first-order equation (equation (3.6)) in Section 3.2, since it gives insight on the proper way to handle equation (3.3) and the equivalent first-order system (3.5).

Incidentally, we note that equation (3.6) has a unique solution when suitable initial and boundary conditions are prescribed. In Section 3.2, we will consider the following ones ensuring problem well-posedness:

Initial and boundary conditions for the transport equation

$$\boxed{\begin{array}{l} u(x,0) = u_0(x)\ \forall x \in [0,L] \text{ and} \\ u(0,t) \ = 0\ \forall t \in (0,\Theta], \text{if } o > 0, \text{or} \\ u(L,t) = 0\ \forall t \in (0,\Theta], \text{if } o < 0. \end{array}} \tag{3.7}$$

Before pursuing, we should emphasise that the vibration of a string is far from being the only application of equation (3.3). For instance, it also governs the phenomenon of wave propagation. For this reason, it is also known as the **wave equation**. For more information on hyperbolic PDEs we refer to reference [109].

Chapter outline: In Section 3.1, we deal with the space–time discretisation of equations (3.1)–(3.2) , while in Section 3.2 we do the same for equations (3.6)–(3.7). In Sections 3.3 and 3.4, we study the stability and the convergence of the corresponding numerical schemes, respectively. Some complements on the equation of the vibrating string are addressed in Section 3.5, where a numerical example is also given.

3.1 Numerical Solution of the Heat Equation

Let us first consider the following semi-discretisation in space of equations (3.1)–(3.2). Given the equally spaced FD grid $\mathcal{G} = [0, h, 2h, \ldots, (n-1)h, nh = L]$, we denote by $\hat{u}_i(t)$ an FD approximation of the solution u at the grid point $x = ih$, $1 \leq i \leq n-1$, at time t . These approximations are supplemented with the exact values $\hat{u}_0(t) := a(t)$ and $\hat{u}_n(t) := b(t)$. Then, like in Section 1.2, we approximate $\partial_{xx} u$ at the grid point $x = ih$ for every t by the FD $[2\hat{u}_i(t) - \hat{u}_{i-1}(t) - \hat{u}_{i+1}(t)]/h^2$. Naturally enough, equation (3.1) at point (ih, t) is approximated by

$$\hat{u}_i'(t) + p[2\hat{u}_i(t) - \hat{u}_{i-1}(t) - \hat{u}_{i+1}(t)]/h^2 = f(ih,t)\ \forall t \in (0,\Theta], \text{for } i = 1, 2, \ldots, n-1, \quad (3.8)$$

where g' represents the first-order derivative of a function $g(t)$. Notice that equation (3.8) is nothing but a system of linear ODEs in terms of the independent variable t , for the $(n-1)$ -component unknown vector $\vec{\hat{u}}(t) := [\hat{u}_1(t), \hat{u}_2(t), \ldots, \hat{u}_{n-1}(t)]^T$, having $\vec{\hat{u}}(0) = [u_0(h), u_0(2h), \ldots, u_0(nh-h)]^T$ as the initial condition. Recalling the notations used in Chapter 1 for the equally spaced FDM, with a constant p and $q = 0$, in concise form this system writes

$$\begin{cases} [\vec{\hat{u}}(t)]' + A_h^+ \vec{\hat{u}}(t) = \vec{b}_h(t)\ \forall t \in (0,\Theta] \\ \text{with } \vec{\hat{u}}(0) = \vec{u}^0, \end{cases} \quad (3.9)$$

where $\hat{u}_0(t) = a(t)$, $\hat{u}_n(t) = b(t)$, $\vec{b}_h(t) := [f(h,t), f(2h,t), \ldots, f(nh-h,t)]^T$ and $\vec{u}^0 := [u_0(h), u_0(2h), \ldots, u_0(nh-h)]^T$. Notice that matrix A_h^+ is similar but not identical to the $n \times n$ matrix introduced in Section 1.2. In order to incorporate the Dirichlet boundary conditions at $x = 0$ and $x = L$, in equation (3.9) A_h^+ is an $(n-1) \times (n+1)$ matrix, whose columns are numbered from 0 through n . The column numbered as the j th column, for j between one and $n-1$, is the same as the one of the matrix corresponding to the FD discretisation of (P_1). The 0th and the n th columns in turn are the vectors of $\Re^{n-1}$ $[-p/h^2, 0, \ldots, 0]^T$ and $[0, \ldots, 0, -p/h^2]^T$, respectively.

In general, an analytic solution to system (3.9) cannot be found. Therefore, we endeavour to solve it numerically as well. As the reader certainly knows, there are countless efficient methods for the numerical solution of initial value first-order systems of ODEs, such as the Euler methods, Runge–Kutta methods, and multistep methods. The literature on the subject is consequently profuse, and if we are to give just one reference we could cite reference [84]. However, here the best choices are those that, while being simple to implement, provide an accuracy compatible with the method employed for the space discretisation. For this reason, many users choose **Euler methods of the first order**, enabling either an **explicit** or an **implicit** solution. Another technique widely in use is the **Crank–Nicolson method**. It is of the second-order in time and gives rise to an implicit resolution. We will exploit these three possibilities hereafter.

3.1.1 Implicit Time Discretisation

Let Θ be finite and l be an integer strictly greater than one. We define a **time step** $\tau = \Theta/l$, with which we associate a (time) FD grid $\mathcal{H} = [0, \tau, 2\tau, \ldots, l\tau = \Theta]^T$. Now, for every $i \in \{1, 2, \ldots, n-1\}$ and for $k \in \{1, 2, \ldots, l\}$, we define an approximation of the time derivative of $\hat{u}_i$ at a certain time $t \in [(k-1)\tau, k\tau]$ to be specified for each method, by the FD:

$$D_i^k(u) := \frac{\hat{u}_i(k\tau) - \hat{u}_i([k-1]\tau)}{\tau}. \tag{3.10}$$

However, since the vector $\vec{\hat{u}}(t)$ is not known, here again the above definition is useless for practical purposes. Instead, we consider that we wish to determine approximations u_i^k of $\hat{u}_i(k\tau)$, and hence of $u(ih, k\tau)$, which satisfy a time-discrete analog of equation (3.9). We consider two possibilities in this subsection.

In the first method, we assign to the FD (equation (3.10)) the role of approximating the time derivative at $t = k\tau$. This results in a fully discretised **implicit** FD scheme to approximate equations (3.1)–(3.2), known as

The Backward Euler scheme (BES)

$$\boxed{\begin{array}{l} \frac{u_i^k - u_i^{k-1}}{\tau} + p\frac{2u_i^k - u_{i-1}^k - u_{i+1}^k}{h^2} = f(ih, k\tau) \\ \text{for } k = 1, 2, \ldots, l \text{ and } i = 1, 2, \ldots, n-1, \\ \text{with } u_i^0 = u_0(ih) \text{ and } u_0^k = a(k\tau), u_n^k = b(k\tau). \end{array}} \tag{3.11}$$

Notice that the underlying problem to solve for each value of k is a SLAE whose vector of unknowns is $\vec{u}^k := [u_1^k, u_2^k, \ldots, u_{n-1}^k]^T$. Assuming at first that $a(t) = b(t) = 0 \ \forall t$, this vector is determined, taking as data both $\vec{b}_h^k := [f(h, k\tau), f(2h, k\tau), \ldots, f(nh - h, k\tau)]^T$ and $\vec{u}^{k-1}$. In compact form, using the $(n-1) \times (n-1)$ identity matrix I, $\vec{u}^k$ is seen to be the solution of the following system:

$$(I/\tau + A_h)\vec{u}^k = \vec{u}^{k-1}/\tau + \vec{b}_h^k. \tag{3.12}$$

where in this case $A_h = \{a_{i,j}\}$ is the $(n-1) \times (n-1)$ symmetric tridiagonal matrix, whose nonzero entries are given by

$$a_{i,i} = 2p/h^2 \text{ for } 1 \leq i \leq n-1 \; a_{i-1,i} = a_{i,i-1} = -p/h^2 \text{ for } 2 \leq i \leq n-1. \tag{3.13}$$

Another possibility is to consider that the FD (equation (3.10)) is an approximation of the time derivative at time $t = (k - 1/2)\tau$. Since we are neither defining nor computing approximations of $u(ih, [k - 1/2]\tau)$, the natural thing to do is to let the mean value $(u_i^{k-1} + u_i^k)/2$ play this role. Setting $f_i^k := f(ih, k\tau)$ for $k = 0, 1, \ldots, l$ and $i = 1, \ldots, n-1$, this leads to the following implicit fully discretised FD scheme to approximate equations (3.1)–(3.2), known as

The Crank–Nicolson scheme (CNS)

$$\begin{array}{l} \frac{u_i^k - u_i^{k-1}}{\tau} + p\frac{2u_i^{k-1/2} - u_{i-1}^{k-1/2} - u_{i+1}^{k-1/2}}{h^2} = f_i^{k-1/2} \\ \text{where } u_i^{k-1/2} := (u_i^{k-1} + u_i^k)/2 \text{ and } f_i^{k-1/2} := (f_i^{k-1} + f_i^k)/2, \\ \text{for } k = 1, 2, \ldots, l \text{ and } i = 1, 2, \ldots, n-1, \\ \text{with } u_i^0 = u_0(ih) \text{ and } u_0^k = a(k\tau), u_n^k = b(k\tau). \end{array} \tag{3.14}$$

The problem to solve for each value of k is again a SLAE, whose vector of unknowns is $\vec{u}^k := [u_1^k, u_2^k, \ldots, u_{n-1}^k]^T$. If we assume again that $a(t) = b(t) = 0 \; \forall t$, $\vec{u}^k$ is the solution of the following system:

$$[I/\tau + A_h/2]\vec{u}^k = [I/\tau - A_h/2]\vec{u}^{k-1} + \vec{b}_h^{k-1/2}. \tag{3.15}$$

where $\vec{b}_h^{k-1/2} := [f_1^{k-1/2}, f_2^{k-1/2}, \ldots, f_{n-1}^{k-1/2}]^T$. Equations (3.12) and (3.15) tell us that for implicit methods, such as those studied in this subsection, at every time step a SLAE has to be solved, but the corresponding matrices remain fixed. Therefore, they can be factorised once for all before the time marching starts. In addition to this advantage, these methods will be shown to be **unconditionally stable** in specific senses, which means that the corresponding schemes are stable whatever discretisation parameters h and τ we choose. This property broadly makes up for the intrinsically necessary computational effort.

Notice that the matrices in both equations (3.12) and (3.15) are tridiagonal, and therefore these systems can be easily solved by iterative methods and advantageously by direct methods, since their matrices can be factorised into the product of bidiagonal triangular matrices once for all at the first step.

The reader may examine as Exercise 3.1 the necessary modifications in both schemes written in matrix form (equations (3.12) and (3.15)), in order to accommodate inhomogeneous boundary conditions. As a guide she or he could use the $(n-1)\times(n+1)$ matrix A_h^+ defined at the beginning of this section.

3.1.2 Explicit Time Discretisation

A tempting alternative to the implicit methods considered in the previous subsection to solve the differential system in equation (3.8) or equivalently equation (3.9) is an **explicit resolution**. By such an expression we mean that, in contrast to the Backward Euler method and the Crank–Nicolson method, at every time step each component of the solution vector $\vec{u}^k$ is determined independently of the others. This can be achieved by considering that the FD (equation (3.10)) is an approximation of the time derivative at time $(k-1)\tau$. This leads to the following explicit fully discretised FD scheme to approximate equations (3.1)–(3.2), namely,

The Forward Euler scheme (FES)

$$\boxed{\begin{array}{l}\frac{u_i^k-u_i^{k-1}}{\tau}+p\frac{2u_i^{k-1}-u_{i-1}^{k-1}-u_{i+1}^{k-1}}{h^2}=f(ih,k\tau-\tau)\\ \text{for } k=1,2,\ldots,l \text{ and } i=1,2,\ldots,n-1,\\ \text{with } u_i^0=u_0(ih) \text{ and } u_0^k=a(k\tau),u_n^k=b(k\tau).\end{array}} \tag{3.16}$$

In the case of homogeneous boundary conditions, at every time step, the above relations in matrix form writes as

$$\vec{u}^k=[I-\tau A_h]\vec{u}^{k-1}+\tau\vec{b}_h^{k-1}, \tag{3.17}$$

where A_h is the same $(n-1)\times(n-1)$ matrix given by equation (3.13). Since there is no matrix multiplying the unknown vector $\vec{u}^k$ on the left side of equation (3.17), it can be determined component by component by sweeping the spatial grid points one after the other. Nevertheless, it would be surprising that such a method works so much better than its implicit counterparts, equations (3.11) and (3.14), without any disadvantage. As we show by means of the example below, there are indeed limits beyond which explicit methods are unreliable.

Let $a = 0$, $b = 0$, $f = 0$ and $u_0(x) = \sin(m\pi x/L)$, m being a non-negative integer. For such data, the solution of problems (3.1)–(3.2) is given by $u(x,t) = e^{-\lambda t}\sin(m\pi x/L)$, with $\lambda = p(m\pi/L)^2$ as one can easily check. Let us apply scheme (3.16) to approximate u . Noticing that the abscissae of the grid points are iL/n for $i = 0, 1, \ldots, n$, at the first time step we obtain

$$u_i^1 = \sin(m\pi i/n) + \tau p\{\sin[m\pi(i+1)/n] - 2\sin(m\pi i/n) + \sin[m\pi(i-1)/n]\}/h^2.$$

On the other hand, from well-known trigonometric identities,

$$\sin[m\pi(i+1)/n] + \sin[m\pi(i-1)/n] = 2\sin(m\pi i/n)\cos(m\pi/n).$$

Thus, $u_i^1 = \{1 - 2p\tau[1 - \cos(m\pi/n)]/h^2\}\sin(m\pi i/n)$, that is, $u_i^1 = \eta(h,\tau,m,L)u_i^0$ for $i = 1, 2, \ldots, n-1$, where $\eta(h,\tau,m,L) = 1 - 2p\tau[1 - \cos(mh\pi/L)]/h^2$.

Since η depends neither on i nor on k , by mathematical induction it follows that necessarily

$$u_i^k = [\eta(h,\tau,m,L)]^k u_i^0 \quad \forall i \in \{1, 2 \ldots, n-1\} \text{ and } \forall k \geq 1. \tag{3.18}$$

Now assuming that $m = n,\ \eta = 1 - 4p\tau/h^2$. Therefore, if $2p\tau/h^2 > 1$, as k varies, the absolute value of the numerical solution will increase indefinitely. This is an inadequate response, since if $f = 0$, the solution of the heat equation is supposed to decrease in absolute value as time increases, as pointed out hereafter. To conclude these preliminary stability considerations, we should add that such a choice of u_0 is not as restrictive as it may seem. This is because every u_0 such that $u_0(0) = u_0(L) = 0$ can be expanded into an absolutely convergent **Fourier series** of the form:

$$u_0(x) = \sum_{m=1}^{\infty} c_m \sin(m\pi x/L)$$

for suitable real coefficients c_m (see e.g. [57]). By linearity, the numerical solution will be an absolutely convergent series whose terms are of the form given in equation (3.18), pre-multiplied by the c_m s. Therefore, if $|\eta| > 1$ for some values of m , with $n > 1$, the corresponding terms will gradually pollute the numerical solution as a whole–so much so that, as k increases, approximate solution absolute values will increase here and there, until the numerical solution becomes completely spoilt. On the other hand, if $|\eta| \leq 1$, these values will gradually decrease. For a detailed explanation about this process, we refer to reference [67].

Summarizing, like most explicit schemes for time integration of time-dependent PDEs, the Forward Euler scheme is only reliable if the time step is bounded above by a constant multiplied by the square of the spatial grid size. Actually, there is an explanation for this, in connection with the expected behaviour of a solution to the heat equation. We will see all this in more detail in the next subsection.

3.1.3 Example 3.1: Numerical Behaviour of the Forward Euler Scheme

In this example, we solve the heat equation by the FES (equation (3.17)), for the data $L = 1$, $p = 1$, $f = 0$, $u_0 = 1$, $a = 0$ and $b = 0$. In Figures 3.1 and 3.2, we illustrate what can be expected from an explicit solution scheme depending on whether a stability condition is satisfied or not. They reflect the numerical solution of the heat equation with a very coarse grid. More precisely, we let k vary from 1 up to 4 taking $h = 1/4$. In Figure 3.1a and 3.1b, we illustrate the numerical solution for τ equal to $1/8$ and $1/16$, respectively, while in Figure 3.2a and 3.2b, we do the same for τ equal to $1/32$ and $1/64$, respectively. As one can see, Figure 3.1 exhibits unstable numerical results, whereas in Figure 3.2 stable responses are depicted. In order to better understand the ongoing processes for these four different values of the time step τ, the curious reader could figure out by her or himself the following behaviours, by pushing the calculations two or three steps further:

- $\tau = 1/8 \rightarrow$ widely oscillating solution;
- $\tau = 1/16 \rightarrow$ moderately oscillating solution;
- $\tau = 1/32 \rightarrow$ fast decreasing solution;
- $\tau = 1/64 \rightarrow$ slowly decreasing solution.

As we will see in Section 3.3, the solution of the heat equation satisfies a **maximum principle** (cf. [156]). In the case where $f = 0$ and $u_0 > 0$, like in the above example, this implies that the temperature decreases until it attains a zero limiting value everywhere, in principle at infinite time. In other words, this means that the final temperature will be the one at which both ends of the bar are maintained[1]. Only smaller values of τ reproduce such a mathematically and physically acceptable temperature evolution. Notice, however, that the smaller the τ, the slower the temperature will decrease, which is logical since larger values of τ correspond to time discretisations not so close to reality.

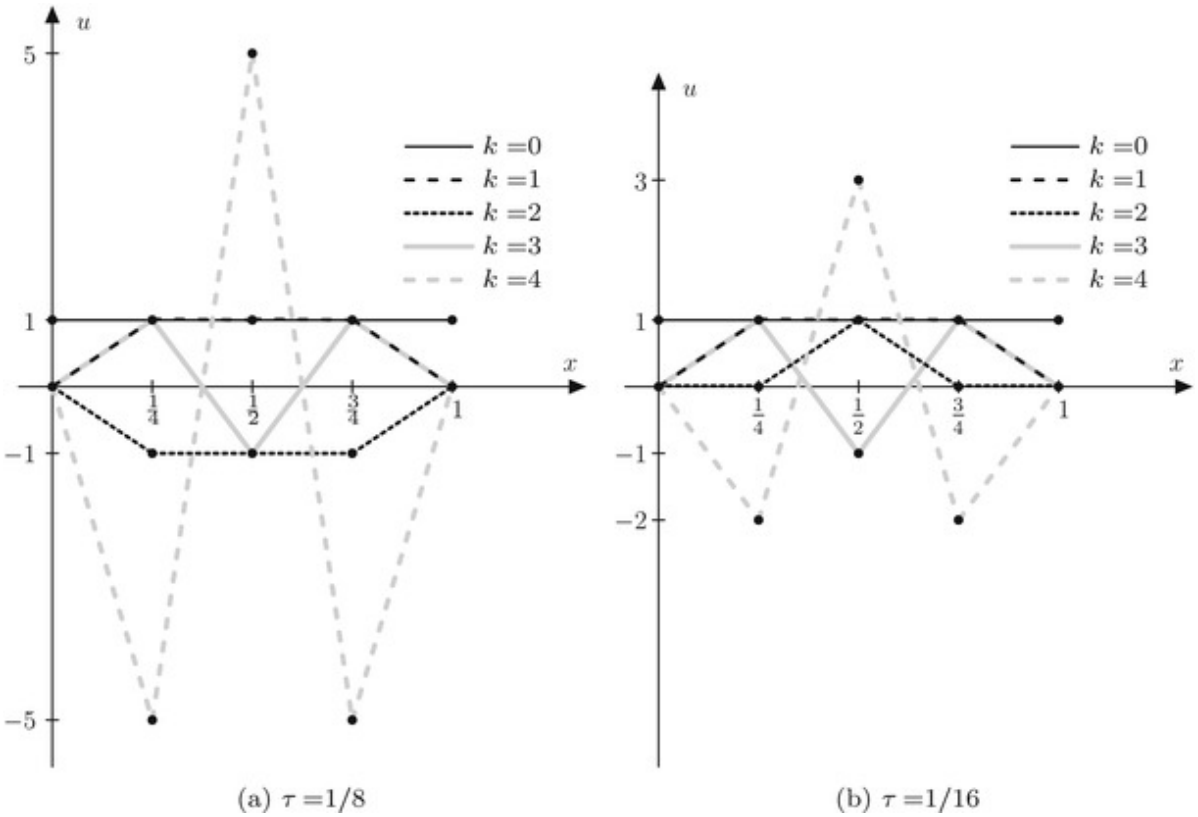

Figure 3.1 Unstable solution of the heat equation by the Forward Euler scheme for $h = 1/4$

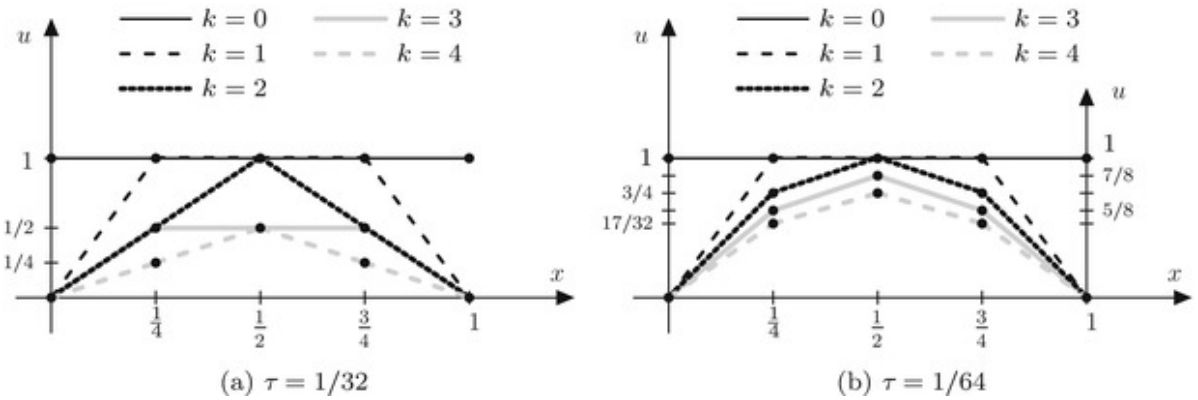

Figure 3.2 Stable solution of the heat equation by the Forward Euler scheme for $h = 1/4$

Summarizing, we have showed in this example that scheme (3.17) is unreliable if $\tau > h^2/(2p)$. The other way around, according to the study to be conducted in Section 3.3, it is always reliable as long as τ is bounded by $h^2/(2p)$. However, this implies that the time step must be very small if one uses fine spatial grids. As a consequence, in this case the numerical resolution will advance very slowly in time. As previously announced, this is the price to pay for using an explicit scheme.

3.2 Numerical Solution of the Transport Equation

From the numerical point of view, considering equations (3.6)–(3.7) will definitively lead to conclusions typical of hyperbolic equations, including second-order ones, such as equations (3.3)–(3.4). That is why we confine ourselves to studying the (first-order) transport equation in its simplest form. This equation carries this name because it governs the transport with velocity o of a