant uncertainty to investigate its influence.

CONCLUSIONS AND MAIN TOPICS TO REMEMBER

If you have followed closely the guidelines of this chapter, you have now available a general 2D finite volume density-based code, which allows you to handle practically

any flow configuration, from low to supersonic speed. You can even simulate incompressible flow conditions, by considering low Mach numbers, say below 0.2, for which the numerical solution is an excellent approximation of incompressible fluid flows.

You also have available another option, with a code based on the pressure correction method, suitable for compressible and incompressible flows, although it is restricted to the subsonic range.

You have certainly experienced, by following the steps of the last two chapters in running the various proposed test cases, that the way to achieve high accuracy and reliability of the CFD results on general grids is a difficult process, requiring a close attention to all the details of the implementation of a selected scheme.

Our main ambition with these two chapters was to introduce you to this awareness and to guide you in your ability to ask the 'right questions' when faced with the development of a CFD code or when using a third party code.

The main topics to remember are summarized in the best practice guidelines of Section 12.6. The main message being that you have to exercise critical judgment at all stages of the code development. If you apply a third party code, your critical judgment should apply to your assessment of all aspects of the schemes and its implementation as proposed by the options you select. Make sure that you have enough information on:

- Formal order of the scheme, but also on its behavior on a non-uniform grid.
- The level of numerical dissipation generated on your grid. This can be obtained by running the same case as an inviscid problem, monitoring the entropy distribution.
- The details of the boundary condition implementation and their effect on the accuracy and convergence.
- Convergence levels of the solution, in terms of residuals, but also by monitoring some of the quantities relevant for the problem you are interested in.

We also hope that these exercises will have stimulated your interest and enthusiasm for the beautiful world of numerical flow simulations.

REFERENCES

Anderson, J.D. (1995). *Computational Fluid Dynamics. The Basics with Applications*. McGraw-Hill, New York.

Casey, M. and Wintergerste, T. (2000). Best Practice Guidelines. *ERCOFTAC Special Interest Group on Quality and Trust in Industrial CFD*. ERCOFTAC, http://www.ercoftac.org.

Chorin, A.J. (1967). A numerical method for solving incompressible viscous flow problems. *J. Comput. Phys*., 2, 12–26.

Chorin, A.J. (1968). Numerical solution of the Navier–Stokes equations. *Math. comput*., 23, 341–54.

Ferziger, J.H. and Peric, M. (1997). *Computational Methods for Fluid Dynamics*. Springer Verlag, Berlin.

Harlow, F.H. and Welch, J.E. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluid*., 8, 2182–2189.

Hemsch, M.J. and Morrison, J.H. (2004). Statistical analysis of CFD solutions from 2nd drag prediction workshop. *42nd AIAA Aerospace Sciences Meeting*, Reno, AIAA Paper 2004-556.

Jameson, A. (1995a). Analysis and design of numerical schemes for gas dynamics. 1. Artificial diffusion, upwind biasing, limiters and their effect on multigrid convergence. *Int. J. Comp. Fluid Dyn*., 4, 171–218.

Jameson, A. (1995b). Analysis and design of numerical schemes for gas dynamics. 2. Artificial diffusion and discrete shock structure. *Int. J. Comp. Fluid Dyn*., 5, 1–38.

Majda, A. and Sethian, J. (1985). The derivation and numerical solution of the equations for zero Mach number combustion. *Combust. Sci. Technol*., 42, 185.

Patankar, S.V. (1980). *Numerical Heat Transfer and Fluid Flow*. Hemisphere Publ. Co., New York.

Patankar, S.V. and Spalding, D.B. (1972). A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transfer*, 15, 1787–1806.

Schlichting, H. (1979). *Boundary Layer Theory*. McGraw-Hill, New York.

Temam, R. (1969). Sur l'approximation de la solution de equations de Navier–Stokes par la methode des pas fractionnnaires. *Arch. Rational Mech. Anal*., 32, 135–153; (II): *Arch. Rational Mech. Anal*., 33, 377–385.

Temam, R. (1977). *Navier–Stokes Equations*. North-Holland, Amsterdam.

Wesseling, P. (2001). *Principles of Computational Fluid Dynamics*. Springer Verlag, Berlin.

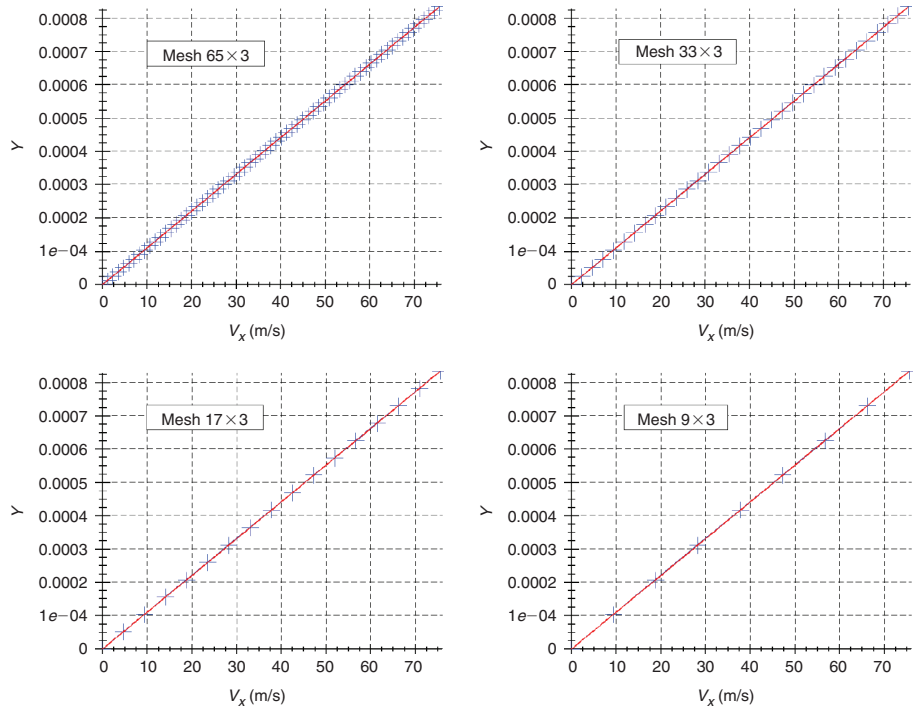

Plate 12.3.3 *Wall-to-wall distribution of axial velocity as deduced from analytical result (continuous line) and from the numerical result (plus signs).*

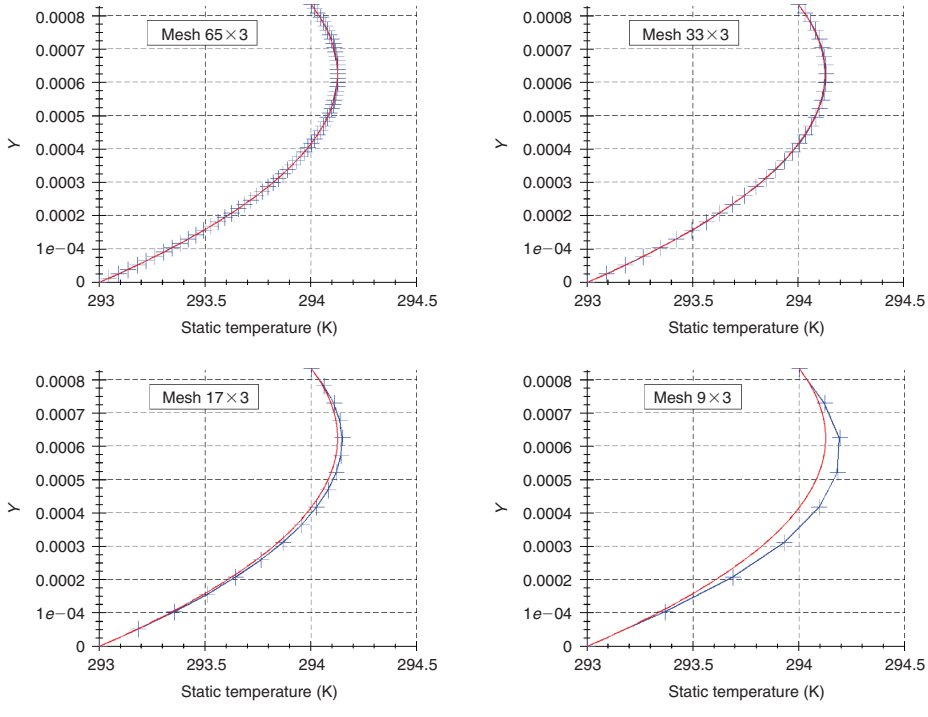

Plate 12.3.4 *Wall-to-wall distribution of static temperature (line with plus signs), compared to the analytical solution (continuous line), for the four different grids.*

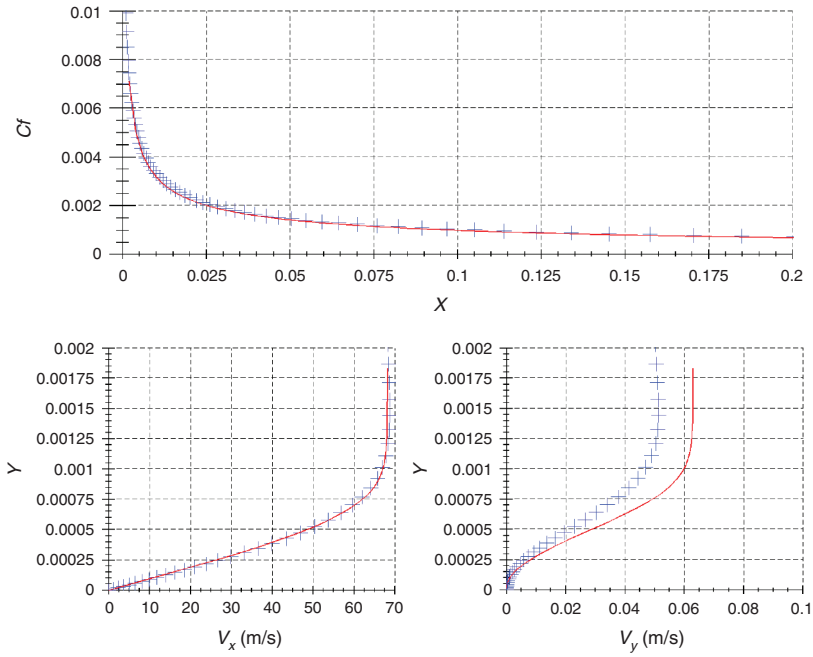

Plate 12.3.8 *Distribution of friction coefficient (upper panel), axial and wall normal velocities at $x = 0.2$ m (lower left and right panels) as obtained from analytical result (continuous line) and from the numerical result on the finest mesh (plus signs). The vertical axis is the distance in meters.*

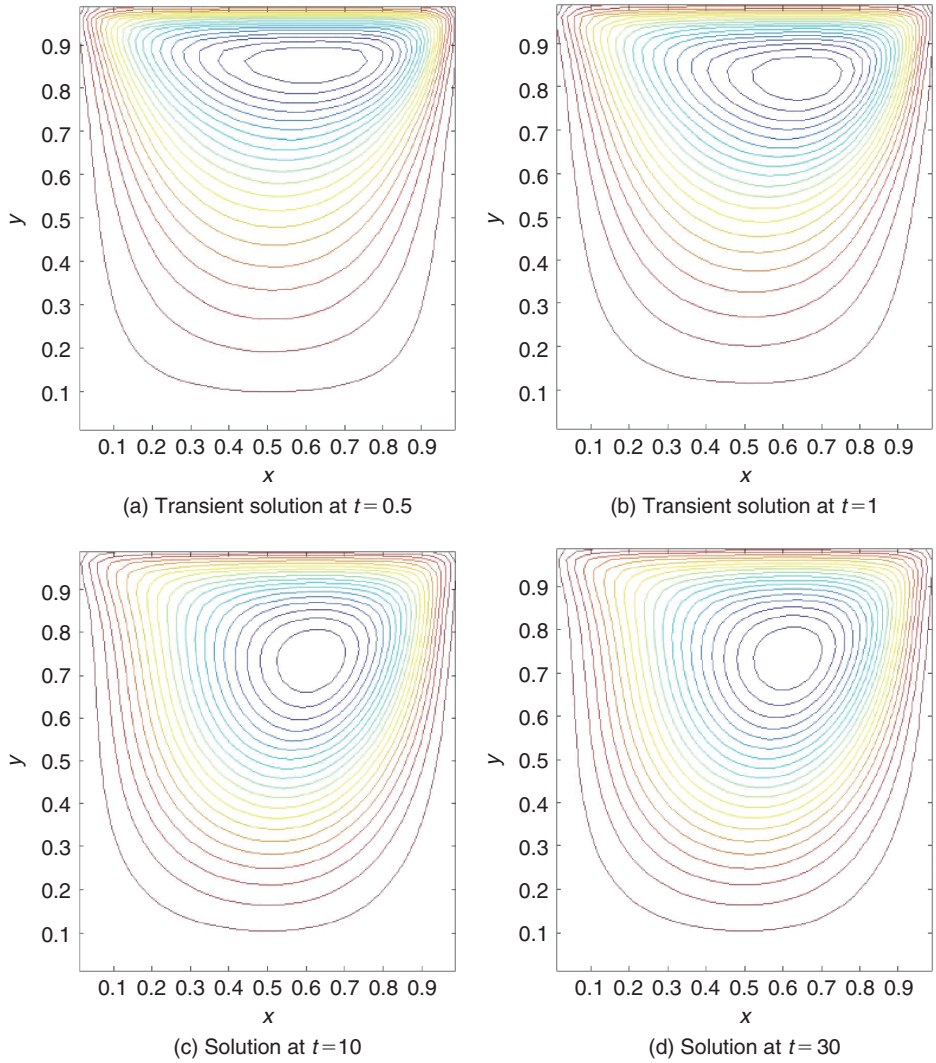

Plate 12.5.2 *Streamlines of the flow field at different transient stages, for t = 0.5, 1, 10 and 30.*