7

Miscellaneous Complements

When it is not in our power to follow what is true, we ought to follow what is most probable.

René Descartes

In this chapter, we address some scattered selected topics on the the FDM, FEM and FVM that do not really fit into the aim and framework of the previous chapters. The guiding criteria of our topic selection are their relevance and the light shed on some complementary aspects of the numerical methods for PDEs, overlooked in the studies already carried out, owing to a lack of pertinence. Of course, the general approach adopted throughout the book is roughly maintained in the material that follows. However, this chapter's four sections are basically disconnected, as far as the corresponding subjects are concerned.

Chapter outline: In Section 7.1, we study fourth-order PDEs, whose treatment involves several new considerations. Section 7.2 is devoted to the advection–diffusion equations, which play a fundamental role in numerical simulations of countless physical phenomena. Section 7.3 addresses the error control through mesh refinement via a posteriori error estimates. In contrast to the material presented so far, we do not attempt to treat this topic in a rigorous manner, but rather draw the reader's attention to this subject as one of the main pillars of contemporary numerical simulations. Finally, we address in Section 7.4 essential aspects of the numerical solution of non-linear PDEs. Two numerical examples are given in this section, incorporating a detailed description of some techniques to handle problem non-linearities.

7.1 Numerical Solution of Biharmonic Equations in Rectangles

A biharmonic equation in two-space variables is a linearised form of fourth-order PDEs that model some fundamental problems in continuum mechanics. Just to give two examples, we could mention the **thin plate-bending** problem (cf. [118]) and plane **viscous incompressible flow** in terms of the **stream function** (see e.g. [157]). In these equations, the dominant differential operator is the square of the Laplacian, that is, the operator $\Delta^2(\cdot) = \Delta[\Delta](\cdot)$. This means that $\Delta^2(\cdot) = \partial_{xxxx}(\cdot) + 2\partial_{xxyy}(\cdot) + \partial_{yyyy}(\cdot)$.

7.1.1 Model Fourth-order Elliptic PDEs

The model problem studied in this section is the linear biharmonic equation, which is posed as follows: Given a function f defined in a bounded two-dimensional domain Ω assumed to have a sufficiently smooth boundary Γ, the problem consists of finding a function u that satisfies

$$\Delta^2 u = f \text{ in } \Omega. \tag{7.1}$$

In the case of thin plate bending, for instance, Ω represents the plate's mid-plane and $u(x, y)$ its deflexion orthogonal to Ω at a point with coordinates $(x; y)$, under the action of an area density of forces proportional to f, pointing in the same direction (see e.g. [118]). Of course, equation (7.1) must be supplemented with suitable boundary conditions, and a minimum regularity of u and f has to be required, otherwise this equation will not have a unique solution. In this section, we will search for u in the space $H^2(\Omega)$, by assuming that $f \in L^2(\Omega)$. Although the latter assumption implies that we can expect more regularity of u, for the purpose of setting up equivalent variational formulations, $H^2(\Omega)$ will be the natural working space as seen hereafter.

In this chapter, we consider only the following essential boundary conditions for equation (7.1):

$$u = g_0;\ \partial_\nu u = g_1 \text{ on } \Gamma, \tag{7.2}$$

where g_0 and g_1 are functions having suitable continuity and differentiability properties on Γ. In the case of the plate-bending problem, most frequently $g_0 = g_1 = 0$ (i.e. the boundary conditions are homogeneous). This corresponds to a **clamped plate**, which means that the plate undergoes no displacement and no rotation along its boundary. In the case of incompressible flow, usually g_0 or g_1 are not zero, but in contrast f is most frequently equal to zero. In order to simplify the presentation, we will focus on homogeneous boundary conditions, but the reader will certainly be able to identify the necessary modifications in the methodology to be studied, in order to accommodate inhomogeneous boundary conditions. In view of this, we will definitively study numerical methods to solve the following:

Model biharmonic equation (7.3)

$$\boxed{\begin{aligned} &\Delta^2 u = f \text{ in } \Omega, \\ &u = \partial_\nu u = 0 \text{ on } \Gamma, \end{aligned}}$$

where $f = g/K$, g being the area density of forces applied to the plate and K a coefficient depending on physical characteristics of the plate. More specifically, $Ed^3/12(1\sigma^2)$, where E is Young's modulus and σ is Poisson's ratio of the material the plate is made of. d in turn is the plate's width. Moreover, the study will be limited to the case of a rectangular Ω.

In Remark 2.4, we explained the difference between essential and natural boundary conditions taking the bar problem (P_1) as a model. In solid mechanics, the solution of an equilibrium problem is usually a kinematic entity (e.g. a deflexion function or a displacement field) that minimises a certain energy functional defined for **kinematically admissible** quantities of the kind. This means that all of them must satisfy certain boundary conditions, which are thus called essential. In contrast to the case of the bar, all the admissible deflexions of a clamped plate must satisfy both deflexion and deflexion normal derivative zero boundary conditions. Hence, in this case, both conditions are essential. The solution itself can eventually satisfy specific boundary conditions carrying a physical meaning, other than essential ones. In this case, the former are natural boundary conditions. Just to give an example, we may consider a rectangular thin plate, whose boundary is simply supported. This means that the deflexion u still vanishes on Γ but $\partial_\nu u$ is not prescribed thereupon (i.e. rotations are allowed along the boundary). But in this case, besides the essential boundary condition satisfied by all the admissible deflexions, the solution will necessarily satisfy the zero momentum boundary condition $\Delta u = 0$, which is thus a natural boundary condition. In this case, the rectangular plate-bending problem reduces to two Poisson equations with homogeneous Dirichlet boundary conditions, that is,

$$\begin{cases} -\Delta w = f \text{ in } \Omega \text{ and } w = 0 \text{ on } \Gamma \\ \text{followed by} \\ -\Delta u = w \text{ in } \Omega \text{ and } u = 0 \text{ on } \Gamma. \end{cases}$$

Of course, it is no point addressing again the solution of Poisson problems in this chapter, and hence we will concentrate on the clamped plate problem.

Remark 7.1

Whenever a simply supported plate has a curved boundary, the problem does not reduce to the solution of a sequence of two Poisson equations. Indeed, in this case the zero momentum boundary condition is $\sigma\Delta u + (1-\sigma)\partial_{nn}u = 0$, where σ *is Poisson's ratio of the material the plate is made of, and* $\partial_{nn}u$ *denotes the second-order outer normal derivative of* u *along the curved boundary. Notice that in principle,* $\partial_{nn}u$ *differs from the derivative in the direction of the outer normal* $\vec{\nu}$ *of the derivative of* u *in the same direction (i.e.,* $\partial_\nu u = (\mathbf{grad}u|\vec{\nu})$*). This is because the continuous variation of* $\vec{\nu}$ *along* Γ *must be taken into account to determine* $\partial_{nn}u$.

7.1.2 The 13-point FD Scheme

Like in the case of second-order PDEs, a simple way to solve equation (7.3) is the FDM. For the sake of simplicity, we consider only the case of uniform grids. Using divided differences, there is no essential difficulty to treat grids with variable spacings, but in the case of the biharmonic equation the description of the scheme becomes a little clumsy.

Let the plate occupy the domain $\Omega = (0, L_x) \times (0, L_y)$ in the plane equipped with a Cartesian coordinate system $(O; x, y)$. The uniform grid has spacings $h_x = L_x/n_x$ and $h_y = L_y/n_y$ for given integers $n_x \geq 3$ and $n_y \geq 3$. Like in Chapter 4, the grid points with coordinates $(ih_x; jh_y)$ are denoted by $M_{i,j}$, for $i = 0, 1, \ldots, n_x$ and $j = 0, 1, \ldots, n_y$.

Let $w = \Delta u$. Denoting by $u_{i,j}$ the approximate values of u at $M_{i,j}$ for $i = 1, \ldots, n_x - 1$ and $j = 1, \ldots, n_y - 1$, we know from Subsection 5.1.1 that second-order approximations of w at the same points are given by

$$w(ih_x, jh_y) \simeq [u_{i,j-1} + u_{i,j+1} - 2u_{i,j}]/h_x^2 + [u_{i-1,j} + u_{i+1,j} - 2u_{i,j}]/h_y^2,$$

where $u_{0,j} = u_{n_x,j} = 0$ for $j = 0, 1, \ldots, n_y$ and $u_{i,0} = u_{i,n_y} = 0$ for $i = 0, 1, \ldots, n_x$. Now, in order to extend the approximation of w to the boundary points $M_{0,j}$, $M_{n_x,j}$ for $j = 1, \ldots, n_y - 1$ and $M_{i,0}$, M_{i,n_y} for $i = 1, \ldots, n_x - 1$, we proceed as follows:

Referring to Figure 7.1, similarly to Chapter 1, we use the fictitious point technique to take into account the boundary conditions $[\partial_x u](M_{0,j}) = [\partial_x u](M_{n_x,j}) = 0$ for $j = 1, \ldots, n_y - 1$ and

$[\partial_y u](M_{i,0}) = [\partial_y u](M_{i,n_y}) = 0$ for $i = 1, \ldots, n_x - 1$. This naturally leads to the definitions $u_{-1,j} := u_{1,j}$, $u_{n_x+1,j} := u_{n_x-1,j}$ for $j = 1, \ldots, n_y - 1$ and $u_{i,-1} := u_{i,1}$, $u_{i,n_y+1} := u_{i,n_y-1}$ for $i = 1, \ldots, n_x - 1$. In doing so w is definitively approximated at $M_{i,j}$ by $\Delta_h(u_{i,j})$ applying the same recipe as in Section 4.2, that is,

$$\begin{cases} \text{For } i = 1, \ldots, n_x - 1 \text{ and } j = 0, 1, \ldots, n_y, \\ \text{and for } i = 0 \text{ or } i = n_x \text{ and } j = 1, \ldots, n_y - 1, \\ \Delta_h(u_{i,j}) := \dfrac{u_{i-1,j} + u_{i+1,j} - 2u_{i,j}}{h_x^2} + \dfrac{u_{i,j-1} + u_{i,j+1} - 2u_{i,j}}{h_y^2}. \end{cases} \tag{7.4}$$

Naturally enough, Δw at an inner grid point $M_{i,j}$ can be handled by the following:

Approximation of $\Delta^2 u$ at $M_{i,j}$

$$\begin{aligned} &\text{For } i = 1, \ldots, n_x - 1 \text{ and } j = 1, \ldots, n_y - 1, \\ &\Delta_h^2(u_{i,j}) := \frac{\Delta_h(u_{i+1,j}) + \Delta_h(u_{i-1,j}) - 2\Delta_h(u_{i,j})}{h_x^2} \\ &+ \frac{\Delta_h(u_{i,j+1}) + \Delta_h(u_{i,j-1}) - 2\Delta_h(u_{i,j})}{h_y^2} \end{aligned} \tag{7.5}$$

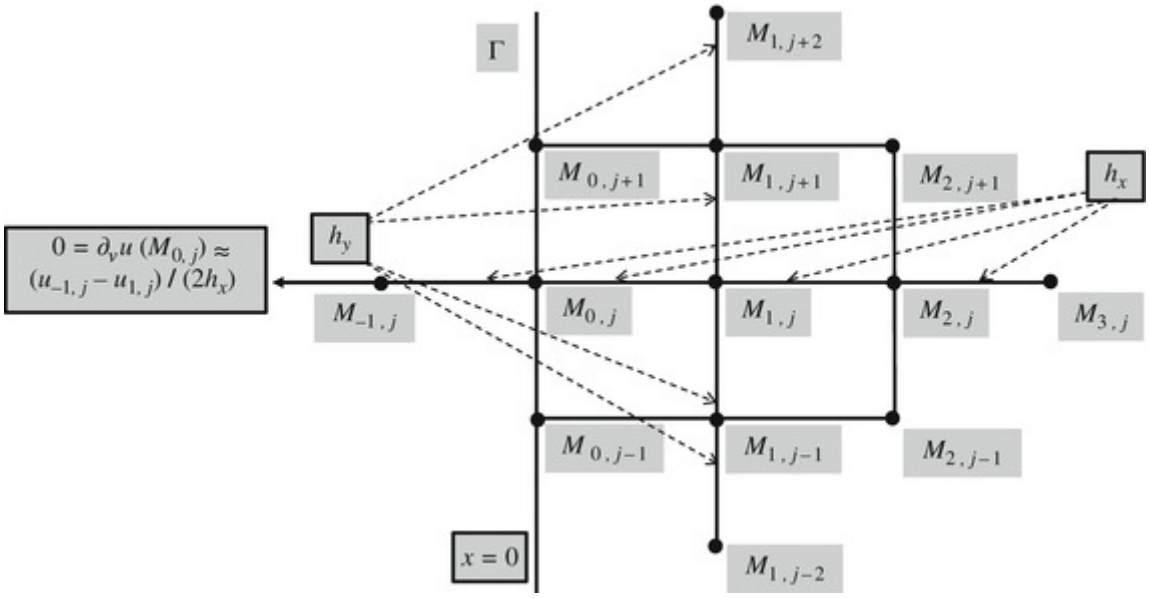


Figure 7.1 The 13-point FD stencil centred at point $M_{1,j}$, for $1 \le j \le n_y - 1$

If we replace the values of $\Delta_h(u_{k,l})$ involved in the expression of $\Delta_h^2(u_{i,j})$ (i.e., values of k (resp. l) equal to either i, $i-1$ or $i+1$ (resp. j, $j-1$ or $j+1$)) by using equation 7.4, we figure out that there are exactly 13 points $M_{k,l}$ involved in the resulting expression. The combination of equations 7.4 and (7.5) yields rather lengthy expressions, which are left to the

reader as a part of Exercise 7.1. Nevertheless, we give below the resulting scheme for the case where Ω is a square and a squared grid is employed, i.e. $n_x = n_y = n$. Setting $h := h_x = h_y$ we have

The 13-point FD scheme with a squared grid for the biharmonic equation in a square

$$\begin{gathered}\text{For } i = 1, \ldots, n-1 \text{ and } j = 1, \ldots, n-1, \\ 20u_{i,j} + 2(u_{i-1,j-1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i+1,j+1}) \\ -8(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) + (u_{i-2,j} + u_{i+2,j} + u_{i,j-2} + u_{i,j+2}) \\ = h^4 f(ih, jh).\end{gathered} \tag{7.6}$$

This scheme is stable in a sense specified in [94], and its local truncation error for a solution sufficiently smooth is an $O(h^2)$ (cf. Exercise 7.1). This yields error estimates in the same sense of order 3/2 (cf. [94]). Further studies on the 13-point FD scheme are proposed to the reader in Exercises 7.1 and 7.5. More details can be found in several classical books dealing with the FDM, such as references [52] or [71] and [94].

7.1.3 Hermite FEM in Intervals and Rectangles

In the two-dimensional case, there are not as many FEM for solving fourth-order boundary value problems as for second-order ones. Indeed, for reasons to be clarified hereafter, Lagrange FEs are not suited to the solution of biharmonic equations or problems alike. In the case of fourth-order boundary value problems, we must resort to **Hermite FE**s. This means that the solution degrees of freedom (i.e. unknowns), involve derivative values besides function values. In this way, continuity of both can be enforced at inter-element boundaries. On the other hand, constructing FEs belonging to this class is much more difficult. As a matter of fact, very few options are available, and we cannot really talk about families of elements like the $\mathcal{P}_k$ and the $\mathcal{Q}_k$ FEs. The only exception is the one-dimensional case, in which a family of Hermite FEs defined upon disjoint intervals, starting from degree three, can be defined. We will work this out in more details in the sequel. For the moment, let us just write equation (7.3) in standard variational form, in order to examine a little better the restrictions pointed out above. Assuming only that Ω has no boundary singularities, first of all we multiply both sides of equation (7.1) with a function v and integrate in Ω. We make the assumption that $v \in H^2(\Omega)$ because we next apply First Green's indentity twice to obtain successively:

$$\int_\Omega \Delta\Delta u\, v\, dxdy = \oint_\Gamma \partial_\nu \Delta u\, v\, ds - \int_\Omega (\mathbf{grad}\Delta u|\mathbf{grad}\, v)\, dxdy = \int_\Omega f\, v\, dxdy,$$

$$\int_\Omega \Delta\Delta u\, v\, dxdy = \oint_\Gamma \partial_\nu \Delta u\, v\, ds - \oint_\Gamma \Delta u\, \partial_\nu v\, ds + \int_\Omega \Delta u \Delta v\, dxdy = \int_\Omega f\, v\, dxdy.$$

Now, analogously to the case of (P_3), we choose v to satisfy $v = 0$ and $\partial_\nu v = 0$ on Γ. In this case, we simply have $\int_\Omega \Delta u \Delta v\, dxdy = \int_\Omega f v\, dxdy$. Denoting by $H_0^2(\Omega)$ the subset of $H^2(\Omega)$ consisting of functions satisfying both boundary conditions[1], we thus have the

Standard Galerkin variational formulation of equation 7.3

$$\boxed{\begin{array}{l}\text{Find } u \in H_0^2(\Omega) \text{ such that} \\ \int_\Omega \Delta u \Delta v\, dxdy = \int_\Omega f v\, dxdy\ \forall v \in H_0^2(\Omega).\end{array}} \tag{7.7}$$

The other way around, the reader can easily check as Exercise 7.2 that equation (7.7) implies (7.3), assuming a suitable regularity of u. Incidentally, equation (7.7) has a unique solution, which is a consequence of the Lax–Milgram theorem (see e.g. [45]). Moreover, provided Ω is sufficiently smooth (and polygons are included in this category), the following stability property holds for problem (7.7):

$$\|H(u)\|_{0,2} \leq C_P^2 \|f\|_{0,2} \tag{7.8}$$

where C_P is the constant of the Friedrichs–Poincaré inequality. Indeed, first we note that $\|H(w)\|_{0,2} = \|\Delta w\|_{0,2}$ for every $w \in H_0^2(\Omega)$. Inspiring herself or himself in the argument of Subsection 5.1.2, the reader can check this equality as Exercise 7.3. Therefore, taking $v = u$, by the Cauchy–Schwarz inequality followed by the Friedrichs–Poincaré inequality, we have, successively,

$$\begin{aligned}\|H(u)\|_{0,2}^2 &\leq \|f\|_{0,2}\|u\|_{0,2}, \\ \|H(u)\|_{0,2}^2 &\leq C_P \|f\|_{0,2}\|\mathbf{grad}\, u\|_{0,2}.\end{aligned} \tag{7.9}$$

Now, we note that by First Green's identity $\|\mathbf{grad}\, u\|_{0,2}^2 = -\int_\Omega \Delta u\, u\, dxdy$. Hence, applying the Cauchy–Schwarz inequality on the right side of this identity followed by the Friedrichs–Poincaré inequality, we easily derive (please check!)

$$\|\mathbf{grad}\, u\|_{0,2} \leq C_P \|H(u)\|_{0,2}. \tag{7.10}$$

Plugging equation (7.10) into (7.9), we obtain (7.8).

The next step is to introduce a FE analog of equation (7.7), which will lead to an approximation u_h of u associated with a given mesh. Why is Hermite interpolation necessary? The answer to this question is to be found in the next two paragraphs.

Two approaches are commonly used to solve a boundary value problem in variational form by the FEM. The **internal approximation** in which the FE space spanned by given shape functions is a subspace of the working space. This gives rise to a **conforming FEM**. Notice that in the case of equation (7.7), the working space is $H_0^2(\Omega)$. The other approach is the **external approximation** in which the underlying FE space is not a subspace of the working space. In this case, the method is called a **non-conforming FEM**. This allows some flexibility and hence simpler FE constructions. For example, a condition like $\partial_\nu u_h = 0$ everywhere on Γ can be relaxed, by enforcing it only at a finite number of boundary points. But in this case, obviously $u_h \notin H_0^2(\Omega)$, and there is a price to pay. First of all, the variational formulation for a non-conforming FEM does not inherit the nice stability property equation (7.8) of the continuous problem, and hence an *ad hoc* stability analysis must be carried out in each case. Furthermore, the easy-to-prove consistency result using the FE interpolate of u cannot be applied to a non-conforming approximation. In this connection, it is generally considered that Iron's **patch-test** (see e.g. [45]) implies the consistency of a non-conforming FEM, but we decline to further elaborate on this issue, because definitively we will only consider a conforming method in this chapter. For further information on non-conforming FEM for the biharmonic equation, the author refers to reference [120], and for conforming ones to reference [45].

So far, only a partial answer to our question on Hermite interpolation has been given: somehow the normal derivative must be a method's degree of freedom, otherwise the vanishing rotation condition for a clamped plate will not be enforced at all. But there is another condition even more stringent: if we are to ensure that a piecewise polynomial function lies in $H^2(\Omega)$, then it must be continuously differentiable at inter-element boundaries (i.e. of the C^1-class). Indeed, if the function is only continuous at such interfaces we can compute its first-order derivatives like in the case of the $\mathcal{P}_1$ FEM. However, the gradient of such a function will be discontinuous at the interfaces, and hence neither $H(u)$ nor Δu will be defined thereupon. Notice that somehow a non-conforming FEM must mimic this C^1 property, by satisfying for instance the continuous differentiability property at least at some points of the inter-element boundaries. But this is only possible if normal derivatives are degrees of freedom in the FE space, and hence in all cases the use of Hermite FE is a must to solve biharmonic problems.

Remark 7.2

The above assertion on Hermite FEs applies to the standard Galerkin formulation (equation). If some other techniques are used, such as ***discontinuous Galerkin methods*** *(see e.g. [48]), then this continuous differentiability condition, and also a mere continuity requirement, can be disregarded even for second-order problems. A little more information about this approach is provided in the Appendix.*

Before pursuing the description of the FE solution of equation (7.7), it is advisable to introduce the FEM based on Hermite interpolation in the framework of the beam-bending model problem posed in an interval $(0, L)$. More precisely, given $f \in L^2(0, L)$ proportional to a force field, find the deflexion u of the beam satisfying

The clamped beam-bending equation

$$\boxed{\begin{aligned} &u^{(iv)} = f \text{ in } (0, L) \\ &u(0) = u(L) = u'(0) = u'(L) = 0. \end{aligned}} \tag{7.11}$$

Of course, unless f is too complex, equation (7.11) can be solved by hand, but this is not the point. The reader certainly understood that we are using this equation just to present some concepts inherent to Hermite FEs. In this aim, let us first set it in variational form. Notice that, as a one-dimensional counterpart of equation (7.3) all the boundary conditions in equation (7.11) are essential, and hence they must be incorporated into the working space. Hence, this space is $H_0^2(0, L)$, that is, the subspace of $H^2(0, L)$ consisting of functions v satisfying $v(0) = v(L) = v'(0) = v'(L) = 0$.

Using elementary integration by parts, it is easy to see that the equivalent Galerkin variational form of equation (7.11) is the symmetric problem

$$\begin{cases} \text{Find } u \in H_0^2(0, L) \text{ such that} \\ \int_0^L u'' v'' \, dx = \int_0^L fv \, dx \; \forall v \in H_0^2(0, L). \end{cases} \tag{7.12}$$

We next consider a mesh $\mathcal{T}_h$ of $(0, L)$ consisting of the closure T_i of n disjoint intervals (x_{i-1}, x_i) for $i = 1, n$ with $0 = x_0 < x_1 < \ldots < x_{n-1} < x_n = L$. We use again the notation h_i for $x_i - x_{i-1}$, together with $h := \max_{1 \le i \le n} h_i$. For the same reasons already pointed out in the

case of equation (7.7), a conforming FEM associated with this mesh must be based on continuously differentiable functions at all the x_i's for $i = 1, 2, \ldots, n-1$. Moreover, we should be able to enforce the essential boundary conditions exactly. Then a natural solution method would be based on spaces V_h associated with $\mathcal{T}_h$ defined as follows for some integer $k > 0$:

$$V_h := \{v |\ v \in C^1[0, L], v \in \mathcal{P}_k(T_i), i = 1, \ldots, n, v(0) = v(L) = v'(0) = v'(L) = 0\}.$$

In the FEM, polynomials of a certain type are defined in the elements independently from each other using certain degrees of freedom. Then they are assembled together by enforcing coincidence of degrees of freedom at inter-element boundaries, so as to satisfy required continuity or differentiability properties. For example, in the case of the $\mathcal{P}_1$ FEM the degrees of freedom are only function values at two points in each interval. These are chosen to be the interval end-points in order to ensure the continuity of the assembled piecewise linear function by requiring coincidence of the function values at these points from each side. In the case of V_h, we must enforce the coincidence not only of function values, but also of first-order derivatives at all interval end-points. For this reason, four degrees of freedom pertaining to each element are necessary. Therefore, the minimum possible value of k is three. Indeed, a cubic function in an interval is defined by a linear combination of the monomials $1, x, x^2, x^3$, and hence there is hope for constructing a cubic function $\varphi \in \mathcal{P}_3(T_i)$ satisfying four conditions, namely,

$$\varphi(x_{i-1}) = d_1;\ \varphi(x_i) = d_2, \varphi'(x_{i-1}) = d_3;\ \varphi'(x_i) = d_4,$$

for any set of given real numbers d_1, d_2, d_3, d_4. Actually, using the auxiliary functions

$$\phi_0(x) := 1 - 3x^2 + 2x^3 \text{ and } \phi_1(x) := x - 2x^2 + x^3,$$

we construct four shape functions, namely,

- $\varphi_1^i(x) = \phi_0([x - x_{i-1}]/h_i)$;
- $\varphi_2^i(x) = \phi_0([x_i - x]/h_i)$;
- $\varphi_3^i(x) = h_i\phi_1([x - x_{i-1}]/h_i)$;
- $\varphi_4^i(x) = h_i\phi_1([x_i - x]/h_i)$.

so that $\varphi := \sum_{l=1}^4 d_l\varphi_l^i$ fulfils the required conditions, as the reader can easily check. The above list indicates that at a global level, there are two types of Hermite shape functions associated with an inner vertex of coordinate $x = x_i$, that is, for $i = 1, \ldots, n-1$, spanning the corresponding

FE space V_h: 'odd' shape functions φ_{2i-1} satisfying $\varphi_{2i-1}(x_j) = \delta_{ij}$ and $\varphi'_{2i-1}(x_j) = 0$ for $j = 0, 1, \ldots, n$, and 'even' shape functions φ_{2i} satisfying $\varphi_{2i}(x_j) = 0$ and $\varphi'_{2i}(x_j) = \delta_{ij}$, for $j = 0, 1, \ldots, n$. Both are illustrated in Figure 7.2. In short, in case $k = 3$, a function in V_h is a linear combination of the shape functions in the set $\{\varphi_k\}_{k=1}^{2(n-1)}$. In particular, the solution u_h of the FE counterpart of equation (7.12) obtained by replacing $H_0^2(0, L)$ with V_h is given by $\sum_{k=1}^{2(n-1)} u_k \varphi_k$, where $u_{2i-1} = u_h(x_i)$ and $u_{2i} = u'_h(x_i)$ for $i = 1, 2, \ldots, n-1$. Checking this assertion is a simple task left to the reader.

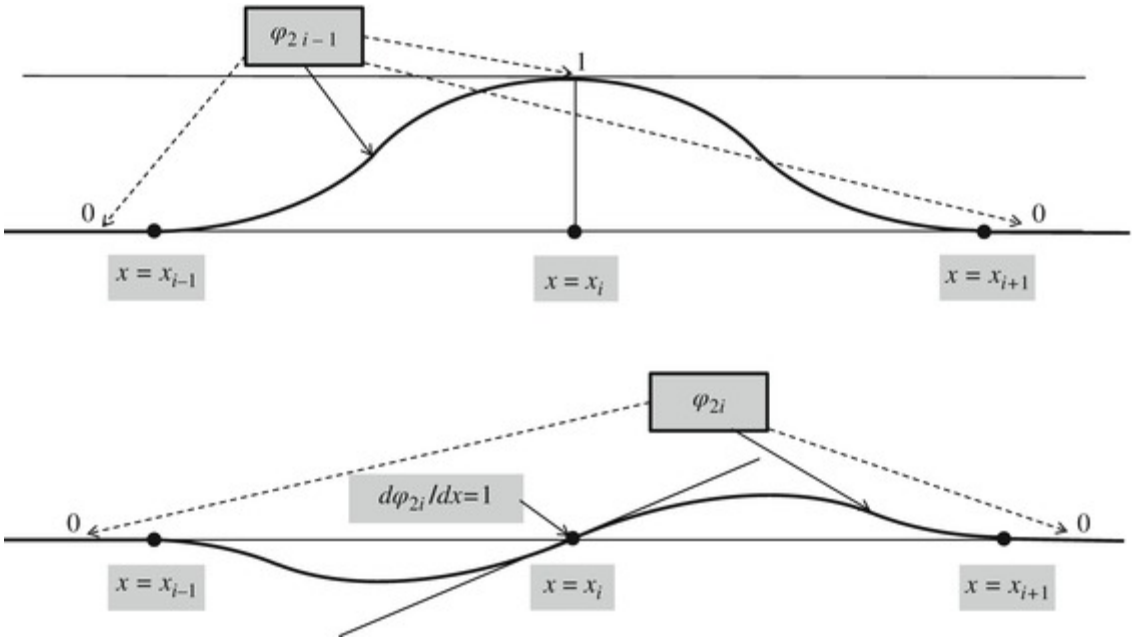


Figure 7.2 The Hermite piecewise cubic shape functions φ_{2i-1} and φ_{2i}, $1 \leq i \leq n-1$

We do not insist on the equivalent SLAE to determine these degrees of freedom of u_h, for it is similar to those we have dealt with so far. We do not elaborate either on FE spaces V_h with $k > 3$, although it is possible to define them in a similar manner whatever k, by enforcing the continuity of shape functions and their derivatives at interval end-points. This is because such constructions are seldom useful in practice.

Remark 7.3

*Using quadratic **splines**, it is possible to construct a function of the* C^1*-class, whose restriction to* T_i *belongs to* $\mathcal{P}_2(T_i)$ for every i. *However, in this case the functions in the approximation space cannot be defined in the intervals independently from each other. As a consequence, a given degree of freedom will lose its local character, since it will influence far away portions of the domain. This situation would be in conflict with the original concept of the FEM. For more information about splines, the reader can consult reference [2].*

After this brief introduction to Hermite FEs in one-dimensional space, we go back to equation (7.3). Still assuming that Ω is the rectangular domain $(0, L_x) \times (0, L_y)$, let $\mathcal{R}_h$ be a non-uniform partition of it into rectangles, satisfying the usual compatibility conditions. Similarly to the non-uniform FD grid introduced in Section 4.2, we denote the x coordinates of the mesh vertices by $0 = x_0 < x_1 < \ldots < x_{n_x-1} < x_{n_x} = L_x$ and its y coordinates by $0 = y_0 < y_1 < \ldots < y_{n_y-1} < y_{n_y} = L_y$. We recall that the mesh spacings are $h_i = x_i - x_{i-1}$ for $i = 1, \ldots, n_x$ and $k_j = y_j - y_{j-1}$ for $j = 1, \ldots, n_y$. We also set $h = \max\{\max_{1\le i\le n_x} h_i, \max_{1\le j\le n_y} k_j\}$.

We will next describe the **Bogner–Fox–Schmit** FEM [31], which is a conforming method based on $\mathcal{Q}_3$ functions in a rectangle. More precisely, we define a space W_h as follows:

$$W_h = \{v |\ v \in C^1(\bar{\Omega}), v_{|R} \in \mathcal{Q}_3(R)\ \forall R \in \mathcal{R}_h\}.$$

We admit for the moment that the set W_h is not empty, and endeavour to construct a subset V_h of W_h with the following characteristics.

Referring to Figure 7.3, where a rectangle $R \in \mathcal{R}_h$ is represented, we first define the local degrees of freedom associated with a function $v \in V_h$ restricted to R to be its values, together with those of its partial derivatives $\partial_x v$, $\partial_y v$ and $\partial_{xy} v$ at the four vertices S_i^R of R, $i = 1, 2, 3, 4$. Let us number these degrees of freedom for a given function in $\mathcal{Q}_3(R)$ as $\mathcal{F}_{ij}$ for $i = 1, 2, 3, 4$ and $j = 1, 2, 3, 4$, where subscript i stands for the vertex number and j for the type of degree of freedom. More precisely, $j = 1$ refers to a function value, $j = 2$ to an x-derivative, $j = 3$ to a y-derivative and $j = 4$ to a cross second-order derivative. Noticing that a function in $\mathcal{Q}_3$ is a linear combination of the 16 monomials

$1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3, x^3y, x^2y^2, xy^3, x^3y^2, x^2y^3$ and x^3y^3, the following property is to be expected, although it must be confirmed: Given a set of 16 real values d_{ij}, $1 \leq i, j \leq 4$ of the degrees of freedom specified above, we can define a unique function $\varphi \in \mathcal{Q}_3$ such that, for $1 \leq i, j \leq 4$,

$$\varphi(S_i^R) = d_{i1};\ [\partial_x \varphi](S_i^R) = d_{i2};\ [\partial_y \varphi](S_i^R) = d_{i3};\ [\partial_{xy} \varphi] = d_{i4}.$$

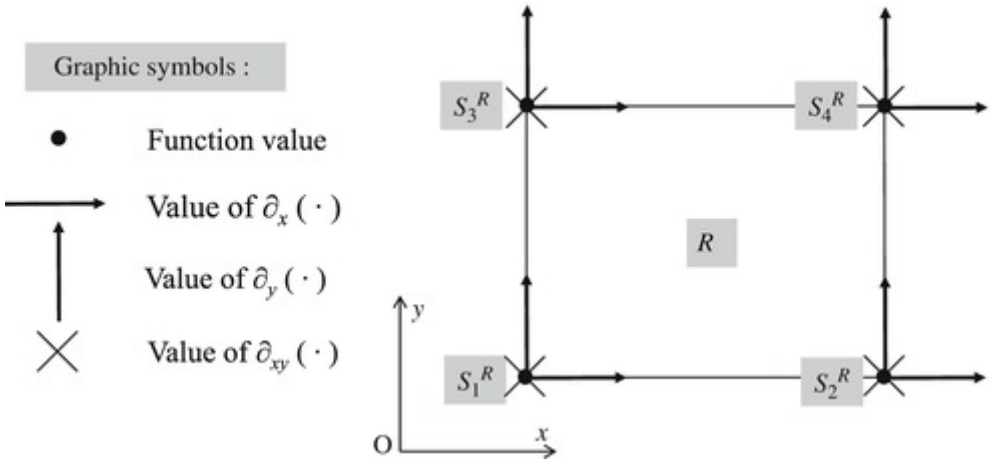


Figure 7.3 The Bogner–Fox–Schmit rectangular element

This is possible indeed, since we can exhibit 16 canonical basis functions $\varphi_{ij}^R \in \mathcal{Q}_3(R)$, for $i, j = 1, 2, 3, 4$, such that

- $\varphi_{i1}^R(S_j^R) = \delta_{ij};\ [\partial_x \varphi_{i1}^R](S_j^R) = [\partial_y \varphi_{i1}^R](S_j^R) = [\partial_{xy} \varphi_{i1}^R](S_j^R) = 0$;
- $[\partial_x \varphi_{i2}^R](S_j^R) = \delta_{ij};\ \varphi_{i2}^R(S_j^R) = [\partial_y \varphi_{i2}^R](S_j^R) = [\partial_{xy} \varphi_{i2}^R](S_j^R) = 0$;
- $[\partial_y \varphi_{i3}^R](S_j^R) = \delta_{ij};\ \varphi_{i3}^R(S_j^R) = [\partial_x \varphi_{i3}^R](S_j^R) = [\partial_{xy} \varphi_{i3}^R](S_j^R) = 0$;
- $[\partial_{xy} \varphi_{i4}^R](S_j^R) = \delta_{ij};\ \varphi_{i4}^R(S_j^R) = [\partial_x \varphi_{i4}^R](S_j^R) = [\partial_y \varphi_{i4}^R](S_j^R) = 0$.

It follows that the function $\varphi := \sum_{i=1}^{4} \sum_{j=1}^{4} d_{ij} \varphi_{ij}^R$ satisfies the required conditions.

The expressions of the basis functions φ_{ij}^R can be found in many books or articles, and in this respect we refer for instance to reference [86].

Now, naturally enough, our FE space V_h consists of functions in W_h that are continuous, as much as their gradients and cross second-order derivatives, at all the inner mesh vertices, and whose values together with those of their gradients and cross second-order derivatives vanish at all the boundary vertices. If these conditions allow for the construction of functions of the C^1-class, then we will have validated a posteriori the initial assumption that W_h is not empty. Let us see how this works.

First of all, we note that the trace of a function in V_h on a mesh inner edge e_x parallel to the Ox axis is a cubic function in terms of x. By construction, this function together with the x derivatives at the edge end-points have the same values in both rectangles sharing e_x. Taking into account what we showed for the one-dimensional Hermite interpolation with cubics, it follows that the cubic traces from both sides of e_x must be the same. The same argument obviously applies to inner edges parallel to the Oy axis. Next, we examine the situation of normal derivative traces along an inner edge e_x. The normal derivative along e_x is a partial derivative with respect to y, and as the reader may check (cf. Exercise 7.4), such traces are also cubic functions in terms of x. Therefore, by the same argument as above, continuity of the normal derivative along e_x will be ensured if this cubic function together with its x partial derivatives at both ends of this edge coincide for both elements sharing it. By construction again, this condition is fulfilled since the y derivatives and their x-derivatives, that is, the cross second-order derivatives, are continuous at all the mesh vertices.

Finally, the reader can check as a part of Exercise 7.4 that, as long as the four degrees of freedom at all the vertices located on Γ vanish, every function in $v \in V_h$ satisfies the boundary conditions $v = \partial_\nu v = 0$ everywhere on Γ. This establishes that V_h is indeed a subspace of $H_0^2(\Omega)$, which means that the following problem is an internal approximation of the biharmonic equation with homogeneous essential boundary conditions, namely,

The Bogner–Fox–Schmit FE counterpart of equation 7.7

$$\boxed{\begin{array}{l}\text{Find } u_h \in V_h \text{ such that} \\ \int_\Omega \Delta u_h \Delta v \, dxdy = \int_\Omega f v \, dxdy \; \forall v \in V_h.\end{array}} \tag{7.13}$$

The total number of degrees of freedom defining a function in V_h is $N_h = 4(n_x - 1)(n_y - 1)$. Since we are dealing with a structured mesh, the unknown numbering may follow the line-by-line systematic already described in similar situations. Here, the only peculiarity is that the numbers of

the four degrees of freedom attached to each vertex should be preferably consecutive: for instance, $4i-3$ for the function value, $4i-2$ for the x derivative, $4i-1$ for the y derivative and $4i$ for the cross second-order derivative, where i sweeps the mesh from one through N_h.

The shape functions underlying the $N_h \times N_h$ matrix and the right-side vector for the SLAE corresponding to equation (7.13) are constructed by putting together basis functions for elements forming a patch of elements surrounding a node. This allows us to proceed in a standard manner. Then both are obtained by assembling 16×16 element matrices and 16 component element vectors. The solution of the resulting SLAE yields the degrees of freedom of u_h, namely, approximations of u, $\mathbf{grad}\, u$ and $\partial_{xy} u$ at the inner mesh vertices. By interpolation using the local shape functions, we can determine u_h everywhere in Ω.

We refrain from elaborating on error estimates for u_h at this introductory level. We just quote known results stating that, as long as all the fourth-order partial derivatives of u belong to $L^2(\Omega)$ and the family of meshes in use is quasi-uniform, then for a constant $C(u)$ independent of h that vanishes if all the fourth-order partial derivatives of u equal zero, it holds that (cf. [45])

$$\| u - u_h \|_{1,2} + \| H(u - u_h) \|_{0,2} \leq C(u) h^2. \tag{7.14}$$

The above error estimate is logical for two reasons. First of all, it means that $u = u_h$ if u happens to belong to $\mathcal{P}_3$ but not to $\mathcal{P}_k$ for $k > 3$. Indeed, it is so because $\mathcal{Q}_3$ contains $\mathcal{P}_3$ but not $\mathcal{P}_4$. On the other hand, second-order convergence is the best we can hope for, since the norm of $H^2(\Omega)$ came into play in the biharmonic case. This implies a one-point downgrade in a method's order, as compared to estimates for second-order equations, whose working norm is $\| \cdot \|_{1,2}$.

Finally, it would be interesting to compare the 13-point FD scheme with the Bogner–Fox–Schmit FE, in the light of their costs. In Exercise 7.5, we propose to the reader the determination of their respective number of unknowns and bandwidths in comparable situations.

Remark 7.4

A result qualitatively equivalent to equation (7.14) holds for the FE approximation of equation (7.11) using the Hermite cubics studied in this subsection.

7.2 The Advection–Diffusion Equation

The advection–diffusion equation, also called the **convection–diffusion equation** depending on context, is the linear paradigm of several non-linear PDEs that model important physical events. Outstanding examples are the incompressible and compressible Navier–Stokes equations governing viscous flow (cf. [15]). The advection–diffusion equation itself has countless applications in engineering and science. The equation carries this name because it models two coupled processes: diffusion and advection of an unknown quantity, which can be as diverse as a pollutant concentration in air or water, or heat distribution in fluids. Once again, it would be out of purpose going into details on phenomenon modelling in this book. Nevertheless, such an aspect of the equation to be studied is briefly addressed hereafter just to keep ideas clear.

7.2.1 A Model One-Dimensional Equation

Let us consider the following problem. We wish to determine the temperature u of a fluid in motion in a straight cylindrical tube whose total length equals L, with mean cross section velocity $\vec{v}(x) = v(x)\vec{e}_x$, where $\vec{e}_x$ is the unit vector parallel to the tube's axis of symmetry, and x is the abscissa along this axis, with $x = 0$ at the tube's ‘left end’. Assuming that heat conduction is also taking place, and that the medium's conductivity is constant equal to $p > 0$, the heat flux accross a section at abscissa x in the sense of increasing x is given by $-pu'(x)$. Further assuming that the velocity v is constant, for any subdomain of the tube, say, $\omega = (a_\omega, b_\omega)$, the incoming and outgoing heat fluxes are given by $pu'(a_\omega) - vu(a_\omega)$ and $-pu'(b_\omega) + vu(b_\omega)$. Moreover, if a distribution g of heat sources is acting along the tube while the heat transfer goes on, the balance of (thermal) energy requires that

$$pu'(a_\omega) - vu(a_\omega) - pu'(b_\omega) + vu(b_\omega) = \int_{a_\omega}^{b_\omega} g\, dx.$$

This readily implies

$$\int_{a_\omega}^{b_\omega} [-pu'' + vu' - g]dx = 0 \ \forall \omega \subset (0, L).$$

Finally, assuming that the temperature is kept fixed at both ends of the tube, say $u(0) = u(L) = 0$, recalling the arguments developed in Section 1.1, we immediately derive the advection–diffusion equation in its simplest expression, namely,

The one-dimensional advection–diffusion equation

$$\boxed{\begin{aligned} -pu'' + vu' &= g \text{ in } (0, L) \\ u(0) &= u(L) = 0 \end{aligned}} \tag{7.15}$$

It is easy to see that this equation has a unique solution, and clearly enough, provided g is not too wild, it can be solved by hand. But once again, this is not the point; (7.15) is just the basic model that will allow us to introduce some popular numerical techniques aimed at overcoming delicate issues often encountered in the solution of advection–diffusion equations. Let us see right away the main source of problems.

First, we set $P := L|v|/p$ and $\epsilon = P^{-1}$. P is a dimensionless parameter called the **Péclet number**. This parameter points in particular to two important situations. The advection-dominant case corresponds to a high value of P. Conversely, whenever P is not so large, the diffusion will influence the phenomenon being modelled practically everywhere in the domain, and convection will not be dominant.

Switching to the dimensionless coordinate $\bar{x} = x/L$ and defining $\bar{u}(\bar{x}) := u(x)$ for all $x \in [0, L]$, a simple manipulation of the ODE in equation (7.15) yields the following equation for $\bar{u}$:

$$-\epsilon \bar{u}'' \pm \bar{u}' = \bar{g} \text{ where } \bar{g}(\bar{x}) = g(x)L/|v|.$$

supplemented with the boundary conditions $\bar{u}(0) = \bar{u}(1) = 0$. Let us drop the bars above the variables x, u and g and definitively work with the **dimensionless equation** (cf. [5]):

$$-\epsilon u'' \pm u' = g \text{ with } u(0) = u(1) = 0. \tag{7.16}$$

Taking $g \equiv 1$, the analytical solution of the above ODE is easily seen to be

$$u(x) = x - \frac{e^{\pm x\epsilon^{-1}} - 1}{e^{\pm\epsilon^{-1}} - 1}. \tag{7.17}$$

In Figure 7.4, we sketch a plot of this function u assuming that $\epsilon \ll 1$: the solution behaves like $y(x) = x$ everywhere for $x \geq 0$, except in a narrow **boundary layer** of width $\simeq \epsilon$ close to $x = 1$, where it abruptly changes of pattern in order to satisfy the boundary condition $u(1) = 0$.

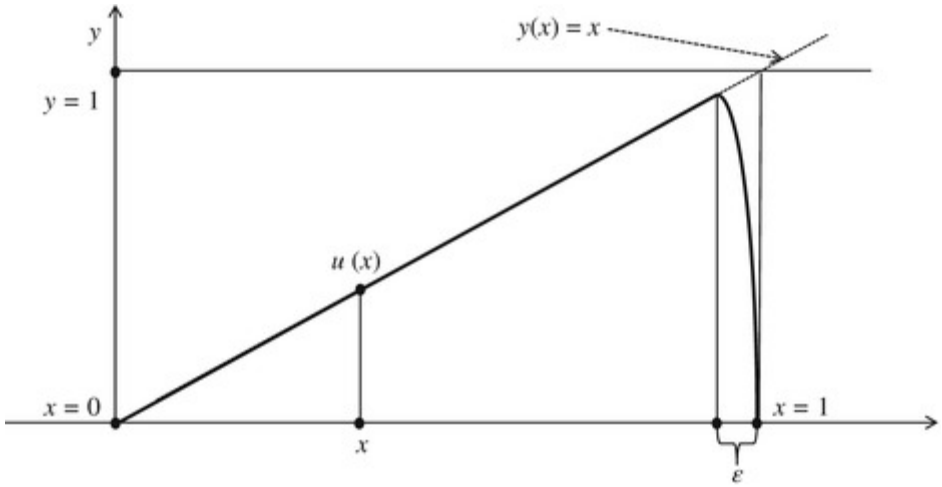


Figure 7.4 Solution of the advection–diffusion equation (7.16) with a boundary layer

Now suppose that, just for fun, we attempt to solve equation (7.16) by one of the numerical methods studied in this book. We guess it is not necessary to describe such procedures, to convince the reader that an accurate numerical solution would require very small grid or mesh sizes in the interior of the boundary layer. Notice that, in practice, the Péclet number can attain very large values, in case convection strongly dominates diffusion. For example, values of P as high as one million or more are perfectly compatible with realistic physical situations. Then the question is: Would it be reasonable to try to tackle a boundary layer behaviour by means of such fine discretisations? The answer seems to be no, but even if it were yes, there are other numerical issues typical of this class of equations that must be handled with much care. That is what we endeavour to show in the remainder of this section. However, before starting, we should make a few comments.

With a few exceptions, the numerical methods for equation (7.15) to be studied are well-known, and at this writing can be considered as classical techniques. Incidentally, there is a rather vast literature on this subject owing to its undeniable importance. However, the presentations in this section are partially homemade, in the sense that several relevant aspects are not to be found elsewhere, to the best of the author's knowledge. There is another reason for this approach. Numerical methods for the advection–diffusion equation can be viewed as an advanced topic. Going into its rigorous treatment would require mathematical resources more sophisticated than those employed elsewhere in this book.

Most advection–diffusion phenomena of practical interest require a two- or a three-dimensional modelling, specially because in general analytical solutions are not available in this case. However, the main obstacles to overcome in order to solve the advection–diffusion equation

numerically show up even in the one-dimensional case. Therefore, for the sake of brevity, we shall restrict to the latter framework the presentation of specific methodology to treat this problem. Nevertheless, we will address in Subsection 7.2.4 the time-dependent counterpart of equation (7.15). Extensions to higher dimensions will be the object of a few comments only.

For comprehensive studies on the topic, the author could cite references [116] and [103], among many others.

7.2.2 Overcoming the Main Difficulties with the FDM

From now on, instead of the dimensionless form (equation (7.16)), we rewrite the ODE in equation (7.15) as the

Working form of the advection–diffusion equation

$$\boxed{-\epsilon u'' + wu' = f \text{ with } u(0) = u(L) = 0,} \tag{7.18}$$

where $w = v/|v|$, $\epsilon = p/|v|$ and f stands for $g/|v|$. This means that we still work with the original variable x sweeping $(0, L)$. This will allow in particular to consider a bounded continuously varying v in $(0, L)$ a little later. In this case, $\|v\|_{0,\infty}$ will replace $|v|$ in the above definitions of w, f and ϵ. Notice that in this case too, equation (7.18) still has a unique solution, according to the well-known theory of linear ODEs.

Since we are dealing with a second-order two-point boundary value ODE, all the methods presented in Chapter 1 can be applied to solve it. However, now the term vu' replaces qu, and this makes a big difference. In this subsection, we will consider a solution commonly adopted in the framework of the FDM. In order to simplify the presentation, we restrict ourselves to uniform grids, leaving the non-uniform case to the next subsections, where the FEM and the FVM will be addressed.

Recalling the uniform grid and pertaining notations of Section 1.2, a natural FDM to solve equation (7.18), is based on centred FD providing a second-order local truncation error, that is,

$$\begin{cases} \text{Setting } u_0 = u_n = 0, \text{ find } u_i \text{ for } i = 1, 2, \ldots, n-1 \text{ such that} \\ \epsilon(2u_i - u_{i-1} - u_{i+1})/h^2 + w(u_{i+1} - u_{i-1})/(2h) = f_i \text{ for } i = 1, 2, \ldots, n-1. \end{cases} \tag{7.19}$$

Unfortunately, this scheme is subject to stringent stability conditions. Let us examine this issue a little closer. We have

$$u_i = h^2 f_i/(2\epsilon) + (1/2 - \alpha)u_{i+1} + (1/2 + \alpha)u_{i-1} \text{ where } \alpha = wh/(4\epsilon).$$

A numerical example is not a proof, but it turns out to be a valid argument to supply convincing indications about the behaviour of the above scheme: we take $L = 1$, $f \equiv 1$ and $\epsilon = 10^{-m}$ for m ranging from 1 to 5. Keeping $h = 10^{-2}$, we display in Table 7.1 $u(1/2)$ together with the corresponding approximate values u_{50} rounded to eight decimals for the five values of ϵ. The sharp increase of the errors so far from the boundary layer as ϵ diminishes does not happen by chance. Such a bad behaviour is inherent to the **Centred FD scheme**. However, it can be remedied, as seen hereafter. As a matter of fact, recalling the stability analyses carried out in Subsection 2.2.1, we may legitimately conjecture that we must have $|\alpha| \leq 1/2$, if we wish to enjoy nice stability properties in the maximum norm for this Centred scheme. But this requires $h \leq 2\epsilon$, which is prohibitive if $\epsilon \ll 1$. Furthermore, our computations strongly suggest that this scheme is suspect. As a conclusion, it is advisable to discard it, at least if ϵ is small.

Table 7.1 u_{50} versu $u(1/2)$ for $\epsilon = 10^{-m}$ and $h = 10^{-2}$, u solves equation (7.18) with $w = f = L = 1$

	0.493307149	0.500000000	0.500000000	0.500000000	0.500000000
	0.493334839	0.500000000	0.499999998	0.380825080	0.049834063

Inspired by the Upwind scheme studied in Chapter 3, let us try the following alternative: Letting $w^+ = \max[w, 0]$ and $w^- = \max[-w, 0]$, we set up

The Upwind FD scheme for equation 7.18

Setting $u_0 = u_n = 0$, find u_i such that for $i = 1, 2, \ldots, n-1$
$\epsilon(2u_i - u_{i-1} - u_{i+1})/h^2 + w^+(u_i - u_{i-1})/h - w^-(u_{i+1} - u_i)/h = f_i.$ (7.20)

Let us see what has changed with respect to the Centred FD scheme we have just discarded. Now u_i satisfies

$$(2\epsilon + hw^+ + hw^-)u_i = h^2 f_i + (\epsilon + hw^+)u_{i-1} + (\epsilon + hw^-)u_{i+1}.$$

Since the coefficients of u_{i-1} and u_{i+1} are both strictly positive, and moreover their sum equals the coefficient of u_i on the left side, there is no way for scheme (7.20) to be unstable. This property is interesting because it holds even if ϵ is very small. An interpretation of this conclusion relies upon the following argument:

Let us add and subtract to the left side of equation (7.20) the term $(w^+ + w^-)(u_{i+1} + u_{i-1})/(2h)$. After straightforward manipulations, taking into account that $w^+ + w^- = |w| = 1$ and $w^+ - w^- = w$, we come up with a different but equivalent form of equation (7.20), namely,

$$\begin{cases} \text{Setting } u_0 = u_n = 0, \text{ find } u_i \text{ such that for } i = 1, 2, \ldots, n-1 \\ (\epsilon + h/2)(2u_i - u_{i-1} - u_{i+1})/h^2 + w(u_{i+1} - u_{i-1})/(2h) = f_i. \end{cases} \tag{7.21}$$

We observe that in fact the Upwind scheme corresponds to the initial centred scheme, if we replace the diffusion coefficient ϵ with $\epsilon_h := \epsilon + h|w|/2$. Otherwise stated, equation (7.20) adds **numerical diffusion** to the equation, sometimes called **artificial diffusion**. This is what makes the difference in terms of stability. Indeed, replacing α by $\alpha_h = wh/(4\epsilon_h)$, the counterpart of condition $|\alpha| \leq 1/2$ is seen to be $h/(4\epsilon + 2h) \leq 1/2$, which holds true for every h. The problem with numerical diffusion is that, if too much of it is added, then big accuracy losses might be observed locally, in particular close to boundary layers. As pointed out at the beginning of this section, it is difficult to simulate accurately in the pointwise sense narrow boundary layers. Hence, in advection–diffusion computations at high Péclet numbers, the practitioner could be satisfied with a good accuracy in the mean-square sense, by giving up very precise representations of the solution in certain narrow regions. In the numerical example using a $\mathcal{P}_1$ FE scheme given in Subsection 7.2.4, we will show this in more detail.

Incidentally, thanks to the fact that the coefficients of the Upwind FD scheme form a partition of unit, it satisfies a DMP of the same kind as the one that holds for the Three-point FDM applied to (P_1). Owing to this similarity, we address this issue rather briefly here, and recommend the reader to refresh her or his understanding of the DMP by reporting to Section 2.2.1 if necessary.

Assume that we are solving the problem $-\epsilon u'' + wu' = f$ in $(0, L)$ with $u(0) = a$ and $u(L) = b$ with the Upwind scheme (7.20) by modifying the homogeneous boundary conditions into $u_0 = a$ and $u_n = b$. Let u_M be the maximum of all the u_i's for $i = 0, 1, \ldots, n$, and assume that $M \neq 0$ and $M \neq n$. Now, if $f_i \leq 0$ for every i, from the property of the scheme's

coefficients, it is easily seen that $u_M \leq \max[u_{M-1}, u_{M+1}]$. Therefore, necessarily $u_{M-1} = u_{M+1} = u_M$ and, like in the case of the Three-point FD scheme, we establish that all the u_i's must be equal. This contradicts the assertion that $M \neq 0$ or $M \neq n$ (it is even absurd if $a \neq b$!). It follows that $\max_{1 \leq i \leq n-1} u_i \leq \max[a, b]$ if $f_i \leq 0 \; \forall i$. This establishes the DMP for equation (7.20).

Again and again, similarly to the case of the Three-point FD scheme, we set $F := \max_{x \in [0,L]} |f(x)|$. Then, the above DMP allows us to derive a

Pointwise stability inequality for scheme 7.20

$$\boxed{\max_{1 \leq i \leq n-1} |u_i| \leq C_{upw} F.} \tag{7.22}$$

The proof of equation (7.22) is left to the reader as Exercise 7.6. As a matter of fact, using the technique of analysis of Subsection 2.2.1, she or he is supposed to find $C_{upw} = L^2/(2\epsilon)$. Strictly speaking, this inequality shows that there cannot be a solution sudden explosion whatever h, but everyone will certainly agree that this bound is not satisfactory if ϵ is very small.

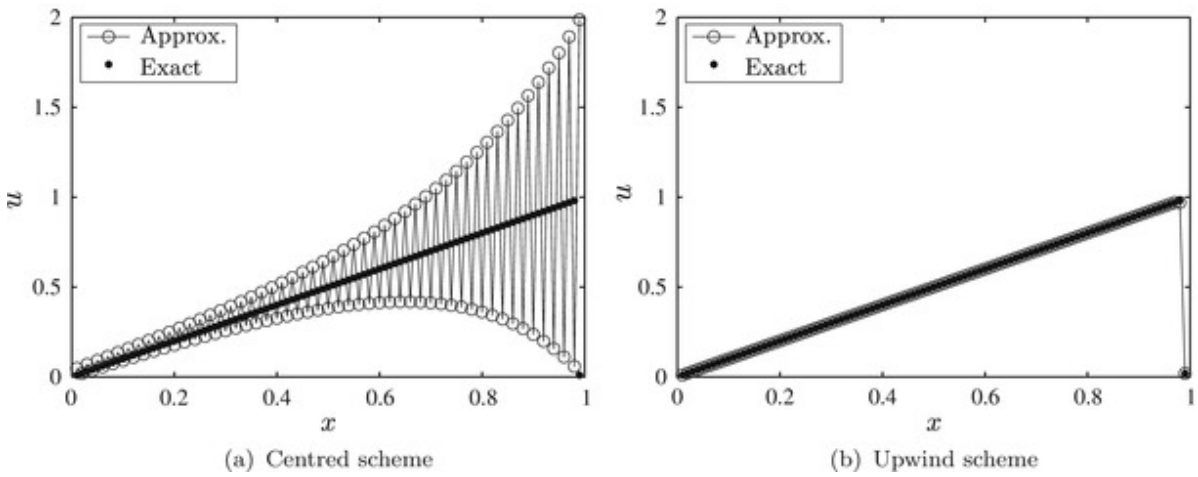
Remark 7.5

A first-order convergence result in the pointwise sense can be established on the basis of equation (7.22). It suffices to derive the local truncation error, assuming that a solution's third-order derivative is bounded in $[0, L]$. *This proof is left to the reader as Exercise 7.7. However, as pointed out in this chapter, such a result is rather of academic use only, in case* ϵ *happens to be very small.*

One can obtain more realistic stability inequalites for practical purposes using mean square norms instead of the maximum norm. It turns out, however, that the Upwind scheme (7.20) can be viewed as a particular case of a Petrov–Galerkin formulation of equation (7.15) combined with a piecewise linear FE discretization, for which a stability result in this sense will be derived in Subsection 7.2.4.

7.2.3 Example 7.1: Numerical Study of the Upwind FD Scheme

In this example, we illustrate the stabilising effect of upwinding. We attempt to approximate the solution of equation (7.16) for $g \equiv 1$ taking the plus sign and $\epsilon = 10^{-4}$. Both the exact solution and the numerical results are plotted in Figures 7.5a and 7.5b at the grid points using two different FDM, namely, the Centred FD scheme (7.19) and scheme (7.20). We computed with a uniform grid containing 101 grid points. As the reader can observe, the centred scheme is blatantly unstable, while the Upwind scheme reproduces the exact solution almost to machine precision at all grid points. Actually, such a high accuracy holds true, except in the region very close to the abscissa $x = 1$, where the FD solution drops to zero within an interval of length $h = 0.01$, instead of the much smaller $O(\epsilon)$ width for the exact solution. This observation just expresses the fact that a very narrow boundary layer can barely be reproduced by a single-scale numerical method[2]. Notice, however, that the main focus of this example is the stability issue, in which respect Figure 7.5 provides clear illustrations.



(a) Centred scheme (b) Upwind scheme

Figure 7.5 FD solution and exact solution for $\epsilon = 10^{-4}$ and $h = 0.01$

7.2.4 The SUPG Formulation

The SUPG is a technique designed to be used in the FE context. The acronym stands for **streamline upwind Petrov–Galerkin**, which corresponds to a formulation of the advection–diffusion equation aimed at obtaining stable and possibly accurate solutions even in high Péclet number simulations. The creators of this technique are acknowledged as Brooks and Hughes in a paper dating back to 1982 [38].

Originally, the Petrov–Galerkin method was used to obtain approximate solutions of PDEs whose highest order term is of odd order. In the variational formulation of this kind of problem, the space in which the solution is searched for (i.e. the space of **trial functions**) and the space of **test functions** cannot be the same. Clearly enough, this restriction needs not to apply to a corresponding FE approximate problem, but in any case the underlying discrete problem will not be symmetric. For example, this would be the case of a basic transport equation – that is, a pure **advection equation**, namely, $ou' = f$ in $(0, L)$, with the boundary condition $u(0) = 0$, where o is a non-vanishing function. If f is given in $L^2(0, L)$, then we have $u \in H^1(0, L)$ and the natural variational formulation of this equation would be

$$\text{Find } u \in U \text{ such that } \int_0^L ou'v\,dx = \int_0^L fv\,dx \;\forall v \in V,$$

where $U := \{u |\ u \in H^1(0, L), u(0) = 0\}$ and $V = L^2(0, L)$. This is a Petrov–Galerkin formulation. Notice that even in the case of the pure advection equation, we can obtain a symmetric formulation by 'testing' both sides of it with the function ov' instead of v, where v now belongs to U. This gives rise to a symmetrised Petrov–Galerkin formulation, that is,

$$\text{Find } u \in U \text{ such that } \int_0^L o^2u'v'\,dx = \int_0^L ofv'\,dx \;\forall v \in U.$$

It is very easy to check that both formulations are equivalent to the original equation. Another possibility is a weighting of both formulations, leading to the asymmetric formulation

$$\text{Find } u \in U \text{ such that } \int_0^L [ou'v + \omega o^2u'v']dx = \int_0^L f(v + \omega ov')dx \;\forall v \in U.$$

where the weight ω is a strictly positive real number. Its role is to balance the importance of the symmetric and asymmetric Petrov–Galerkin formulations, but we note that still in this case the trial and test function spaces remain the same. This weighted formulation lies on the basis of the SUPG technique. Incidentally, we note that nothing prevents the weight ω from varying with x.

Strictly speaking, the Petrov–Galerkin formulation is not well-suited to the advection–diffusion equation (7.15) since it is a second-order boundary value problem. However, if convection strongly dominates diffusion, to a large extent u behaves like the solution of a pure advection equation, as we saw in the introductory part of this section. That is where the SUPG formulation comes into play. However, in contrast to the model one-dimensional pure advection equation, the SUPG formulation is designed to act directly on the FE approximate problem. This is because the

weighting function ω will decrease with the mesh sizes, as seen below. This means that in some sense in the limit as the mesh size goes to zero, only the original asymmetric formulation will remain. Let us see how this works.

We are given a uniform mesh $\mathcal{T}_h$ with mesh size $h = L/n$ for a given integer $n > 2$, and a $\mathcal{P}_1$ FE discretisation of equation (7.15). The corresponding FE space is

$$V_h := \{v |\ v \in C^0[0, L], v_{|T} \in \mathcal{P}_1(T)\ \forall T \in \mathcal{T}_h, v(0) = v(L) = 0\}.$$

The reader should pay attention to the fact that, here, V_h is a subspace of the space denoted in the same manner in Chapters 1, 2 and 3.

Now, we choose the weight ω to be Ch for a suitable constant $C > 0$. Then, we set the

SUPG formulation of equation 7.18

Find $u_h \in V_h$ such that $\forall v \in V_h$
$$\int_0^L [\epsilon u_h' v' + w u_h' v] dx + Ch \int_0^L w^2 u_h' v' dx = \int_0^L f v\, dx + Ch \int_0^L w f v' dx. \quad (7.23)$$

Using the $\mathcal{P}_1$ shape functions φ_i and writing as usual $u_h = \sum_{j=1}^{n-1} u_j \varphi_j$, and then setting $v = \varphi_i$ for $i = 1, \ldots, n-1$, the reader can check as Exercise 7.8 that in case $w = \pm 1$, equation (7.23) is equivalent to the following:

SUPG $\mathcal{P}_1$ FE scheme to solve equation 7.18

Find u_i satisfying for $i = 1, \ldots, n-1$,
$$(\epsilon + Ch)\frac{2u_i - u_{i-1} - u_{i+1}}{h} + w\frac{u_{i+1} - u_{i-1}}{2} = \int_{(i-1)h}^{ih} f(x)\left[\frac{x - ih + h}{h} + Cw\right] dx + \int_{ih}^{(i+1)h} f(x)\left[\frac{x - ih - h}{h} - Cw\right] dx. \quad (7.24)$$

Now, assuming that f is continuous, we set $f_i = f(ih)$. Then, if we use the trapezoidal rule to compute the integrals on the right side of equation (7.24), we obtain

$$(\epsilon + Ch)\frac{2u_i - u_{i-1} - u_{i+1}}{h} + w\frac{u_{i+1} - u_{i-1}}{2} = hf_i + hCw\frac{f_{i-1} - f_{i+1}}{2}.$$

Then, we note that if $C = 1/2$ and we divide both sides of it by h, except for the second term on the right side, scheme (7.24) reproduces the Upwind FD scheme (7.20). Notice that, as long as f is continuously differentiable, the factor $(f_{i-1} - f_{i+1})/2$ is an $O(h)$, and therefore, for not so large C, the SUPG term $Cw[f_{i-1} - f_{i+1}]/2$ on the right side represents just a small perturbation of f_i. We conclude that the SUPG technique is a sort of FE counterpart of the Upwind FD scheme (7.20). Of course, in the case of equation (7.23) we go much further, since this scheme can be applied to more general problems by taking different velocities w. Actually, similarly to Chapter 1, if suitable numerical quadrature formulae are employed to approximate integrals involving this function, we can get upwind FD analogs of equation (7.15), from the SUPG $\mathcal{P}_1$ formulation for variable w. We do not further elaborate on this issue for it has been developed in detail in Section 1.4.

Anyhow, owing to this equivalence, the SUPG FE scheme enjoys the same stability properties as its FD counterpart with $w = \pm 1$. We do not insist very much on this property either, as far as the maximum norm is concerned. On the other hand, it is possible to obtain an exploitable stability inequality in the mean-square sense. Indeed, taking $v = u_h$ in equation (7.23), we get

$$\int_0^L (\epsilon + Ch)u_h^{'2}dx + \int_0^L wu_h^{'}u_h dx = \int_0^L fu_h\, dx + Ch\int_0^L fwu_h^{'}dx. \tag{7.25}$$

Now using integration by parts, since $u_h(0) = u_h(L) = 0$ and w is constant, we note that $\int_0^L wu_h^{'}u_h dx = -\int_0^L wu_h^{'}u_h dx = 0$. Hence, using the Cauchy–Schwarz inequality together with the Friedrichs–Poincaré inequality, after some trivial manipulations on the right side of equation (7.25) that we suggest the reader to carry out as Exercise 7.9, we obtain the

Stability inequalities for scheme 7.23 in the mean-square sense

$$\boxed{\begin{array}{l}(\epsilon + Ch) \parallel u_h^{'}\parallel_{0,2} \leq \parallel f \parallel_{0,2}(C_P + hC) \\ (\epsilon + Ch) \parallel u_h\parallel_{0,2} \leq \parallel f \parallel_{0,2}C_P(C_P + hC)\end{array}} \tag{7.26}$$

Notice that we can distinguish two important cases, namely, $\epsilon << Ch$ and $h \leq \epsilon$, in order to make sure that the solution norms remain under control. The stability inequality (7.26) could sound strange, for it is ineffective if $h \leq \epsilon$ and ϵ is small. However, this situation is utopian in practice, while in contrast, computations with $h >> \epsilon/C$ turn out to be quite realistic.

Now, as far as consistency is concerned, it is worth examining a little closer the situation of the SUPG scheme, for after all it is based on a variational approach. We do this for an arbitrary w with $\| w\|_{0,\infty} = 1$, assuming that $u'' \in L^2(\Omega)$ and that the integrals are computed exactly, for simplicity.

Recalling Subsection 2.3.2, we replace u_h by the $\mathcal{P}_1$ interpolate $\tilde{u}_h$ of u defined in Chapter 2, to obtain two residual functions $R_h(u)$ and $S_h(u)$, satisfying $\forall v \in V_h$,

$$\epsilon(\tilde{u}_h'|v')_0 + (w\tilde{u}_h'|v)_0 + Ch(w^2\tilde{u}_h'|v')_0 - (f|v + Chwv')_0 = (R_h(u)|v)_0 + (S_h(u)|v')_0. \quad (7.27)$$

Using the fact that u is the solution of equation (7.15), these residual functions are found to be $R_h(u) = w[\tilde{u}_h - u]'$ and $S_h(u) = \epsilon Chwu'' + (\epsilon + Chw^2)[\tilde{u}_h - u]'$.

Since $\| w\|_{0,\infty}$ is bounded, estimates in terms of an $O(h)$ can be easily derived for both residual functions in the L^2-norm, respectively. This, combined with equation (7.26), yields first-order error estimates for $\| u - u_h\|_{1,2}$. The calculations leading to this result from equation (7.27) on should be carried out in detail as Exercise 7.10, until the final estimate is specified. Notice that, once again, the error estimates to be found are rather academic if ϵ is very small.

In complement to the material presented in this subsection and in the preceding one, we should add a few words about artificial diffusion.

We saw that the Upwind FD scheme adds artificial diffusion to the equation. Due to the fact that this method is practically identical to the SUPG FE scheme for $C = 1/2$, at least in this case the latter adds the same amount of artificial diffusion. Then some questions immediately arise: What happens for different values of C? Is it possible to reduce artificial diffusion to an optimum without penalising stability? The answer to the first question can be given as follows. A value of C in the interval $[0, 1/2)$ corresponds to a weighted (convex) combination of the Upwind FD scheme and the Centred FD scheme, in the sense that the more we approach zero, the less artificial diffusion is added, but at the price of downgrading stability. The answer to the second question is yes, but it requires a more careful analysis. Actually, optimality is attained if the constant C in the added Petrov–Galerkin terms is replaced by a certain function $C_{opt}(h)$ (see e. g. [78]). More specifically, C_{opt} is the

Optimal coefficient of the SUPG added terms

$$\boxed{C_{opt} = \frac{1}{2}\left[coth(P_h) - \frac{1}{P_h}\right] \quad \text{with } P_h = \frac{h}{2\epsilon}.} \tag{7.28}$$

We remind the reader that the function $coth(x)$ is the hyperbolic cotangent given by $\frac{e^{2x}+1}{e^{2x}-1}$. The coefficient P_h in turn is called the **element Péclet number**.

As shown in reference [78], if C_{opt} replaces C in equation (7.23), then the nodal values supposed to approximate the test solution of Figure 7.4 are exact. This is the usual argument for concluding the optimality of this choice.

Following similar principles, we finally note that the best results for the FDM are obtained with a suitable convex combination of the centred and the upwind approximations of the term wu'. This means tuning added artificial diffusion to the optimal level.

Remark 7.6

*The SUPG formulation is of the **Galerkin least-squares** type, in the sense that it is a linear combination of the standard Galerkin formulation and the **least-squares formulation** of the advection–diffusion equation. The acronym **GLS** is commonly used when referring to it. A pure least-squares formulation of the latter is a symmetric formulation, which would write as follows for a suitable space V_h:*

$$\sum_{i=1}^{n}\int_{x_{i-1}}^{x_i}[\epsilon u_h'' + w u_h'][\epsilon v'' + w v']dx = \sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} f[\epsilon v'' + w v']dx \;\forall v \in V_h.$$

Notice that in the above formulation, the equation is tested with an expression of the same type in terms of the test functions v. For more explanations about least-squares FEM of PDEs, the author refers the reader to reference [30].

7.2.5 Example 7.2: Numerics of the SUPG Formulation for the $\mathcal{P}_1$ FEM

The aim of this example is to show the performance of the SUPG formulation for different values of the constant C. Two test problems were solved with both a uniform and a non-uniform mesh

of $(0, 1)$ consisting of 1024 intervals. For the uniform mesh, $h \simeq 0.00098$ and $C_{opt} = 0.489760$. The elements in the non-uniform mesh are sequentially numbered from the leftmost to the rightmost. Their lengths decrease from left to right by groups of 256 or 128 intervals with equal lengths. The groups together with the corresponding interval lengths are specified in Table 7.2. In the same table, we supply the value of C_{opt} for each interval length.

Table 7.2 Values of C_{opt} for different sizes of the variable FE mesh

Element number							
Mesh size	0.00213	0.00142	0.00071	0.00057	0.00043	0.00028	0.00014
	0.495307	0.492960	0.485920	0.482400	0.476533	0.464800	0.429601

Test case 1: We first tested the SUPG method in the solution of equation (7.15) under the same conditions as in Example 7.1. We recall that in this case the exact solution is given by equation (7.17), and that the SUPG method reproduces the exact solution at the nodes, in case the constant C equals C_{opt} for all the intervals. In Figure 7.6a, we display the error measured in both the L^2-norm and the discrete L^∞-norm in terms of the constant C taken in the computations for all the intervals. As one can see, the L^2 errors are very small for every C. As for the L^∞-errors, a minimum is attained for $C = C_{opt}$ in the case of the uniform mesh, while in the case of the non-uniform mesh the best results are obtained with C close to the value of C_{opt} corresponding to the smallest mesh size.

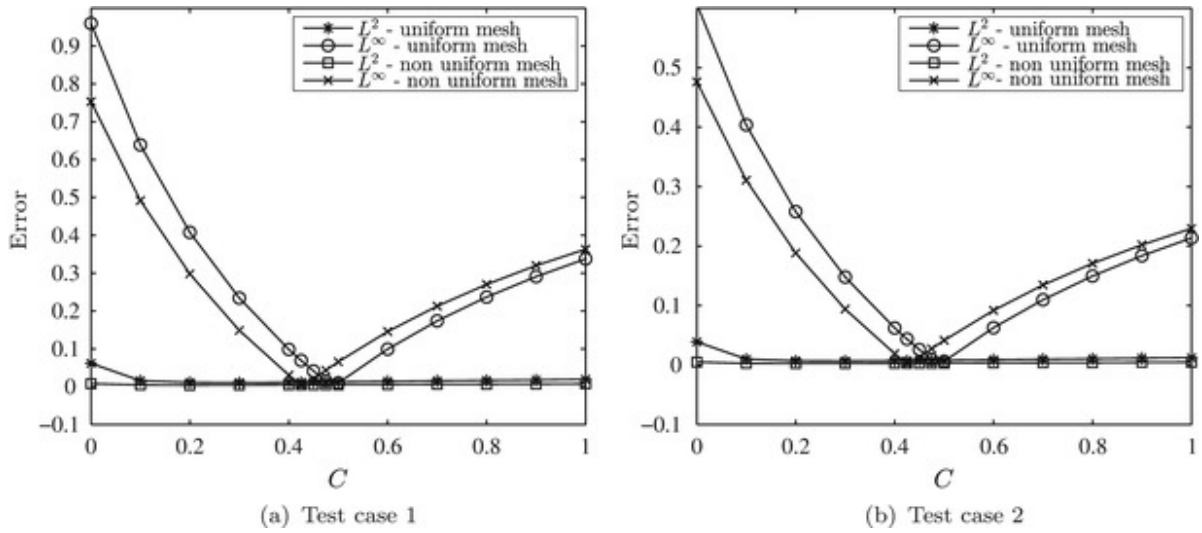


Figure 7.6 Absolute errors in L^2 and L^∞ for $\epsilon = 10^{-5}$ and $n = 1024$ in terms of C

Test case 2: We also tested the SUPG method in the solution of equation (7.15) for $f(x) = -e^{-x}$ and $w \equiv 1 - \epsilon$. In this case, the exact solution is given by

$$u(x) = e^{-x} + \frac{e^{-\frac{1}{\epsilon}} - 1 + (e-1)e^{\frac{(1-\epsilon)x-1}{\epsilon}}}{1 - e^{1-\frac{1}{\epsilon}}}.$$

Here again, we computed with constant values of C for all mesh intervals. Figure 7.6b reflects the very same behaviour as in test case 1, even though the numerical values are no longer exact at the mesh nodes.

Summarizing, we emphasised in this example the importance of doing simulations of advection–diffusion phenomena using the SUPG technique with the formulation's constant C close to C_{opt} everywhere in the domain.

7.2.6 An Upwind FV Scheme

Following the ideas presented in Subsection 7.2.2, we next study a (Cell-centred) FVM to solve equation (7.18). In this aim, we use a non-uniform mesh and pertaining notations for this kind of method, introduced in Subsection 1.4.2. For the sake of simplicity, instead of using the values w^+ and w^-, we assume that $w = 1$. Of course, the case $w = -1$ can be treated in a similar manner, by taking corresponding approximations of u' in the opposite upwind direction. In this way, we study in this subsection as a simplified model, an Upwind scheme very close to the one considered in reference [68], but not quite the same.

Setting $u_{-1/2} = 0$, $u_{n+1/2} = 0$ and $f_{j-1/2} := \int_{x_{j-1}}^{x_j} f(x)\,dx/h_j$, we set the

Upwind FV scheme for equation 7.18

Find $u_{j-1/2}$ such that for $j = 1, 2, \ldots, n$,

$$\epsilon \left[\frac{u_{j-1/2} - u_{j-3/2}}{h_{j-1/2}} - \frac{u_{j+1/2} - u_{j-1/2}}{h_{j+1/2}} \right] + u_{j-1/2} - u_{j-3/2} = h_j f_{j-1/2} \quad (7.29)$$

As everyone now certainly knows, the first step in the reliability analysis of a numerical scheme is to establish a stability inequality for it. Like in the case of Cell-centred FV schemes proposed in the preceding chapters, the above FV scheme is not consistent in the usual FD sense. More

precisely, except for special meshes, the local truncation errors are an $O(1)$ in the case of equation (7.29), as the reader can check (cf. Exercise 7.11). Hence, we shall resort to the concept of stability in the mean-square sense, by applying the scheme to a problem more general than equation (7.29), namely,

$$\begin{cases} \text{Find } u_{j-1/2} \text{ such that, for } j = 1, 2, \ldots, n, \\ \epsilon \left[\dfrac{u_{j-1/2} - u_{j-3/2}}{h_{j-1/2}} - \dfrac{u_{j+1/2} - u_{j-1/2}}{h_{j+1/2}} \right] + u_{j-1/2} - u_{j-3/2} \\ = h_j f_{j-1/2} + \mathcal{F}^+_{j-1/2} + \mathcal{F}^-_{j-1/2} \end{cases} \tag{7.30}$$

where $\mathcal{F}^+_{j-1/2}$ and $\mathcal{F}^-_{j-1/2}$ stand for given incoming and outgoing fluxes with respect to CV $T_j = (x_{j-1}, x_j)$. These fluxes are required to be bounded independently of the mesh and conservative in the sense that $\mathcal{F}^+_{j-1/2} = -\mathcal{F}^-_{j+1/2}$ for $j = 1, 2, \ldots, n-1$, and are completed by $\mathcal{F}^+_{-1/2} := -\mathcal{F}^-_{1/2}$ and $\mathcal{F}^-_{n+1/2} := -\mathcal{F}^+_{n-1/2}$.

Our goal is to derive a stability inequality in the mean-square sense. We should stress beforehand that this is not only because the discretisation is not uniform. The reason is inherent to FV schemes, for in general they are not consistent in the FD sense.

Here again, we resort to a functional framework close to the one of Part B of Subsection 5.1.5. In this aim, we need some definitions and preliminary results.

First of all, we define a piecewise constant function u_h by

$$u_h = \sum_{j=1}^{n} u_{j-1/2} \chi_{T_j},$$

where χ_{T_j} is the **characteristic function** of T_j. Next, we define a discrete derivative u'^h of u_h, namely, a constant function in every interval $(x_{j-3/2}, x_{j-1/2}]$ with value equal to $(u_{j-1/2} - u_{j-3/2})/h_{j-1/2}$, for $j = 1, 2, \ldots, n+1$. We recall that the L^2-norm of a function v_h whose value is v_j in every element of a partition of $(0, L)$ into m disjoint intervals I_j with length k_j for $j = 1, \ldots, m$ is given by $\|v_h\|_{0,2} = \left[\sum_{j=1}^{m} k_j v_j^2 \right]^{1/2}$. Taking down this expression, the reader is invited to establish as Exercise 7.12, the validity of the following discrete Friedrichs–Poincaré inequality:

$$\|u_h\|_{0,2} \leq L \, \|u_h'\|_{0,2}. \tag{7.31}$$

Now, let us multiply both sides of equation (7.30) with $u_{j-1/2}$ and add up the resulting relations from $j = 1$ through $j = n$. Let $\mathcal{F}_j$ be the outgoing flux $\mathcal{F}^+_{j-1/2}$ for $j = 1, \ldots, n-1$, $\mathcal{F}_0 := \mathcal{F}^-_{1/2}$ and $\mathcal{F}_n := \mathcal{F}^+_{n-1/2}$. Rearranging the summations, similarly to Part B of Subsection 5.1.5, after some straightforward manipulations to be checked by the reader, we obtain

$$\begin{cases} \displaystyle\sum_{j=2}^{n} \left[\epsilon \frac{(u_{j-1/2} - u_{j-3/2})^2}{h_{j-1/2}} + (u^2_{j-1/2} - u_{j-1/2}u_{j-3/2}) \right] + \frac{u^2_{1/2}(\epsilon + h_{1/2})}{h_{1/2}} + \\ \displaystyle\frac{\epsilon u^2_{n-1/2}}{h_{n+1/2}} = \sum_{j=1}^{n} h_j f_{j-1/2} u_{j-1/2} - \sum_{j=1}^{n-1} \mathcal{F}_j (u_{j+1/2} - u_{j-1/2}) + \mathcal{F}_0 u_{1/2} + \mathcal{F}_n u_{n-1/2}. \end{cases} \tag{7.32}$$

Next, we apply the same trick as in Part B of Subsection 5.1.5. to the flux terms on the right side of equation (7.32). More precisely, we apply Young's inequality to the term $-u_{j-3/2}u_{j-1/2}$ on the left side with $\delta = 1$. After straightforward calculations,

$$\begin{cases} \displaystyle \epsilon \sum_{j=1}^{n+1} \frac{(u_{j-1/2} - u_{j-3/2})^2}{h_{j-1/2}} + \frac{u^2_{n-1/2} + u^2_{1/2}}{2} \\ \displaystyle \leq \sum_{j=1}^{n} h_j f_{j-1/2} u_{j-1/2} + \mathcal{F}_{\max} \sum_{j=0}^{n} \sqrt{h_{j+1/2}} \frac{|u_{j+1/2} - u_{j-1/2}|}{\sqrt{h_{j+1/2}}}. \end{cases} \tag{7.33}$$

where $\mathcal{F}_{\max} := \max\limits_{0 \leq j \leq n} |\mathcal{F}_j|$. Notice that $\sum\limits_{j=0}^{n} h_{j+1/2} = L$. Thus, setting $f_h := \sum\limits_{j=1}^{n} f_{j-1/2} \chi_{T_j}$ and using the definition of u_h together with the Cauchy–Schwarz inequality on the right side of equation (7.33), we further obtain

$$\begin{cases} \displaystyle \epsilon \sum_{j=1}^{n+1} h_{j-1/2} \frac{(u_{j-1/2} - u_{j-3/2})^2}{h^2_{j-1/2}} \leq \|f_h\|_{0,2} \, \|u_h\|_{0,2} \\ \displaystyle + \sqrt{L} \mathcal{F}_{\max} \left[\sum_{j=1}^{n+1} h_{j-1/2} \frac{(u_{j-1/2} - u_{j-3/2})^2}{h^2_{j-1/2}} \right]^{1/2}. \end{cases} \tag{7.34}$$

Finally, recalling the expression of u_h' and using equation (7.31), equation (7.34) yields

$$\epsilon \, \|u_h'\|^2_{0,2} \leq \|f_h\|_{0,2} \, \|u_h\|_{0,2} + \sqrt{L} \mathcal{F}_{\max} \, \|u_h'\|_{0,2}. \tag{7.35}$$

and therefore two

Mean-square stability inequalities for the Upwind FV scheme

$$\boxed{\begin{array}{l}\|u_h^{'h}\|_{0,2} \le [L\ \|f_h\|_{0,2} + \sqrt{L}\mathcal{F}_{\max}]/\epsilon, \\ \|u_h\|_{0,2} \le [L^2\ \|f_h\|_{0,2} + \sqrt{L^3}\mathcal{F}_{\max}]/\epsilon.\end{array}} \quad (7.36)$$

Next, we examine the consistency of scheme (7.29). Similarly to Part B of Subsection 5.1.5, all we have to do is to establish the consistency of the fluxes across the boundaries of the CVs, when $u_{j-1/2}$ is replaced in the scheme by $u(x_{j-1/2})$ for pertaining values of j. For CV T_j, $j = 1, 2, \ldots, n$, the exact and resulting approximate fluxes are, respectively,

- Incoming flux: $\tilde{F}^-_{j-1/2} = \epsilon u'(x_{j-1}) - u(x_{j-1});\ F^-_{j-1/2} = \epsilon\frac{u(x_{j-1/2})-u(x_{j-3/2})}{h_{j-1/2}} - u(x_{j-3/2}),$
- Outgoing flux:

$$\tilde{F}^+_{j-1/2} = -\epsilon u'(x_j) + u(x_j);\quad F^+_{j-1/2} = \epsilon\frac{u(x_{j-1/2})-u(x_{j+1/2})}{h_{j+1/2}} + u(x_{j-1/2}),$$

completed by $\tilde{F}^+_{-1/2} = -\tilde{F}^-_{1/2};\ F^+_{-1/2} = -F^-_{1/2}$, and $\tilde{F}^-_{n+1/2} = -\tilde{F}^+_{n-1/2};\ F^-_{n+1/2} = -F^+_{n-1/2}$. It is clear that the property $F^+_{j-1/2} = -F^-_{j+1/2}$ (resp. $\tilde{F}^+_{j-1/2} = -\tilde{F}^-_{j+1/2}$) holds for $j = 1, \ldots, n-1$.

Assuming that u is twice continuously differentiable, let us evaluate the numerical flux residuals $R^+_{j-1/2}(u) := \tilde{F}^+_{j-1/2} - F^+_{j-1/2} = -R^-_{j+1/2}(u)$ at the end-points x_j for $n > j > 0$, that is, from the right and from the left of x_j, respectively. By elementary Taylor expansions, we have

$$\begin{array}{l} u(x_{j-1/2}) = u(x_j) - h_j u'(x_j)/2 + h_j^2 u''(\xi_j^-)/8 \\ u(x_{j+1/2}) = u(x_j) + h_{j+1} u'(x_j)/2 + h_{j+1}^2 u''(\xi_j^+)/8, \end{array}$$

where $x_{j-1/2} \le \xi_j^- \le x_j$ and $x_j \le \xi_j^+ \le x_{j+1/2}$. Recalling that $h_{j+1/2} = (h_j + h_{j+1})/2$, after simple manipulations, it follows that

$$R^+_{j-1/2}(u) = \epsilon[h_{j+1}^2 u''(\xi_j^+) - h_j^2 u''(\xi_j^-)]/(8h_{j+1/2}) + h_j u'(x_j)/2 - h_j^2 u''(\xi_j^-)/8. \quad (7.37)$$

These relations are completed with the residuals at $x = 0$ and $x = L$, namely, $R^+_{-1/2}(u) = \epsilon[-u'(0) + u(x_{1/2})/h_{1/2}]$ and $R^+_{n-1/2}(u) = \epsilon[-u'(L) - u(x_{n-1/2})/h_{n+1/2}] - u(x_{n-1/2})$, which yield for $0 \le \xi_0^+ \le h_{1/2}$ and $L - h_{n-1/2} \le \xi_n^- \le L$,

$$\begin{cases} R^+_{-1/2}(u) = \epsilon h_{1/2} u''(\xi_0^+)/2, \text{ and} \\ R^+_{n-1/2}(u) = -\epsilon h_{n+1/2} u''(\xi_n^-)/2 + h_{n+1/2} u'(L) - h^2_{n+1/2} u''(\xi_n^-)/2. \end{cases} \tag{7.38}$$

Now we introduce the function $\bar{u}_h := \sum_{j=1}^{n} \bar{u}_{j-1/2} \chi_{T_j}$, with $\bar{u}_{j-1/2} := u(x_{j-1/2}) - u_{j-1/2}$. By straightforward calculations, $\bar{u}_h$ is seen to solve equation (7.30) when we set $f_{j-1/2} = 0$ for all j, and take $\mathcal{F}^+_{j-1/2} = R^+_{j-1/2}$ and $\mathcal{F}^-_{j-1/2} = R^-_{j-1/2}$. Noticing that $h_{j+1/2} \geq \max[h_j, h_{j+1}]/2$ for $j = 1, \ldots, n-1$, $h_{1/2} = h_1/2$ and $h_{n+1/2} = h_n/2$ and setting $h := \max_{1 \leq j \leq n} h_j$, on the basis of equations (7.37) and (7.38), we can assert that the corresponding value of $\mathcal{F}_{\max}$ is bounded above by $[\epsilon h/2 + h^2/8] \, \|u''\|_{0,\infty} + h \, \|u'\|_{0,\infty}/2$ (please check!). Recalling equation (7.36), the function $\bar{u}_h$ is readily seen to satisfy

$$\begin{cases} \|\bar{u}'_h\|_{0,2} \leq \sqrt{L} \left[\left(\dfrac{h}{2} + \dfrac{h^2}{8\epsilon} \right) \|u''\|_{0,\infty} + \dfrac{h}{2\epsilon} \|u'\|_{0,\infty} \right], \\ \|\bar{u}_h\|_{0,2} \leq \sqrt{L^3} \left[\left(\dfrac{h}{2} + \dfrac{h^2}{8\epsilon} \right) \|u''\|_{0,\infty} + \dfrac{h}{2\epsilon} \|u'\|_{0,\infty} \right]. \end{cases} \tag{7.39}$$

Now, we further define $\tilde{u}_h := \sum_{j=1}^{n} u(x_{j-1/2}) \chi_{T_j}$. Using elementary interpolation theory, the reader should not find it difficult to prove as Exercise 7.13 that

$$\|\tilde{u}_h - u\|_{0,2} \leq \sqrt{L} h \, \|u'\|_{0,\infty}/2. \tag{7.40}$$

Finally, combining equation (7.39) and (7.40), by the triangle inequality we derive the

Mean-square error estimate for the Upwind FV scheme

$$\boxed{\|u_h - u\|_{0,2} \leq \sqrt{L^3} \left[\left(\frac{h}{2} + \frac{h^2}{8\epsilon} \right) \|u''\|_{0,\infty} + \frac{(L+\epsilon)h}{2\epsilon L} \|u'\|_{0,\infty} \right].} \tag{7.41}$$

As a conclusion, the Upwind FV scheme is first-order convergent in the L^2-norm for a non-uniform mesh. Notice that, here again, this is a qualitative result of little quantitative use, in case ϵ is very small.

7.2.7 A FE Scheme for the Time-dependent Problem

In the last part of this section, we deal with a FE scheme to solve the time-dependent advection–diffusion equation that can be easily applied to the case of multiple space variables. Like in the previous subsections, however, we will focus on the one-dimensional case for the sake of

conciseness. Before starting, let us make a brief review of different strategies to treat advection in a variational framework.

One of the first FE techniques employed to model advection is the Lesaint–Raviart method (cf. [124]). As for convection–diffusion, the earliest known contribution is due to Heinrich et al. [97]. A little later, the Japanese school gave relevant contributions to the subject, as it is well reported in [103]. In this respect, the pioneering work of Tabata (cf. [187]) together with [115] among others should be quoted. Since the mid-1980s a widespread manner to deal with dominant advection has been the use of stabilizing procedures based on the space mesh parameter, among which the SUPG technique described in Subsection 7.2.3 (see also [38]) appears to be the most popular. Among many others, an interesting approach based on this technique was proposed in reference [82]. An interesting reference on the FEM to deal with advection-diffusion problems and related topics is the book of Knabner and Angermann [116].

As far as time-dependent problems are concerned, it turns out that the time step plays a better stabilizing role, provided a formulation well suited to the equations to be solved is employed. A good illustration of this assertion in the case of the time-dependent Navier–Stokes equations can be found in reference [50]. The author himself and co-workers gave a contribution in this direction, in the case of the advection–diffusion equations with dominant advection, discretised in space with piecewise linear FEs using a classical Galerkin approach, combined with a non-standard Forward Euler scheme for the time integration. Actually, this subsection is devoted to this method, which follows similar principles to the one long exploited by Kawahara et al. for simulating convection dominated phenomena (cf. [114], among several other earlier or later papers by this author and co-workers). Actually, an introduction to this technique was supplied in Section 3.2.4 restricted to the one-dimensional transport equation, which is a pure advection equation.

An outline of this subsection is as follows. We first specify the problem to solve and make some assumptions on the data. Next, we describe the type of discretisation corresponding to the FE scheme, and especially the weighted manner to deal with the **mass matrices**[3] on both sides of the discrete equations. In the sequel, we give stability results for this scheme in the sense of the space and time maximum norm, and quote the error estimates that hold for this scheme, derived in reference [172]. Finally, we illustrate the scheme's performance by means of some numerical tests in one space dimension.

The time-dependent advection–diffusion equation consists of finding a scalar valued function $u(\boldsymbol{x},t)$ defined in $\bar{\Omega}\times[0,\Theta]$, Ω being a bounded open subset of $\Re^N$ with boundary Γ, for $N=1,2$ or 3, such that

$$\begin{cases}\partial_t u+(\vec{v}\,|\,\textbf{grad}\ u)-p\Delta u=f\ \text{ in }\Omega\times(0,\Theta]\\ u=g\ \text{on}\ \Gamma\times(0,\Theta]\\ u=u_0\ \text{in}\ \Omega\ \text{for}\ t=0.\end{cases}\tag{7.42}$$

where p is a (positive) diffusivity constant and $\vec{v}$ is a given (advective) convective velocity at every time t, assumed to be uniformly bounded in $\bar{\Omega}\times[0,\Theta]$. The data f and g are, respectively, a given forcing function belonging to $L^\infty[\Omega\times(0,\Theta)]$, and a prescribed boundary value $\forall t$.

Henceforth, we confine ourselves to the case where $\Omega=(0,L)$, in which g reduces to a pair of functions $[a(t);b(t)]$ such that $u(0,t)=a(t)$ and $u(L,t)=b(t)\ \forall t$. We further assume that $u_0\in C[0,L]$ and that $a(t)$ and $b(t)$ are bounded in $[0,\Theta]$. Moreover, we consider only normalised dimensionless lengths, time and velocity $w=v/\mathrm{V}$ where $\mathrm{V}:=\max\limits_{x\in[0,L];\ t\in[0,\Theta]}|v(x,t)|$. In doing so, p is replaced by ϵ in equation (7.42) and v is replaced by w. Then like before, the inverse of ϵ is the Péclet number $P:=\mathrm{V}L/p$. Finally, without loss of essential aspects, we assume that $L=1$.

Taking all these simplifications into account, equation (7.42) reduces to

$$\begin{cases}\partial_t u+w\partial_x u-\epsilon\partial_{xx}u=f\ \text{ in }(0,1)\times(0,\Theta]\\ u(0,t)=a(t)\ \text{and}\ u(1,t)=b(t)\ \forall t\in(0,\Theta]\\ u(x,0)=u_0(x)\ \forall x\in(0,1).\end{cases}\tag{7.43}$$

We will work with the equivalent standard Galerkin formulation of equation (7.43), namely,

$$\begin{cases}\text{For every}\ t\in(0,\Theta],\ \text{find}\ u(\cdot,t)\in H^1(0,1)\ \text{with}\ \partial_t u(\cdot,t)\in L^2(0,1),\\ u(0,t)=a(t),u(1,t)=b(t)\ \forall t\in(0,\Theta]\ \text{and}\ u(x,0)=u_0(x)\ \forall x\in(0,1)\\ \text{such that,}\ \int_0^1[\partial_t u+w\partial_x u]v\,dx+\epsilon\int_0^1\partial_x u v'dx=\int_0^1 fv\,dx\ \forall v\in H_0^1(0,1).\end{cases}$$

Next we consider a partition $\mathcal{T}_h$ of $(0,1)$ into intervals T_j with maximum edge length equal to h. We also need a second mesh parameter ρ, namely, the minimum length of all the elements in $\mathcal{T}_h$. We assume that $\mathcal{T}_h$ belongs to a quasi-uniform family of meshes, which means that there exists a constant $\tilde{c}>0$ independent of h such that $h/\rho\le\tilde{c}$ for all meshes in such a family.

Recalling the notation $\mathcal{P}_1(T)$ introduced in Section 1.3, for $T\in\mathcal{T}_h$, we also revisit the following spaces associated with $\mathcal{T}_h$:

$$W_h := \{v \mid \ v \in C^0[0,1] \ \text{ and } \ v|_T \in \mathcal{P}_1(T), \forall T \in \mathcal{T}_h\},$$
$$V_h := W_h \cap H_0^1(0,1).$$

We further introduce for any pair $g(t) := (a(t); b(t))$ of real functions a and b bounded in $[0, \Theta]$, the following manifold of W_h:

$$V_h^g(t) := \{v \in V_h \mid v(0) = a(t), v(1) = b(t)\}.$$

Now, let u_h^0 be the field of $V_h^g(0)$ that interpolates u_0 at the mesh nodes, and $\tau > 0$ be a time step given by $\tau = \Theta/l$ where l is a non-negative integer. Defining $g^k := [a(k\tau); b(k\tau)]$, together with f^k by $f^k(\cdot) = f(\cdot, k\tau)$ and $w^k(\cdot) := w(\cdot, k\tau)$ in $(0,1)$, for $k = 1, 2, \ldots, l$, ideally we wish to determine approximations $u_h^k(\cdot)$ of $u(\cdot, k\tau)$ for $k = 1, 2, \cdots$, by solving the following FE discrete set of equations, corresponding to the Forward Euler scheme:

$$\begin{cases} \text{For } k = 1, 2, \ldots, l, \text{ find } u_h^k \in V_h^g(k\tau) \text{ satisfying } \forall v \in V_h, \\ \displaystyle\int_0^1 u_h^k v \, dx = \int_0^1 u_h^{k-1} v \, dx \\ \displaystyle +\tau \left[\int_0^1 f^{k-1} v \, dx - \int_0^1 w^k [u_h^{k-1}]' v \, dx - \epsilon \int_0^1 [u_h^{k-1}]' v' \, dx \right]. \end{cases} \tag{7.44}$$

Now, we expand u_h^k into a sum of the form, $u_h^k = \sum_{j=0}^{n} u_j^k \varphi_j$, where φ_j is the shape function of V_h associated with the jth node of $\mathcal{T}_h$, say S_j, $u_j^k \in \Re$ is the value of u_h^k at S_j, the dimension of V_h being $n-1$. We assume that the nodes S_j are numbered in such a manner that the corresponding abscissae are x_j satisfying $0 = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = 1$. Now, we choose v successively equal to φ_i, for $i = 1, 2, \ldots, n-1$, and we approximate for every k, $\int_0^1 w^k [\varphi_j]' \varphi_i dx$ by $\int_0^1 w_i^k [\varphi_j]' \varphi_i dx$ and $\int_0^1 f^k \varphi_i dx$ by $\int_0^1 f_i^k \varphi_i dx$, where $w_i^k := w^k(S_i)$, $f_i^k := f^k(S_i)$.

Still denoting the resulting values of $u_h^k(S_j)$ by u_j^k for $j = 1, \ldots, n-1$ and setting $u_0^k = a(k\tau)$ and $u_n^k = b(k\tau)$, the unknown coefficients u_j^k for $j = 1, 2, \ldots, n-1$ and $k = 1, 2, \ldots, l$, are recursively determined by solving the following SLAE:

$$\sum_{j=1}^{n-1} m_{i,j}^C u_j^k = \sum_{j=0}^{n} [m_{i,j}^C - \tau a_{i,j}^{k-1}] u_j^{k-1} + \tau b_i^k, \text{ for } i = 1, \ldots, n-1, \tag{7.45}$$

where the coefficients $m_{i,j}^C$, $a_{i,j}^k$ and b_i^k are given by

$$m_{i,j}^C = \int_0^1 \varphi_j \varphi_i dx; \quad a_{i,j}^k = \int_0^1 [w_i^k \varphi_j' \varphi_i \, dx + \epsilon \varphi_j' \varphi_i'] dx; \quad b_i^k = \int_0^1 f_i^{k-1} \varphi_i dx. \tag{7.46}$$

Actually, since for every $1 \leq i \leq n-1$, φ_i vanishes at $x = 0$ and $x = 1$ and w_i^k is constant, by integration by parts we easily derive $\int_0^1 w_i^k \cdot [\varphi_i]' \varphi_i = 0$. Hence, we may rewrite a_{ij}^k in equation (7.46) as

$$a_{i,j}^k = \int_0^1 [(1 - \delta_{ij}) w_i^k \varphi_j' \varphi_i + \epsilon \varphi_j' \varphi_i'] dx. \tag{7.47}$$

The pointwise stability of scheme (7.44) is not ensured in general. Therefore, stabilizing techniques have been introduced such as upwinding (cf. [187] and [10]), in which the integral corresponding to the advection term is computed only in the element(s) situated upwind to the node S_i, with respect to w_i^k. The strategy adopted here is based on the use of different quadrature formulae, according to the side of equation (7.44), to approximate the **consistent mass matrix**, that is, the matrix whose coefficients are the $m_{i,j}^C$'s, where for convenience we let j vary from zero through n, while $i = 1, 2, \ldots, n-1$. In the particular choice made in this work, on the left side of equation (7.45), the integral in the expression of $m_{i,j}^C$ is approximated by the trapezoidal rule, and on the right side the approximate value of $m_{i,j}^C$ denoted by $m_{i,j}^W$ is obtained by an asymmetric quadrature formula (at least for non-uniform meshes) specified below. We recall that the trapezoidal rule consists of approximating the integral of a continuous function ψ in $(0, 1)$ by

$$\mathcal{J}_h(\psi) = \sum_{T \in \mathcal{T}_h} \frac{length(T)}{2} \sum_{m=1}^{2} \psi(S_m^T),$$

where the S_m^T's denote the end-points of T, with $m = 1, 2$.

We recall that the **support** of φ_i is the closure of the set in which this shape function does not vanish identically (i.e. the interval $J_i := [x_{i-1}, x_{i+1}]$ for $i = 1, \ldots, n-1$). Then, setting $\Pi_i := x_{i+1} - x_{i-1}$, the approximation of the consistent mass matrix on the left side of equation (7.45) is nothing but the well-known **lumped mass matrix**, $\{m_{i,j}^L\}$, whose coefficients are $m_{i,j}^L = \Pi_i \delta_{ij}/2$, $0 \leq j \leq n$ and $1 \leq i \leq n-1$.

In doing so and scaling the linear equations by the factor $\Pi_i/2$, the unknown nodal values of u_h^k still denoted by u_j^k are determined by

The weighted mass explicit scheme to solve equation 7.43

$$\boxed{\begin{array}{l}\text{Compute recursively for } k = 1, 2, \ldots, l, \\ u_i^k = \sum_{j=0}^{n} [\tilde{m}_{i,j} - \tau \tilde{a}_{i,j}^{k-1}] u_j^{k-1} + \tau \tilde{b}_i^k, \text{ for } i = 1, \ldots, n-1, \\ \text{where } \tilde{m}_{i,j} = 2m_{i,j}^W / \Pi_i, \tilde{a}_{i,j}^k = 2a_{i,j}^k / \Pi_i \text{ and } \tilde{b}_i^k = f_i^{k-1}\end{array}} \tag{7.48}$$

as one easily concludes for $\tilde{b}_i^k$. The coefficients $m_{i,j}^W$ on the right side are determined as follows: An inner node S_i has two neighbouring nodes S_{i-1} and S_{i+1}. Let $\Pi_{i\pm1}^i$ be the measure fractions associated with $S_{i\pm1}$ given by

$$\Pi_{i-1}^i = \frac{h_i}{2} \text{ and } \Pi_{i+1}^i = \frac{h_{i+1}}{2} \tag{7.49}$$

where $h_i = x_i - x_{i-1}$, $i = 1, 2, \ldots, n$. Notice that

$$\Pi_{i-1}^i + \Pi_{i+1}^i = \frac{\Pi_i}{2}. \tag{7.50}$$

Let $\omega_{i\pm1}^i$ be strictly positive weights satisfying

$$\omega_{i-1}^i \Pi_{i-1}^i + \omega_{i+1}^i \Pi_{i+1}^i = \frac{\Pi_i}{6} \tag{7.51}$$

Now, we define the $(n-1) \times (n+1)$ **weighted mass matrix** $M^W := \{m_{i,j}^W\}$ by

The weighted mass matrix M^W

$$\boxed{m_{i,i\pm1}^W = \omega_{i\pm1}^i \Pi_{i\pm1}^i; \quad m_{i,i}^W = m_{i,i}^C = \frac{\Pi_i}{3}; \quad m_{i,j}^W = 0 \text{ for } |i-j| > 1.} \tag{7.52}$$

It is interesting to note that equation (7.52) implies that

$$m_{i,i-1}^W + m_{i,i}^W + m_{i,i+1}^W = m_{i,i}^L, \forall i \in \{1, 2, \ldots, n-1\}. \tag{7.53}$$

We observe as well that the choice of the weights is not unique. For instance, if we take $\omega_{i\pm1}^i = \frac{1}{3}$ for every node S_i, the coefficients $m_{i,j}^W$ will be nothing but the $m_{i,j}^C$'s. However, except for the case of uniform meshes, this is not the right choice if one wishes to have a consistent scheme, as seen hereafter.

Next, we show how τ must be related to the spatial mesh parameter, in such a way that scheme (7.48) is stable in the sense of L^∞. First, we have to define the following quantities:

- $W = \max\limits_{t\in[0,\Theta]} \max\limits_{1\le i\le n-1} |w(x_i,t)|$; (here $W=1$, but we leave W as such for more general problems);
- $\omega = \min\limits_{1\le i\le n-1} \min[\omega_{i-1}^i, \omega_{i+1}^i]$.

It is interesting to note that equations (7.51) and (7.50) imply that $\omega \le \frac{1}{3}$.

Now we claim that if τ fulfil

$$\tau \le \frac{\omega\rho^2}{W\rho+\epsilon}, \tag{7.54}$$

then the $(n-1)\times(n+1)$ matrix $C^k = \{c_{i,j}^k\}$ given by $c_{i,j}^k = \tilde{m}_{i,j} - \tau\tilde{a}_{i,j}^{k-1}$, is a non-negative matrix having a unit row-norm $\|\cdot\|_1$:

$$\begin{cases} c_{i,j}^k \ge 0 \quad \forall i\in\{1,2,\dots,n-1\} \text{ and } \forall j\in\{0,1,\dots,n\} \\ \sum\limits_{j=0}^{n} c_{i,j}^k = 1 \qquad\qquad \forall i\in\{1,2,\dots,n-1\} \end{cases} \tag{7.55}$$

Let us justify this property. First, we treat the coefficients $c_{i,i}^k$, which are given by

$$c_{i,i}^k = \frac{2}{\Pi_i}(m_{i,i}^W - \tau a_{i,i}^{k-1}), \tag{7.56}$$

where $m_{i,i}^W$ is defined by equation (7.52) and $a_{i,i}^k$ is given by $a_{i,i}^k = \int_{x_{i-1}}^{x_{i+1}} \epsilon|\varphi_i'|^2 dx \ \forall k$. Then, straightforward calculations lead to

$$a_{i,i}^{k-1} \le \epsilon\rho^{-2}\Pi_i. \tag{7.57}$$

It follows from equation (7.52) and (7.57) that $c_{i,i}^k \ge 0\, \forall i\in\{1,2,\dots,n-1\}$ if $\tau \le \frac{\rho^2}{3\epsilon}$. Since $\omega \le 1/3$, this condition is satisfied if equation (7.54) holds.

Next, we switch to the coefficients $c_{i,j}^k$ for $i\ne j$. Noticing that $c_{i,j}^k = 0$ if S_j does not belong to J_i, for $j = i\pm 1$, we have

$$c_{i,i\pm1}^k = \frac{2}{\Pi_i}(m_{i,i\pm1}^W - \tau a_{i,i\pm1}^{k-1}), \tag{7.58}$$

where $m_{i,i\pm1}^W$ is defined by equation (7.52) and $a_{i,i\pm1}^k$ is given by

$$a_{i,i\pm1}^k = \int_0^1 [w_i^k \varphi_{i\pm1}^{'} \varphi_i + \epsilon \varphi_{i\pm1}^{'} \varphi_i^{'}] dx. \tag{7.59}$$

From equation (7.52), equation (7.49) and the definition of ω, we trivially have $m_{i,i\pm1}^W \geq \frac{\omega}{2}|x_i - x_{i\pm1}|$. Moreover, we note that $\epsilon \int_0^1 \varphi_{i\pm1}^{'} \varphi_i^{'} dx < 0, \forall i$. Hence, referring to Exercise 3.4, after straightforward calculations, we conclude that for all k, $c_{i,i\pm1}^k \geq 0$ provided $\tau \leq \frac{\omega\rho}{W}$. This condition in turn is fulfilled if equation (7.54) holds.

To complete the argument, we note that, according to well-known properties of the shape functions φ_j, we have $\sum_{j=0}^n \varphi_j = 1$ everywhere in $[0,1]$. Hence, by linearity, we easily derive

$$\sum_{j=0}^n a_{i,j}^k = 0, \forall i \in \{1,2,\ldots,n-1\} \text{ and } \forall k = 1,2,\ldots,l. \tag{7.60}$$

Moreover, from equation (7.53) we derive $\sum_{j=0}^n m_{i,j}^W = m_{i,i}^W + m_{i,i-1}^W + m_{i,i+1}^W = m_{i,i}^L$. Thus,

$$\sum_{j=0}^n m_{i,j}^W = \frac{\Pi_i}{2} \; \forall i \in \{1,2,\ldots,n-1\}. \tag{7.61}$$

Then, using the definitions $\tilde{a}_{i,j}^k = 2\frac{a_{i,j}^k}{\Pi_i}$ and $\tilde{m}_{i,j} = 2\frac{m_{i,j}^W}{\Pi_i}$, together with equations (7.58), (7.60) and (7.61), we readily conclude that for every $i \in \{1,2,\ldots,n-1\}$, $\sum_{j=0}^n c_{i,j}^k = 1$.

As a consequence of the property we just established, the FE solution sequence $\{u_h^k\}_k$ generated by equation (7.48), given by $u_h^k = \sum_{j=0}^n u_j^k \varphi_j$, satisfies the

Maximum norm stability inequality for schemes 7.48–7.52

Under the condition (7.54), we have for $1 \leq k \leq l$,

$$\| u_h^k \|_{0,\infty} \leq \max \left\{ \max_{1\leq m\leq k} \max[|a(m\tau)|, |b(m\tau)|], \| u^0 \|_{0,\infty} \right\} + \tau \sum_{m=1}^k \| f^{m-1} \|_{0,\infty} \tag{7.62}$$

This result is proven in reference [172] in a wider context. The reader can use arguments very close to those in Section 3.2 for the Forward Euler scheme, to establish the validity of equation (7.62) as Exercise 7.14.

It is interesting to observe that condition (7.54) reflects the particular nature of equation (7.43). If ϵ is very small as compared to ρ, the time step τ should be roughly an $O(\rho)$, like in the case of explicit schemes for hyperbolic equations. On the other hand, in the diffusion dominant case, or if ρ is small enough to be close to ϵ, then stability requires a time step proportional to the square of the mesh size. Notice that this is the case of explicit schemes for parabolic equations.

Finally, we specify a condition on the weights $\omega_{i\pm1}^{i}$ under which a variant of schemes (7.48)–(7.52) proposed in reference [172] to solve equation (7.43) is consistent. In this variant, the weighted mass matrix coefficient $m_{i,j}^{W}$ is replaced by a convex combination of it with the coefficient $m_{i,j}^{L}$ of the lumped mass matrix, more precisely by $\beta m_{i,j}^{L} + (1-\beta)m_{i,j}^{W}$ with $\beta = \epsilon/(\epsilon+\rho)$. Consistency is achieved by enforcing $\omega_{i-1}^{i}\Pi_{i-1}^{i} = \omega_{i+1}^{i}\Pi_{i+1}^{i}$, which combined with equation (7.51) leads to a

Consistency condition on the weights

$$\boxed{\omega_{i+1}^{i} = \frac{h_i}{3h_{i+1}};\quad \omega_{i-1}^{i} = \frac{h_{i+1}}{3h_i}} \tag{7.63}$$

Now we make the following assumptions on f, w and u:

- f, w and $\partial_x w$ are bounded in $L^{\infty}[(0,1)\times(0,\Theta)]$;
- $\partial_{xx}u$, $\partial_{xt}u$, $\partial_{xxt}u$ and $\partial_{tt}u$ are bounded in $L^{\infty}[(0,1)\times(0,\Theta)]$.

Under these assumptions together with equation 7.63, we are lead to the following:

Error estimate for the weighted mass scheme in reference [172] satisfying equation 7.63

$$\boxed{\begin{aligned}&\text{If } \tau \le \kappa\tfrac{\rho^2}{3} \text{ with } \kappa = \min\left[\tfrac{1}{\varepsilon}, \tfrac{1}{W(\varepsilon+\rho)}\right], \text{ for } C>0 \text{ independent of } u,h,\tau:\\ &\max_{1\le k\le l} \| u^k - u_h^k \|_{0,\infty} \le Ch \max_{0\le s\le\Theta}\{\| \partial_{xx}u(\cdot,s)\|_{0,\infty} + \| \partial_{xt}u(\cdot,s)\|_{0,\infty}\\ &+h\| \partial_{xxt}u(\cdot,s)\|_{0,\infty} + h\| \partial_{tt}u(\cdot,s)\|_{0,\infty} + \| \partial_x w(\cdot,s)\|_{0,\infty}(\| \partial_x u(\cdot,s)\|_{0,\infty}\\ &+h\| \partial_{xx}u(\cdot,s)\|_{0,\infty}) + \| w(\cdot,s)\|_{0,\infty}\| \partial_{xx}u(\cdot,s)\|_{0,\infty} + \| \partial_x f(\cdot,s)\|_{0,\infty}\}.\end{aligned}} \tag{7.64}$$

As a matter of fact, equation 7.64 is a particular case of a more general result proven in reference [172]. The reader can figure out how complex issues are inherent to the error analysis of scheme (7.48)–(7.52)–(7.63) by solving Exercise 7.15.

7.2.8 Example 7.3: Numerical Study of the Weighted Mass FE Scheme

To close this section, we show some numerical results for a one-dimensional problem with a sharp boundary layer.

Two schemes are experimented and compared: scheme (7.48)–(7.52)–(7.63) and the one resulting from a combination of the lumped mass matrix on the left side and the consistent mass matrix on the right side. This corresponds to a first-order consistent counterpart of the second-order consistent method considered in reference [114] for uniform meshes. Hereafter, we call this scheme the **basic Kawahara scheme**.

More specifically, problem (7.43) was solved with $a(t) = b(t) = 0\ \forall t \in (0, \Theta]$, $u_0(x) = 0\ \forall x \in (0, 1)$ and $w = 1$. Setting $u_\epsilon(x) := x - \dfrac{1 - e^{x/\epsilon}}{1 - e^{1/\epsilon}}$, which is nothing but the function depicted in Figure 7.4, the manufactured exact solution is given by $u(x, t) = (1 - e^{-t})u_\epsilon(x)$. Akin to u_ϵ, u presents a boundary layer of width $O(\epsilon)$ close to the point $x = 1$ for every $t > 0$. The corresponding right-side datum f is given by $f(x, t) := 1 + e^{-t}[u_\epsilon(x) - 1]$. The results supplied below are restricted to a given time, namely, $\Theta = 1$, as they are sufficiently representative of the behaviour of the numerical methods being experimented. This is due to the solution's decaying exponential term.

In order to figure out the influence of the schemes in the numerical results, double precision was used combined with non-uniform spatial meshes with $n + 1$ nodes, n being an even number. The corresponding step sizes are h_i for $i = 1, 2, \ldots, n$ where $h_{2k} = h/R$ and $h_{2k-1} = h$, for $k = 1, 2, \ldots, n/2$, R being a real number greater than one. It follows that $\rho = h/R = 2/[n(1 + R)]$. We determine $\tau = \Theta/l$ for each n, as the largest possible value that satisfies equation (7.54). Notice that the pair of weights for the weighted mass scheme are either $(R; R^{-1})/3$ or $(R^{-1}; R)/3$, according to the parity of the node subscript. As for the basic Kawahara scheme, both weights are equal to $1/3$ for every node. In the tables that follow, the weighted mass scheme is referred to as **WMS** and the basic Kawahara scheme as **BKS**.

First, we take $\epsilon = 10^{-2}$ and $R = 5$. We display in Tables 7.3 and 7.4 the relative errors for the indicated values of n, in $L^\infty(0, 1)$ and in $L^2(0, 1)$, respectively. It is noteworthy observing that the maximum errors are not attained at the leftmost inner mesh node (i.e. the closest inner node to the abscissa $x = 1$, as one might expect), but rather at a point not so far from it. Both points are different for each scheme. Anyway, the error at such points is rather large, since they lie inside

the boundary layer. Next, we compute approximate solutions for $\epsilon = 10^{-5}$ taking again $R = 5$. We display in Table 7.5 the absolute errors of the approximations of $u(0.5, 1)$ computed with both schemes being experimented for increasing n. The corresponding relative errors measured in the norm of $L^2(0,1)$ at time $t = \Theta$ in terms of n are shown in Table 7.6.

Table 7.3 Relative errors in $L^\infty(0,1)$ at time $\Theta = 1$, for $\epsilon = 10^{-2}$ and $R = 5$

	64	128	256	512	1024

Table 7.4 Relative errors in $L^2(0,1)$ at time $\Theta = 1$ for $\epsilon = 10^{-2}$ and $R = 5$

	64	128	256	512	1024

Table 7.5 Absolute errors of approximations of $u(0.5, 1)$ for $\epsilon = 10^{-5}$ and $R = 5$

	256	512	1024	2048	4096

Table 7.6 Relative errors in $L^2(0,1)$ at time $\Theta = 1$ for $\epsilon = 10^{-5}$ and $R = 5$

	256	512	1024	2048	4096

In the case of a moderately dominant convection (i.e. for $\epsilon = 10^{-2}$, we can assert that WMS performs globally better than BKS, as could be expected. Indeed, the errors for the former are significantly smaller than for the latter. Moreover, convergence is observed for both schemes in the L^2- and L^∞-norms, but at better rates for WMS. As for the convection largely dominant case with $\epsilon = 10^{-5}$, it is not possible to detect convergence in the maximum norm as n increases, for either schemes. As a matter of fact, we observed that the maximum errors occur at the grid point next to $x = 1$, and therefore this effect is not surprising at all. After all, it is well-known that the

mesh has to be extremely refined locally, in order to reduce numerical errors in the interior of such a narrow boundary layer, which was not done here. We refer to similar numerical examples given in reference [172] for more details on this issue. Nevertheless, as one moves away from the boundary layer, WMS is found to be very accurate in the pointwise sense; BKS in turn is also accurate but much less than WMS, as long as $R \neq 1$. The results given in Table 7.5 are particularly representative of this behaviour of the schemes being experimented. On the other hand, from Table 7.6, we infer that for this value of ϵ both schemes seem to converge in the sense of L^2, with a slight advantage of BKS over WMS in terms of error magnitudes, at least up to the degree of refinement we have attained.

An interesting conclusion about the experiments reported here is that schemes of the type extensively exploited by Kawahara et al. (see e.g. [114] and references therein) are inexpensive numerical tools and acceptably accurate in all cases, as much as their weighted mass modification [172]. The observation that the smaller the value of ϵ, the better the performance of BKS in the mean square sense is particularly impressive. In spite of this fact, BKS does not seem to be optimally convergent in the strict mathematical sense for non-uniform meshes. WMS in turn is an asserted reliable alternative from this point of view, at an equivalent cost.

7.3 Basics of a Posteriori Error Estimates and Adaptivity

Several error estimates were established for the methods we studied throughout the seven chapters of this book. All these results are **a priori error estimates** in the sense that they predict the behaviour of the numerical solution, in terms of both stability and order of convergence. However, unless empirical techniques are employed such as Richardson's extrapolation like in Example 4.2, these estimates provide only a qualitative answer on what can be expected of a numerical method. Incidentally, we recall that Richardson's extrapolation is a strategy to come very close to a limiting unknown exact solution, on the basis of a bunch of numerical results obtained for the same problem with different discretisation levels (see e.g. [35]). As a matter of fact, in principle, an a priori error bound depends on the exact solution of the differential equation, which, except for very particular cases, is not known. Therefore, since long authors endeavoured to find useful **a posteriori error estimates** for a numerical solution. This means error bounds computed only from problem data and the numerical solution itself. The FE school was very active in this field, and several important works emerged in the late 1970s and the 1980s. Among early contributions in this direction, we could quote references [12], [117], [18],

[20] and [211]. A posteriori error estimation is still a subject of active research for both the FEM and the FVM. They are usually applied in the framework of technical problems far more complex than most of the linear differential models we studied in this book. In addition to this, even for linear problems the rigorous study of these estimations requires some knowledge most readers might not master. From the beginning, it was not our intention to address mathematical problems that fall into this category. However, owing to its great importance in contemporary numerical solution of PDEs, this topic was included in this book's final chapter.

As a complement, we address as briefly an important corollary of a posteriori error estimation, namely, **mesh adaptivity**. As we should clarify, most **a posteriori error estimators** are local in the sense that they evaluate the numerical errors at the level of elements or to the most in their immediate vicinity. In general, the error distribution is not uniform. For instance, the numerical solution may have steep variations in certain regions, thereby representing an expected physical reality. In this case, it seems reasonable to try to improve thus revealed less smooth behaviour of the solution, by increasing the number of mesh nodes in such regions. That is where the concept of mesh adaptivity comes into play: If the number of nodes increase in certain regions, eventually to the detriment of others, in case the total number of nodes is to remain constant, then there must be **remeshing**. As a consequence, if the problem is linear for example, a new system matrix and right-side vector must be computed, and the underlying SLAE solved once again for the newly located unknowns. Notice that this procedure can be repeated several times until acceptable error bounds based on the a posteriori error estimation are obtained.

The principle of mesh adaptivity is explained in rather simple terms in reference [110]. In contrast, as pointed out above, even by simplifying things with all one's might, a rigorous and complete treatment of a posteriori error estimates requires concepts not well suited to this book's level. Therefore, in the remainder of this subsection, we give just a quick overview of both connected subjects.

7.3.1 A Posteriori Error Estimates

Let us consider the solution by the $\mathcal{P}_1$ FEM of the Poisson equation in a polygon Ω, with mixed homogeneous Dirichlet–inhomogeneous Neumann boundary conditions. More precisely, $u = 0$ on Γ_0 and $\partial_\nu u = g$ on Γ_1 where $\Gamma_0 \cap \Gamma_1 = \emptyset$, $\Gamma_0 \cup \Gamma_1 = \Gamma$, $length(\Gamma_0) > 0$. Like before, we assume that Γ_0 contains its transition points with Γ_1.

V being the space defined in Section 4.3, we recall that the variational form of this problem writes $a(u,v) = F(v)\ \forall v \in V$, where

$$a(u,v) := \int_\Omega (\mathbf{grad}\ u | \mathbf{grad}\ v) dxdy \text{ and } F(v) := \int_\Omega fv\ dxdy + \oint_{\Gamma_1} gv\ ds.$$

V_h is the $\mathcal{P}_1$ FE subspace of V associated with a family of triangulations $\mathcal{T}_h$ of Ω satisfying the compatibility conditions specified in Section 4.3. We assume the $\mathcal{T}_h$ belongs to a quasi-uniform family of triangulations of Ω. Let $T \in \mathcal{T}_h$ and h_T be the maximum edge length of T. We recall that $h = \max_{T \in \mathcal{T}_h} h_T$.

The corresponding approximate problem consists of finding $u_h \in V_h$ such that $a(u_h, v) = F(v)\ \forall v \in V_h$. We would like to determine **computable upper and lower bounds** for the error function $u - u_h$ in the norm of V, that is, $\|u - u_h\|_{1,2}$. Following reference [201], there are at least six different techniques to achieve this. Here, we will focus on two of them, namely, the **explicit residual based method** and the **averaging method** (also called the **gradient recovery method**).

The residual based method:

First, we note that $a(u - u_h, v) = F(v) - a(u_h, v)\ \forall v \in V$. The right side is the **residual** in the exact problem when u is replaced with u_h tested with $v \in V$. We denote it by $r(u_h, v)$, that is, r is a bilinear form with the first argument in V_h and second argument in V. We denote the boundary of T by ∂T, and the unit normal vector along any edge e of the triangulation by $\vec{\nu}_e$ taken in a sole direction for elements intersecting at e, except for a boundary edge where it is necessarily oriented outwards the domain. The set of all the edges of the triangles in $\mathcal{T}_h$ are assigned to three disjoint sets, namely, $\mathcal{E}_h$ consisting only of inner edges, $\mathcal{E}_{h1}$ consisting of edges contained in Γ_1 and the complementary set $\mathcal{E}_{h0}$ which will play no role. Let $e \in \mathcal{E}_h$, $h_e := length(e)$ and T and T' be two triangles of $\mathcal{T}_h$ whose intersection is e. Assuming that $\vec{\nu}_e$ is oriented from T onto T', we denote by $[\ |\partial_{\nu_e} u_h|\ ]_e$ the (constant) jump of the normal derivative of u_h across $e \in \mathcal{E}_h$, that is, $([\mathbf{grad}\ u_h]_{|T} - [\mathbf{grad}\ u_h]_{|T'} | \vec{\nu}_e)$. Applying First Green's identity in each element T, since $\Delta u_h = 0$ in every T, after straightforward calculations, we obtain

$$r(u_h, v) = \int_\Omega fv \, dxdy + \sum_{e \in \mathcal{E}_{h1}} \int_e [g - \partial_{\nu_e} u_h] v \, ds + \sum_{e \in \mathcal{E}_h} \int_e [\, |\partial_{\nu_e} u_h| \,]_e v ds. \tag{7.65}$$

Notice that the two summation terms in the residual express the fact that, contrary to the expected behaviour of exact solution normal derivatives, those of the approximate solution have jumps across inter-element boundaries. Therefore, it makes sense to evaluate those jumps as a measure of how much a numerical solution differs from the exact solution at a local level. In order to do this, we further define a constant function f_T in every $T \in \mathcal{T}_h$ given by $f_T = \frac{\int_T f \, dxdy}{area(T)}$ together with another constant function g_e in every $e \in \mathcal{E}_{1h}$ by $g_e = \frac{\int_e g \, ds}{length(e)}$. We denote by $\|\cdot\|_{0,T}$ and by $\|\cdot\|_{0,e}$ the norms in $L^2(T)$ and $L^2(e)$, respectively. Finally, for $T \in \mathcal{T}_h$, let $\mathcal{E}_T$ be the set of edges of T belonging to $\mathcal{E}_h$ and $\mathcal{E}_{T1}$ be the set of edges of T belonging to $\mathcal{E}_{1h}$. In doing so, we define a quantity η_T, namely, the

Local explicit residual a posteriori error estimator

$$\eta_T = \left[h_T^2 \, \| f_T \|_{0,T}^2 + \frac{1}{2} \sum_{e \in \mathcal{E}_T} h_e \, \| [|\partial_{\nu_e} u_h|]_e \|_{0,e}^2 + \sum_{e \in \mathcal{E}_{T1}} h_e \, \| g_e - \partial_{\nu_e} u_h \|_{0,e}^2 \right]^{\frac{1}{2}}. \tag{7.66}$$

η_T is intended to represent an upper bound for the residual in the sense that $r(u_h, v) \leq C_T \eta_T \|\mathbf{grad}\, v\|_{0,T}$, if we take $v \in V$ vanishing everywhere in the domain but in a small subdomain surrounding T, C_T being a constant independent of h_T. In this respect, it is a good error indicator to assess the quality of the numerical solution locally, and thus be used for mesh refinement.

Now, if we define the

Global residual a posteriori error estimator

$$\eta := \left[\sum_{T \in \mathcal{T}_h} \left(\eta_T^2 + h_T^2 \, \| f - f_T \|_{0,T}^2 \right) + \sum_{e \in \mathcal{E}_{T1}} h_e \, \| g - g_e \|_{0,e}^2 \right]^{1/2}, \tag{7.67}$$

it can be asserted that there exists a mesh-independent constant c_T such that

$$\|u - u_h\|_{1,2} \leq c_{\mathcal{T}}\eta.$$

This is the final a posteriori error estimate using the explicit residual-based method. The constant $c_{\mathcal{T}}$ can be estimated in order to obtain more or less sharp estimates, depending on the case (cf [200]). Moreover, η_T also provides local lower bounds for the error $u - u_h$ in a patch of elements surrounding T, measured in a suitable natural norm. For further details, we refer to reference [200].

The averaging method:

In this method, the same fundamentals of the residual-based a posteriori error estimation for the Poisson equation are exploited, but in a more heuristic manner. As pointed out above, in general the gradient of the approximate solution is discontinuous at inter-element boundaries. Here, instead of setting forth estimations in terms of normal derivative jumps, we attempt to smoothen the gradient of u_h into a continuous field $\mathbf{G}(u_h)$ determined by local averaging. Actually, for those well acquainted with concepts and ordinary operations in Hilbert spaces (see e.g. [52]), in principle $\mathbf{G}(u_h)$ is the **orthogonal projection** of $\mathbf{grad}\ u_h$ onto the space of continuous fields $W_h \times W_h$, where W_h is the $\mathcal{P}_1$ FE space incorporating no boundary conditions. As a matter of fact, irrespective of mastering the concept of orthogonal projection or not, the reader should take note that $\mathbf{G}(u_h)$ fulfils,

$$\mathbf{G}(u_h) \in W_h \times W_h \text{ and } (\mathbf{G}(u_h)|\mathbf{w})_0 = (\mathbf{grad}\ u_h|\mathbf{w})_0\ \forall \mathbf{w} \in W_h \times W_h. \tag{7.68}$$

Notice that we have $\mathbf{G}(u_h) = [G_x(u_h), G_y(u_h)]^T$. Thus, we must solve two SLAEs to determine $G_x(u_h)$ and $G_y(u_h)$, each one of them being at least as costly as the system we must solve to compute u_h itself. This is because there are two unknowns per mesh node instead of one per mesh node not belonging to Γ_0 in the case of u_h. Therefore, the procedure as such is not so reasonable, if we are to apply it recursively in order to successive refine the mesh. However, recalling the lumped mass technique, we can diagonalise the matrix underlying equation (7.68). In this case, $\mathbf{G}(u_h)$ can be determined in a straightforward manner, that is, without solving SLAEs. Let us see how this works.

First of all, we recall the two-dimensional counterpart of the trapezoidal rule, yielding an approximation $\mathcal{J}_h(\psi)$ of the integral of a continuous function ψ in Ω, namely,

$$\mathcal{J}_h(\psi) = \sum_{T \in \mathcal{T}_h} \frac{area(T)}{3} \sum_{m=1}^{3} \psi(S_m^T), \tag{7.69}$$

where S_m^T, $m = 1, 2, 3$ are the vertices of a triangle T. Let us apply this formula to compute the left side of equation (7.68), taking $\mathbf{w}$ equal to either $[\varphi_i, 0]^T$ or $[0, \varphi_i]^T$, φ_i being the shape function associated with mesh node S_i. Notice that in the former case for instance, this integral is nothing but the integral in Ω of the product $G_x(u_h)\varphi_i$, whereas in the latter case it equals the integral of $G_y(u_h)\varphi_i$ in Ω. Let Π_i be the union of mesh triangles having S_i as a vertex. Appying equation (7.69), from the fundamental properties of the shape functions it easily follows that

$$\begin{cases} G_x(S_i) \sum\limits_{T \subset \Pi_i} area(T)/3 = (\partial_x u_h | \varphi_i)_0, \\ G_y(S_i) \sum\limits_{T \subset \Pi_i} area(T)/3 = (\partial_y u_h | \varphi_i)_0. \end{cases} \tag{7.70}$$

Now, using the notation $[u_{h,x}^T, u_{h,y}^T]^T$ for $[\mathbf{grad}\ u_h]_{|T}$, we observe that in each triangle $T \subset \Pi_i$ this field is constant. Hence, taking into account that $\int_T \varphi_i dxdy = area(T)/3$ if $T \subset \Pi_i$, the right sides of equation (7.70) are given by

$$\begin{cases} (\partial_x u_h | \varphi_i)_0 = \sum\limits_{T \subset \Pi_i} u_{h,x}^T area(T)/3, \\ (\partial_y u_h | \varphi_i)_0 = \sum\limits_{T \subset \Pi_i} u_{h,y}^T area(T)/3. \end{cases} \tag{7.71}$$

Combining equations (7.70) and (7.71), we immediately conclude that

$$\begin{cases} G_x(S_i) = \sum\limits_{T \subset \Pi_i} u_{h,x}^T area(T)/area(\Pi_i), \\ G_y(S_i) = \sum\limits_{T \subset \Pi_i} u_{h,y}^T area(T)/area(\Pi_i). \end{cases} \tag{7.72}$$

Equation (7.72) can be interpreted as follows:

$$[\mathbf{G}(u_h)](S_i) = \sum_{T \subset \Pi_i} [\mathbf{grad}\ u_h]_{|T} area(T)/area(\Pi_i), \text{ for } i = 1, 2, \ldots, K_h, \tag{7.73}$$

where K_h is the number of mesh nodes. This means that the value of the orthogonal projection $\mathbf{G}(u_h)$ of $\mathbf{grad}\ u_h$ at mesh node S_i is obtained by the averaging of $\mathbf{grad}\ u_h$ in the domain Π_i about vertex S_i specified on the right side of equation (7.73).

It turns out that the quantity $\gamma_T := \|[\mathbf{G}(u_h) - \mathbf{grad}\ u_h]_{|T}\|_{0,T}$ related to an element $T \in \mathcal{T}_h$, that is,

$$\gamma_T = \left[\int_T \left\{ |[G_x(u_h)]_{|T} - u_{h,x}^T|^2 + |[G_y(u_h)]_{|T} - u_{h,y}^T|^2 \right\} dxdy \right]^{1/2} \tag{7.74}$$

is an excellent easy-to-compute local error estimator [3].

7.3.2 Mesh Adaptivity: h, p and h–p Methods

Skipping details for the sake of brevity, we see below some highlights on how γ_T or η_T can be used to improve FEM's accuracy through **mesh adaptivity**. FE and FV users have seen the importance of this technique steadily increase since the 1980s. In many branches of activity, it became an inseparable companion. For example, this is true of **inverse problems** governed by a PDE, whose solution is aimed at reconstructing field data. For applications of mesh adaptivity in this framework, we refer to the work of Beilina *et al*. [24].

Let us be given a small tolerance $\epsilon > 0$ for the quantities γ_T. First, we assume that only one mesh triangle, say $T_0 \in \mathcal{T}_h$, fulfils $\gamma_{T_0} \geq \epsilon$. Referring to Figure 7.7, we can subdivide T_0 into four triangles using the edge mid-points of T_0. Of course, if M_h is the total number of triangles in $\mathcal{T}_h$, the new triangulation with $M_h + 3$ triangles will not be a compatible mesh in the FE (or the FV) sense. Then, assuming that T_0 does not have any edge contained in Γ_1 for simplicity, we may adjust the three triangles T_1, T_2 and T_3 sharing an edge with T_0 in such a way that each one of them is split into two triangles by the segment joining the edge mid-point of T_0 to the opposite vertex of the neighbouring triangle (cf. Figure 7.7). Then, we come up with a final compatible mesh with $M_h + 6$ triangles, which is then used for the computation of a new approximate solution, say $\underline{u}_h$. Quantities $\underline{\gamma}_T$ related to $\underline{u}_h$ are computed in the same manner as γ_T for the original mesh, and the tolerance criterion is applied again. In case there are still triangles T in the locally refined mesh such that $\underline{\gamma}_T \geq \epsilon$, we repeat the remeshing procedure, until the tolerance criterion is fulfilled.

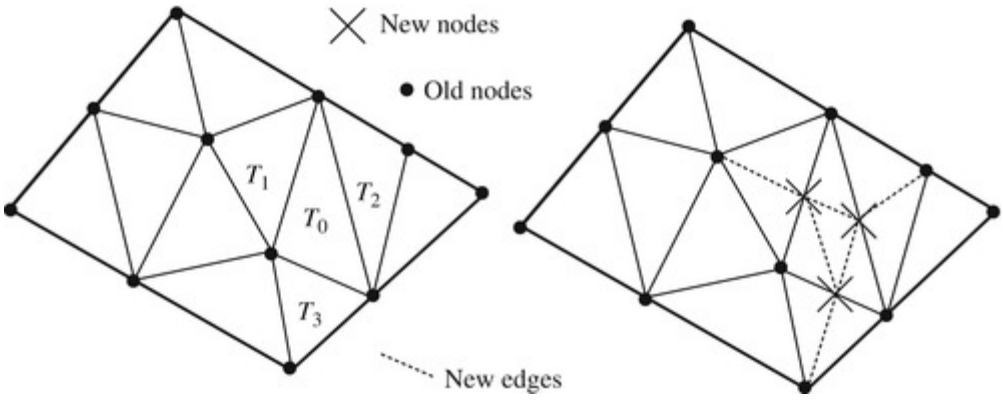


Figure 7.7 Subdivision of T_0 into four triangles and of T_1, T_2 and T_3 into two triangles

The above procedure was described in a deliberately simplified context, just to illustrate the principles that mesh adaptivity is based upon. Some remarks on everyday mesh refinement are in order to complete this brief introduction to the subject.

First of all, we note that the procedure we have just described is the so-called ***h*-version** of mesh adaptivity. Provided the exact solution is sufficiently smooth in the mesh portion to be refined, though with steep gradients, instead of subdividing the triangles not fulfilling the tolerance criterion into four or two triangles, we may change the approximate solution representation in each one of them using Lagrange quadratic interpolation having as additional nodes the edge mid-points (cf. Subsection 6.1.3). However, we cannot do the same for the triangles ensuring the transition from quadratic interpolation to linear interpolation, corresponding to those subdivided into two triangles in the refinement h-version. But this is not really a problem. It suffices to assume that the quadratic function in those transition triangles is such that their restriction to edges common to another triangle where only linear interpolation is performed is itself a linear function. This reduces to considering that the values of the quadratic function at such edge mid-points is the mean value of those at the respective end-points. In this approach, the computational refinement can go on by increasing the degree p of the interpolating function in triangles that do not fulfil the tolerance criterion. This is the so-called *p*-**version** of adaptivity. It is also possible to combine both techniques, thereby giving rise to the *h*-*p*-**version** of adaptivity. We do not further elaborate on procedures based on Lagrange interpolation of increasing order, since we did not present FE spaces of degree higher than two in two dimensions. We refer to reference [45] for a description of Lagrange interpolation of arbitrary order in triangles and to reference [13] for a thorough study on the h, p and $h-p$ methods. It is also noteworthy that, in principle, every step of a mesh refinement procedure affects a large number of mesh elements. For this reason, a sensible choice of remeshing techniques is recommended. There are plenty of them in the

literature, and it would be difficult to be exhaustive in this respect. A remarkable example of efficient mesh refinement algorithms can be found in reference [164] and the references therein.

7.4 A Word about Non-linear PDEs

As mathematical models of countless problems of interest in real-life applications, PDEs are mostly non-linear. For this reason, in contemporary numerical simulations of underlying physical events, practitioners have to deal with the solution of non-linear systems of algebraic equations of increasing complexity. In most cases, it is not reasonable to attempt to examine all details pertaining to an adopted numerical procedure, and users are satisfied with the a posteriori validation of their approach based on the coherence of the results, often as per their own judgment. This means that emphasis is given to the solution procedure in the aim of reducing costs, rather than on formal reliability studies. That is roughly what we intend to show in this book's final section, even though this does not mean that we approve such an attitude. Whatever a problem's complexity, there should be concern about accuracy control. However, we have to overlook such aspects here as a matter of choice, owing to obvious limitations to treat such a vast subject in a single section.

Before starting, a few comments on the relationship between linear and non-linear, as far as numerical methods for PDEs are concerned, are of paramount importance.

This book was intended to give insight on the most used numerical methods for solving PDEs. However as a beginner-oriented text, the presentation of these techniques was confined to the linear paradigms of the three types of PDEs in both one and two space variables, with a short extension to the three-dimensional case. In many cases, these linear PDEs are the **linearised form** of the true non-linear equation. For this reason too, the adopted approach in this book is mandatory in some sense. Indeed, methods that fail to work for a linearised equation will certainly work even worse if one attempts to apply it to the full non-linear form of the equation. But then the question is: to which extent can the reliability of a numerical method for a certain type of linear PDE predict its performance as applied to a complete non-linear PDE of a similar nature? The study of non-linear PDEs opens the gate to a much more complex universe, and for this reason unfortunately the answer to this question cannot be given in simple terms. For instance, even if the existence of a solution to a non-linear PDE is guaranteed, is general its uniqueness is not. There are phenomena known as bifurcation or turning points depending on the variation of a real parameter defining in some sense the importance of non-linear terms. Generally

speaking, if this parameter is close to zero the non-linear equation has a behaviour similar to the underlying linearised model. In particular, this means an equation with a unique solution. On the other hand, as this parameter goes beyond critical values the solution will no longer be unique. Some non-linear PDEs also behave in a way rather difficult to deal with. Even smooth data and domains can give rise to discontinuous solutions. A particularly representative example is the apparently gentle **Bürgers equation** of gas dynamics, which is a sort of non-linear transport equation, whose transport velocity at every $(x;t)$ is the solution at this point itself. In an infinite one-dimensional domain, this equation writes,

$$\partial_t u + u\partial_x u = 0 \text{ for } t > 0 \text{ with given } u_0(x) \text{ such that } u(x,0) = u_0(x)\ \forall x.$$

If we take $u_0(x) = x$, then a possible solution is given by $u(x,t) = x/(t+1)$ for $t > 0$, as one can easily check. Now let us take $u_0(x) = -x$. If we change $t+1$ into $t-1$ in the above expression of $u(x,t)$, in principle the new function satisfies the Bürgers equation, together with the initial condition. However, what happens when $t = 1$? The gas particles located at the abscissae x at time $t = 0$, and transported with initial velocity equal to $-x$ will all collide at time $t = 1$! From this time on, the unknown function u will continue to evolve, even though not according to the above analytic expression. This example was rather academic, but in practice a phenomenon known as *shock wave* occurs for flows modelled by the Bürgers equation. In mathematical terms, this consists of a curve $t = g(x)$ in the $x - t$ plane, along which the solution is discontinuous. We refer for instance to references [110] and [67] for more explanations about the Bürgers equation and an introduction to numerical methods to deal with it, even in the presence of shock waves.

As a rule, whenever the dominant differential operator in terms of order of differentiation is linear, a non-linear PDE behaves somehow like its linear counterpart, provided the data are sufficiently small. Just to give an example, let us consider the following equation in a bounded two-dimensional domain Ω with boundary Γ:

$$-\Delta u + u^3 = f \text{ in } \Omega \text{ with } u = 0 \text{ on } \Gamma.$$

Multiplying both sides of this equation by u and integrating in Ω, and further using the Cauchy–Schwarz and the Friedrichs–Poincaré inequalities, followed by Young's inequality with $\delta = 1$, we obtain

$$\|\mathbf{grad}\ u\ \|_{0,2}^2 + 2\ \|u^2\|_{0,2}^2 \leq C_P^2\ \|f\|_{0,2}^2\ .$$

The solution gradient is seen to be bounded by the norm of f times a constant, but better than this, the norm of the squared solution in $L^2(\Omega)$ is also bounded by the same quantity. Then, provided the latter is not so large (i.e. f is sufficiently small), we can study the numerical solution of the above equation following recipes in all similar to those we exploited throughout the text, based on the Lax–Richtmyer equivalence theorem. Of course, this is not as direct and simple as for the linearised form $-\Delta u + cu = f$ for a certain non-negative constant c. However, the simple fact that $u^3 = u^2 \times u$, combined with the bound for the norm of u^2 which also holds for the numerical counterpart u_h, allows for this. We observe, however, that even so the reliability analysis becomes fairly more complicated than in the linear case, and for this reason we do not further elaborate on them.

A final comment to conclude this brief introduction to the non-linear world is as follows. For the general case, a certain number of mathematical tools not considered in this text are available, to handle non-linear PDEs together with their numerical solutions, such as **Brower's fixed point theorem**. We refer to reference [47] for more details on this matter.

7.4.1 Example 7.4: Solving Non-linear Two-point Boundary Value Problems

Along the same line of thought as in the introduction of this section, we next endeavour to solve a very simple non-linear ODE as a model. However, it is the author's expectation that the reader will be able to extrapolate and figure out what could be done to solve any other non-linear boundary value PDE.

More precisely, we consider the numerical solution by the FDM of the following non-linear boundary value ODE:

Given $p, r, f \in C^0[0, L]$, find u such that

$$-(pu')' + ru - (1+r)u^2 + u^3 = f \text{ in } (0, L), u(0) = 0, u'(L) = 0. \qquad (7.75)$$

This equation models a special case of combustion, in which a reaction–diffusion process takes place along a straight cylindrical channel with length L, occupied by a certain medium, without transversal effects [209]. In this case, p and r are the (local) diffusion and reaction coefficients, respectively, both being strictly positive. Notice that for suitably small data, u will be necessarily small enough for the non-linear terms to be negligible. In such a case, the model reduces to the

linear ODE (P_1) with $q = r$.

Taking $p = r = 1$, let us be given a uniform FD grid with size $h = L/n$ for a certain integer $n > 1$. Denoting $f(ih)$ by f_i and the approximations of $u(ih)$ by u_i for $i = 1, 2, \ldots, n$, similarly to Section 1.2, the latter satisfy

$$\frac{2u_i - u_{i-1} - u_{i+1}}{h^2} + u_i - 2u_i^2 + u_i^3 = f_i, \text{ for } i = 1, 2, \ldots, n \text{ with } u_{n+1} = u_{n-1}, u_0 = 0.$$

These equations correspond to a non-linear system of n algebraic equations with n unknowns. Leaving aside deeper analyses, we assume that it has a solution $\vec{u}_h = [u_1, u_2, \ldots, u_n]^T$ close to an initial guess $\vec{u}_h^0 = [u_1^0, u_2^0, \ldots, u_n^0]^T$, more precisely satisfying $|\vec{u}_h^0 - \vec{u}_h| \leq \rho$ for ρ sufficiently small. There are many methods to solve this non-linear system, and in this respect we refer to classical books on the subject such as reference [150]. Here, we select Newton's method as one of the most effective. For a given n-component vector $\vec{v} = [v_1, v_2, \ldots, v_n]^T$, we set

$$F_i(\vec{v}) := \frac{2v_i - v_{i-1} - v_{i+1}}{h^2} + v_i - 2v_i^2 + v_i^3 - f_i, \text{ for } i = 1, 2, \ldots, n-1,$$

together with

$$F_n(\vec{v}) := \frac{2(v_n - v_{n-1})}{h^2} + v_n - 2v_n^2 + v_n^3 - f_n.$$

Notice that the system to solve is $F_i(\vec{u}_h) = 0$ for $i = 1, 2, \ldots, n$. Now we define the gradient of $F_i(\vec{v})$ to be the n-component vector whose jth component is $G_{i,j}(\vec{v}) := \partial F_i(\vec{v})/\partial v_j$. This definition gives rise to an $n \times n$ **Jacobian matrix** $\mathcal{G}(\vec{v}) = \{G_{i,j}(\vec{v})\}$ associated with $\vec{v}$, which is nothing but the gradient of the n-component field $\mathcal{F}(\vec{v}) = [F_1(\vec{v}), \ldots, F_n(\vec{v})]^T$. Notice that $G_{i,j}(\vec{v}) = 0$ if $|j - i| > 1$. Therefore, $\mathcal{G}(\vec{v})$ is a tridiagonal matrix whose entries are $G_{i,i}(\vec{v}) = 2/h^2 + 1 - 4v_i + 3v_i^2$, for $i = 1, 2, \ldots, n$, $G_{i,i-1}(\vec{v}) = -1/h^2$ for $i = 2, \ldots, n-1$ and $G_{n,n-1}(\vec{v}) = -2/h^2$, $G_{i,i+1}(\vec{v}) = -1/h^2$ for $i = 1, 2, \ldots, n-1$. Then the first iteration of Newton's method yields a vector $\vec{u}_h^1 = [u_1^1, \ldots, u_n^1]^T$ satisfying,

$$\mathcal{G}(\vec{u}_h^0)[\vec{u}_h^1 - \vec{u}_h^0] = -\vec{\mathcal{F}}(\vec{u}_h^0).$$

The same procedure is recursively applied to determine approximations $\vec{u}_h^{k+1}$ of $\vec{u}_h$ for $k = 1, 2, \cdots$, by **Newton's iterations**, that is,

$$\mathcal{G}(\vec{u}_h^k)[\vec{u}_h^{k+1} - \vec{u}_h^k] = -\vec{\mathcal{F}}(\vec{u}_h^k),$$

until the maximum norm of $\vec{u}_h^k - \vec{u}_h^{k+1}$ becomes smaller than a given tolerance δ, or, for better scaling, the product of δ with the maximum norm of $\vec{u}_h^{k+1}$. This means that in the latter case, the stop criterion is based on the relative increment between two successive iterations, and on the absolute increment in the former case. Notice that a sufficiently large maximum number of iterations must be stipulated, in order to provide for the eventuality of iteration divergence. Anyway, as long as the initial guess is not too far from an expected solution of the non-linear system, as a rule convergence of this procedure to this solution does occur at an asymptotic rate better than linear, except for especial cases, and to the most quadratic (see e.g. [46]). More precisely, $\lim_{k\to\infty} \frac{\|\vec{u}_h^{k+1} - \vec{u}_h\|_\infty}{\|\vec{u}_h^k - \vec{u}_h\|_\infty^\alpha}$ equals a constant for some α, $1 < \alpha \leq 2$. The well-posedness issue of the SLAE to solve at the kth iteration deserves careful attention. In case the matrix $\mathcal{G}(\vec{u}_h^k)$ is singular or very ill-conditioned (cf. [46]), a possible way out is to shift the components of $\vec{u}_h^k$ by a small quantity and recompute the matrix.

In order to check the performance of the combination of Newton's method and the FD discretisation, we next give some numerical results obtained for a **manufactured solution**. We take $u(x) = \sin(\pi x/2)$, which satisfies the prescribed boundary conditions at $x = 0$ and $x = 1$. Then, we calculate $f = -u'' + u - 2u^2 + u^3$ to obtain $f(x) = \sin(\pi x/2)\{(\pi/2)^2 + 1 - \sin(\pi x/2)[2 - [\sin(\pi x/2)]\}$. We ran a MATLAB code supplied hereafter for solving this ODE by Newton's iterations, starting from a zero initial guess and taking a tolerance for the absolute increment between two successive iterations equal to 10^{-5}. Two different meshes were tested, namely, a very coarse one with $n = 5$ and a finer one with $n = 100$. In both cases, the stop criterion was satisfied after only five iterations, which is undoubtedly a remarkable performance.

Figures 7.8a and 7.8b displays the exact solution and the approximate solution at the grid points for $n = 5$ and $n = 100$, respectively. As one can infer from both figures, the discretisation errors are very small, even for the coarse mesh.

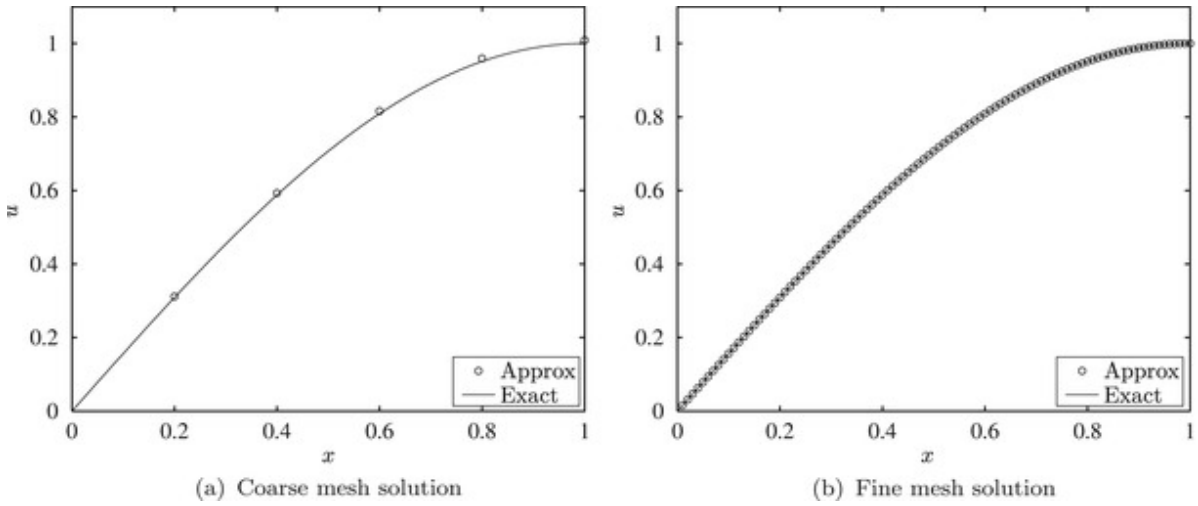


Figure 7.8 Exact and FD solution of a non-linear ODE

The following MATLAB code was used to generate these numerical results. It is rather self-explanatory, but the reader is advised to read it through in order to consolidate her or his understanding of Newton's iterations to solve non-linear systems of algebraic equations. Such a task, among others, is the object of Exercise 7.16.

```
function main
MaxNumNewtonIter = 100; delta = 1e-5;  n = 5;
a                = 0;   b     = 1;     h = (b-a)/n; u0 = 0;
x  = zeros(n,1); uold = zeros(n,1);  unew  = zeros(n,1);
DF = zeros(n,n); F    = zeros(n,1);
for k = 1:MaxNumNewtonIter
  x(1)    = h;
  DF(1,1) = 2/h^2 + 1 - 4*uold(1) + 3*uold(1)^2;
  DF(1,2) = -1/h^2;
  F(1)    = -(- u0/h^2 + ( 2*uold(1)/h^2 + uold(1) - 2*uold(1)^2
          + uold(1)^3 ) ... - uold(2)/h^2 - f(x(1)) ) ;
  for i = 2:n-1
    x(i) = i*h;
    DF(i,i-1) = -1/h^2;
    DF(i,i  ) = 2/h^2 + 1 - 4*uold(i) + 3*uold(i)^2;
    DF(i,i+1) = -1/h^2;
    F(i)      = -(- uold(i-1)/h^2 + ( 2*uold(i)/h^2 + uold(i)
```

```
            -2*uold(i)^2 + ... uold(i)^3 ) - uold(i+1)/h^2 - f(x(i)) ) ;
 end
 x(n)      = b;
 DF(n,n-1) = -1/h^2;
 DF(n,n  ) = ( 2/h^2 + 1 - 4*uold(n) + 3*uold(n)^2 )/2;
 F(n)      = -( - uold(n-1)/h^2 + ( 2*uold(n)/h^2 + uold(n)
           - 2*uold(n)^2 + ... uold(n)^3 ) - uold(n-1)/h^2 - f(x(n)) )/2 ;
 unew = linsolve( DF, F );
 max=0;
 for i=1:n
  if ( abs(unew(i))> max )
    max = abs(unew(i));
  end
 end
 unew = unew + uold;
 if ( max < delta )
  break
 end
 uold = unew;
end
figure(1);
z = linspace( 0, 1, 100 );
hfig = plot( x, unew, 'ko', z, sin(pi*z/2), '-k' );
xlabel('$x$','FontSize', 18,'interpreter','latex');
ylabel('$u$','FontSize', 18,'interpreter','latex');
legend('Approx','Exact','Location','southeast');
end
function y = f( x )
  y = sin(pi*x/2)*( pi^2/4 + 1 - 2*sin(pi*x/2) + (sin(pi*x/2))^2 );
end
```

To conclude, it is important to point out that non-linear differential equations discretised by the FVM and the FEM can be solved in a similar manner. For example, in the case of the above model ODE, if the FVM is used, the non-linear terms will be replaced with their integrals in the

CVs. A field $\mathcal{F}$ and corresponding Jacobian matrix $\mathcal{G}$ very similar to those we determine above for the FDM will result from such a discretisation. In the case of the $\mathcal{P}_1$ FEM, however, the non-linear algebraic equations will not quite have the same form, since they will result from the following variational form: $a(u,v) = F(v)\ \forall v \in V$, where

$$a(u,v) := \int_0^L [pu'v' + (u - 2u^2 + u^3)v]dx;\ F(v) := \int_0^L fv\,dx,$$

V being the space defined in Section 1.3, together with its FE counterpart V_h. Actually, if V is replaced by V_h and u by $u_h \in V_h$ in the above equations, a system of n non-linear algebraic equations $F_i(\vec{u}_h) = 0$ will have to be solved, each F_i corresponding to the choice $v = \varphi_i$, $i = 1, 2, \ldots, n$ (i.e. the $\mathcal{P}_1$ FE shape functions). However, now the off-diagonal coefficients of the Jacobian matrix will no longer be constant because second and third powers of $u_h = \sum_{j=1}^{n} u_j\varphi_j$ will couple products of powers of u_i and u_j if $|j - i| \leq 1$. The reader is invited to write down the non-linear systems to solve at each Newton's iteration for both the Vertex-centred FVM and the $\mathcal{P}_1$ FEM for the same mesh with n equally spaced CVs or elements, as Exercise 7.17.

7.4.2 Example 7.5: A Quasi-explicit Method for the Navier–Stokes Equations

Newton's method is a a sort of master key to be employed in the efficient solution of discretised non-linear PDEs, with very few exceptions. However, the underlying procedure couples all the unknowns in a sequence of varying non-linear systems, whose solution can be very expensive in terms of computational effort, depending on the space dimension and mesh or grid size. Therefore, practitioners are often searching for less costly though reliable solution algorithms, well adapted to a particular type of non-linear PDE they have to deal with. Once more, we do not intend to go into details, but to conclude this section it seems important to minimally address what solution algorithms for non-linear PDEs other than Newton's iterations are about. This will be done in the framework of a *homemade* method to solve the **Navier–Stokes equations** [171], which model stationary or time-dependent flows of an incompressible Newtonian fluid (see e.g. [15]) in a bounded N-dimensional domain Ω, $N = 2$ or $N = 3$, with boundary Γ. Here, only numerical results based on the $\mathcal{P}_1$ FEM matter, which illustrate an inexpensive solution strategy as compared to Newton's method.

Skipping details for the sake of conciseness, we consider only stationary flows which are governed by

The stationary incompressible Navier–Stokes equations

$$\left.\begin{array}{l}\text{Find a velocity } \vec{u} = [u_1, \ldots, u_N]^{\mathcal{T}} \text{ and a pressure } p \text{ such that} \\ -\mu\Delta\vec{u} + [\mathbf{grad}\ \vec{u}]\vec{u} + \mathbf{grad}\ p = \vec{f}, \\ div\ \vec{u} = 0, \\ \vec{u} = \vec{g}\end{array}\right\} \begin{array}{l}\\ \text{in } \Omega; \\ \\ \text{on } \Gamma\end{array} \tag{7.76}$$

where $\vec{f}$ is a given density of forces, and $\vec{g}$ is a prescribed boundary velocity satisfying the global conservation property $\oint_\Gamma (\vec{g}|\vec{\nu})dS = 0$ (because $div\ \vec{u} = 0$). Like in the linear elasticity system, **grad** $\vec{u}$ is a second-order $N \times N$ tensor. For $N = 3$, its $i - j$ component is $\partial u_i/\partial x_j$ where $x_1 = x$, $x_2 = y$ and $x_3 = z$. The unknowns $\vec{u}$, p and domain are assumed to be in normalised dimensionless form. In this case, the coefficient μ is the inverse of the **Reynolds number** Re (cf. [15]), which is given by $Re = \rho VL/\eta$ where ρ is fluid's density, V and L are characteristic velocity and length, and η is fluid's viscosity. Notice that, similarly to the advection–diffusion equation, μ measures the importance of the terms containing second-order derivatives with respect to those containing first-order derivatives. More precisely, a large μ indicates dominant viscous effects, while a small μ points to dominance of mass transfer by inertia. Notice that the pressure is determined up to an additive constant, which can be fixed by prescribing for instance $p = 0$ at a given point of Ω.

The method employed to obtain the computational results shown hereafter incorporates the **deviator Cauchy stress tensor** σ as an additional unknown field. For a Newtonian viscous fluid, this tensor is expressed by $\sigma = 2\mu D(\vec{u})$, $D(\vec{u})$ being the symmetric gradient of $\vec{u}$ given by $D(\vec{u}) := [\mathbf{grad}\ \vec{u} + (\mathbf{grad}\ \vec{u})^{\mathcal{T}}]/2$, known as the **strain rate tensor**. In doing so, the stationary incompressible Navier–Stokes equations can be rewritten as follows:

$$\left\{\begin{array}{l}\text{Find } \vec{u}, p \text{ and } \sigma \text{ such that} \\ -\mathbf{Div}\sigma + [\mathbf{grad}\ \vec{u}]\vec{u} + \mathbf{grad}\ p = \vec{f}, \\ div\ \vec{u} = 0 \\ \sigma = 2\mu D(\vec{u}), \\ \vec{u} = \vec{g} \text{ on } \Gamma \text{ and } p(M) = 0.\end{array}\right\} \text{ in } \Omega; \tag{7.77}$$

where M is a given point of Ω. A popular technique to linearise time-independent non-linear PDEs like equation (7.76) is based on a pseudo time integration. This consists of applying a FD discretisation to a fictitious first-order time derivative added to the left side of the equation, with (pseudo) time step τ. In the case of equation (7.77), among other possibilities, this can be achieved by adding $\partial_t \vec{u}$ to the first equation and $\lambda\partial_t\sigma$ to the third equation, where $\lambda > 0$ is a

constant numerical parameter. Then these terms are approximated by $[\vec{u}^k - \vec{u}^{k-1}]/\tau$ and $\lambda[\sigma^k - \sigma^{k-1}]/\tau$, for $k = 1, 2, \cdots$, starting from a suitably chosen $\vec{u}^0$ satisfying $\vec{u}^0 = \vec{g}$ on Γ, and setting $\sigma^0 := 2\mu D(\vec{u}^0)$.

Schematically, assuming that $\vec{u}^{k-1}$ and σ^{k-1} are known, we solve at the kth iteration,

$$\left\{\begin{array}{l} \text{Find } \vec{u}^k, p^k \text{ and } \sigma^k \text{ such that} \\ \vec{u}^k = \vec{u}^{k-1} + \tau\{\mathbf{Div}\sigma^{k-1} - [\mathbf{grad}\ \vec{u}^{k-1}]\vec{u}^{k-1} - \mathbf{grad}\ p^k + \vec{f}\}, \\ div\ \vec{u}^k = 0 \\ \sigma^k = \frac{1}{\tau+\lambda}\left[\lambda\sigma^{k-1} + 2\mu\tau D(\vec{u}^k)\right], \\ \vec{u}^k = \vec{g} \text{ on } \Gamma \text{ and } p^k(M) = 0. \end{array}\right\} \text{ in } \Omega; \tag{7.78}$$

for $k \geq 1$. Solution algorithm (7.78) is explicit but for the coupling of $\vec{u}^k$ and p^k in the first two equations. Both unknowns can be decoupled, taking the divergence of all the terms in the first equation except $\vec{u}^k$. In doing so, we come up with a new second equation to replace $div\ \vec{u} = 0$. Recalling that $div\ \mathbf{grad}\ p = \Delta p$, interchanging the resulting second equation and the first equation, after straightforward manipulations, instead of equation (7.78) we solve

$$\left\{\begin{array}{l} \text{Find } \vec{u}^k, p^k \text{ and } \sigma^k \text{ such that} \\ -\Delta p^k = -div\ \vec{u}^{k-1}/\tau - div(\mathbf{Div}\sigma^{k-1} - [\mathbf{grad}\ \vec{u}^{k-1}]\vec{u}^{k-1} - \vec{f}), \\ \vec{u}^k = \vec{u}^{k-1} + \tau(\mathbf{Div}\sigma^{k-1} - [\mathbf{grad}\ \vec{u}^{k-1}]\vec{u}^{k-1} - \mathbf{grad}\ p^k + \vec{f}), \\ \sigma^k = \frac{1}{\tau+\lambda}\left[\lambda\sigma^{k-1} + 2\tau\mu D(\vec{u}^k)\right]. \\ \vec{u}^k = \vec{g} \text{ on } \Gamma \text{ and } p^k(M) = 0. \end{array}\right\} \text{ in } \Omega; \tag{7.79}$$

Notice that, taking the divergence of both sides of the second equation of (7.79) and adding up both sides of the resulting relation to the first equation multiplied by τ, we derive $div\ \vec{u}^k = 0$. Equation (7.79) is said to be quasi-explicit since only the pressure p^k is determined by solving a PDE. More specifically, p^k is the solution of a Poisson equation with inhomogeneous Neumann boundary conditions, which are formally determined by multiplying both sides of the second equation by $\vec{\nu}$ (cf. [171] and references therein). However, since we are dealing with a variational (weak) formulation, we do not need to bother about these boundary conditions, for they will be automatically fulfilled, though only in a weak sense.

Convergence of $(\vec{u}^k; p^k; \sigma^k)$ to $(\vec{u}; p; \sigma)$ in an appropriate norm is ensured, provided μ is sufficiently large or τ is sufficiently small (cf. [171]).

Equation (7.79) is set in a suitable GLS variational form, and then a classical discretisation of $\vec{u}^k$, p^k and σ^k with $\mathcal{P}_1$ FEs is performed, thereby yielding corresponding approximations $\vec{u}_h^k$, p_h^k and σ_h^k. We refer to reference [171] for the adaptation to this formulation of the quasi-explicit solution algorithm (7.79), among other pertaining details. In particular, the reader should be

aware that the solution method for this system of equations in Galerkin formulation with linear FE representations of the three fields turns out to be unstable. This is due to the violation of two Babuška–Brezzi (inf-sup) conditions (see e.g. [169]).

It is important to stress the fact that the mass lumping technique is used to determine $\vec{u}_h^k$ and σ_h^k from the FE analog of the second and third equations of (7.79). This means that the nodal values of both fields are determined by sweeping the mesh in a completely explicit manner (i.e. node by node and component by component). p_h^k in turn is computed by solving the SLAE underlying the discretised first equation of (7.79). Storage requirements are reduced to a minimum, if this SLAE is solved by an iterative method. Actually, in these computations we used the **pre-conditioned conjugate gradient method** (see e.g. [63]) with a pre-conditioning matrix obtained by incomplete Cholesky factorisation. This means that the latter has the same sparsity structure as the original matrix. Notice that in this case, only the nonzero coefficients of fixed matrices have to be stored along the iterations, in the total amount of circa l times the number of nodes multiplied by two to take into account the pre-conditioning matrix. l accounts for the half number of neighbours of a typical node, since the matrix is symmetric. The value of l is about 5 (resp. 14) in two (resp. three) space dimensions. Moreover, since for every k we take p_h^{k-1} as an initial guess, convergence of this linear system solver occurs after one or two iterations, except for the very first values of k.

Next we supply some numerical results illustrating the performance of this algorithm, taking as test problem the **circular Couette flow** or **Taylor–Couette flow** (see e.g. [44]). In this particular problem, a viscous incompressible fluid is confined between two concentric cylinders with radii r_i and $r_e > r_i$. The outer cylinder stands still, while the inner cylinder is rotating with angular velocity ω. If the magnitude of a dimensionless parameter Ta called the Taylor number does not exceed a critical value Ta_c, the flow will be laminar (i.e. not turbulent) and invariant with respect to the axial coordinate z. Ta_c in turn is attained for a critical value of Re depending on r_i and r_e (see e.g. [6]). If $Ta \leq Ta_c$, it is possible to determine exact analytic expressions for $\vec{u}$ and p, and consequently for σ, depending only on the radial coordinate of the polar coordinate system $(O; r, \theta)$ of the concentric cylinders' cross-section plane, whose origin O lies on their axis of symmetry. Moreover, we may consider that the Navier–Stokes equations hold for $\vec{u}$ and p, in an annulus Ω with inner radius r_i and outer radius r_e. Γ is the union of the concentric circles with radii r_i and r_e. The velocity field components in the polar coordinate basis $[\vec{e}_r; \vec{e}_\theta]$ are u_r and u_θ, respectively, for which the following Dirichlet boundary conditions apply: $u_r = 0$ on Γ,

$u_\theta = \omega r_i$ for $r = r_i$ and $u_\theta = 0$ for $r = r_e$. Field $\vec{f}$ vanishes identically, and the exact solution is given by

$$u_r \equiv 0;\ u_\theta = A(r_e^2/r - r);\ p = A^2(r^2 - 4r_e^2 \log r - r_e^4/r^2 + 4r_e^2 \log r_e)/2,$$

with $A = \omega r_i^2/(r_e^2 - r_i^2)$. Notice that the pressure vanishes for $r = r_e$ and this is the condition we use to fix the additive constant up to which p is defined.

In a battery of experiments with $Ta < Ta_c$, we took $r_i = 1/2$, $r_e = 1$, $\omega = 1/2$ and $Re = 18.75$, We worked in Cartesian coordinates $(O; x, y)$ of the plane of Ω. This means that we search for two velocity components u_x and u_y in the Cartesian frame in terms of x and y. This is because a priori a computer code ignores symmetries if they are not prescribed. It is noteworthy that in some cases, solving a non-linear problem by forcing symmetries can be harder than without doing it.

In order to observe the convergence of the three fields to the exact solution in the L^2-norm, we solved the problem with meshes containing an equal number of elements in each radial level corresponding to n_r equal subdivisions of annulus' radius, and the subdivision of the domain in the azimuthal sense into n_θ equal sectors, for increasing values of n_r and n_θ. The triangles are obtained by subdividing the thus-generated $n_r \times n_\theta$ trapezoids by means of one of their two diagonals, in such a way that a given diagonal never shares its end-points with the other ones both in the same radial level and in the same sector (see Figure 7.9). In doing so, the mesh contains m triangles with $m = 2n_r \times n_\theta$ but has symmetries with respect to neither x nor y. The mesh in a quarter annulus is illustrated in Figure 7.9 for $n_r = 2$ and $n_\theta = 16$.

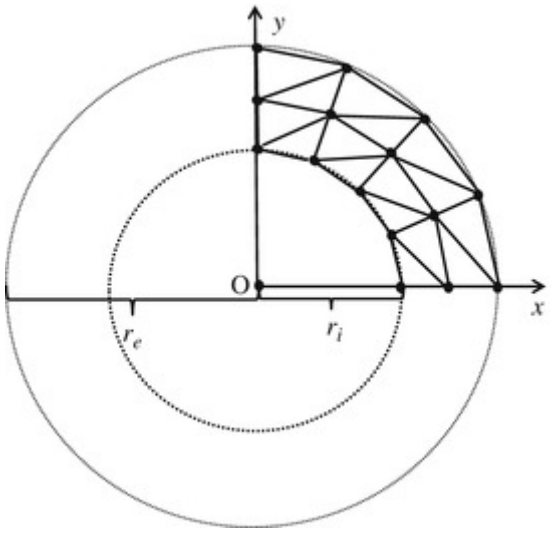


Figure 7.9 Mesh of a quarter annulus for the simulation of circular Couette flow

Notice that the union of the mesh triangles is not contained in Ω, since Γ consists of both convex and concave portions. The value of τ decreases with the mesh size in the way indicated in Table 7.7, where we display relative errors in the L^2-norm of $\vec{u}$, p and σ, for $n_r = 2, 4, 8, 16$ and $n_\theta = 16, 32, 64, 128$. A tolerance of 0.2×10^{-6} for the velocity maximum increment between two successive iterations was used, and we took $\lambda = .5$. The total number of iterations k_f to attain the tolerance is also shown, rounded to 1000. In order to speed up the convergence in those experiments, we used the combination of mass lumping on the left side with the consistent mass on the right side in the velocity equation (cf. Subsection 7.2.7), in the middle of the simulations. As one can infer from Table 7.8, τ must diminish roughly linearly as the mesh is refined, while the total number of iterations to satisfy the tolerance criterion sharply increases. The main conclusion of the above results is the fact that the approximations of the three unknown fields generated by the combination of a $\mathcal{P}_1$ FE –GLS formulation and the algorithm (7.79) can be qualified as reasonable. This is because convergence can be detected as the number of elements increase, though at an unclear rate. On the other hand, the method'smain limitation in the solution of this non-linear problem is the rather large number of iterations necessary to attain convergence. However, this is often the case of numerical methods to solve non-linear PDEs designed to reduce costs. Notice that if Newton's iterations were used to solve the same problem by coupling all the three fields, much less iterations would be necessary to attain convergence. But in this case, an $m \times m$ matrix would have to be computed at every iteration, where m equals 6 (resp. 10) times the number of nodes in two (resp. three) space dimensions, in principle in the form of a banded matrix. Notice that this matrix would have to be factorised, in case a direct method or even some iterative method was employed, and this can be really very costly. In short, the solution of large systems resulting from the discretisation of non-linear PDEs requires good compromises between cost and accuracy. That is what we intended to show by means of this test case, among a multitude of other examples to be found in the literature.

Table 7.7 Relative errors of $\vec{u}$, p and σ in the L^2-norm

	64	**256**	**1024**	**4096**
	0.00050	0.00040	0.00002	0.00001
	8	15	47	112
	0.05159502	0.04407774		0.00823048
		0.07072221	0.02191661	0.00143939
	0.26958132	0.13792677	0.06432909	0.02446557

For a comprehensive study on the numerical solution of the incompressible Navier–Stokes equations by the FEM, the reader can consult Glowinski's book [90].

7.5 Exercises

7.1 Show that, as long as the solution of the biharmonic equation 7.3 is sufficiently smooth, the local truncation error of FD scheme (7.6) is an $O(h^2)$ with $h = h_x = h_y$. Then develop (7.4)–(7.5) to obtain the 13-point FD scheme analogous to equation 7.6, for solving the biharmonic equation in a rectangle with $h_x \neq h_y$. What happens to the local truncation error in this case?

7.2 Check that equation 7.7 implies equation 7.3, assuming a suitable regularity of u. To be rigorous, the reader should further admit the validity of the following **density property**: For every function $g \in L^2(\Omega)$, given a real number $\epsilon > 0$ arbitrarily small, there exists a function $v \in H_0^2(\Omega)$ such that $\| g - v\|_{0,2} \leq \epsilon$ (see e.g. Adams, 1975).

7.3 Show that $\| H(w)\|_{0,2} = \| \Delta w\|_{0,2}$ for every $w \in H_0^2(\Omega)$ (although the result is perfectly true, in the calculations the reader may assume a little more regularity of w).

7.4 Check that the traces of the normal derivative of a function v in the space V_h associated with the Bogner–Fox–Schmit FE along a mesh edge parallel to the x-axis (resp. y-axis) are a cubic function in terms of x (resp. y). Conclude that, as long as the four values $v(P)$, $[\partial_x v](P)$, $[\partial_y v](P)$ and $[\partial_{xy} v](P)$ at all the mesh vertices P located on the boundary Γ vanish, every function in $v \in V_h$ satisfies the boundary conditions $v = \partial_\nu v = 0$ everywhere on Γ.

7.5 Determine the number of unknowns and matrix bandwidth for the 13-point FD scheme with a $2n_x \times 2n_y$ uniform grid, and the Bogner–Fox–Schmit (BFS) FE to solve equation 7.3 with the corresponding $n_x \times n_y$ mesh. Assume an unknown numbering minimising the maximum bandwidths, and take $n_y \leq n_x$. Compare also matrix sparsity for both methods in the same situation.

7.6 Prove the stability inequality (7.22) for scheme (7.20) taking $L = 1$ (hint: use a technique inspired by the stability analysis for the Three-point FD scheme leading to equation (2.3)).

7.7 Assuming that the solution to the advection–diffusion equation has a continuous third-order derivative in $[0, L]$, prove the first-order convergence in the pointwise sense of FD scheme (7.20) starting from equation 7.22.

7.8 Assuming that $w = \pm 1$, check that equation 7.23 gives rise to equation 7.24.

7.9 Starting from equation 7.25, prove the stability inequalities (7.26) for the SUPG FE scheme with $w = \pm 1$.

7.10 Derive an estimate for $|u - u_h|_{1,2}$, assuming that w is a differentiable function everywhere in $[0, L]$, having a non-positive first-order derivative in $(0, L)$. Specify the underlying constant based on the fact that necessarily $w \in L^\infty(0, L)$.

7.11 Check that the local truncation error of the FV scheme (7.29) in the FD sense is an $O(1)$. Consider the case of a non-uniform mesh.

7.12 Establish the validity of the discrete Friedrichs–Poincaré inequality (7.31).

7.13 Prove that estimate (7.40) holds.

7.14 Establish the validity of equation 7.62 (hint: use arguments similar to those exploited in Section 3.2 in the stability analysis of the Forward Euler scheme).

7.15 Check that the local truncation error of schemes (7.48)–(7.52) is a term of the form $\tau O(1)$, even under suitable solution regularity conditions and assuming that equation 7.63 holds.

7.16 Study the MATLAB code supplied in Example 7.4, until all its steps and instructions are fully understood. Modify the code in order to treat inhomogeneous Dirichlet and Neumann boundary conditions at the right and left ends, respectively, and recover FD approximations of another manufactured solution satisfying them.

7.17 Write down the systems of non-linear algebraic equations to solve at each Newton's iteration, for both a Vertex-centred FV and a $\mathcal{P}_1$ FE discretisation applied to equation 7.75 for $r = 0$ and a constant p. Consider a mesh consisting of n equally spaced intervals.

The lesson is over, the time is up.

Notes

1 Both conditions are perfectly compatible with functions in $H^2(\Omega)$ (see e.g. [126]).

2 A **multiscale** approach could yield better results, using for instance **subscales** (see e.g. [49]). These techniques are playing an increasingly important role in modern numerical simulation of certain phenomena, but their presentation has to be left to higher level studies.

3 The expression mass matrix is commonly used in the engineering community. It stems from the fact that this matrix applies to discretised time derivatives, which multiplied by a mass density accounts for the inertia of the medium under study. In fluid mechanics, this derivative is the first-order time derivative, while in solid mechanics it is usually a second-order time derivative. On the other hand, in solid mechanics, the matrix related to body's deformation, that is, to terms involving second- or fourth-order spatial partial derivatives depending on the model, is called the **stiffness matrix**.