Contents

Chapter 1

Geometric integration

In the previous chapter, we considered time discretization of the scalar linear equation

$$y' = \lambda y, \qquad \lambda \in \mathbb{C}^-, \qquad \text{(i.e., } \mathrm{Re}(\lambda) < 0\text{)}. \tag{1.1}$$

In that case, we know that the equation is dissipative, i.e.,

$$\lim_{t\to\infty} y(t) = 0.$$

We showed that all time discretization methods that we discussed so far are able to satisfy a similar criterion for the time discretized solution

$$\lim_{n\to\infty} y^n = 0, \tag{1.2}$$

in which $y^n \approx y(t^n)$ and $t^n = n \cdot h$, with h the step size of the time discretization method. We say that time integration is *stable* when(1.2) is satisfied. (This is the case when the quantity $h\lambda$ is contained within the stability region $\mathcal{D}$ of the method.) Different methods have different stability regions. For many methods, ensuring stability is only possible by requiring that the step size h is small enough, and we studied in the previous chapter how the stability regions of the method affects the choice of suitable time steps.

In particular, we studied *stiff systems* of differential equations, in which there are multiple time scales, corresponding to multiple values of λ of (very) different order of magnitude. In stiff systems, there are some fast time scales that are (very) quickly damped, such that, after a very short time, they are no longer present in the system. Simultaneously, there are some slow time scales that describe the evolution of the system on longer time intervals. The slow modes need to integrated accurately, whereas, for the fast modes, we only care about quick damping: for the fast time scales (corresponding to $\lambda \in \mathbb{C}^-$ and $|\lambda| \gg 1$), we only want condition (1.2) to be satisfied.

This desire led to the definition of A-stability, which is a desirable property of a numerical method when the problem is stiff, because it allows to choose

the time step h solely based on accuracy considerations for the slow part of the solution. Most importantly, we argued (but did not demonstrate rigorously) that, to study time discretization methods for stiff systems, a *linear* stability analysis for equation (1.1) is sufficient, because, for a nonlinear stiff system, a local linearization can be performed at every time step.

In this chapter, we will study another class of problems for which one needs to be careful in the selection of a suitable time integration method: problems with conserved quantities. To set the stage, we first consider a linear harmonic oscillator in Section 1.1 and relate the behaviour of some common time discretization methods for this equation to the linear stability analysis of the previous chapter. We then generalize to nonlinear Hamiltonian problems in Section 1.2 and show how some, but not all, conclusions carry over to the nonlinear case. It will turn out that Hamiltonian systems have a special (*symplectic*) structure, which we will need to conserve during time discretization. We conclude in Section 1.3 with an introduction to symplectic time discretization schemes.

1.1 The harmonic oscillator

1.1.1 Model problem and properties

1.1.1.1 Differential equation and exact solution

Let us consider numerical time integration of the following system of differential equations, corresponding to a harmonic oscillator:

$$y_1' = y_2, \qquad y_2' = -y_1, \tag{1.3}$$

or, equivalently,

$$\boldsymbol{y}' = A\boldsymbol{y}, \qquad \text{with } A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \tag{1.4}$$

with initial condition $\boldsymbol{y}(0) = \boldsymbol{y}_0$.

Since the eigenvalues of the matrix A are $\lambda_{1,2} = \pm i$, the exact solution of (1.4) can be written as

$$\boldsymbol{y}(t) = c_1 \exp(it) \begin{bmatrix} 1 \\ i \end{bmatrix} + c_2 \exp(-it) \begin{bmatrix} 1 \\ -i \end{bmatrix}, \tag{1.5}$$

with c_1 and c_2 such that the initial conditions are satisfied. In particular, when choosing $y_1(0) = 1$ and $y_2(0) = 0$, the exact solution is given as

$$y_1(t) = \cos(t), \qquad y_2(t) = -\sin(t). \tag{1.6}$$

1.1.1.2 Conservation properties

For the harmonic oscillator, we see that an undamped oscillation (with constant amplitude) results as the exact solution. This implies that, in the (y_1, y_2) phase

plane, the solution lies on a circle with center 0 and radius that depends on the initial condition. Thus, we have that, for all $t > 0$,

$$H(y_1, y_2) = \frac{1}{2}\left((y_1(t))^2 + (y_2(t))^2\right) = \frac{1}{2}\left((y_1(0))^2 + (y_2(0))^2\right) = c, \quad (1.7)$$

in which we have introduced the symbol H to denote the conserved quantity. (The reason for choosing the symbol H and introducing the factor $1/2$ we will become clear in section 1.2.) To see that (1.7) is true, we compute

$$\frac{dH}{dt} = \frac{\partial H}{\partial y_1} \cdot y_1' + \frac{\partial H}{\partial y_2} \cdot y_2' = y_1 \cdot y_1' + y_2 \cdot y_2' = y_1 \cdot y_2 - y_2 \cdot y_1 = 0.$$

in which we first used the chain rule, then the definition of H in (1.7), and finally the definition of the harmonic oscillator (1.3).

1.1.1.3 Conservation and symmetry

The above conservation property can also be derived in matrix notation, which sheds some additional light on the underlying structure of equation (1.4) from which it results. To this end, observe that $H(y_1, y_2) = \boldsymbol{y}^T\boldsymbol{y}/2$, and that (1.7) implies that

$$\frac{dH}{dt} = \frac{1}{2}\boldsymbol{y}^T \cdot \frac{d\boldsymbol{y}}{dt} + \frac{1}{2}\left(\frac{d\boldsymbol{y}}{dt}\right)^T \cdot \boldsymbol{y} = 0. \quad (1.8)$$

To show in an alternative way why (1.8) is true, we write

$$\begin{aligned}\frac{dH}{dt} &= \frac{1}{2}\boldsymbol{y}^T \cdot \frac{d\boldsymbol{y}}{dt} + \frac{1}{2}\left(\frac{d\boldsymbol{y}}{dt}\right)^T \cdot \boldsymbol{y} \\ &= \frac{1}{2}\boldsymbol{y}^T \cdot A\boldsymbol{y} + \frac{1}{2}y^T A^T \cdot \boldsymbol{y} = 0.\end{aligned}$$

In the last line, we have use a crucial property of the harmonic oscillator that results in the conservation of the quantity H, namely the skew-symmetry of the matrix A:

$$A^T = -A. \quad (1.9)$$

Thus, the above derivation holds for *any* linear system with skew-symmetric system matrix A.

1.1.2 Forward and backward Euler integration

1.1.2.1 Numerical experiments

Forward Euler integration Let us first consider a forward Euler integration of equation (1.3) (or, equivalently, (1.4)):

$$\begin{cases} y_1^{n+1} &= y_1^n + hy_2^n \\ y_2^{n+1} &= y_2^n - hy_1^n \end{cases}, \qquad \text{or also} \qquad \boldsymbol{y}^{n+1} = (I + h \cdot A)\boldsymbol{y}^n. \quad (1.10)$$

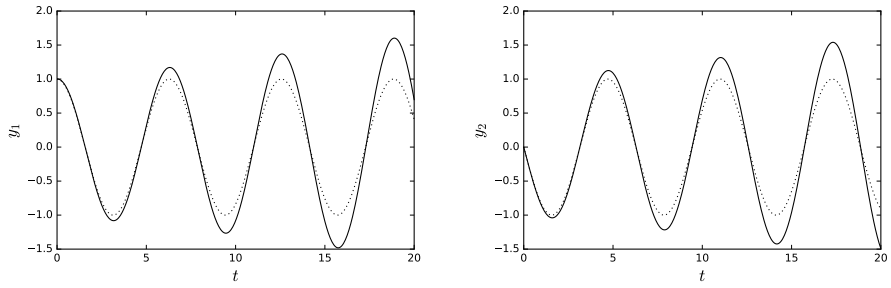

Figure 1.1: Forward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) as a function of time, using $h = 5 \cdot 10^{-2}$.

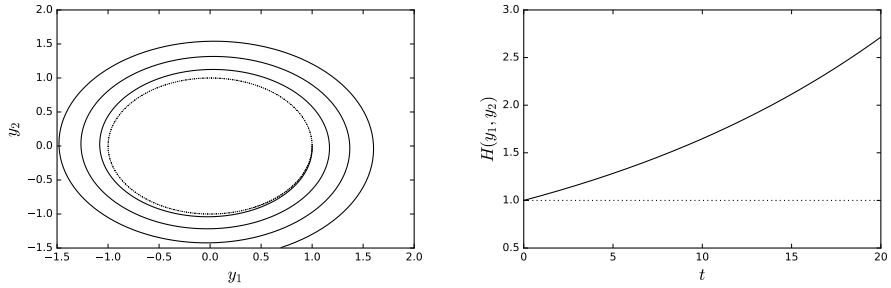

Figure 1.2: Left: Forward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) in the (y_1, y_2) phase plane, using $h = 5 \cdot 10^{-2}$. Right: Evolution of the quantity $H(y_1, y_2)$ as a function of time for the forward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3).

We perform a numerical simulation of equation (1.3) with $h = 5 \cdot 10^{-2}$ on the time interval $[0, 20]$. The results are shown in Figure 1.1. We clearly see that the amplitude of the forward Euler solution increases as a function of time, whereas the amplitude of the exact solution remains constant (as expected). Figure 1.2 gives a different view on the same observation. On the left, we see a phase space view of trajectories. The exact solution is periodic and manifests itself as a circle in this phase space view, whereas the forward Euler solution spirals outwards. On the right, we see the conserved quantity $H(y_1, y_2)$, defined in equation (1.7), as a function of time. Clearly, the forward Euler method does not conserve $H(y_1, y_2)$; instead, $H(y_1, y_2)$ increases as a function of time.

Backward Euler integration Now, we consider a backward Euler integration of equation (1.3) (or, equivalently, (1.4)):

$$\begin{cases} y_1^{n+1} &= y_1^n + h y_2^{n+1} \\ y_2^{n+1} &= y_2^n - h y_1^{n+1} \end{cases}, \qquad \text{or also} \qquad \boldsymbol{y}^{n+1} = (I - h \cdot A)^{-1} \boldsymbol{y}^n. \tag{1.11}$$

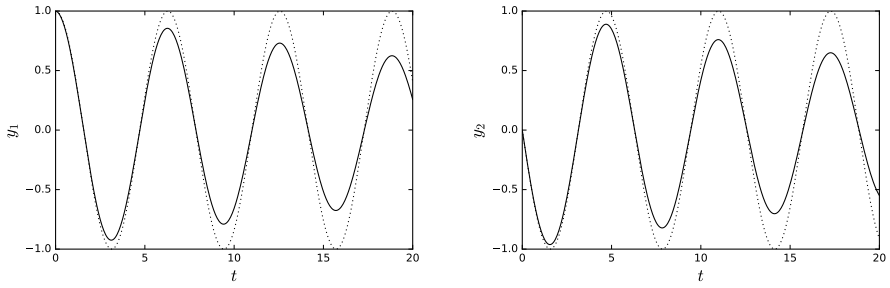

Figure 1.3: Backward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) as a function of time, using $h = 5 \cdot 10^{-2}$.

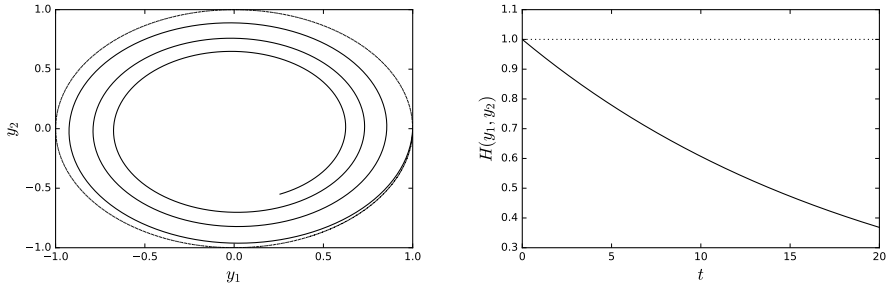

Figure 1.4: Left: Backward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) in the (y_1, y_2) phase plane, using $h = 5 \cdot 10^{-2}$. Right: Evolution of the quantity $H(y_1, y_2)$ as a function of time for the backward Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3).

We again perform a numerical simulation of equation (1.3) with $h = 5 \cdot 10^{-2}$ on the time interval $[0, 20]$. The results are shown in Figure 1.3. In contrast with the forward Euler integration, we now clearly see that the amplitude of the backward Euler solution *decreases* as a function of time. Figure 1.4 gives a different view on the same observation. In the phase space view of trajectories, the backward Euler solution spirals inwards; the backward Euler method also does not conserve $H(y_1, y_2)$, which now diminishes as a function of time.

1.1.2.2 A linear analysis

From the above simulations, we conclude that both the forward and the backward Euler method exhibit behavior that is qualitatively different from that of the exact solution. Instead of a periodic solution, they result in an oscillation that is either excited (forward Euler) or damped (backward Euler). It should be stressed that these issues *do not imply* that the methods are not convergent. We have proved convergence of the forward and backward Euler methods for general

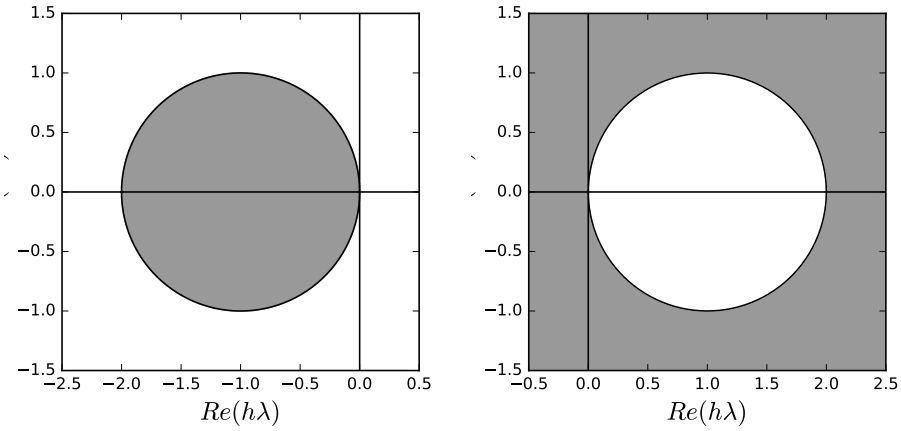

Figure 1.5: Left: stability region of the forward Euler method. Right: stability region of backward Euler method. The stability domain is marked in grey.

equations. Obviously, this proof is also valid for the harmonic oscillator (1.3). And indeed, if we let the step size h tend to zero, the numerical solution will get closer and closer to the exact solution. Nevertheless, the problem shown above exists *for any finite value of the step size* h (and for many more equations than the linear harmonic oscillator!). It becomes particularly relevant if one is interested in time integration over long time intervals, because, no matter how small h is chosen, the problem will always be too large to ignore as soon as one performs a long enough time integration.

To get more insight in the reasons behind this undesired behaviour, we return to the linear stability analysis that we have performed in the previous chapter. Figure 1.5 shows again the linear stability regions of the forward and backward Euler methods. Given that the eigenvalues of the matrix A in (1.4) are $\lambda = \pm i$, we need to look at values $h\lambda$ that are on the imaginary axis. For the forward Euler method, these values of $h\lambda$ are *outside* of the stability region, regardless of the value of h. This explains the growth of the amplitude of the oscillations in figures 1.1 and 1.2. For the backward Euler method, these values of $h\lambda$ are *inside* the stability regions. The harmonic oscillator thus experiences numerical damping due to the time discretization, as observed in figures 1.3 and 1.4.

1.1.3 Trapezoidal and implicit midpoint methods

1.1.3.1 Numerical experiments

From the previous experiments, we may infer that, to preserve the qualitative behaviour of the harmonic oscillator, one should choose a time discretization scheme for which the imaginary axis is on the boundary of the stability domain. We have encountered two such methods in the previous chapter, namely the

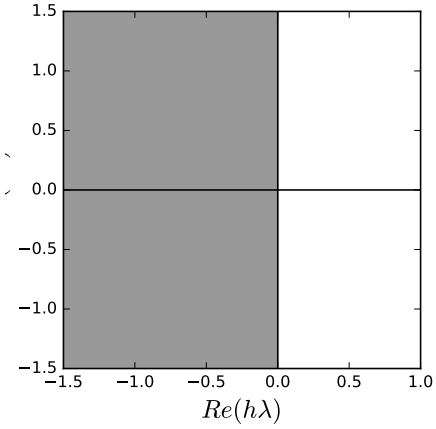

Figure 1.6: Stability region of the trapezoidal and implicit midpoint methods.

trapezoidal rule and the implicit midpoint rule. (See figure 1.6 for a reminder of the stability region of the trapezoidal and implicit midpoint rules.)

It is easy to check that the implicit midpoint rule and the trapezoidal rule are identical for linear ODEs, see also the Appendix. In this section, we continue with the trapezoidal rule, which reads:

$$\boldsymbol{y}^{n+1} = \boldsymbol{y}^n + h\left(\frac{1}{2}A\boldsymbol{y}^n + \frac{1}{2}A\boldsymbol{y}^{n+1}\right), \tag{1.12}$$

or, equivalently,

$$\boldsymbol{y}^{n+1} = \left(I - \frac{h}{2}\cdot A\right)^{-1}\left(I + \frac{h}{2}\cdot A\right)\boldsymbol{y}^n. \tag{1.13}$$

The resulting numerical solution can then be seen to satisfy:

$$\boldsymbol{y}^n = c_1\left(\frac{1+hi/2}{1-hi/2}\right)^n\begin{bmatrix}1\\ i\end{bmatrix} + c_2\left(\frac{1-hi/2}{1+hi/2}\right)^n\begin{bmatrix}1\\ -i\end{bmatrix}, \tag{1.14}$$

with c_1 and c_2 such that the initial conditions are satisfied. It can easily be verified that

$$\left|\frac{1+hi/2}{1-hi/2}\right| = 1,$$

regardless of h and therefore, indeed, no damping or excitation should take place. To see this, we go to the exponential form:

$$\frac{1+hi/2}{1-hi/2} = \frac{(1+hi/2)^2}{(1-hi/2)(1+hi/2)} = \frac{1-h^2/4}{1+h^2/4} + i\frac{h}{1+h^2/4} = r_h \exp i\theta_h, \tag{1.15}$$

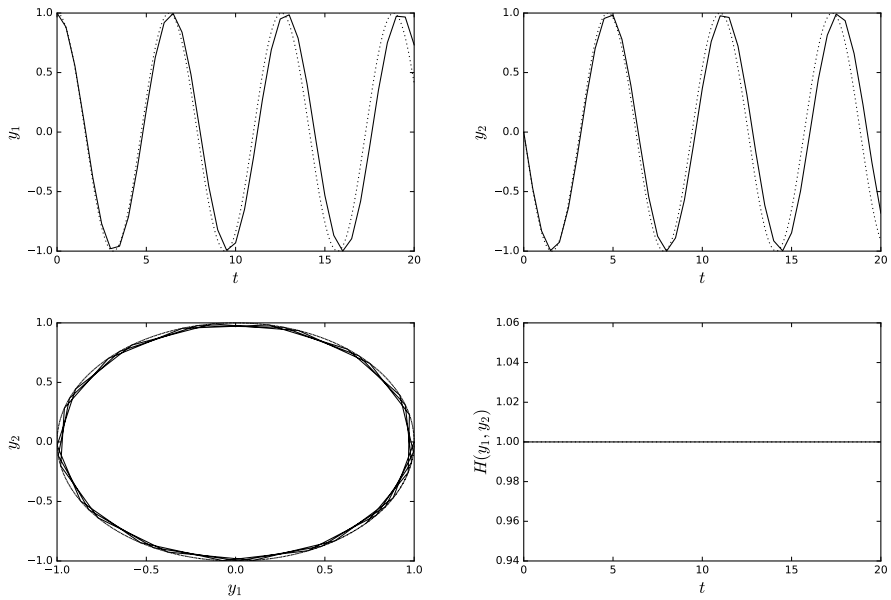

Figure 1.7: Top: Trapezoidal (solid) and exact solution (dotted) of the harmonic oscillator (1.3) as a function of time, using $h = 5{\cdot}10^{-1}$. Bottom left: Trapezoidal (solid) and exact solution (dotted) of the harmonic oscillator (1.3) in the (y_1, y_2) phase plane, using $h = 5 \cdot 10^{-1}$. Bottom right: Evolution of the quantity $H(y_1, y_2)$ as a function of time for the trapezoidal (solid) and exact solution (dotted) of the harmonic oscillator (1.3).

with $r = 1$ and

$$\theta_h = \tan^{-1}\left(\frac{h}{1 - h^2/4}\right). \tag{1.16}$$

A similar calculation can be done for the second amplification factor in (1.14).

To verify this, we again perform a numerical simulation of equation (1.3), this time with $h = 5 \cdot 10^{-1}$, on the time interval $[0, 20]$. (If we would use $h = 5 \cdot 10^{-2}$ as before, the exact and numerical solution would be indistinguishable on the plot.) The results are shown in Figure 1.7. We indeed observe that the numerical solution is not damped and the quantity $H(y_1, y_2)$ is conserved exactly by the scheme.

A second observation is that the period of the oscillation is slightly altered by the time discretization. This can be understood by looking back at the amplification factors of the trapezoidal rule in exponential form (1.15). Let us for simplicity compare the exact solution (1.5) with the numerical solution (1.14) for an initial condition $\boldsymbol{y}_0$ that corresponds to $c_1 = 1$ and $c_2 = 0$. (The argument can easily repeated in the general case, at the expense of some extra lines of text.) In this case, we know that the exact solution after time integration over

a time interval of size h is given by $\boldsymbol{y}(h) = \exp(ih)\boldsymbol{y}_0$, whereas one step with the trapezoidal rule results in

$$\boldsymbol{y}^1 = \exp(i\theta_h)\boldsymbol{y}_0,$$

with θ_h in (1.16). A Taylor expansion yields

$$\begin{aligned}\theta_h &= \tan^{-1}\left(\frac{h}{1-h^2/4}\right)\\ &= \frac{h}{1-h^2/4} - \frac{1}{3}\left(\frac{h}{1-h^2/4}\right)^3 + \frac{1}{5}\left(\frac{h}{1-h^2/4}\right)^5 + \text{h.o.t.},\end{aligned}$$

in which "h.o.t." stands for higher order terms. Further using

$$\frac{h}{1-h^2/4} = h(1 + h^2/4 + (h^2/4)^2 + \text{h.o.t.}),$$

we see that the time integration scheme results in a rotation on the unit circle with a slightly different frequency $\theta_h = h + O(h^3)$ as the exact solution.

1.1.3.2 Conservation and symmetry

It will become clear in the next section that this argument, based on a linear stability analysis, is not sufficient to understand the non-linear setting. Instead, we will need to look directly into the conservation of the quantity $H(y_1, y_2)$. Let us therefore also prove the conservation of $H(y_1, y_2)$ directly, without using the linear stability region. We start by introducing the following shorthand notation for the trapezoidal rule:

$$\boldsymbol{y}^{n+1} = \Phi_h \boldsymbol{y}^n, \qquad \Phi_h = \left(I - \frac{h}{2}A\right)^{-1}\left(I + \frac{h}{2}A\right). \tag{1.17}$$

Observe that we have the following property:

$$\Phi_h^{-1} = \Phi_{-h}, \tag{1.18}$$

i.e., the inverse of taking a step with step size h is identical to taking a (backward) step with size $-h$. We call time integration methods that satisfy this property *symmetric*. It is clear that any time-continuous problem is symmetric (integrating backward in time simply means retracing your steps). Equation (1.18) shows that the trapezoidal rule is symmetric. (This can also be checked for the general nonlinear case.) However, not every time discretization method is symmetric. The forward and backward Euler methods, for instance, are not.

We now show that symmetry of the time discretization, combined with the skew-symmetry of the matrix A in (1.8), implies that a time discretization method conserves $H = \boldsymbol{y}^T\boldsymbol{y}$. We have:

$$\left(\boldsymbol{y}^{n+1}\right)^T \boldsymbol{y}^{n+1} = \left(\boldsymbol{y}^n\right)^T \Phi_h^T \Phi_h \boldsymbol{y}^n.$$

We thus only need to show that

$$\Phi_h^T = \Phi_h^{-1} = \Phi_{-h}. \tag{1.19}$$

By writing

$$\Phi_h^T = \left(I + \frac{h}{2}A\right)^T \left(I - \frac{h}{2}A\right)^{-T},$$

and subsequently using the skew-symmetry of A, we get

$$\Phi_h^T = \left(I - \frac{h}{2}A\right) \left(I + \frac{h}{2}A\right)^{-1},$$

from which (1.19) follows by realizing that the two matrices in the product commute.

1.1.4 The symplectic Euler method

In the previous sections, we have shown how the forward and backward Euler method fail to capture the qualitative behaviour of the harmonic oscillator. We have related these observations to the shapes of the linear stability domains of these methods. We then proceeded to the trapezoidal and implicit midpoint rules and showed that these methods are able to capture the correct qualitative behaviour. This follows from a study of the linear stability domains and from a direct study of the conservation properties of the methods.

At the end of section 1.1.3, we indicated that the symmetry property (1.18) allowed us to prove exact conservation of the quantity H. However, both the trapezoidal rule and the implicit midpoint rule are second order methods. It is therefore tempting to think that the superiority of the trapezoidal and implicit midpoint rules can be attributed to their higher order accuracy. This is not true! To conclude this section on the harmonic oscillator, we therefore consider one additional method that is only of first order but nevertheless has an interesting conservation property: the *symplectic Euler method*. We will show that this method also captures the correct qualitative behaviour, such that we can be convinced that the conservation properties of these methods are more important than their order.

The symplectic Euler method for the harmonic oscillator (1.3) is given as:

$$\begin{cases} y_1^{n+1} &= y_1^n + h y_2^n \\ y_2^{n+1} &= y_2^n - h y_1^{n+1} \end{cases}. \tag{1.20}$$

Note that is is not a classical method, in the sense that we treat the different components of the system (the variables y_1 and y_2) differently. Let us look at this method in more detail. We can make the following observations:

- The variable y_2 is treated explicitly and the variable y_1 implicitly – the method is *semi-implicit*;

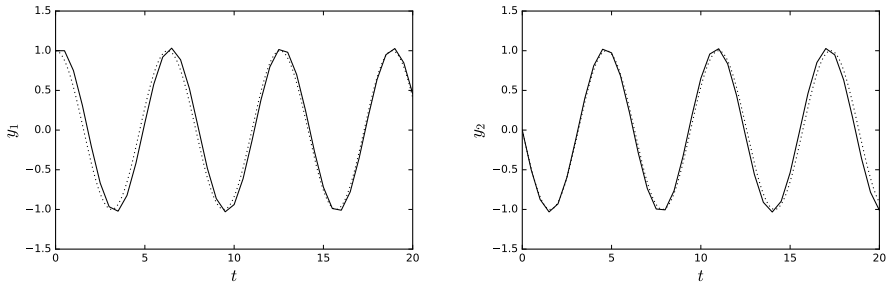

Figure 1.8: Symplectic Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) as a function of time, using $h = 1 \cdot 10^{-1}$.

- The first equation is solved using forward Euler, the second using backward Euler – the method is of order one;
- Since y_1^{n+1} is solved explicitly first, and used in the second step to compute y_2^{n+1} – the semi-implicit method can be implemented as an explicit one.

Let us now look at the numerical behaviour of the symplectic Euler method. We again perform a numerical simulation of equation (1.3), this time using $h = 1 \cdot 10^{-1}$, on the time interval $[0, 20]$. The results are shown in Figure 1.8. We again observe that, while the solution is less accurate than with the second order trapezoidal and implicit midpoint rules, the periodicity of the solution is qualitatively preserved. Figure 1.9 gives a different view on the same observation. Note, however, the curious evolution of the quantity H. The symplectic Euler method does not preserve H exactly, nor does it systematically decrease or increase H. Instead, it appears that the quantity H itself also evolves periodically! We will postpone a discussion of why this occurs to a later section.

1.2 Hamiltonian systems

In this section, we will generalize our observations from section 1.1 to general Hamiltonian systems. We define Hamiltonian systems in section 1.2.1, and turn to a more detailed study of their geometrical structure in section 1.2.2. This will lead to the concept of symplecticity, of which we discuss the importance in more detail in section 1.2.3.

1.2.1 Definition of Hamiltonian systems

Hamiltonian systems are a special class of ODEs for the unknowns $\boldsymbol{p}$ and $\boldsymbol{q}$ with the following equations of motion:

$$\boldsymbol{p}' = -\frac{\partial H(\boldsymbol{p}, \boldsymbol{q})}{\partial \boldsymbol{q}}, \qquad \boldsymbol{q}' = \frac{\partial H(\boldsymbol{p}, \boldsymbol{q})}{\partial \boldsymbol{p}}, \tag{1.21}$$

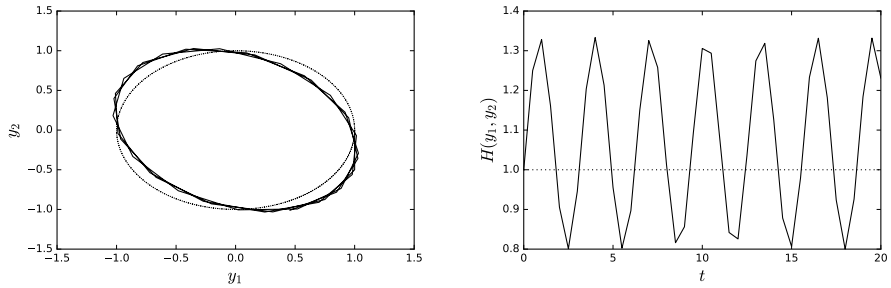

Figure 1.9: Left: Symplectic Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3) in the (y_1, y_2) phase plane, using $h = 5 \cdot 10^{-2}$. Right: Evolution of the quantity $H(y_1, y_2)$ as a function of time for the symplectic Euler (solid) and exact solution (dotted) of the harmonic oscillator (1.3).

for $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{R}^d$.

These equations of motion ensure that the quantity $H(\boldsymbol{p}, \boldsymbol{q})$ is constant along solution trajectories, as can be seen from the following simple computation (compare (1.8)):

$$\begin{aligned} \frac{dH(\boldsymbol{p},\boldsymbol{q})}{dt} &= \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{p}}^T \boldsymbol{p}' + \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}}^T \boldsymbol{q}' \\ &= -\frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{p}}^T \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}} + \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}}^T \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{p}} = 0 \end{aligned}$$

While their structure might seem very specific, equations of the form (1.21) often appear in practice. We give a few examples:

Example 1.1 (Pendulum)**.** Consider a pendulum with length $\ell = 1$ and mass $m = 1$ attached to the ceiling at the origin and subject only to gravity. Denote furthermore by α the angle it makes with respect to the vertical axis. When introducing $q = \alpha$ and $p = \alpha'$ (in which we have omitted the boldface because p and q are scalars), we have

$$q' = p, \qquad p' = -\sin(q), \tag{1.22}$$

and, consequently,

$$H(p, q) = \frac{1}{2}p^2 - \cos(q), \tag{1.23}$$

in which we clearly recognize the contributions of kinetic and potential energy.

Example 1.2 (Interacting particles)**.** Consider a system of N particles in D spatial dimensions, with positions $\boldsymbol{q} \in \mathbb{R}^d$ and velocities $\boldsymbol{p} \in \mathbb{R}^d$, $d = N \cdot D$. Each particle moves according to Newton's laws of motion, i.e.,

$$\boldsymbol{q}' = \boldsymbol{p}, \qquad \boldsymbol{p}' = -\nabla_q V(\boldsymbol{q}), \tag{1.24}$$

which state that the acceleration of the particles is given by the gradient of a potential with respect to the positions. Think, for instance, of the gravitational force if the particles are stars or planets, or interatomic forces if they are atoms. We again recognize the contributions of kinetic and potential energy in the Hamiltonian:

$$H(\boldsymbol{p}, \boldsymbol{q}) = \frac{1}{2}\boldsymbol{p}^T\boldsymbol{p} + V(\boldsymbol{q}). \tag{1.25}$$

Note that the harmonic oscillator in the previous section is also an example of a Hamiltonian system when choosing $p = y_2$ and $q = y_1$ an defining the Hamiltonian $H(p, q) = (1/2)(p^2 + q^2)$.

It is clear from the above examples that we will often encounter situations with *separable* Hamiltonians of the form:

$$H(\boldsymbol{p}, \boldsymbol{q}) = T(\boldsymbol{p}) + V(\boldsymbol{q}). \tag{1.26}$$

It will turn out that this structure allows for some significant simplifications in the proposed numerical methods.

1.2.2 Symplecticity

Based on the linear example of section 1.1, one could be tempted to search for numerical methods that, just like the continuous dynamics, also preserve $H(\boldsymbol{p}, \boldsymbol{q})$ after time discretization, but this would be a litte too hasty. There are two reasons for this:

- Exact conservation of the Hamiltonian will, in general, not be possible – it will luckily not be necessary to obtain numerical solution with a desirable qualitative features;
- Hamiltonian systems have a much deeper and richer structure than simply conserving the quantity $H(\boldsymbol{p}, \boldsymbol{q})$ – a structure that we *will* be able to conserve, and that is so important that it deserves the special name *sympecticity*.

The first point might come as a surprise, but we have already seen indications that pointed in this direction. In section 1.1.4, we already observed that the symplectic Euler method did not exactly conserve $H(p, q)$ for the harmonic oscillator. Nevertheless, the numerical simulation captured nicely the qualitative behaviour of the exact solution. We are now almost ready to see why.

1.2.2.1 Flow maps and area preservation

To make the notation more concise, we write, from here on, $\boldsymbol{y} = (\boldsymbol{p}, \boldsymbol{q})$, $\nabla H(\boldsymbol{y}) = (\partial H(\boldsymbol{p}, \boldsymbol{q})/\partial \boldsymbol{p}, \partial H(\boldsymbol{p}, \boldsymbol{q})/\partial \boldsymbol{q})$ and

$$\boldsymbol{y}' = J^{-1}\nabla H(\boldsymbol{y}), \qquad J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \tag{1.27}$$

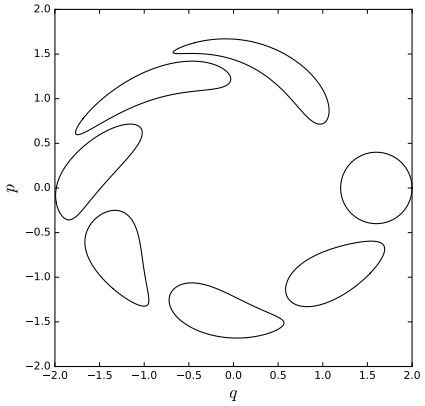

Figure 1.10: An initial disc Ω, centered at $(8/5, 0)$ with radius $2/5$, is mapped by unit time intervals with the flow map $\varphi_t(\Omega)$. All resulting regions have the same area.

Additionally, we also define the *flow map* $\varphi_t(\boldsymbol{y}_0)$ as the solution $\boldsymbol{y}(t; \boldsymbol{y}_0)$ of (1.27), starting from the initial condition $\boldsymbol{y}_0$. We need this additional definition for two reasons. First, we will use it to describe the evolution of the volume that is covered by a set of initial conditions. Second, to define symplecticity in the next section, we will also need the Jacobian of the flow map $\varphi_y(\boldsymbol{y})$ with respect to the initial condition y.

We can extend the definition of a flow map from a specific initial condition $\boldsymbol{y}_0$ to measurable sets $\Omega \subset \mathbb{R}^d$ as follows:

$$\varphi_t(\Omega) = \{\boldsymbol{y}(t) : \boldsymbol{y}(0) \in \Omega\}\,. \tag{1.28}$$

Let us now look at the evolution of $\Omega = \left\{y \in \mathbb{R}^2 : (q - 8/5)^2 + p^2 \leq 4/25\right\}$ under the dynamics of the pendulum, equation (1.22). The results are shown in Figure 1.10: each closed curve forms the boundary of the set of states that the pendulum reaches, starting from an initial condition in Ω at $t = 0$, at times $t = 1$, $t = 2$, …, $t = 6$. One sees that these areas move clockwise and become increasingly distorted. The most important observation, however, is that their area does not change over time. This is how symplecticity manifests itself in $\mathbb{R}^2$: the flow map for Hamiltonian systems with $p, q \in \mathbb{R}$ is area-preserving.

1.2.2.2 Symplecticity in higher dimensions

One could suspect that, in higher dimensions, symplecticity manifests itself as volume preservation, but this would be a bit too simple. Instead, a sum of areas is conserved. Consider an element $\boldsymbol{\omega} = (p_1, p_2, \ldots, p_d, q_1, \ldots, q_d)$ in the domain

$\Omega \subset \mathbb{R}^{2d}$, then we define the two-dimensional domain Ω_k as the projection of Ω onto the coordinates (p_k, q_k):

$$\Omega_k = \left\{ \begin{bmatrix} p_k \\ q_k \end{bmatrix}, \boldsymbol{\omega} \in \Omega \right\}.$$

Introducing $|\cdot|$ to denote the area of a two-dimensional domain, symplecticity then corresponds to the conservation of $\sum_{k=1}^{d} |\Omega_k|$.

Clearly, the above characterization is cumbersome to prove, and it would be convenient to have a different definition of symplecticity that is easier to verify, from which the above area preservation follows. Luckily, such a definition exists:

Definition 1.3 (Symplectic map)**.** Consider a map $\varphi : \Omega \to \mathbb{R}^{2d} : \boldsymbol{y} \mapsto \varphi(\boldsymbol{y})$, with $\Omega \subset \mathbb{R}^{2d}$, and denote its Jacobian as

$$\Psi(\boldsymbol{y}) = \frac{\partial \varphi}{\partial \boldsymbol{y}}(\boldsymbol{y}). \tag{1.29}$$

We say that φ is *symplectic* if

$$\Psi(\boldsymbol{y})^T J \Psi(\boldsymbol{y}) = J, \qquad \text{with} \qquad J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}. \tag{1.30}$$

We will not prove in these notes that Definition 1.3 implies the desired area conservation properties.

1.2.3 The equivalence of symplecticity and Hamiltonian structure

Why do we care so much about symplecticity? After all, is conservation of the Hamiltonian $H(\boldsymbol{p}, \boldsymbol{q})$ not sufficient to understand these systems? There are (at least) two reasons to study this concept more closely:

- As you might have expected, we will show below that Hamiltonian systems are indeed symplectic. What we will not prove rigorously, but only claim, is that the converse is also true: all symplectic maps can be written as the flow map of a Hamiltonian system. Thus, symplecticity and Hamiltonian structure are, in some deep sense, exactly the same thing!

- It will turn out (in section 1.3) that exact conservation of the Hamiltonian $H(\boldsymbol{p}, \boldsymbol{q})$ by numerical methods will be very difficult to achieve (if not impossible) for general Hamiltonian systems. However, it is possible to construct time discretization schemes that are *time-discrete symplectic maps*. This implies that they are the exact solution to *some* (modified) Hamiltonian system. One can then make statements about the accuracy of such methods by studying how close this modified Hamiltonian is to the real Hamiltonian.

Let us now show that Hamiltonian systems are indeed symplectic.

Theorem 1.4 (Poincaré theorem)**.** *If H is twice continuously differentiable, then the flow map φ_t of the Hamiltonian system (1.27) is symplectic.*

Proof. We start from (1.27) and recall the definition of the Jacobian $\Psi_t(\boldsymbol{y})$ from (1.29), i.e., $\Psi_t(\boldsymbol{y})$ is the derivative of the flow map $\varphi_t(\boldsymbol{y})$ with respect to the initial condition $\boldsymbol{y}$, in which we have added a subscript t because we have a family of maps (one for each value of t). Using the chain rule, we see that $\Psi_t(\boldsymbol{y})$ satisfies the following differential equation (which we call the *variational equations*):

$$\frac{d\Psi_t}{dt} = J^{-1}\nabla^2 H(\varphi_t(\boldsymbol{y}))\Psi_t(\boldsymbol{y}). \tag{1.31}$$

We see that equation (1.31) is linear in $\Psi_t(\boldsymbol{y})$, with time-dependent coefficients. To be precise, this equation is the linearization of the Hamiltonian system around the solution trajectory with initial condition $\boldsymbol{y}$. From (1.31), we obtain

$$\begin{aligned}\frac{d}{dt}\left(\Psi_t^T J \Psi_t\right) &= \left(\frac{d\Psi_t}{dt}\right)^T J\Psi_t + \Psi_t^T J\left(\frac{d\Psi_t}{dt}\right)\\ &= \left(\Psi_t(\boldsymbol{y})^T \nabla^2 H(\varphi_t(\boldsymbol{y}))J^{-T}\right) J\Psi_t + \Psi_t^T J\left(J^{-1}\nabla^2 H(\varphi_t(\boldsymbol{y}))\Psi_t(\boldsymbol{y})\right),\end{aligned}$$

where we have just used the product rule and (1.31), as well as the observation that $\nabla^2 H$ is symmetric. Now, it is easy to check that $J^{-T}J = -I$, such that

$$\frac{d}{dt}\left(\Psi_t^T J \Psi_t\right) = 0. \tag{1.32}$$

Hence, $\Psi_t^T J\Psi_t = \Psi_0^T J\Psi_0$ for all $t > 0$, and the proof is concluded by noting that $\Psi_0 = I$. □

The converse statement would be that one can construct a Hamiltonian system corresponding to every symplectic flow map. This statement is also true, but we will not prove it here.

1.3 Symplectic time discretization

1.3.1 Definition of symplectic time discretization

Given that symplecticity gives such an important geometric structure to Hamiltonian systems, it is not surprising that we want to preserve it during time discretisation. We start from the following notation for an arbitrary time discretization,

$$\boldsymbol{y}^{n+1} = \Phi_h(\boldsymbol{y}^n). \tag{1.33}$$

(Compare (1.17) for the specific choice of the trapezoidal rule for a linear system.) Following Definition 1.3, this time discretization is symplectic if we have

$$\Psi_h(\boldsymbol{y})^T J \Psi_h(\boldsymbol{y}) = J, \qquad \text{with} \qquad \Psi_h(\boldsymbol{y}) = \frac{\partial \Phi_h}{\partial \boldsymbol{y}}(\boldsymbol{y}). \tag{1.34}$$

Based on the previous numerical experiments for the harmonic oscillator, we have reasons to believe that the trapezoidal rule and the implicit midpoint rule are symplectic due to the fact that they are symmetric, see section 1.1.3.2. It may be surprising to find out that this is not true! The implicit midpoint rule will turn out to be symplectic, whereas the trapezoidal rule is not. Only for linear systems we have the following equivalence:

Lemma 1.5. *For* linear *Hamiltonian ODEs that correspond to a quadratic Hamiltonian $H = \boldsymbol{y}^T C \boldsymbol{y}$ with a symmetric matrix C, the following two statements are equivalent:*

- *the time discretization is symmetric, i.e., $\Phi_{-h} = \Phi_h^{-1}$;*
- *the time discretization is symplectic.*

For nonlinear ODEs, this equivalence will not hold in general, and the condition (1.34) needs to be checked.

1.3.2 Symplecticity of the implicit midpoint rule

Consider the implicit midpoint rule for the Hamiltonian system (1.27), which we write for the occasion as:

$$\begin{aligned} \boldsymbol{\xi} &= \boldsymbol{y} + \frac{h}{2} J^{-1} \nabla H(\boldsymbol{\xi}) \\ \Phi_h(\boldsymbol{y}) &= \boldsymbol{y} + h J^{-1} \nabla H(\boldsymbol{\xi}) \end{aligned}. \tag{1.35}$$

To show that this method is symplectic, we need to prove (1.34). When introducing $\Xi = \partial \boldsymbol{\xi} / \partial \boldsymbol{y}$, we see that

$$\begin{aligned} \Psi_h(\boldsymbol{y})^T J \Psi_h(\boldsymbol{y}) &= \left(I + h J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi\right)^T J \left(I + h J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi\right) \\ &= J + h \left(\Xi^T \nabla^2 H(\boldsymbol{\xi}) J^{-T} J + J J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi\right) \\ &\quad + h^2 \left(\Xi^T \nabla^2 H(\boldsymbol{\xi})^T J^{-T} J J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi\right) \\ &= J + h \left(-\Xi^T \nabla^2 H(\boldsymbol{\xi}) + \nabla^2 H(\boldsymbol{\xi}) \Xi\right) \\ &\quad + h^2 \left(\Xi^T \nabla^2 H(\boldsymbol{\xi})^T J^{-T} \nabla^2 H(\boldsymbol{\xi}) \Xi\right). \end{aligned} \tag{1.36}$$

Working out Ξ further, we get

$$\Xi = \frac{\partial \xi}{\partial \boldsymbol{y}} = I + \frac{h}{2} J^{-1} \nabla H^2(\boldsymbol{\xi}) \Xi. \tag{1.37}$$

We will not easily be able to continue with an explicit formula for Ξ. Instead, we will perform a weird-looking trick: we will write

$$I = \left(I - \frac{h}{2} J^{-1} \nabla H^2(\xi) \right) \Xi.$$

and further elaborate the term of order h in equation (1.36). The first component is:

$$\begin{aligned} -\Xi^T \nabla^2 H(\boldsymbol{\xi}) &= -\Xi^T \nabla^2 H(\boldsymbol{\xi}) I \\ &= -\Xi^T \nabla^2 H(\boldsymbol{\xi}) \left(I - \frac{h}{2} J^{-1} \nabla H^2(\boldsymbol{\xi}) \right) \Xi \\ &= -\Xi^T \nabla^2 H(\boldsymbol{\xi}) \Xi + \frac{h}{2} \Xi^T \nabla^2 H(\boldsymbol{\xi}) J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi \\ &= -\Xi^T \nabla^2 H(\boldsymbol{\xi}) \Xi - \frac{h}{2} \Xi^T \nabla^2 H(\boldsymbol{\xi}) J^{-T} \nabla^2 H(\boldsymbol{\xi}) \Xi, \end{aligned} \tag{1.38}$$

in which we used $J^{-T} = -J^{-1}$ in the last line.

The second component becomes:

$$\begin{aligned} \nabla^2 H(\boldsymbol{\xi}) \Xi &= \Xi^T \left(I - \frac{h}{2} J^{-1} \nabla H^2(\boldsymbol{\xi}) \right)^T \nabla^2 H(\boldsymbol{\xi}) \Xi \\ &= \Xi^T \nabla^2 H(\boldsymbol{\xi}) \Xi - \frac{h}{2} \Xi^T \nabla H^2(\boldsymbol{\xi}) J^{-T} \nabla^2 H(\boldsymbol{\xi}) \Xi. \end{aligned} \tag{1.39}$$

Combining (1.38), and (1.39), we see that the terms of order h and order h^2 cancel out:

$$\begin{aligned} \Psi_h(\boldsymbol{y})^T J \Psi_h(\boldsymbol{y}) &= J + h \left(-\Xi^T \nabla^2 H(\boldsymbol{\xi}) + \nabla^2 H(\boldsymbol{\xi}) \Xi \right) \\ &\quad + h^2 \left(\Xi^T \nabla^2 H(\boldsymbol{\xi})^T J^{-1} \nabla^2 H(\boldsymbol{\xi}) \Xi \right) \\ &= J + h \left(-\Xi^T \nabla^2 H(\boldsymbol{\xi}) \Xi + \Xi^T \nabla^2 H(\boldsymbol{\xi}) \Xi \right) \\ &\quad - 2 \frac{h^2}{2} \left(\Xi^T \nabla^2 H(\boldsymbol{\xi})^T J^{-T} \nabla^2 H(\boldsymbol{\xi}) \Xi \right) \\ &\quad + h^2 \left(\Xi^T \nabla^2 H(\boldsymbol{\xi})^T J^{-T} \nabla^2 H(\boldsymbol{\xi}) \Xi \right) \qquad (1.40) \\ &= J, \qquad (1.41) \end{aligned}$$

which concludes the proof.

1.3.3 Symplecticity of the symplectic Euler method

Now we are ready to explain the numerical experiments in section 1.1.4, that showed very satisfying behaviour of the symplectic Euler method despite the fact that it did not exactly conserve the Hamiltonian. As the name of the method blatantly suggests, the symplectic Euler method will turn out to be symplectic for Hamiltonian systems with a separable Hamiltonian of the form (1.26). This

implies that the symplectic Euler method is the exact solution to some other Hamiltonian system, hence the qualitative agreement with the exact solution.

For separable Hamiltonian systems, the symplectic Euler scheme reads

$$\begin{aligned} \boldsymbol{q}^{n+1} &= \boldsymbol{q}^n + h\partial_{\boldsymbol{p}} T(\boldsymbol{p}^n) \\ \boldsymbol{p}^{n+1} &= \boldsymbol{p}^n - h\partial_{\boldsymbol{q}} V(\boldsymbol{q}^{n+1}) \end{aligned} \tag{1.42}$$

But why exactly is this method symplectic? To see this, we need to introduce an additional property of symplectic maps:

Proposition 1.6 (Composition)**.** *Consider two symplectic maps φ_1 and φ_2. Then, the composition of these two maps, $\varphi = \varphi_2 \circ \varphi_1$, is also symplectic.*

Because the Hamiltonian (1.26) is the sum of two Hamiltonians (one that depends only on $\boldsymbol{p}$ and one that depends only on $\boldsymbol{q}$), we can associate two Hamiltonian systems with these two separate Hamiltonians:

$$\begin{cases} \boldsymbol{q}' &= \partial_{\boldsymbol{p}} T(\boldsymbol{p}) \\ \boldsymbol{p}' &= 0 \end{cases} \quad \text{and} \quad \begin{cases} \boldsymbol{q}' &= 0 \\ \boldsymbol{p}' &= -\partial_{\boldsymbol{q}} V(\boldsymbol{q}) \end{cases}. \tag{1.43}$$

Let us denote by φ_t^T the flow map corresponding to the exact solution of the first Hamiltonian, and by φ_t^V the flow map corresponding to the exact solution of the second. We have, for these flow maps

$$\begin{cases} \boldsymbol{q}(t) &= \boldsymbol{q} + t\partial_{\boldsymbol{p}} T(\boldsymbol{p}) \\ \boldsymbol{p}(t) &= \boldsymbol{p} \end{cases} \quad \text{and} \quad \begin{cases} \boldsymbol{q}(t) &= \boldsymbol{q} \\ \boldsymbol{p}(t) &= \boldsymbol{p} - t\partial_{\boldsymbol{q}} V(\boldsymbol{q}) \end{cases} \tag{1.44}$$

It is not difficult to see that we can introduce an intermediate stage $(\boldsymbol{q}^{n,*}, \boldsymbol{p}^{n,*})$ such that we can write the symplectic Euler method as the composition of two symplectic maps:

$$(\boldsymbol{q}^{n,*}, \boldsymbol{p}^{n,*}) = \varphi_h^T(\boldsymbol{q}^n, \boldsymbol{p}^n) \tag{1.45}$$

$$(\boldsymbol{q}^{n+1}, \boldsymbol{p}^{n+1}) = \varphi_h^V(\boldsymbol{q}^{n,*}, \boldsymbol{p}^{n,*}). \tag{1.46}$$

or, equivalently,

$$(\boldsymbol{q}^{n+1}, \boldsymbol{p}^{n+1}) = \left(\varphi_h^V \circ \varphi_h^T\right)(\boldsymbol{q}^n, \boldsymbol{p}^n). \tag{1.47}$$

The observation that the symplectic Euler method is the composition of two symplectic maps is sufficient to conclude that it is a symplectic method for separable Hamiltonian systems.

1.3.4 Splitting methods and the Störmer-Verlet method

The symplectic Euler method is a particular example of a splitting method. The idea of splitting (which is very general) is to write the ODE of interest (not necessarily a Hamiltonian system),

$$\boldsymbol{y}' = \boldsymbol{f}(\boldsymbol{y}),$$

as the sum of two terms:

$$\boldsymbol{y}' = \boldsymbol{f}_1(\boldsymbol{y}) + \boldsymbol{f}_2(\boldsymbol{y}),$$

and we construct (approximations of) the flow maps of each of the resulting ODEs

$$\boldsymbol{y}' = \boldsymbol{f}_1(\boldsymbol{y}), \qquad \text{and} \qquad \boldsymbol{y}' = \boldsymbol{f}_2(\boldsymbol{y})$$

as φ_t^1 and φ_t^2. In the case of the splitting (1.43), we can explicitly solve the resulting ODEs, but, in general, one can also work with time discretizations. By Taylor expansion, one can show that $(\varphi_h^1 \circ \varphi_h^2)(\boldsymbol{y}) = \varphi_h(\boldsymbol{y}) + (O)(h^2)$, such that simply the act of performing the splitting results in a first order method, even when each of the substeps is solved exactly.

Based on the above reasoning, it is not so hard to construct additional explicit symplectic methods for separable Hamiltonian systems. A particularly popular example is the method of Störmer and Verlet, which is obtained by the following splitting (also called *Strang splitting*):

$$\Phi_h(\boldsymbol{y}) = \left(\varphi_{h/2}^V \circ \varphi_h^T \circ \varphi_{h/2}^V\right)(\boldsymbol{y}), \tag{1.48}$$

or, equivalently:

$$\begin{cases} \boldsymbol{p}^{n+1/2} &= \boldsymbol{p}^n - \dfrac{h}{2}\partial_{\boldsymbol{q}} V(\boldsymbol{q}^n) \\ \boldsymbol{q}^{n+1} &= \boldsymbol{q}^n + h\partial_{\boldsymbol{p}} T(\boldsymbol{p}^{n+1/2}) \\ \boldsymbol{p}^{n+1} &= \boldsymbol{p}^{n+1/2} - \dfrac{h}{2}\partial_{\boldsymbol{q}} V(\boldsymbol{q}^{n+1}) \end{cases} \tag{1.49}$$

It can be shown that the Störmer-Verlet method is of order 2.

Chapter 2

The finite element method

In this chapter, we discuss the basics of the finite element method, which is the most commonly used method in publicly available simulation software, and therefore also in scientific and engineering applications. We start by detailing the construction of the finite element method in section 2.1 in the very simple setting of a two-point boundary value problem. Given the importance of the finite element method in practice and its elegant formulation that we are about to discover, one might wonder why so much attention is given to finite differences in this course. As one might expect, part of the answer lies in the observation that finite difference methods allow for a much more direct approach, allowing a quantitative understanding of all important numerical behaviour without excessive technicality in the derivations. (In principle, we have only used Taylor expansions up to this point in the course.) Moreover, many of the numerical properties of finite element methods are similar to those of finite difference methods. We give a more detailed comparison of finite difference and finite element methods in section 2.2, before turning to a more general formulation of the finite element method in higher dimensions in section 2.3. In that section, we also comment on some of the most important considerations to keep in mind when implementing or using a finite element method. We comment on higher order methods in section 2.4 and conclude with some remarks on time-dependent problems in section 2.5.

2.1 Two-point boundary value problems

Consider, for simplicity, the following linear two-point boundary value problem in one space variable:

$$-\frac{d}{dx}\left(a(x)\frac{d}{dx}u(x)\right) + b(x)u(x) = f(x), 0 \leq x \leq 1, \tag{2.1}$$

in which the function $u(x)$ is the unknown and the (known) functions a, b and f satisfy: $a(x) > 0$ (and differentiable) and $b(x) \geq 0$. Equation (2.1) is

supplemented with Dirichlet boundary conditions:

$$u(0) = u_0, \qquad u(1) = u_1, \tag{2.2}$$

with $u_{0,1}$ some specified constants. Such two-point boundary value problems arise in many applications. Let us here simply point out that the solution $u(x)$ of equation (2.1) can be seen to be the steady state solution of the reaction-diffusion PDE:

$$\frac{\partial}{\partial t}u(x,t) = \frac{\partial}{\partial x}\left(a(x)\frac{\partial}{\partial x}u(x,t)\right) - b(x)u(x,t) + f(x), \qquad 0 \le x \le 1. \tag{2.3}$$

In this section, we will, however, not pin ourselves to a particular origin of the problem (2.1), instead viewing it merely as a prototypical equation for which it is relatively straightforward to develop a finite element method.

2.1.1 Approximation in a finite-dimensional subspace

In finite difference methods, the basic discretization principle is that we represent the solution $u(x)$ by its values on a grid (a discrete set of points) and subsequently replace every derivative in sight by a finite difference approximation on this grid. The finite element method starts from a different point of view: *first approximate the solution in a finite-dimensional space*. To this end, we need to introduce a basis that defines the finite-dimensional space. We choose to define a function φ_0 that satisfies the boundary conditions (2.2), and a set of M linearly independent functions $\{\varphi_m\}_{m=1}^M$ that satisfy the homogeneous Dirichlet boundary conditions $\varphi_m(0) = \varphi_m(1) = 0$, $1 \le m \le M$. For the time being, we do not specify any further the possible choices for these basis functions. We will come back to this point later (in section 2.1.4).

Given the functions φ_0 and $\{\varphi_m\}_{m=1}^M$, we may write the numerical solution

$$u_M(x) = \varphi_0(x) + \sum_{m=1}^{M} c_m\varphi_m(x), \qquad 0 \le x \le 1, \tag{2.4}$$

and the problem reduces to finding the coefficients $\{c_m\}_{m=1}^M$ such that the function $u_M(x)$ approximates, in some sense, the solution of the original problem (2.1).

The function φ_0 is special. Since it is only there to ensure that u_M satisfies the boundary conditions (2.2), there is no unknown coefficient c_0 attached to it. As a consequence, we will define the linear space as

$$\overset{\circ}{\mathbb{H}}_M := \mathrm{Sp}\left\{\varphi_1, \varphi_2 \ldots, \varphi_M\right\}, \tag{2.5}$$

the span of the basis $\{\varphi_m\}_{m=1}^M$(i.e., the set of all linear combinations of these functions). We then search the function

$$u_M - \varphi_0 \in \overset{\circ}{\mathbb{H}}_M,$$

such that u_M is an approximation to the solution of (2.1) in some sense.

2.1.2 Approximation criterion

The next step now is to give a precise meaning to the word 'approximation'. How do we choose the coefficients $\{c_m\}_{m=1}^M$ in equation (2.4) adequately?

2.1.2.1 The collocation method

One idea is to use the collocation method that we have also encountered during the construction of implicit Runge–Kutta methods. One then forces the differential equation (2.1) to be satisfied exactly by the numerical solution u_M in a set of M distinct *collocation points* $\{x_m\}_{m=1}^M$. This results in the following linear system of equations:

$$\begin{aligned} -\frac{d}{dx}\left(a(x_m)\left\{\frac{d}{dx}\varphi_0(x_m)+\sum_{m'=1}^{M} c_{m'}\frac{d}{dx}\varphi_{m'}(x_m)\right\}\right) \\ +b(x_m)\left\{\varphi_0(x_m)+\sum_{m'=1}^{M} c_{m'}\varphi_{m'}(x_m)\right\}=f(x_m), \qquad 1\le m\le M, \end{aligned} \tag{2.6}$$

Given that the basis functions φ_m (and hence their derivatives) are known explicitly, the only unknowns in (2.6) are the coefficients $\{c_m\}_{m=1}^M$, which can be readily solved for.

2.1.2.2 Galerkin projection

An alternative approach is the Galerkin projection, which tries to use information on the whole interval $(0,1)$, instead of only a discrete set of points. To this end, we introduce the *residual*

$$r_M(x) := -\frac{d}{dx}\left(a(x)\frac{d}{dx}u_M(x)\right)+b(x)u_M(x)-f(x). \tag{2.7}$$

If u_M, by a incredible stroke of luck, would happen to be the exact solution to the differential equation (2.1), we know that $r_M \equiv 0$. Hence, in general, we can guess that the numerical solution u_M will approximate the exact solution u well when r_M is close to zero. We will therefore propose to make r_M as small as possible in an appropriate norm.

One way to ensure that the residual is as small as possible is to require that it is *orthogonal* to the space $\overset{\circ}{\mathbb{H}}_M$. To this end, we introduce a scalar product $\langle\cdot,\cdot\rangle$ on the space $\overset{\circ}{\mathbb{H}}_M$ and search for the set of coefficients $\{c_m\}_{m=1}^M$ that ensure that the following orthogonality conditions are satisfied:

$$\langle r_M, \varphi_m\rangle = 0, \qquad 1\le m\le M. \tag{2.8}$$

The equations (2.8) enforce the residual r_M to be orthogonal to each of the basis functions φ_m and, consequently, to the complete space $\overset{\circ}{\mathbb{H}}_M$; they are called the

Galerkin conditions. A typical choice for the scalar product $\langle \cdot, \cdot \rangle$ is the standard Euclidean inner product (the L_2 inner product):

$$\langle \varphi, \phi \rangle = \int_0^1 \varphi(x)\phi(x)dx. \tag{2.9}$$

We will use (2.9), as is often done in practice, but it should be clear that other choices are very well possible and lead to different finite element approximations.

2.1.2.3 Relation to best approximation

In approximation theory in Hilbert spaces, the *best approximation* u_M of a known function u in a finite-dimensional subspace such as $\overset{\circ}{\mathbb{H}}_M$ is obtained by requiring the *error* $e_M = u_M - u$ to be orthogonal to the subspace $\overset{\circ}{\mathbb{H}}_M$, i.e., by requiring the equations

$$\langle e_M, \varphi_m \rangle_H = 0, \qquad 1 \leq m \leq M, \tag{2.10}$$

to be satisfied for some suitably defined scalar product $\langle \cdot, \cdot \rangle_H$ (with associated norm $\| \cdot \|_H$). For the solution of the differential equation (2.1), however, the function u to approximate itself is unknown, so that we cannot solve the equations (2.10)! The Galerkin conditions are only a *surrogate* for the uncomputable 'best approximation'!

Remark 2.1 (On scalar products)**.** Note that we have introduced a second scalar product here, which can (and usually will) be different from the scalar product $\langle \cdot, \cdot \rangle$ that was used in the Galerkin projection (2.8)! We will discuss the relation between these two scalar products below.

There is some theory required to deduce that the Galerkin conditions provide a *good* surrogate for the best approximation. The main theorem on this matter, of which we only mention the name as an element of general culture, is the *Lax-Milgram theorem.* It requires a number of ingredients from functional analysis that we will not be able to introduce in these notes – hence, a thorough theoretical convergence analysis of the finite element method is out of scope here[1]. The missing ingredients mostly concern the introduction of suitable function spaces (so-called *Sobolev spaces*) and the associated scalar products (and norms).

For the two-point boundary value problem (2.1), the Lax-Milgram theorem shows that we really have

$$\|u_M - u\|_H = \min_{v_M \in \varphi_0 + \overset{\circ}{\mathbb{H}}_M} \|v_M - u\|_H \,, \tag{2.11}$$

for a suitable choice of the norm $\| \cdot \|_H$, which is called the "energy norm". We will be able to specify the energy norm more precisely in section 2.1.3.

[1] Whether this is fortunate or unfortunate probably depends on your point of view. For engineering applications, it is probably sufficient to realize that such an analysis exists. For the development of new finite element methods for particular problems, it might be worthwhile to delve a bit deeper into this theory in a follow-up course.

Thus, the Galerkin projection indeed gives the best approximation in the finite-dimensional subspace when measuring the error in the energy norm.

However, be careful: while the above statement is true for the two-point boundary value problem (2.1), it is not true in general! For more general two-point boundary value problems (with some suitable assumptions that we – again! – not detail at this point), the Galerkin projection usually *does not* yield the best approximation in any norm. Nevertheless, it gives some sort of 'near-best' solution, characterised by the inequality:

$$\|u_M - u\|_H \leq C \min_{v_M \in \varphi_0 + \mathring{\mathbb{H}}_M} \|v_M - u\|_H \,, \tag{2.12}$$

with the norm $\|\cdot\|_H$ the appropriate Sobolev norm[2]. Thus, the Lax-Milgram theorem can rigorously bound the 'non-optimality' of the approximation. Also, equation (2.12) ensures convergence when enlarging the approximation space $\mathring{\mathbb{H}}_M$ (i.e., letting the number of basis functions M tend to infinity)[3].

2.1.3 Weak solutions

When filling in the finite-dimensional approximation (2.4) in the definition of the residual (2.7), we get

$$r_M = -\left[(a\varphi_0')' + \sum_{m=1}^{M} c_m \left(a\varphi_m'\right)'\right] + b\left[\varphi_0 + \sum_{m=1}^{M} c_m \varphi_m\right] - f, \tag{2.13}$$

in which we have used the notation $d/dx\,(\cdot) \equiv (\cdot)'$ for conciseness. Then, we can write the orthogonality conditions (2.8) as

$$\begin{aligned} &\sum_{m'=1}^{M} c_{m'} \left[\left\langle -\left(a\varphi_{m'}'\right)', \varphi_m \right\rangle + \langle b\varphi_{m'}, \varphi_m \rangle\right] \\ &\qquad = \langle f, \varphi_m \rangle - \left[\left\langle -\left(a\varphi_0'\right)', \varphi_m \right\rangle + \langle b\varphi_0, \varphi_m \rangle\right], \qquad 1 \leq m \leq M. \end{aligned} \tag{2.14}$$

A next step in the derivation of the finite element method is to use partial integration to get rid of the second order derivatives of the basis functions $\{\varphi_m\}_{m=0}^{M}$. There are a number of reasons to want to do this:

1. Generally speaking, when fewer derivatives of the basis functions φ_m appear in the equations to solve, more options remain open to choose from.

[2]The energy norm only exists for problems such as (2.1), which are *self-adjoint* – another term that we employ here without further explanation.

[3]This is the final point at which we are lacking some functional analysis background. To make this statement rigorous, we need to ensure that the linear space $\mathring{\mathbb{H}}_M$ is *complete* when M tends to infinity, such that every possible solution of (2.1) can be represented by an element in it. But since we have not even described in which function space solutions of the boundary value problem (2.1) lie, it would clearly be multiple bridges too far to go into these issues here.

2. In particular, it will turn out that some of the most commonly used and most practical choices indeed have reasonably poor smoothness properties. Most common choices are only piecewise differentiable in $[0, 1]$.

3. To be very precise, since the value of an integral is independent of the values of the integrand on a finite set of points, even piecewise linear basis functions are suitable, as soon as only first derivatives appear in the integrand.

Remark 2.2 (Weak solutions)**.** Note that, by allowing numerical solutions (2.4) that are only piecewise differentiable and not twice differentiable at all, the computed numerical solution cannot be a solution to the two-point boundary value problem 2.1 in the strict sense. We call solutions in the strict sense *strong solutions*, and – by contrast – we call the solutions that we compute using Galerkin projection *weak solutions*. We will come back to this point in section 2.1.5.

We thus want to get rid of the second derivatives in (2.14), and we will achieve this (as announced) via partial integration. To perform this partial integration, we will need to be specific about the scalar product we use. Up to now, we had only suggested the L_2 inner product (2.9) as one of the options. At this point, we will go further: we *will* from now on choose (2.9) as our inner product! With this choice, equation (2.14) becomes

$$\sum_{m'=1}^{M} c_{m'} \left[-\int_0^1 \left(a(x)\varphi'_{m'}(x)\right)' \varphi_m(x)dx + \int_0^1 b(x)\varphi_{m'}(x)\varphi_m(x)dx \right] = \int_0^1 f(x)\varphi_m(x)dx - \left[-\int_0^1 \left(a(x)\varphi'_0(x)\right)' \varphi_m(x)dx + \int_0^1 b(x)\varphi_0(x)\varphi_m(x)dx \right], \quad 1 \le m \le M. \tag{2.15}$$

Now we perform partial integration, using the fact (by construction) that the functions φ_m vanish at $x = 0$ and $x = 1$:

$$-\int_0^1 \left(a(x)\varphi'_{m'}(x)\right)' \varphi_m(x)dx = -a(x)\varphi'_{m'}(x)\varphi_m(x)\big|_0^1 + \int_0^1 a(x)\varphi'_{m'}(x)\varphi'_m(x)dx = \int_0^1 a(x)\varphi'_{m'}(x)\varphi'_m(x)dx. \tag{2.16}$$

When introducing the notation

$$a_{m,m'} = \int_0^1 a(x)\varphi'_{m'}(x)\varphi'_m(x) + b(x)\varphi_{m'}(x)\varphi_m(x)dx, \tag{2.17}$$

and

$$f_m = \int_0^1 f(x)\varphi_m(x)dx - a_{m,0} \tag{2.18}$$

it is easy to see that (2.15) reduces to

$$\sum_{m'=1}^{M} a_{m,m'} c_{m'} = f_m, \qquad 1 \leq m \leq M, \tag{2.19}$$

which we write even shorter in matrix notation as

$$\boldsymbol{A}\boldsymbol{c} = \boldsymbol{f}, \tag{2.20}$$

in which we have introduced

$$\boldsymbol{A} = (a_{m,m'})_{m,m'=1}^{M}, \qquad \boldsymbol{c} = (c_m)_{m=1}^{M}, \qquad \boldsymbol{f} = (f_m)_{m=1}^{M}. \tag{2.21}$$

Remark that equation (2.17) introduces a specific inner product

$$\langle u, v \rangle_H = \int_0^1 a(x)u'(x)v'(x) + b(x)u(x)v(x)dx, \tag{2.22}$$

from which the energy norm $\|u\|_H = \sqrt{\langle u, u \rangle_H}$ can be derived.

Since we have now minimised the differentiability requirements on $\{\varphi_m\}_{m=1}^{M}$, and (as expected) all quantities $a_{m,m'}$ can be computed once the basis functions are chosen, equation (2.19) is the form of the orthogonality conditions that will find its way into to finite element software.

2.1.4 Local basis functions

2.1.4.1 The principle of local basis functions

To compute the finite element approximation (2.4), any finite element software needs to perform two steps:

- *construct* the linear system (2.20), i.e., compute the matrix elements $a_{m,m'}$, $1 \leq m, m' \leq M$ and the right hand sides f_m, $1 \leq m \leq M$;
- *solve* the resulting linear system.

It should be clear that there is an interest in choosing the basis functions such that many matrix elements $a_{m,m'}$ become zero. One natural thought after a course on numerical approximation theory is then to choose the $\{\varphi_m\}_{m=1}^{M}$ such that they form an 'orthogonal basis' for the space $\mathring{\mathbb{H}}_M$ with respect to the scalar product (2.22). However, this is usually not done in finite element software, since it is impractical: requiring orthogonality with respect to (2.22) would require the computation of the orthogonal basis *for every equation that one wants to solve*, since this inner product depends on $a(x)$ and $b(x)$[4].

[4] Additionally, for more general problems, the partial integration that leads to (2.17) does not necessarily imply the definition of a scalar product. This is related to the fact that the energy norm only exists when the problem is self-adjoint, see a previous footnote. When (2.17) does not induce a scalar product, constructing an orthogonal basis is not only impractical, it is even impossible.

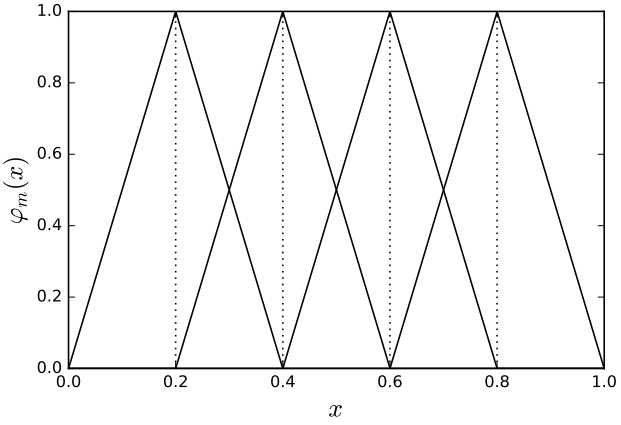

Figure 2.1: Basis functions $\varphi_m(x)$, $m = 1, \ldots, 4$ (solid lines), as well as an indication of the element boundaries (dotted lines).

Instead, we will limit the number of non-zero matrix elements $a_{m,m'}$ by choosing the basis functions $\{\varphi_m\}_{m=1^M}$ to have only local support (choose them to be only non-zero locally), such that only for a limited number of values m and m' the product $\varphi_m \varphi_{m'}$ are non-zero. In this discussion, we will only consider *piecewise linear* basis functions on an equidistant mesh. We start by defining the mesh size $h = 1/(M+1)$ and the mesh points $\{x_m\}_{m=1}^{M}$, with $x_m = mh$. We attach a basis function to each mesh point x_m as follows:

$$\varphi_m(x) = \begin{cases} (x - (x_m - h))/h, & x \in [x_{m-1}, x_m] \\ (-x + (x_m + h))/h, & x \in [x_m, x_{m+1}] \\ 0, & x \notin [x_{m-1}, x_{m+1}] \end{cases} \tag{2.23}$$

(Note that, because φ_m is continuous on $[0, 1]$, it does not matter that we have two function prescriptions at the mesh points $x_{m\pm1}$.) Figure 2.1 shows these basis functions for the case with $M = 4$, which corresponds to $h = 0.2$. We see that these mesh points divide the interval $[0, 1]$ into $M + 1$ *finite elements* (indicated by the dotted lines). We can make a number of observations:

- In each of the internal elements, only two basis functions are non-zero; due to the boundary conditions only one basis function is non-zero in the first and last element.
- In each element, the solution can be any linear function.
- The solution is continuous, but not differentiable, across elements.
- The support of the basis function φ_m only overlaps with the support of φ_{m-1} and φ_{m+1}.

As a consequence of the last observation, we see that the matrix $\boldsymbol{A}$ in equation (2.20) becomes tridiagonal for this choice of basis functions. Thus, only $O(M)$ elements $a_{m,m'}$ need to be computed (instead of $O(M^2)$) and the linear system can readily be solved by the Thomas algorithm that we have encountered in an earlier chapter.

2.1.4.2 Explicit computation of matrix elements

Let us now explicitly compute the system matrix $\boldsymbol{A}$ for the two-point boundary value problem (2.1). For simplicity of the computations below, we choose $b(x) \equiv b$, independent of x, and homogeneous Dirichlet boundary conditions $u(0) = u(1) = 0$. We start with the computation of $a_{m,m-1}$. From (2.17), we see that

$$\begin{aligned} a_{m,m-1} &= \int_0^1 a(x)\varphi'_{m-1}(x)\varphi'_m(x) + b\varphi_{m-1}(x)\varphi_m(x)dx \\ &= \int_{x_{m-1}}^{x_m} a(x)\varphi'_{m-1}(x)\varphi'_m(x) + b\varphi_{m-1}(x)\varphi_m(x)dx, \end{aligned}$$

in which we can restrict the domain of integration to the interval $[x_{m-1}, x_m]$ because this is the only element in which both $\varphi_{m-1}(x)$ and $\varphi_m(x)$ are non-zero.

We can compute these integrals much more conveniently by introducing the *local coordinate* ξ, which is chosen such that $x = x_{m-1} + h\xi$. Using this coordinate, we obtain (locally in the element $[x_{m-1}, x_m]$)

$$\varphi_{m-1}(x) = \varphi_{m-1}(x_{m-1} + h\xi) = 1 - \xi, \qquad \varphi_m(x) = \varphi_m(x_{m-1} + h\xi) = \xi. \tag{2.24}$$

and

$$\varphi'_{m-1}(x) = \frac{d}{d\xi}\varphi_{m-1}(x_{m-1} + h\xi) \cdot \frac{d\xi}{dx} = -\frac{1}{h}, \qquad \varphi'_m(x) = \frac{1}{h} \tag{2.25}$$

We then have, using the above formulas and $dx = hd\xi$,

$$\begin{aligned} a_{m,m-1} &= -\frac{1}{h^2}\int_0^1 a(x_{m-1} + h\xi)hd\xi + \int_0^1 b(1-\xi)\xi hd\xi \\ &= -\frac{1}{h}A_{m-1} + \frac{1}{6}bh, \end{aligned} \tag{2.26}$$

in which we have introduced

$$A_{m-1} = \int_0^1 a(x_{m-1} + h\xi)d\xi. \tag{2.27}$$

which still needs to be computed. For a general function $a(x)$, this can typically not be done analytically. In practice, one therefore often uses a quadrature rule, for instance the trapezoidal rule:

$$A_{m-1} = \frac{1}{2}\left(a(x_{m-1}) + a(x_m)\right). \tag{2.28}$$

Note that, given the piecewise linear basis functions, a higher order quadrature rule at this point does not really make much sense.

With a very similar reasoning, we obtain

$$a_{m,m+1} = -\frac{1}{h}A_m + \frac{1}{6}bh. \tag{2.29}$$

The computation of $a_{m,m}$ follows the same principle, albeit that the integrand is now non-zero on *two* elements: $[x_{m-1}, x_m]$ and $[x_m, x_{m+1}]$. We thus obtain:

$$\begin{aligned} a_{m,m} &= \int_0^1 a(x)\varphi'_m(x)\varphi'_m(x) + b\varphi_m(x)\varphi_m(x)dx \\ &= \int_{x_{m-1}}^{x_m} a(x)\varphi'_m(x)\varphi'_m(x) + b\varphi_m(x)\varphi_m(x)dx \\ &\quad + \int_{x_m}^{x_{m+1}} a(x)\varphi'_m(x)\varphi'_m(x) + b\varphi_m(x)\varphi_m(x)dx, \end{aligned}$$

in which we have split the integral into the contributions stemming from the individual elements. We now perform the local coordinate transform in each element individually to obtain (using (2.24) and (2.25)):

$$\begin{aligned} a_{m,m} &= \frac{1}{h^2}\int_0^1 a(x_{m-1} + h\xi)hd\xi + \int_0^1 b\xi^2 hd\xi \\ &\quad + \frac{1}{h^2}\int_0^1 a(x_m + h\xi)hd\xi + b\int_0^1 (1-\xi)^2 hd\xi, \\ &= \frac{1}{h}\left(A_{m-1} + A_m\right) + 2\cdot\frac{1}{3}bh \end{aligned}$$

Finally, we are left with the computation of the elements f_m, defined in (2.18). We thus start from

$$f_m = \int_0^1 f(x)\varphi_m(x)dx - a_{m,0}, \tag{2.30}$$

and notice that f_m depends on the basis function φ_0 through the quantity $a_{m,0}$. This is therefore the place where we need to worry about the boundary conditions, because these will affect the choice for φ_0. However, given that we have imposed homogeneous Dirichlet boundary conditions, we can safely choose $\varphi_0 \equiv 0$, such that $a_{m,0} = 0$. We thus continue as follows (again using (2.24) and (2.25)):

$$f_m = \int_0^1 f(x)\varphi_m(x)dx \tag{2.31}$$

$$= \int_{x_{m-1}}^{x_m} f(x)\varphi_m(x)dx + \int_{x_m}^{x_{m+1}} f(x)\varphi_m(x)dx \tag{2.32}$$

$$= \int_0^1 f(x_{m-1} + h\xi)\xi hd\xi + \int_0^1 f(x_m + h\xi)(1-\xi)hd\xi. \tag{2.33}$$

We again observe that the integrals cannot be computed analytically. If we also use the trapezoidal rule here, we get:

$$f_m = \frac{h}{2}\left(f(x_{m-1})\cdot 0 + f(x_m)\cdot 1\right) + \frac{h}{2}\left(f(x_m)\cdot 1 + f(x_m)\cdot 0\right) = hf(x_m). \tag{2.34}$$

Thus, after dividing away a factor h that is common to all the terms, equation (2.19) reduces to:

$$\left(-\frac{1}{h^2}A_{m-1} + \frac{1}{6}b\right)c_{m-1} + \left(\frac{1}{h^2}\left(A_{m-1} + A_m\right) + \frac{2}{3}b\right)c_m + \left(-\frac{1}{h^2}A_m + \frac{1}{6}b\right)c_{m+1} = f(x_m). \tag{2.35}$$

Let us stand still for a moment on the two crucial tricks that made the above computation relatively simple to perform:

- We have split the integral over the domain into integrals over individual elements.
- We introduced a local coordinate ξ inside each element to perform the individual integrations.

Any finite element code, regardless of the complexity of the problem, will use these tricks to efficiently construct the matrices $\boldsymbol{A}$ and $\boldsymbol{f}$. The fact that this is always possible is one of the powers of the finite element method. It, for instance, allows to readily introduce adaptive grids. (The only change then is that the value of h changes from element to element, but the above procedure carries out without problems.)

Remark 2.3 (Interpretation of coefficients)**.** Due to the choice of the basis functions, we see that c_m is an approximation to the solution $u(x_m)$, which allows us to visualise the solution without the need to evaluate (2.4) in the grid points.

Remark 2.4 (Relation to finite difference methods)**.** The resulting discretization (2.35) is very similar to a finite difference discretization of the same problem. One can recognize the diffusive part and the right hand side. A peculiarity is the fact that the local term $bu(x)$ is spread out over three terms. Note, however, that for the one-dimensional Poisson equation $u'' = f$, equation (2.35) reduces to

$$-\frac{1}{h^2}c_{m'-1} + \frac{2}{h^2}c_{m'} - \frac{1}{h^2}c_{m'+1} = f(x_m), \tag{2.36}$$

which is *identical* to the finite difference approximation.

Remark 2.5 (Contrast with spectral methods)**.** Since the choice of basis functions is up to the user, one might consider other criteria than locality to be

important. One reasoning would be to consider global basis functions that are so representative of the solution of the boundary value problem that only a limited number of them are needed to adequately represent the solution. One then needs to compare the computational cost of solving a relatively small (but full) linear system with that of a sparse (but potentially very large) linear system. Given that finite element methods with piecewise linear basis functions result in a space discretization error of $O(h^2)$, this might be a potentially interesting avenue to follow. (We have not shown – or even discussed – the order of finite element methods here, but a quick look at Remark 2.4 might convince you of the $O(h^2)$ convergence.) That avenue leads to so-called *spectral methods*, and we will not discuss such methods in these notes.

2.1.5 Variational formulation

In the previous sections, we have developed the finite element method for a very simple two-point boundary value problem. Before we will generalize the method to more complex situations and really discuss its virtues over the finite difference method, we will make a detour that derives the Galerkin equations (2.8) in an alternative way.

Let us consider a *variational problem*, in which we are given a functional $\mathcal{J}(u) : \mathbb{H} \to \mathbb{R}$, in which $\mathbb{H}$ is some function space, that we – again! – do not specify further here, except to note that $\mathbb{H}$ only contains functions that satisfy the boundary conditions (2.2). We are now interested in finding a function $u \in \mathbb{H}$ that and minimizes $\mathcal{J}$:

$$u = \arg\min_{v \in \mathbb{H}} \mathcal{J}(v).$$

In particular, we will consider here the functional

$$\mathcal{J}(v) := \int_0^1 \left[a(x) \left(v'(x)\right)^2 + b(x) \left(v(x)\right)^2 - 2f(x)v(x) \right] dx, \tag{2.37}$$

with the functions $a(x)$, $b(x)$ and $f(x)$ (not coincidentally) satisfying the same conditions as those in the boundary value problem (2.1). It is possible to prove that the minimizer u exists, such that the variational problem always has a solution. As can be expected by now, we will not do this here.

We also introduce the space $\overset{\circ}{\mathbb{H}}$ of all functions v that obey homogeneous Dirichlet boundary conditions. Then, given that $u \in \mathbb{H}$ minimizes $\mathcal{J}$, we know that we have

$$\mathcal{J}(u) \leq \mathcal{J}(u + v), \qquad v \in \overset{\circ}{\mathbb{H}}. \tag{2.38}$$

Let us choose $v \neq 0$ and $\epsilon \neq 0$. We can then write

$$\begin{aligned}
\mathcal{J}(u+\epsilon v) &= \int_0^1 \left[a\left(u'+\epsilon v'\right)^2 + b\left(u+\epsilon v\right)^2 - 2f(u+\epsilon v)\right] dx \\
&= \int_0^1 \left[a\left(\left(u'\right)^2 + 2\epsilon u'v' + \epsilon^2\left(v'\right)^2\right) + b\left(u^2 + 2\epsilon uv + \epsilon^2 v^2\right) - 2f\left(u+\epsilon v\right)\right] dx \\
&= \int_0^1 \left[a\left(u'\right)^2 + bu^2 - 2fu\right] dx + 2\epsilon \int_0^1 \left[au'v' + buv - fv\right] dx \\
&\qquad\qquad + \epsilon^2 \int_0^1 \left[a\left(v'\right)^2 + bv^2\right] dx \\
&= \mathcal{J}(u) + 2\epsilon \int_0^1 \left[au'v' + buv - fv\right] dx + \epsilon^2 \int_0^1 \left[a\left(v'\right)^2 + bv^2\right] dx.
\end{aligned}$$

Using (2.38), we see that

$$2\epsilon \int_0^1 \left[au'v' + buv - fv\right] dx + \epsilon^2 \int_0^1 \left[a\left(v'\right)^2 + bv^2\right] dx \geq 0,$$

for all non-zero values of ϵ. As $|\epsilon|$ can be made arbitrarily small, we can safely neglect the second order term. Moreover, this condition is valid for *any* ϵ, and we are allowed to replace ϵ by $-\epsilon$ without violating the condition. Thus, we deduce that, necessarily,

$$\int_0^1 \left[a(x)u'(x)v'(x) + b(x)u(x)v(x) - f(x)v(x)\right] dx = 0, \forall v \in \overset{\circ}{\mathbb{H}}. \tag{2.39}$$

While we have only shown that (2.39) is necessary, it can also be proved to be sufficient. (We will, of course, not do this here...)

The above formulation can be seen to be equivalent to the *weak form* of the two-point boundary value problem (2.1), which is obtained by requiring the residual of the solution u of (2.1) to be orthogonal to the whole space $\overset{\circ}{\mathbb{H}}$:

$$\int_0^1 \left[-\left(a(x)u'(x)\right)' + b(x)u(x) - f(x)\right] v(x)dx. \tag{2.40}$$

In principle, this is the same as requiring that the residual is identically zero, since it needs to be orthogonal to the whole space. However, there is a point in doing this, as we can perform partial integration (as we have done in section 2.1.3) to reduce the smoothness requirements on u. This can be seen to result in (2.39).

One can therefore also recover the Galerkin conditions by searching for the minimizer u_M in the finite dimensional subspace $\mathbb{H}_M$ (see equation (2.4)):

$$u = \arg\min_{v \in \mathbb{H}_M} \mathcal{J}(v).$$

Since v is now a finite-dimensional object, we can proceed in the classical way to formulate optimality conditions. We write

$$\mathcal{J}_M(\boldsymbol{c}) := \mathcal{J}\left(\varphi_0 + \sum_{m=1}^{M} c_m \varphi_m\right), \qquad \boldsymbol{c} \in \mathbb{R}^M,$$

and compute the coefficients $\boldsymbol{c}$ by requiring $\nabla_{\boldsymbol{c}} \mathcal{J}_M(\boldsymbol{c}) = 0$. Writing down these equations, one recovers exactly the Galerkin conditions (2.19). Indeed, we have

$$\begin{aligned}\frac{1}{2}\frac{\partial \mathcal{J}_M}{\partial c_m} = &\sum_{m'=1}^{M} c_{m'} \int_0^1 (a\varphi'_{m'}\varphi'_m + b\varphi_{m'}\varphi_m)\, dx \\ &+ \int_0^1 (a\varphi'_0\varphi'_m + b\varphi_0\varphi_m)\, dx - \int_0^1 f\varphi_m dx, \qquad 1 \le m \le M, \quad (2.41)\end{aligned}$$

which results in exactly the Galerkin conditions (2.19).

2.2 Comparison with finite difference methods

2.2.1 Why mainly finite differences in this course?

Now that the finite difference and the finite element method have both been introduced, it is clear that the finite difference method has a number of clear advantages over the finite element method, mainly in terms of simplicity:

- Discretizing all derivatives in sight on a grid is relatively straightforward to do, even without a thorough mathematical background;
- Also performing a numerical analysis of the properties (consistency, stability) of the resulting schemes is straightforward, and a Fourier analysis of the resulting schemes does not require advanced mathematical tools.

Clearly, the mathematical setup of the finite element method is much more involved. For this reason, most of the theory on numerical methods for PDEs in this course concentrated on finite difference schemes. For the finite element method, we restricted ourselves to a detailed description of its construction, merely hinting at some of the subtleties that should be accounted for in a proper convergence analysis. Nevertheless, the finite element method is much more commonly used in practice. For this to make sense, there must be an advantage that comes with this increased complexity!

2.2.2 Practical advantages of finite element methods

There are a number of practical advantages that follow immediately from the element-wise formulation of the finite element method. We announce those advantages here:

- *Flexibility.* The finite element method can easily handle complex geometries and boundary conditions, whereas the finite difference method is usually restricted to (a variation of) a rectangular domain. The finite element method will turn out to discretize the domain into triangles, which allow more flexibility than the rectangles that are typically required in finite difference schemes. Usually, a separate software program to triangularize the computational domain can be used to generate a suitable mesh for a given simulation.
- *Adaptivity.* Additionally, the flexibility of the element shapes and sizes allows for a straightforward *local mesh refinement* in regions of the spatial domain in which the error is large (or can be expected to be large *a priori*. Also the introduction of higher-order methods is relatively straightforward by simply considering local polynomial approximations of higher order.
- *Generality.* The finite element method is very general, and allows one to easily simulate (in one simulation) coupled multiphysics problems (for instance, fluid-structure interaction). Here, different PDEs represent the physics in different parts of the spatial domain – the coupling is done with appropriate boundary conditions. The relative ease with which boundary conditions are constructed is a significant factor in solving such complex problems.

In section 2.3, we will formulate the finite element method in a more general (multi-dimensional) situation. There, we will illustrate these advantages in more detail.

2.2.3 Theoretical advantages of finite element methods

It would lead us too far to discuss at full length the theory that exists for finite element method. In view of the very brief discussion above, we only state the following theoretical advantages of the finite element method, that all stem from the principle of approximation in a finite-dimensional space:

- There exists a large literature on a priori error estimation and convergence analysis, which can be used during discretization and mesh generation.
- The numerical approximation is guaranteed to be reasonable in every point of the spatial domain (also in between the mesh points).
- There is also a large literature on *a posteriori* error estimation, i.e., obtaining upper bounds on the error once the solution is computed. These results can be used to adaptively refine the grid where needed.

2.3 Finite element methods in higher dimensions

Now that we have discussed the formulation and principles of the finite element method for the specific case of a one-dimensional two-point boundary value

Figure 2.2: A triangle and a non-conforming triangle.

problem, we are ready to generalize. Let us, in these notes, restrict ourselves to two space dimensions, such that we search for the solution $u(x, y)$ of the following PDE:

$$\mathcal{L}u = f, \qquad (x, y) \in \Omega, \tag{2.42}$$

with some suitable boundary conditions that we will specify later. The direct two-dimensional generalization of the two-point boundary value problem (2.1) is obtained by choosing

$$\mathcal{L} := -\nabla \left(a(x, y) \nabla \right) + b(x, y),$$

in which $a(x, y) > 0$ and differentiable, and $b(x, y) > 0$.

2.3.1 Finite-dimensional space and local basis functions

In two space dimensions, we look for a piecewise linear approximation on a *set of two-dimensional subdomains* that cover the computational domain. In each subdomain, the solution is then represented by a plane in the three-dimensional space (x, y, u). Since a plane is completely specified by three points, the subdomains will need to be triangles (as we need to specify the solution in exactly three points per subdomain to have a unique representation). (Hence, we will not use rectangles for now, as is typically done with finite difference schemes. Some remarks on the use of rectangles are given at the end of this section.) To ensure the continuity of the solution in the whole domain, solutions need to be continuous on triangle edges. As a consequence, all vertices need to be shared among triangles; no vertex of a triangle can lie on an edge of a different triangle, see Figure 2.2.

When partitioning the domain Ω into a set of J triangles $\{\Omega_j\}_{j=1}^{J}$, this immediately implies that the boundary of the domain $\partial\Omega$ will consist of a finite number of straight segments. Thus, the equation will be solved on *an approximation of the domain* and with boundary conditions that are imposed at *an approximation of the boundary* $\partial\Omega$.

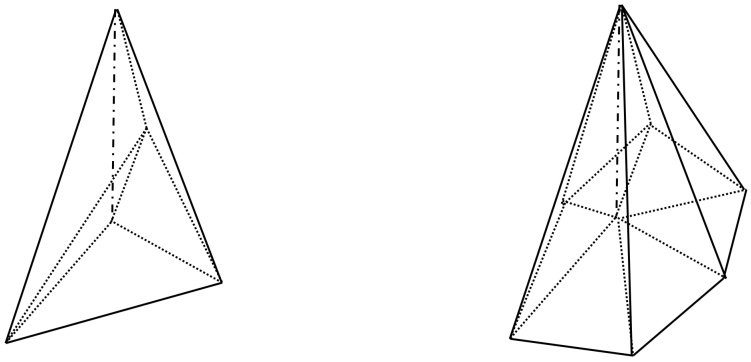

Figure 2.3: Different configurations of chapeau functions in two space dimensions.

For one-dimensional problems, we defined the basis functions as the piecewise linear functions that vanish at all mesh points except for one. In two dimensions, this corresponds to the so-called *chapeau functions*[5]. A chapeau function is one in exactly one vertex, zero in all other vertices and has support only in the elements surrounding the vertex in which it takes the value one.

As a consequence (and unfortunately), making explicit use of these chapeau functions is very impractical, because many different configurations are possible, and the number of elements in each support (and hence the shape of the basis function) may change from vertex to vertex, see Figure 2.3.

At the same time, we know from the derivations in section (2.1.4) that it will be convenient to compute the integrals that appear in the matrix $\boldsymbol{A}$ (see equation (2.20)) on an element-by-element basis. Therefore, all we need is a representation of the solution *inside each element* Ω_j. Let us denote by $\left\{(x_j^\ell, y_j^\ell)\right\}_{\ell=1}^3$ the three vertices of triangle Ω_j. Since, inside Ω_j, the approximate solution is linear, we can write it as

$$u_j(x,y) = \alpha_j + \beta_j x + \gamma_j y, \qquad (x,y) \in \Omega_j, \tag{2.43}$$

in which the values of α_j, β_j and γ_j can be found by solving the linear system:

$$\alpha_j + \beta_j x_j^\ell + \gamma_j y_j^\ell = u_j^\ell, \qquad \ell = 1,2,3 \tag{2.44}$$

with $\left\{u_j^\ell\right\}_{\ell=1}^3$ the solution values in the vertices. Both the values α_j, β_j and γ_j and the three solution values $\left\{u_j^\ell\right\}_{\ell=1}^3$ specify completely the approximate solution in the elements. In particular, the values $\left\{u_j^\ell\right\}_{\ell=1}^3$ are not known yet:

[5]By using the French word "chapeau" instead of the English word "hat", the intention is to indicate that the chapeau functions are constructed using the same principle as the hat functions in the one-dimensional case, but are nonetheless slightly more complicated – or sophisticated, just like the French language apparently is.

they are exactly what we are trying to compute! The linear system (2.44) shows the equivalence between the solution values in the vertices $\{u_j^\ell\}_{\ell=1}^3$ and the parameters α_j, β_j and γ_j.

Remark 2.6 (Quadrilateral elements)**.** Sometimes, one prefers quadrilateral elements (rectangles aligned with the axes). One can then interpolate using functions of the form:

$$u_j(x,y) = u_j^1(x)u_j^2(y), \qquad \text{with} \qquad u_j^1(x) = \alpha_j + \beta_j x, \quad u_j^2(y) = \gamma_j + \delta_j y. \tag{2.45}$$

Clearly, we have now four parameters, which is perfectly suited to interpolation at the four corners of a rectangle. Along both horizontal edges, $u_j^2(y)$ is constant. Since $u_j^1(x)$ is completely specified by the values at the corners, the function $u_j(x,y)$ is independent of the interpolated values elsewhere on the mesh. A similar argument holds for the vertical edges. This ensures global continuity of the resulting approximation on the whole domain.

2.3.2 Weak formulation in higher dimensions

Now that we know how we will represent the numerical solution inside each element, we need to write down the equations that need to be solved to solve the PDE (2.42). As in the one-dimensional case, we will perform a Galerkin projection (requiring orthogonality of the residual to the approximation space, see equation (2.8)) and perform "partial integration" to get rid of the second order derivative.

In the two-dimensional case, we will need to use various multivariate counterparts of integration by parts. To recall a few examples of these, we denote by $\boldsymbol{x} = (x,y)$ a point in the domain Ω, by $\boldsymbol{s}$ a point on the boundary $\partial\Omega$ and by $\partial v(\boldsymbol{s})/\partial n$ the directional derivative of v on the boundary $\partial\Omega$ in the direction of the outward normal to the boundary. One particular case is the *divergence theorem*:

$$\int_\Omega \nabla\cdot[a(\boldsymbol{x})\nabla v(\boldsymbol{x})]\, w(\boldsymbol{x})d\boldsymbol{x} = \int_{\partial\Omega} a(\boldsymbol{s})w(\boldsymbol{s})\frac{\partial v(\boldsymbol{s})}{\partial n}d\boldsymbol{s} - \int_\Omega a(\boldsymbol{x})\,[\nabla v(\boldsymbol{x})]\,[\nabla w(\boldsymbol{x})]\,d\boldsymbol{x}. \tag{2.46}$$

Another example is *Green's formula*:

$$\int_\Omega \left[\nabla^2 v(\boldsymbol{x})\right] w(\boldsymbol{x})d\boldsymbol{x} + \int_\Omega [\nabla v(\boldsymbol{x})]\,[\nabla w(\boldsymbol{x})]\,d\boldsymbol{x} = \int_{\partial\Omega} \frac{\partial v(\boldsymbol{s})}{\partial n} w(\boldsymbol{s})d\boldsymbol{s}. \tag{2.47}$$

Both formulas are special cases of *Stokes's theorem*, which we will not cover in detail in these notes.

Let us make use of the above formulas to derive the weak formulation of the two-dimensional *Poisson equation*:

$$-\nabla^2 u = f \qquad \text{or} \qquad -\left(\frac{\partial^2}{\partial x^2}u(x,y) + \frac{\partial^2}{\partial y^2}u(x,y)\right) = f(x,y), \tag{2.48}$$

for $(x, y) \in \Omega$ with homogeneous Dirichlet boundary conditions. Writing this equation in weak form and using (2.47), we get

$$-\int_\Omega \nabla^2 u \, v d\boldsymbol{x} = \int_\Omega \nabla u \cdot \nabla v d\boldsymbol{x} - \int_{\partial\Omega} v(\boldsymbol{s}) \frac{\partial u(\boldsymbol{s})}{\partial n} d\boldsymbol{s} \tag{2.49}$$

$$= \int_\Omega \nabla u \cdot \nabla v d\boldsymbol{x} = \int_\Omega f v d\boldsymbol{x}, \tag{2.50}$$

since we again only allow test functions v that vanish on the boundary $\partial\Omega$.

The finite element formulation then amounts to a finite-dimensional version of (2.50), i.e., we formally write

$$u_M(\boldsymbol{x}) = \varphi_0(\boldsymbol{x}) + \sum_{m=1}^{M} c_m \varphi_m(\boldsymbol{x}), \tag{2.51}$$

in which the functions φ_m are the chapeau functions mentioned earlier.

Remark 2.7 (On the chapeau functions)**.** As announced in section 2.3.1 will never make direct use of these basis functions. Instead, we will – as in the one-dimensional case – always make use of local representations inside each element. That is the reason we say we "formally write". How the direct use of the chapeau functions is avoided, will become clear in section 2.3.3.

With the formal representation (2.51), we obtain the following equations,

$$\int_\Omega \nabla u_M \cdot \nabla \varphi_m d\boldsymbol{x} = \int_\Omega f \varphi_m d\boldsymbol{x}, \qquad 1 \le m \le M, \tag{2.52}$$

which reduce to the linear system

$$\sum_{m'=1}^{M} c_{m'} \int_\Omega \nabla \varphi_{m'} \cdot \nabla \varphi_m d\boldsymbol{x} = \int_\Omega f \varphi_m d\boldsymbol{x} - \int_\Omega \nabla \varphi_0 \cdot \nabla \varphi_m d\boldsymbol{x}, \qquad 1 \le m \le M, \tag{2.53}$$

which, again, results in the linear system

$$\sum_{m'=1}^{M} a_{m,m'} c_{m'} = f_m, \qquad 1 \le m \le M, \tag{2.54}$$

with

$$a_{m,m'} = \int_\Omega \nabla \varphi_{m'} \cdot \nabla \varphi_m d\boldsymbol{x}, \qquad f_m = \int_\Omega f \varphi_m d\boldsymbol{x} - \int_\Omega \nabla \varphi_0 \cdot \nabla \varphi_m d\boldsymbol{x}. \tag{2.55}$$

Compare the above equations with (2.19) for the one-dimensional case.

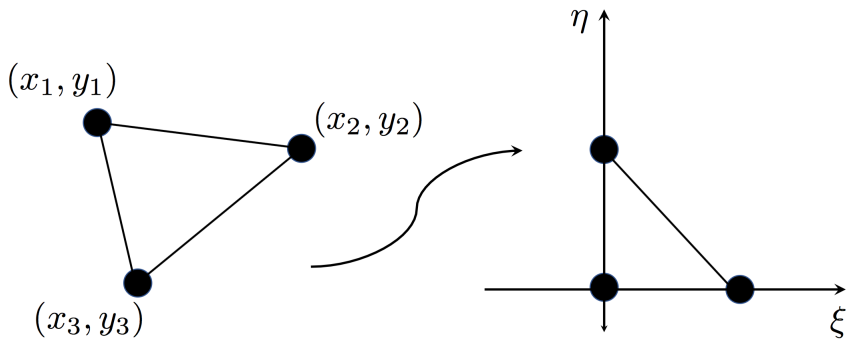

Figure 2.4: Local coordinate transform in 2D.

2.3.3 Assembling the stiffness matrix

While the above equations are written in terms of the chapeau functions φ_m, we have already indicated in section 2.3.1 that these functions are not ideal for the computation of the matrix elements $a_{m,m'}$ because their shape depends significantly on the number of triangles in which a given vertex participates. To circumvent this problem, we again employ the two tricks that were also used in the one-dimensional case in section 2.1.4:

- We split the integral into the contributions per element;
- We introduce a local coordinate system in each element.

We will limit ourselves to an outline of the general principles.

Let us first elaborate the computation of the coefficients

$$\begin{aligned} a_{m,m'} &= \int_\Omega \nabla\varphi_{m'} \cdot \nabla\varphi_m d\boldsymbol{x} = \sum_{j=1}^{J} \int_{\Omega_j} \nabla\varphi_{m'} \cdot \nabla\varphi_m d\boldsymbol{x} \\ &= \sum_{j \in \mathcal{J}_{m,m'}} \int_{\Omega_j} \nabla\varphi_{m'} \cdot \nabla\varphi_m d\boldsymbol{x}, \end{aligned}$$

in which the set $\mathcal{J}_{m,m'}$ consists of the indices of all elements Ω_j in which both the basis functions $\varphi_{m'}$ and φ_m are non-zero. We have now split the integral into the element-wise contributions and restricted ourselves to the elements in which these contributions are non-zero.

The second step is the introduction of a local coordinate system $\boldsymbol{\xi} = (\xi, \eta)$. This is done by mapping each element (with vertices $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$) onto a reference triangle with vertices at $\boldsymbol{x}_1 = (0, 0)$, $\boldsymbol{x}_2 = (1, 0)$ and $\boldsymbol{x}_3 = (0, 1)$, see also Figure 2.4. When introducing the notation $|\Omega_j|$ for the area of the element

Ω_j, the following transformations are straightforward to check:

$$\xi = \frac{1}{2\,|\Omega_j|}\left((y_3 - y_1)(x - x_1) - (x_3 - x_1)(y - y_1)\right), \tag{2.56}$$

$$\eta = \frac{1}{2\,|\Omega_j|}\left(-(y_2 - y_1)(x - x_1) + (x_2 - x_1)(y - y_1)\right). \tag{2.57}$$

and the three basis functions in this local coordinate system can be seen to be:

$$\psi_1(\xi, \eta) = 1 - \xi - \eta, \qquad \psi_2(\xi, \eta) = \xi, \qquad \psi_3(\xi, \eta) = \eta. \tag{2.58}$$

Then, as in the one-dimensional case, the required integrals can be computed by taking the appropriate products of two of these basis functions inside each element. We will not work out the details here.

Remark 2.8 (Computation of the right hand side)**.** As in the one-dimensional case (see equation (2.18)), the computation of f_m cannot be done analytically, but requires some numerical quadrature formula. Also in this case, it is convenient to use the element-wise formulation and local coordinates.

2.4 Higher order methods

2.4.1 Higher order polynomials in each element

One natural way to improve the order of accuracy of the finite element method is to consider higher order polynomials to approximate the solution inside each element. As before, we want a representation of the solution that satisfies the following properties:

- Global continuity of the numerical approximation, i.e., continuity on the element edges in particular;
- A representation that allows an element-by-element computation of all necessary matrix entries.

Both requirements can be fulfilled by choosing appropriate vertices and representing the numerical solution inside an element in terms of local basis functions that are each a polynomial of the desired degree that is zero in all of these vertices except one.

Consider a general quadratic function in $\mathbb{R}^2$ of the form

$$u_j(x, y) = \alpha_j + \beta_j x + \gamma_j y + \delta_j x^2 + \eta_j xy + \zeta_j y^2, \tag{2.59}$$

which represents the numerical solution inside the element Ω_j. We see that we need to determine six parameters; we will therefore need six local basis functions and six vertices in a triangle. We also know that the solution is uniquely determined on each element edge as soon as three point on this edge are fixed. (This is because the solution is a quadratic function on every straight

Figure 2.5: Schematic picture of higher order finite elements (edges and vertices). Left: second order element; right: third order element. .

line, and therefore also along element edges.) As a result, a second order element can be graphically depicted as the left triangle in Figure 2.5. Similarly, a cubic function in two space dimensions has ten parameters (check this!) and we therefore need ten basis functions inside one element. Since we need to specify function values at four points along each edge to ensure global continuity of the solution, we have one degree of freedom left that cannot be placed along an element edge. We will place it at the center of mass of the triangle. This leads to the element that is schematically depicted as the right triangle in Figure 2.5.

2.4.2 Higher-order continuity along element edges

Piecewise continuity of the numerical solution is sufficient in most common situations. However, for some equations (for instance, including the biharmonic operator ∇^4), one also requires the first derivative of the basis functions be be continuous across element boundaries. This can be ensured by introducing an element of the form in figure 2.6, in which one interpolates the solution value and both spatial derivatives in each of the three vertices (which gives nine conditions), together with the solution value in the center of mass of the triangle.

2.4.3 Periodic table of finite element methods

We have only covered a small portion of finite element theory and practice. As mentioned before, a lot of effort in finite element analysis goes out to error analysis, error control and adaptivity. Another branch of research deals with the choice of the basis functions and representations. It is not necessary to represent the solution in terms of the solution values on the vertices. This is a choice – a common choice in many situations, but other choices are possible nevertheless.

Depending on the form of the equation, one can have a special interest to preserve qualitative features of the solution, dependencies between different variables in a specific model, etc. To illustrate the richness of finite element practice would be out of scope in these notes. Here, we only refer to the following

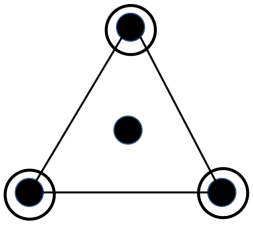

Figure 2.6: Schematic picture of a finite element that results in basis functions that have continuous derivatives across element edges. The interpolation is based on the function value and (when an extra circle is present around the vertex) both spatial derivatives.

website that aims at collecting and systematizing many finite element methods "out there": `http://femtable.org`. In these notes, we have only covered the P-type finite elements in one and two space dimensions.

2.5 Time-dependent problems

2.5.1 General derivation

The use of the finite element method can be extended to time-dependent problems of the form:

$$\partial_t u(x,t) + \mathcal{L}u(x,t) = f(x,t). \tag{2.60}$$

(Compare this equation with (2.42).)

The key idea is to *first* discretize the equation in space, leaving time continuous, and to only discretize time afterwards. In the finite element method, this can be achieved by introducing a numerical approximation in the finite-dimensional subspace *at each moment in time*, i.e., we write

$$u_M(x,t) = \varphi_0(x) + \sum_{m=1}^{M} c_m(t)\varphi_m(x), \tag{2.61}$$

in which the only change is that we have made the coefficients $\{c_m(t)\}_{m=1}^M$ dependent on time. Since the number of coefficients M is finite, the spatial discretization using finite elements transforms the PDE (2.60) into a (potentially very large) system of ordinary differential equations for the coefficients $\{c_m(t)\}_{m=1}^M$.

Let us now define the residual as

$$r_M(x,t) = \partial_t u_M(x,t) + \mathcal{L}u_M(x,t) - f(x,t), \tag{2.62}$$

and again perform the Galerkin projection onto the basis functions:

$$\langle r_M, \varphi_m \rangle = 0, \qquad 1 \leq m \leq M, \tag{2.63}$$

which we can expand to

$$\begin{aligned}\langle r_M, \varphi_m \rangle &= \langle \partial_t u_M, \varphi_m \rangle + \langle \mathcal{L} u_M, \varphi_m \rangle - \langle f, \varphi_m \rangle \\ &= \sum_{m'=1}^{M} \frac{d}{dt} c_{m'}(t) \langle \varphi_{m'}, \varphi_m \rangle + \sum_{m'=1}^{M} c_{m'}(t) \langle \mathcal{L}\varphi_{m'}, \varphi_m \rangle + \langle \mathcal{L}\varphi_0, \varphi_m \rangle - \langle f, \varphi_m \rangle\end{aligned}$$

to obtain the following system of ordinary differential equations for $c_m(t)$:

$$\sum_{m'=1}^{M} \frac{d}{dt} c_{m'}(t) \langle \varphi_{m'}, \varphi_m \rangle + \sum_{m'=1}^{M} c_{m'}(t) \langle \mathcal{L}\varphi_{m'}, \varphi_m \rangle = \langle f, \varphi_m \rangle - \langle \mathcal{L}\varphi_0, \varphi_m \rangle . \tag{2.64}$$

As in (2.17) (for the one-dimensional setting) and equation (2.55) (for the two-dimensional case), we again introduce the notation

$$a_{m,m'} = \langle \mathcal{L}\varphi_{m'}, \varphi_m \rangle , \qquad f_m = \langle f, \varphi_m \rangle - \langle \mathcal{L}\varphi_0, \varphi_m \rangle$$

(Remember that the coefficients $a_{m,m'}$ are typically computed using partial integration or a multidimensional variant.) The additional quantities to compute now are the coefficients

$$b_{m,m'} = \langle \varphi_{m'}, \varphi_m \rangle ,$$

resulting in the shorthand notation

$$\sum_{m'=1}^{M} \frac{d}{dt} c_{m'}(t) b_{m,m'} + \sum_{m'=1}^{M} c_{m'}(t) a_{m,m'} = f_m. \tag{2.65}$$

Compare equation (2.65) with equation (2.19) for the time-independent case. As we did in the one-dimensional case in equation (2.20), we can again write these equations in matrix form as:

$$\boldsymbol{B} \frac{d}{dt} \boldsymbol{c}(t) + \boldsymbol{A}\boldsymbol{c}(t) = \boldsymbol{f}. \tag{2.66}$$

in which the matrix $\boldsymbol{B} = (b_{m,m'})_{m,m'=1}^{M}$ is called the *mass matrix*. For the one-dimensional two-point boundary value problem (2.1) and the piecewise linear basis functions (2.23), it can easily be checked (by following the computations in section 2.1.4.2), that the matrix B is tridiagonal, with elements

$$b_{m,m-1} = 1/6, \qquad b_{m,m} = 4/6, \qquad b_{m,m+1} = 1/6.$$

Remark 2.9 (Mass lumping)**.** Sometimes, the appearance of the mass matrix $\boldsymbol{B}$ is considered a nuisance, and one would have preferred the system (2.66) to read

$$\frac{d}{dt} \boldsymbol{c}(t) + \boldsymbol{A}\boldsymbol{c}(t) = \boldsymbol{f}, \tag{2.67}$$

i.e., the matrix $\boldsymbol{B}$ is replaced by the identity matrix. This is called *mass lumping* and it has advantages if one wants to use a software for time discretization that does not allow adding a mass matrix. When one discretizes (2.67) with forward Euler in time, one can show that this corresponds to only applying the finite element discretization in space and combining it with a finite difference approximation in time.

2.5.2 Artificial viscosity methods for advection-dominated problems

The theory on finite difference methods allows to easily understand the behaviour of space-time discretizations of various kinds of PDEs (using only a von Neumann stability analysis and the principle of modified equations). One (very important) example is the introduction of *upwinding*. With finite elements, upwinding is very hard to encode in the choice of basis functions. However, one can make use of the observation that upwinding results in an additional diffusion term with a diffusion coefficient that depends on the mesh width. In finite element methods for advection dominated problems, one then simply modifies the equations to explicitly contain such an artificial diffusion term. (The name "artificial viscosity" arises because, in fluid flow, the quantity that is modeled is usually momentum, and diffusion of momentum is called viscosity.)

Chapter 3

Additional notes

3.1 Interpolation formula on page 110

On page 110, below equation (6.11), it is written that the interpolation when changing the step size in a linear multistep formula can be obtained as

$$w_1^{new} = \frac{1}{8}\left(3w_2 + 6w_1 - y_{n-2}\right). \tag{3.1}$$

The values w are the temporary quantities that are stored in the linear multistep method, and we have that $w_2 = y_n$ and $w_1 = y_{n-1}$. We are now interested in computing an approximation halfway in between these two points, w_1^{new}.

The easiest way to proceed is to write down the Langrange formulation of the interpolating quadratic polynomial $w(t)$ through the points $(0, y_{n-2})$, (h, w_1) and $(2h, w_2)$. (The value of time is irrelevant as it cancels out in the computations, so we can put $t = 0$ at y_{n-2} during these calculations). We have

$$w(t) = y_{n-2}\frac{(t-h)(t-2h)}{(-h)(-2h)} + w_1\frac{t(t-2h)}{h(-h)} + w_2\frac{t(t-h)}{2h(h)} \tag{3.2}$$

and we need to evaluate $w(t)$ at $t = (3/2)h$. This results in (3.1).

3.2 Convergence of functional iteration in a simple case

These notes consider the functional iteration to solve the (non-)linear systems arising in each time step of an implicit time-stepping discretization. We specifically discuss the linear scalar test equation:

$$y' = \lambda y, \qquad y(0) = y_0, \qquad \lambda \in \mathbb{C}^-. \tag{3.3}$$

For this equation, the trapezoidal rule reads:

$$\begin{aligned} y^{n+1} &= y^n + h\left(\frac{1}{2}\lambda y^n + \frac{1}{2}\lambda y^{n+1}\right) \\ &= y^n + \frac{h\lambda}{2}\left(y^n + y^{n+1}\right). \end{aligned}$$

When solving this linear equation for y^{n+1} using functional iteration, we start from an initial guess $w^{[0]}$ and perform the following iteration:

$$w^{[s+1]} = y^n + \frac{h\lambda}{2}\left(y^n + w^{[s]}\right). \tag{3.4}$$

To study the convergence of this iteration, we introduce the notation $e^{[s]} = w^{[s]} - y^{n+1}$, and write

$$\begin{aligned} e^{[s+1]} &= y^n + \frac{h\lambda}{2}\left(y^n + w^{[s]}\right) - y^{n+1} \\ &= y^n + \frac{h\lambda}{2}\left(y^n + w^{[s]}\right) - \left(y^n + \frac{h\lambda}{2}\left(y^n + y^{n+1}\right)\right) \\ &= \frac{h\lambda}{2}\left(w^{[s]} - y^{n+1}\right) = \frac{h\lambda}{2}e^{[s]}. \end{aligned}$$

We see that the error gets multiplied with a factor $\dfrac{h\lambda}{2}$ in each iteration. Only if $\left|\dfrac{h\lambda}{2}\right| < 1$, the error tends to zero as the number of iterations s tends to infinity. We see that this constraint results in a condition on the time step h that is identical to the constraint for the explicit Euler method.

This is a special case of the Banach fixed point theorem.

3.3 On the relation between time discretization for ODEs and PDEs

Consider $u(x,t)$ to be the solution of the heat equation in one space dimension, on the domain $[0,1]$ with periodic boundary conditions:

$$\partial_t u(x,t) = \partial_{xx} u(x,t), \qquad u(x,0) = u^0(x), \qquad u(0,t) = u(1,t). \tag{3.5}$$

Let us first consider a discretization of $u(x,t)$ in space only, keeping time continuous. We thus consider the solution to be computed only on the grid points $x_j = j\Delta x$, $1 \le j \le J$, with $J\Delta x = 1$. We then have a finite set of pointwise solution values $u_j(t) \approx u(x_j,t)$. By using finite differences to approximate the spatial derivatives, the heat equation reduces to a system of ordinary differential equations:

$$u_j'(t) = \frac{1}{\Delta x^2}\left(u_{j+1}(t) - 2u_j(t) + u_{j-1}(t)\right), \qquad 1 \le j \le J, \tag{3.6}$$

and introducing the convention that $u_0(t) = u_J(t)$, $u_{-1}(t) = u_{J-1}(t)$ and $u_{J+1}(t) = u_1(t)$.

We can now discretize the system of ODEs (3.6) using any time discretization method that we have considered so far:

- When using the forward Euler method, we obtain:

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\Delta x^2}\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right), \tag{3.7}$$

  which is identical to the explicit method, equation (2.19), on page 11 of the handbook on PDE methods.

- When using the backward Euler method, we obtain:

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{\Delta x^2}\left(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}\right), \tag{3.8}$$

  which is identical to the implicit method, equation (2.63), on page 23 of the handbook on PDE methods.

- When using the trapezoidal rule, we recover the Crank-Nicholson scheme, see pages 29 and 30 of the handbook on PDE methods.

In the handbook on PDE methods, a *von Neumann analyis* is performed to obtain a condition on the time step Δt such that time integration remains stable, see section 2.7, starting on page 19. This stability condition is a direct consequence of the linear stability analysis of the forward Euler method, as studied in Chapter 4 of the handbook on ODEs. The confusing part is that the symbol λ is used differently in the ODE and PDE parts of the course. In these notes, I will continue using λ as in Chapter 4 of Iserles's book. The λ in the book of Morton and Mayers (see equation (2.50) on page 19) will be written as ρ in these notes.

To see this, we write $\boldsymbol{U} = (u_j)_{j=1}^J$, and notice that (3.6) is equivalent to

$$\boldsymbol{U}' = \boldsymbol{A}\boldsymbol{U}, \qquad \boldsymbol{A} = \frac{1}{\Delta x^2}\begin{bmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ 1 & & & 1 & -2 \end{bmatrix}, \tag{3.9}$$

in which the elements in $\boldsymbol{A}$ that are left blank are equal to zero. To use the linear stability analysis for ODEs, we need to compute the eigenvalues of the matrix $\boldsymbol{A}$, which we denote by λ_k, $k = 1, \ldots, J$. We will make life easier in this section by using the fact that the corresponding eigenvectors $\boldsymbol{U}_k$ are given by

$$\boldsymbol{U}_k = (u_{k,j})_{j=1}^J, \qquad u_{k,j} = \exp(ik\, j\Delta x). \tag{3.10}$$

(The computation of the corresponding eigenvalue λ_k will confirm that $\boldsymbol{U}_k$ is indeed an eigenvector.) When filling in the ansatz (3.10) in the semi-discretization (3.6), we obtain

$$\begin{aligned} u'_{k,j}(t) &= \frac{1}{\Delta x^2}\left(\exp(ik\,(j+1)\Delta x) - 2\exp(ik\,j\Delta x) + \exp(ik\,(j-1)\Delta x)\right) \\ &= \frac{1}{\Delta x^2}\left(\exp(ik\Delta x) - 2 + \exp(-ik\Delta x)\right)\exp(ik\,j\Delta x) \\ &= \frac{1}{\Delta x^2}\left(\exp(ik\Delta x) - 2 + \exp(-ik\Delta x)\right)u_{k,j}(t), \end{aligned}$$

independently of j, from which we deduce that

$$\lambda_k = \frac{1}{\Delta x^2}\left(\exp(ik\Delta x) - 2 + \exp(-ik\Delta x)\right), \tag{3.11}$$

which can be further elaborated (see the handbook of PDE methods).

For forward Euler to be stable, we need to ensure that all λ_k are within the stability region of the forward Euler method. Given that all λ_k are real (which follows directly from the symmetry of $\boldsymbol{A}$), this condition is satisfied if

$$-2 < \rho_k < 0, \qquad \rho_k = (1 + \lambda_k \Delta t), \tag{3.12}$$

which is identical to the constraint that $\Delta t \leq (1/2)\Delta x^2$ (compare equation (2.51) on page 20 of the PDE handbook). Hence, the linear stability analysis in the ODE part of the course and the stability analysis of the explicit method for the heat equation are the same thing.

A last question to answer is then: is the semi-discretization (3.9) a stiff system? Since stiffness is only defined informally, the answer to this question cannot simply be yes or no. We highlight a few points:

- Given the values of λ_k, $(k = 1, \ldots, K)$, we see that the stiffness ratio is typically $1/\Delta x^2$. (The rightmost eigenvalue λ_k is close to zero, whereas the leftmost eigenvalue corresponds to $1/\Delta x^2$.) Hence, the problem becomes more stiff as Δx tends to zero.

- The fact that Δt needs to be proportional to Δx^2 implies that a refinement of the mesh with a factor two induces a reduction of the size of the time step with a factor 4. Hence, as soon as Δx becomes small enough, the fast modes will become the bottleneck in the computation. As the cost of solving the tridiagonal linear systems in an implicit method is of order J, whereas the time step Δt is inversely proportional to $1/J^2$, the implicit method will be advantageous when Δx is small enough.