Chapter 9
Diffusion Equations and Parabolic Problems

We now begin to study finite difference methods for time-dependent partial differential equations (PDEs), where variations in space are related to variations in time. We begin with the heat equation (or diffusion equation) introduced in Appendix E,

$$u_t = \kappa u_{xx}. \tag{9.1}$$

This is the classical example of a *parabolic* equation, and many of the general properties seen here carry over to the design of numerical methods for other parabolic equations. We will assume $\kappa = 1$ for simplicity, but some comments will be made about how the results scale to other values of $\kappa > 0$. (If $\kappa < 0$, then (9.1) would be a "backward heat equation," which is an ill-posed problem.)

Along with this equation we need initial conditions at some time t_0, which we typically take to be $t_0 = 0$,

$$u(x, 0) = \eta(x), \tag{9.2}$$

and also boundary conditions if we are working on a bounded domain, e.g., the Dirichlet conditions

$$\begin{aligned} u(0, t) &= g_0(t) \quad \text{for } t > 0, \\ u(1, t) &= g_1(t) \quad \text{for } t > 0 \end{aligned} \tag{9.3}$$

if $0 \le x \le 1$.

We have already studied the steady-state version of this equation and spatial discretizations of u_{xx} in Chapter 2. We have also studied discretizations of the time derivatives and some of the stability issues that arise with these discretizations in Chapters 5 through 8. Next we will put these two types of discretizations together.

In practice we generally apply a set of finite difference equations on a discrete grid with grid points (x_i, t_n), where

$$x_i = ih, \qquad t_n = nk.$$

Here $h = \Delta x$ is the mesh spacing on the x-axis and $k = \Delta t$ is the time step. Let $U_i^n \approx u(x_i, t_n)$ represent the numerical approximation at grid point (x_i, t_n).

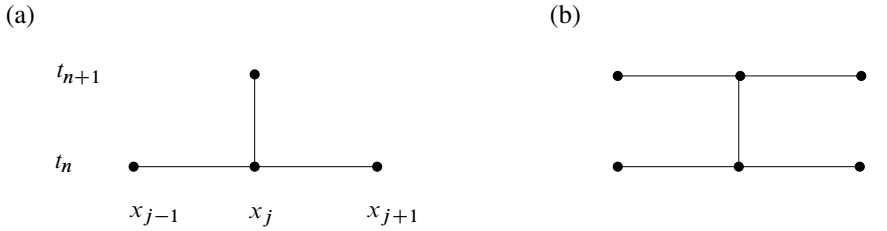

Figure 9.1. *Stencils for the methods* (9.5) *and* (9.7).

Since the heat equation is an evolution equation that can be solved forward in time, we set up our difference equations in a form where we can march forward in time, determining the values U_i^{n+1} for all i from the values U_i^n at the previous time level, or perhaps using also values at earlier time levels with a multistep formula.

As an example, one natural discretization of (9.1) would be

$$\frac{U_i^{n+1} - U_i^n}{k} = \frac{1}{h^2}(U_{i-1}^n - 2U_i^n + U_{i+1}^n). \tag{9.4}$$

This uses our standard centered difference in space and a forward difference in time. This is an *explicit* method since we can compute each U_i^{n+1} explicitly in terms of the previous data:

$$U_i^{n+1} = U_i^n + \frac{k}{h^2}(U_{i-1}^n - 2U_i^n + U_{i+1}^n). \tag{9.5}$$

Figure 9.1(a) shows the *stencil* of this method. This is a one-step method in time, which is also called a two-level method in the context of PDEs since it involves the solution at two different time levels.

Another one-step method, which is much more useful in practice, as we will see below, is the *Crank–Nicolson* method,

$$\begin{aligned}\frac{U_i^{n+1} - U_i^n}{k} &= \frac{1}{2}(D^2 U_i^n + D^2 U_i^{n+1}) \\ &= \frac{1}{2h^2}(U_{i-1}^n - 2U_i^n + U_{i+1}^n + U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}),\end{aligned} \tag{9.6}$$

which can be rewritten as

$$U_i^{n+1} = U_i^n + \frac{k}{2h^2}(U_{i-1}^n - 2U_i^n + U_{i+1}^n + U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}) \tag{9.7}$$

or

$$-rU_{i-1}^{n+1} + (1+2r)U_i^{n+1} - rU_{i+1}^{n+1} = rU_{i-1}^n + (1-2r)U_i^n + rU_{i+1}^n, \tag{9.8}$$

where $r = k/2h^2$. This is an *implicit* method and gives a tridiagonal system of equations to solve for all the values U_i^{n+1} simultaneously. In matrix form this is

$$\begin{bmatrix} (1+2r) & -r & & & & & \\ -r & (1+2r) & -r & & & & \\ & -r & (1+2r) & -r & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -r & (1+2r) & -r & \\ & & & & -r & (1+2r) \end{bmatrix} \begin{bmatrix} U_1^{n+1} \\ U_2^{n+1} \\ U_3^{n+1} \\ \vdots \\ U_{m-1}^{n+1} \\ U_m^{n+1} \end{bmatrix} \tag{9.9}$$

$$= \begin{bmatrix} r(g_0(t_n) + g_0(t_{n+1})) + (1-2r)U_1^n + rU_2^n \\ rU_1^n + (1-2r)U_2^n + rU_3^n \\ rU_2^n + (1-2r)U_3^n + rU_4^n \\ \vdots \\ rU_{m-2}^n + (1-2r)U_{m-1}^n + rU_m^n \\ rU_{m-1}^n + (1-2r)U_m^n + r(g_1(t_n) + g_1(t_{n+1})) \end{bmatrix}.$$

Note how the boundary conditions $u(0,t) = g_0(t)$ and $u(1,t) = g_1(t)$ come into these equations.

Since a tridiagonal system of m equations can be solved with $O(m)$ work, this method is essentially as efficient per time step as an explicit method. We will see in Section 9.4 that the heat equation is "stiff," and hence this implicit method, which allows much larger time steps to be taken than an explicit method, is a very efficient method for the heat equation.

Solving a parabolic equation with an implicit method requires solving a system of equations with the same structure as the two-point boundary value problem we studied in Chapter 2. Similarly, a multidimensional parabolic equation requires solving a problem with the structure of a multidimensional elliptic equation in each time step; see Section 9.7.

9.1 Local truncation errors and order of accuracy

We can define the local truncation error as usual—we insert the exact solution $u(x,t)$ of the PDE into the finite difference equation and determine by how much it fails to satisfy the discrete equation.

Example 9.1. The local truncation error of the method (9.5) is based on the form (9.4): $\tau_i^n = \tau(x_i, t_n)$, where

$$\tau(x,t) = \frac{u(x,t+k) - u(x,t)}{k} - \frac{1}{h^2}(u(x-h,t) - 2u(x,t) + u(x+h,t)).$$

Again we should be careful to use the form that directly models the differential equation in order to get powers of k and h that agree with what we hope to see in the global error. Although we don't know $u(x,t)$ in general, if we assume it is smooth and use Taylor series expansions about $u(x,t)$, we find that

$$\tau(x,t) = \left(u_t + \frac{1}{2}ku_{tt} + \frac{1}{6}k^2u_{ttt} + \cdots\right) - \left(u_{xx} + \frac{1}{12}h^2u_{xxxx} + \cdots\right).$$

Since $u_t = u_{xx}$, the $O(1)$ terms drop out. By differentiating $u_t = u_{xx}$ we find that $u_{tt} = u_{txx} = u_{xxxx}$ and so

$$\tau(x,t) = \left(\frac{1}{2}k - \frac{1}{12}h^2\right) u_{xxxx} + O(k^2 + h^4).$$

This method is said to be *second order accurate in space* and *first order accurate in time* since the truncation error is $O(h^2 + k)$.

The Crank–Nicolson method is centered in both space and time, and an analysis of its local truncation error shows that it is second order accurate in both space and time,

$$\tau(x,t) = O(k^2 + h^2).$$

A method is said to be *consistent* if $\tau(x,t) \to 0$ as $k,\ h \to 0$. Just as in the other cases we have studied (boundary value problems and initial value problems for ordinary differential equations (ODEs)), we expect that consistency, plus some form of stability, will be enough to prove that the method converges at each fixed point (X, T) as we refine the grid in both space and time. Moreover, we expect that for a stable method the global order of accuracy will agree with the order of the local truncation error, e.g., for Crank–Nicolson we expect that

$$U_i^n - u(X,T) = O(k^2 + h^2)$$

as $k,\ h \to 0$ when $ih \equiv X$ and $nk \equiv T$ are fixed.

For linear PDEs, the fact that consistency plus stability is equivalent to convergence is known as the *Lax equivalence theorem* and is discussed in Section 9.5 after an introduction of the proper concept of stability. As usual, it is the definition and study of stability that is the hard (and interesting) part of this theory.

9.2 Method of lines discretizations

To understand how stability theory for time-dependent PDEs relates to the stability theory we have already developed for time-dependent ODEs, it is easiest to first consider the so-called method of lines (MOL) discretization of the PDE. In this approach we first discretize in space alone, which gives a large system of ODEs with each component of the system corresponding to the solution at some grid point, as a function of time. The system of ODEs can then be solved using one of the methods for ODEs that we have previously studied.

This system of ODEs is also often called a *semidiscrete* method, since we have discretized in space but not yet in time.

For example, we might discretize the heat equation (9.1) in space at grid point x_i by

$$U_i'(t) = \frac{1}{h^2}(U_{i-1}(t) - 2U_i(t) + U_{i+1}(t)) \quad \text{for } i = 1,\ 2,\ \ldots,\ m, \tag{9.10}$$

where prime now means differentiation with respect to time. We can view this as a coupled system of m ODEs for the variables $U_i(t)$, which vary continuously in time along the lines shown in Figure 9.2. This system can be written as

$$U'(t) = AU(t) + g(t), \tag{9.11}$$

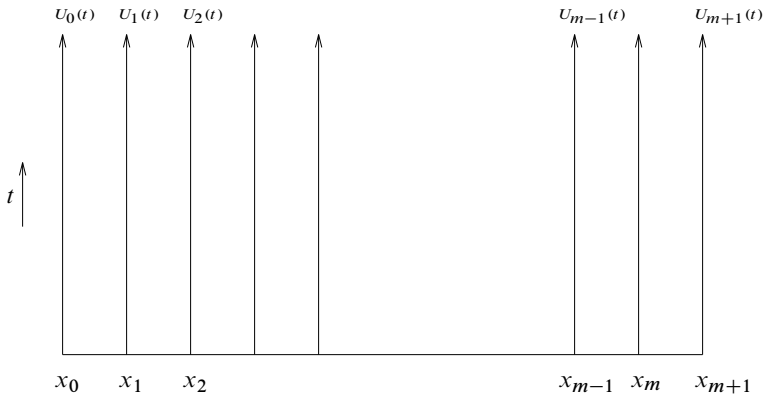

Figure 9.2. *Method of lines interpretation. $U_i(t)$ is the solution along the line forward in time at the grid point x_i.*

where the tridiagonal matrix A is exactly as in (2.9) and $g(t)$ includes the terms needed for the boundary conditions, $U_0(t) \equiv g_0(t)$ and $U_{m+1}(t) \equiv g_1(t)$,

$$A = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{bmatrix}, \qquad g(t) = \frac{1}{h^2}\begin{bmatrix} g_0(t) \\ 0 \\ 0 \\ \vdots \\ 0 \\ g_1(t) \end{bmatrix}. \tag{9.12}$$

This MOL approach is sometimes used in practice by first discretizing in space and then applying a software package for systems of ODEs. There are also packages that are specially designed to apply MOL. This approach has the advantage of being relatively easy to apply to a fairly general set of time-dependent PDEs, but the resulting method is often not as efficient as specially designed methods for the PDE. See Section 11.2 for more discussion of this.

As a tool in understanding stability theory, however, the MOL discretization is extremely valuable, and this is the main use we will make of it. We know how to analyze the stability of ODE methods applied to a linear system of the form (9.11) based on the eigenvalues of the matrix A, which now depend on the spatial discretization.

If we apply an ODE method to discretize the system (9.11), we will obtain a fully discrete method which produces approximations $U_i^n \approx U_i(t_n)$ at discrete points in time which are exactly the points (x_i, t_n) of the grid that we introduced at the beginning of this chapter.

For example, applying Euler's method $U^{n+1} = U^n + kf(U^n)$ to this linear system results in the fully discrete method (9.5). Applying instead the trapezoidal method (5.22) results in the Crank–Nicolson method (9.7). Applying a higher order linear multistep or Runge–Kutta method would give a different method, although with the spatial discretization (9.10) the overall method would be only second order accurate in space. Replacing

the right-hand side of (9.10) with a higher order approximation to $u_{xx}(x_i)$ and then using a higher order time discretization would give a more accurate method.

9.3 Stability theory

We can now investigate the stability of schemes like (9.5) or (9.7) since these can be interpreted as standard ODE methods applied to the linear system (9.11). We expect the method to be stable if $k\lambda \in \mathcal{S}$, i.e., if the time step k multiplied by any eigenvalue λ of A lies in the stability region of the ODE method, as discussed in Chapter 7. (Note that A is symmetric and hence normal, so eigenvalues are the right thing to look at.)

We have determined the eigenvalues of A in (2.23),

$$\lambda_p = \frac{2}{h^2}(\cos(p\pi h) - 1) \quad \text{for } p = 1, 2, \ldots, m, \tag{9.13}$$

where again m and h are related by $h = 1/(m+1)$. Note that there is a new wrinkle here relative to the ODEs we considered in Chapter 7: the eigenvalues λ_p depend on the mesh width h. As we refine the grid and $h \to 0$, the dimension of A increases, the number of eigenvalues we must consider increases, and the values of the eigenvalues change.

This is something we must bear in mind when we attempt to prove convergence as $k, h \to 0$. To begin, however, let's consider the simpler question of how the method behaves for some fixed k and h, i.e., the question of absolute stability in the ODE sense. Then it is clear that the method is absolutely stable (i.e., the effect of past errors will not grow exponentially in future time steps) provided that $k\lambda_p \in \mathcal{S}$ for each p, where $\mathcal{S}$ is the stability region of the ODE method, as discussed in Chapter 7.

For the matrix (9.12) coming from the heat equation, the eigenvalues lie on the negative real axis and the one farthest from the origin is $\lambda_m \approx -4/h^2$. Hence we require that $-4k/h^2 \in \mathcal{S}$ (assuming the stability region is connected along the negative real axis up to the origin, as is generally the case).

Example 9.2. If we use Euler's method to obtain the discretization (9.5), then we must require $|1 + k\lambda| \le 1$ for each eigenvalue (see Chapter 7) and hence we require $-2 \le -4k/h^2 \le 0$. This limits the time step allowed to

$$\frac{k}{h^2} \le \frac{1}{2}. \tag{9.14}$$

This is a severe restriction: the time step must decrease at the rate of h^2 as we refine the grid, which is much smaller than the spatial width h when h is small.

Example 9.3. If we use the trapezoidal method, we obtain the Crank–Nicolson discretization (9.6). The trapezoidal method for the ODE is absolutely stable in the whole left half-plane and the eigenvalues (9.13) are always negative. Hence the Crank–Nicolson method is stable for *any* time step $k > 0$. Of course it may not be accurate if k is too large. Generally we must take $k = O(h)$ to obtain a reasonable solution, and the unconditional stability allows this.

9.4 Stiffness of the heat equation

Note that the system of ODEs we are solving is quite stiff, particularly for small h. The eigenvalues of A lie on the negative real axis with one fairly close to the origin, $\lambda_1 \approx -\pi^2$

for all h, while the largest in magnitude is $\lambda_m \approx -4/h^2$. The "stiffness ratio" of the system is $4/\pi^2 h^2$, which grows rapidly as $h \to 0$. As a result the explicit Euler method is stable only for very small time steps $k \le \frac{1}{2}h^2$. This is typically much smaller than what we would like to use over physically meaningful times, and a method designed for stiff problems will be more efficient.

The stiffness is a reflection of the very different time scales present in solutions to the physical problem modeled by the heat equation. High frequency spatial oscillations in the initial data will decay very rapidly due to rapid diffusion over very short distances, while smooth data decay much more slowly since diffusion over long distances takes much longer. This is apparent from the Fourier analysis of Section E.3.3 or is easily seen by writing down the exact solution to the heat equation on $0 \le x \le 1$ with $g_0(t) = g_1(t) \equiv 0$ as a Fourier sine series:

$$u(x,t) = \sum_{j=1}^{\infty} \hat{u}_j(t) \sin(j\pi x).$$

Inserting this in the heat equation gives the ODEs

$$\hat{u}_j'(t) = -j^2\pi^2 \hat{u}_j(t) \qquad \text{for } j = 1, 2, \ldots. \tag{9.15}$$

and so

$$\hat{u}_j(t) = e^{-j^2\pi^2 t}\hat{u}_j(0)$$

with the $\hat{u}_j(0)$ determined as the Fourier coefficients of the initial data $\eta(x)$.

We can view (9.15) as an infinite system of ODEs, but which are decoupled so that the coefficient matrix is diagonal, with eigenvalues $-j^2\pi^2$ for $j = 1, 2, \ldots$. By choosing data with sufficiently rapid oscillation (large j), we can obtain arbitrarily rapid decay. For general initial data there may be some transient period when any high wave numbers are rapidly damped, but then the long-time behavior is dominated by the slower decay rates. See Figure 9.3 for some examples of the time evolution with different sets of data.

If we are solving the problem over the long periods needed to track this slow diffusion, then we would ultimately (after any physical transients have decayed) like to use rather large time steps, since typically the variation in time is then on roughly the same scale as variations in space. We would generally like to have $k \approx h$ so that we have roughly the same resolution in time as we do in space. A method that requires $k \approx h^2$ forces us to take a much finer temporal discretization than we should need to represent smooth solutions. If $h = 0.001$, for example, then if we must take $k = h^2$ rather than $k = h$ we would need to take 1000 time steps to cover each time interval that should be well modeled by a single time step. This is the same difficulty we encountered with stiff ODEs in Chapter 8.

Note: The remark above that we want $k \approx h$ is reasonable assuming the method we are using has the same order of accuracy in both space and time. The method (9.5) does not have this property. Since the error is $O(k + h^2)$ we might want to take $k = O(h^2)$ just to get the same level of accuracy in both space and time. In this sense the stability restriction $k = O(h^2)$ may not seem unreasonable, but this is simply another reason for not wanting to use this particular method in practice.

Note: The general diffusion equation is $u_t = \kappa u_{xx}$ and in practice the diffusion coefficient κ may be different from 1 by many orders of magnitude. How does this affect our conclusions above? We would expect by scaling considerations that we should take

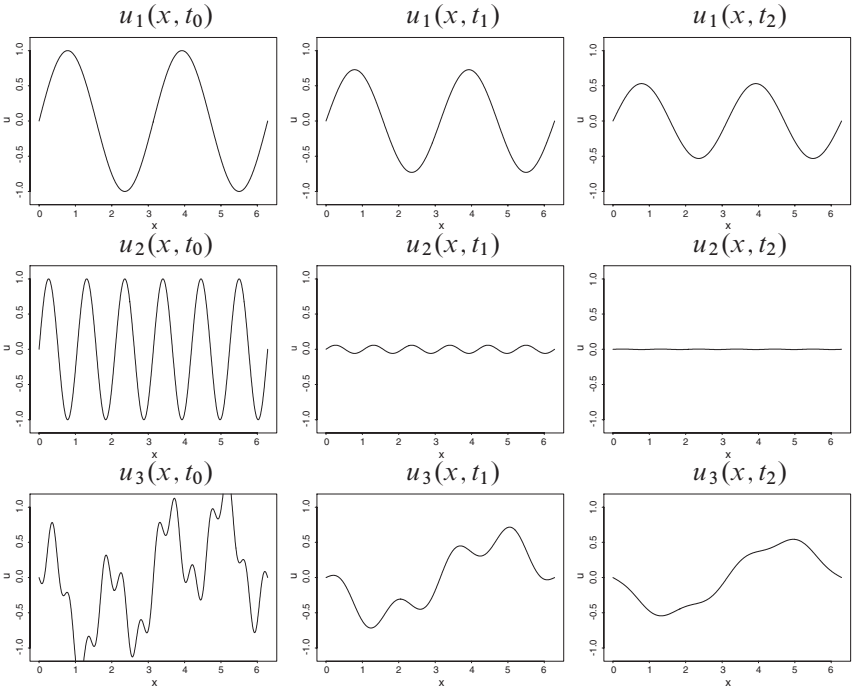

Figure 9.3. *Solutions to the heat equation at three different times (columns) shown for three different sets of initial conditions (rows). In the top row $u_1(x,t_0)$ consists of only a low wave number, which decays slowly. The middle row shows data consisting of a higher wave number, which decays more quickly. The bottom row shows data $u_3(x,t_0)$ that contains a mixture of wave numbers. The high wave numbers are most rapidly damped (an initial rapid transient), while at later times only the lower wave numbers are still visible and decaying slowly.*

$k \approx h/\kappa$ in order to achieve comparable resolution in space and time, i.e., we would like to take $\kappa k/h \approx 1$. (Note that $\hat{u}_j(t) = \exp(-j^2\pi^2\kappa t)\hat{u}_j(0)$ in this case.) With the MOL discretization we obtain the system (9.11) but A now has a factor κ in front. For stability we thus require $-4\kappa k/h^2 \in \mathcal{S}$, which requires $\kappa k/h^2$ to be order 1 for any explicit method. This is smaller than what we wish to use by a factor of h, regardless of the magnitude of κ. So our conclusions on stiffness are unchanged by κ. In particular, even when the diffusion coefficient is very small it is best to use an implicit method because we then want to take very long time steps $k \approx h/\kappa$.

These comments apply to the case of pure diffusion. If we are solving an advection-diffusion or reaction-diffusion equation where there are other time scales determined by other phenomena, then if the diffusive term has a very small coefficient we may be able to use an explicit method efficiently because of other restrictions on the time step.

Note: The physical problem of diffusion is "infinitely stiff" in the sense that (9.15) has eigenvalues $-j^2\pi^2$ with arbitrarily large magnitude, since j can be any integer. Luckily the discrete problem is not this stiff. It is not stiff because, once we discretize in space, only a finite number of spatial wave numbers can be represented and we obtain the finite

set of eigenvalues (9.13). As we refine the grid we can represent higher and higher wave numbers, leading to the increasing stiffness ratio as $h \to 0$.

9.5 Convergence

So far we have only discussed absolute stability and determined the relation between k and h that must be satisfied to ensure that errors do not grow exponentially as we march forward in time on this fixed grid. We now address the question of convergence at a fixed point (X, T) as the grid is refined. It turns out that in general exactly the same relation between k and h must now be required to hold as we vary k and h, letting both go to zero.

In other words, we cannot let k and h go to zero at arbitrary independent rates and necessarily expect the resulting approximations to converge to the solution of the PDE. For a particular sequence of grids (k_1, h_1), (k_2, h_2), ..., with $k_j \to 0$ and $h_j \to 0$, we will expect convergence only if the proper relation ultimately holds for each pair. For the method (9.5), for example, the sequence of approximations will converge only if $k_j/h_j^2 \leq 1/2$ for all j sufficiently large.

It is sometimes easiest to think of k and h as being related by some fixed rule (e.g., we might choose $k = 0.4h^2$ for the method (9.5)), so that we can speak of convergence as $k \to 0$ with the understanding that this relation holds on each grid.

The methods we have studied so far can be written in the form

$$U^{n+1} = B(k)U^n + b^n(k) \tag{9.16}$$

for some matrix $B(k) \in \mathbb{R}^{m\times m}$ on a grid with $h = 1/(m+1)$ and $b^n(k) \in \mathbb{R}^m$. In general these depend on both k and h, but we will assume some fixed rule is specified relating h to k as $k \to 0$.

For example, applying forward Euler to the MOL system (9.11) gives

$$B(k) = I + kA, \tag{9.17}$$

where A is the tridiagonal matrix in (9.12). The Crank–Nicolson method results from applying the trapezoidal method to (9.11), which gives

$$B(k) = \left(I - \frac{k}{2}A\right)^{-1}\left(I + \frac{k}{2}A\right). \tag{9.18}$$

To prove convergence we need consistency and a suitable form of stability. As usual, consistency requires that the local truncation error vanishes as $k \to 0$. The form of stability that we need is often called Lax–Richtmyer stability.

Definition 9.1. *A linear method of the form* (9.16) *is* Lax–Richtmyer stable *if, for each time T, there is a constant $C_T > 0$ such that*

$$\|B(k)^n\| \leq C_T \tag{9.19}$$

for all $k > 0$ and integers n for which $kn \leq T$.

Theorem 9.2 (Lax Equivalence Theorem). *A consistent linear method of the form* (9.16) *is convergent if and only if it is Lax–Richtmyer stable.*

For more discussion and a proof see [75]. The main idea is the same as our proof in Section 6.3.1 that Euler's method converges on a linear problem. If we apply the numerical method to the exact solution $u(x,t)$, we obtain

$$u^{n+1} = Bu^n + b^n + k\tau^n, \tag{9.20}$$

where we suppress the dependence on k for clarity and where

$$u^n = \begin{bmatrix} u(x_1,t_n) \\ u(x_2,t_n) \\ \vdots \\ u(x_m,t_n) \end{bmatrix}, \qquad \tau^n = \begin{bmatrix} \tau(x_1,t_n) \\ \tau(x_2,t_n) \\ \vdots \\ \tau(x_m,t_n) \end{bmatrix}.$$

Subtracting (9.20) from (9.16) gives the difference equation for the global error $E^n = U^n - u^n$:

$$E^{n+1} = BE^n - k\tau^n,$$

and hence, after N time steps,

$$E^N = B^N E^0 - k\sum_{n=1}^{N} B^{N-n}\tau^{n-1},$$

from which we obtain

$$\|E^N\| \le \|B^N\|\|E^0\| + k\sum_{n=1}^{N} \|B^{N-n}\|\,\|\tau^{n-1}\|. \tag{9.21}$$

If the method is Lax–Richtmyer stable, then for $Nk \le T$,

$$\begin{aligned} \|E^N\| &\le C_T\|E^0\| + TC_T \max_{1\le n\le N}\|\tau^{n-1}\| \\ &\to 0 \text{ as } k\to 0 \text{ for } Nk \le T, \end{aligned}$$

provided the method is consistent ($\|\tau\| \to 0$) and we use appropriate initial data ($\|E^0\| \to 0$ as $k \to 0$).

Example 9.4. For the heat equation the matrix A from (9.12) is symmetric, and hence the 2-norm is equal to the spectral radius, and the same is true of the matrix B from (9.17). From (9.13) we see that $\|B(k)\|_2 \le 1$, provided (9.14) is satisfied, and so the method is Lax–Richtmyer stable and hence convergent under this restriction on the time step. Similarly, the matrix B of (9.18) is symmetric and has eigenvalues $(1 + k\lambda_p/2)/(1 - k\lambda_p/2)$, and so the Crank–Nicolson method is stable in the 2-norm for any $k > 0$.

For the methods considered so far we have obtained $\|B\| \le 1$. This is called *strong stability*. But note that this is not necessary for Lax–Richtmyer stability. If there is a constant α so that a bound of the form

$$\|B(k)\| \le 1 + \alpha k \tag{9.22}$$

holds in some norm (at least for all k sufficiently small), then we will have Lax–Richtmyer stability in this norm, since

$$\|B(k)^n\| \le (1 + \alpha k)^n \le e^{\alpha T}$$

for $nk \le T$. Note that the matrix $B(k)$ depends on k, and its dimension $m = O(1/h)$ grows as $k, h \to 0$. The general theory of stability in the sense of uniform power boundedness of such families of matrices is often nontrivial.

9.5.1 PDE versus ODE stability theory

It may bother you that the stability we need for convergence now seems to depend on absolute stability, and on the shape of the stability region for the time-discretization, which determines the required relationship between k and h. Recall that in the case of ODEs all we needed for convergence was "zero-stability," which does not depend on the shape of the stability region except for the requirement that the point $z = 0$ must lie in this region.

Here is the difference: with ODEs we were studying a fixed system of ODEs of fixed dimension, and the fixed set of eigenvalues λ was independent of k. For convergence we needed $k\lambda$ in the stability region as $k \to 0$, but since these values all converge to 0 it is only the origin that is important, at least to prove convergence as $k \to 0$. Hence the need for zero-stability. With PDEs, on the other hand, in our MOL discretization the system of ODEs grows as we refine the grid, and the eigenvalues λ grow in magnitude as k and h go to zero. So it is not clear that $k\lambda$ will go to zero, and zero-stability is not sufficient. For the heat equation with k/h^2 fixed, these values do not go to zero as $k \to 0$. For convergence we must now require that these values at least lie in the region of absolute stability as $k \to 0$, and this gives the stability restriction relating k and h. If we keep k/h fixed as $k, h \to 0$, then $k\lambda \to -\infty$ for the eigenvalues of the matrix A from (9.12). We must use an implicit method that includes the entire negative real axis in its stability region.

We also notice another difference between stability theory for ODEs and PDEs that for the ODE $u'(t) = f(u(t))$ we could prove convergence of standard methods for any Lipschitz continuous function $f(u)$. For example, the proof of convergence of Euler's method for the linear case, found in Section 6.3.1, was easily extended to nonlinear functions in Section 6.3.3. In the PDE case, the Lax equivalence theorem is much more limited: it applies only to linear methods (9.16), and such methods typically only arise when discretizing linear PDEs such as the heat equation. It is possible to prove stability of many methods for nonlinear PDEs by showing that a suitable form of stability holds, but a variety of different techniques must be used, depending on the character of the differential equation, and there is no general theory of the sort obtained for ODEs.

The essential difficulty is that even a linear PDE such as the heat equation $u_t = \partial_x^2 u$ involves an operator on the right-hand side that is not Lipschitz continuous in a function space norm of the sort introduced in Section A.4. Discretizing on a grid replaces $\partial_x^2 u$ by $f(U) = AU$, which is Lipschitz continuous, but the Lipschitz constant $\|A\|$ grows at the rate of $1/h^2$ as the grid is refined. In the nonlinear case it is often difficult to obtain the sort of bounds needed to prove convergence. See [40], [68], [75], or [84] for further discussions of stability.

9.6 Von Neumann analysis

Although it is useful to go through the MOL formulation to understand how stability theory for PDEs is related to the theory for ODEs, in practice there is another approach that will sometimes give the proper stability restrictions more easily.

The von Neumann approach to stability analysis is based on Fourier analysis and hence is generally limited to constant coefficient linear PDEs. For simplicity it is usually applied to the *Cauchy problem*, which is the PDE on all space with no boundaries, $-\infty < x < \infty$ in the one-dimensional case. Von Neumann analysis can also be used to study the stability of problems with *periodic boundary conditions*, e.g., in $0 \le x \le 1$ with $u(0,t) = u(1,t)$ imposed. This is generally equivalent to a Cauchy problem with periodic initial data.

Stability theory for PDEs with more general boundary conditions can often be quite difficult, as the coupling between the discretization of the boundary conditions and the discretization of the PDE can be very subtle. Von Neumann analysis addresses the issue of stability of the PDE discretization alone. Some discussion of stability theory for initial boundary value problems can be found in [84], [75]. See also Section 10.12.

The Cauchy problem for linear PDEs can be solved using Fourier transforms—see Section E.3 for a review. The basic reason this works is that the functions $e^{i\xi x}$ with wave number $\xi =$ constant are eigenfunctions of the differential operator ∂_x,

$$\partial_x e^{i\xi x} = i\xi e^{i\xi x},$$

and hence of any constant coefficient linear differential operator. Von Neumann analysis is based on the fact that the related grid function $W_j = e^{ijh\xi}$ is an eigenfunction of any translation-invariant finite difference operator.[1] For example, if we approximate $v'(x_j)$ by $D_0 V_j = \frac{1}{2h}(V_{j+1} - V_{j-1})$, then in general the grid function $D_0 V$ is not a scalar multiple of V. But for the special case of W, we obtain

$$\begin{aligned} D_0 W_j &= \frac{1}{2h}\left(e^{i(j+1)h\xi} - e^{i(j-1)h\xi}\right) \\ &= \frac{1}{2h}\left(e^{ih\xi} - e^{-ih\xi}\right) e^{ijh\xi} \\ &= \frac{i}{h}\sin(h\xi) e^{ijh\xi} \\ &= \frac{i}{h}\sin(h\xi) W_j. \end{aligned} \tag{9.23}$$

So W is an "eigengridfunction" of the operator D_0, with eigenvalue $\frac{i}{h}\sin(h\xi)$.

Note the relation between these and the eigenfunctions and eigenvalues of the operator ∂_x found earlier: W_j is simply the eigenfunction $w(x)$ of ∂_x evaluated at the point x_j, and for small $h\xi$ we can approximate the eigenvalue of D_0 by

[1] In this section $i = \sqrt{-1}$ and the index j is used on the grid functions.

$$\frac{i}{h}\sin(h\xi) = \frac{i}{h}\left(h\xi - \frac{1}{6}h^3\xi^3 + O(h^5\xi^5)\right)$$
$$= i\xi - \frac{i}{6}h^2\xi^3 + \cdots.$$

This agrees with the eigenvalue $i\xi$ of ∂_x to $O(h^2\xi^3)$.

Suppose we have a grid function V_j defined at grid points $x_j = jh$ for $j = 0, \pm 1, \pm 2, \ldots$, which is an l_2 function in the sense that the 2-norm

$$\|U\|_2 = \left(h\sum_{j=-\infty}^{\infty} |U_j|^2\right)^{1/2}$$

is finite. Then we can express V_j as a linear combination of the grid functions $e^{ijh\xi}$ for all ξ in the range $-\pi/h \le \xi \le \pi/h$. Functions with larger wave number ξ cannot be resolved on this grid. We can write

$$V_j = \frac{1}{\sqrt{2\pi}}\int_{-\pi/h}^{\pi/h} \hat{V}(\xi)e^{ijh\xi}\,d\xi,$$

where

$$\hat{V}(\xi) = \frac{h}{\sqrt{2\pi}}\sum_{j=-\infty}^{\infty} V_j e^{-ijh\xi}.$$

These are direct analogue of the formulas for a function $v(x)$ in the discrete case.

Again we have *Parseval's relation*, $\|\hat{V}\|_2 = \|V\|_2$, although the 2-norms used for the grid function V_j and the function $\hat{V}(\xi)$ defined on $[-\pi/h, \pi/h]$ are different:

$$\|V\|_2 = \left(h\sum_{j=-\infty}^{\infty} |V_j|^2\right)^{1/2}, \qquad \|\hat{V}\|_2 = \left(\int_{-\pi/h}^{\pi/h} |\hat{V}(\xi)|^2\,d\xi\right)^{1/2}.$$

To show that a finite difference method is stable in the 2-norm by the techniques discussed earlier in this chapter, we would have to show that $\|B\|_2 \le 1 + \alpha k$ in the notation of (9.22). This amounts to showing that there is a constant α such that

$$\|U^{n+1}\|_2 \le (1 + \alpha k)\|U^n\|_2$$

for all U^n. This can be difficult to attack directly because of the fact that computing $\|U\|_2$ requires summing over all grid points, and each U_j^{n+1} depends on values of U^n at neighboring grid points so that all grid points are coupled together. In some cases one can work with these infinite sums directly, but it is rare that this can be done. Alternatively one can work with the matrix B itself, as we did above in Section 9.5, but this matrix is growing as we refine the grid.

Using Parseval's relation, we see that it is sufficient to instead show that

$$\|\hat{U}^{n+1}\|_2 \le (1 + \alpha k)\|\hat{U}^n\|_2,$$

where $\hat{U}^n$ is the Fourier transform of the grid function U^n. The utility of Fourier analysis now stems from the fact that after Fourier transforming the finite difference method, we obtain a recurrence relation for each $\hat{U}^n(\xi)$ that is decoupled from all other wave numbers. For a two-level method this has the form

$$\hat{U}^{n+1}(\xi) = g(\xi)\hat{U}^n(\xi). \tag{9.24}$$

The factor $g(\xi)$, which depends on the method, is called the *amplification factor* for the method at wave number ξ. If we can show that

$$|g(\xi)| \le 1 + \alpha k,$$

where α is independent of ξ, then it follows that the method is stable, since then

$$|\hat{U}^{n+1}(\xi)| \le (1 + \alpha k)|\hat{U}^n(\xi)| \quad \text{for all } \xi$$

and so

$$\|\hat{U}^{n+1}\|_2 \le (1 + \alpha k)\|\hat{U}^n\|_2.$$

Fourier analysis allows us to obtain simple scalar recursions of the form (9.24) for each wave number separately, rather than dealing with a system of equations for U_j^n that couples together all values of j.

Note: Here we are assuming that $u(x,t)$ is a scalar, so that $g(\xi)$ is a scalar. For a system of s equations we would find that $g(\xi)$ is an $s \times s$ matrix for each value of ξ, so some analysis of matrix eigenvalues is still required to investigate stability. But the dimension of the matrices is s, independent of the grid spacing, unlike the MOL analysis, where the matrix dimension increases as $k \to 0$.

Example 9.5. Consider the method (9.5). To apply von Neumann analysis we consider how this method works on a single wave number ξ, i.e., we set

$$U_j^n = e^{ijh\xi}. \tag{9.25}$$

Then we expect that

$$U_j^{n+1} = g(\xi)e^{ijh\xi}, \tag{9.26}$$

where $g(\xi)$ is the amplification factor for this wave number. Inserting these expressions into (9.5) gives

$$\begin{aligned} g(\xi)e^{ijh\xi} &= e^{ijh\xi} + \frac{k}{h^2}\left(e^{i\xi(j-1)h} - 2e^{ijh\xi} + e^{i\xi(j+1)h}\right) \\ &= \left(1 + \frac{k}{h^2}\left(e^{-i\xi h} - 2 + e^{i\xi h}\right)\right)e^{ijh\xi}, \end{aligned}$$

and hence

$$g(\xi) = 1 + 2\frac{k}{h^2}(\cos(\xi h) - 1).$$

Since $-1 \le \cos(\xi h) \le 1$ for any value of ξ, we see that

$$1 - 4\frac{k}{h^2} \le g(\xi) \le 1$$

for all ξ. We can guarantee that $|g(\xi)| \le 1$ for all ξ if we require

$$4\frac{k}{h^2} \le 2.$$

This is exactly the stability restriction (9.14) we found earlier for this method. If this restriction is violated, then the Fourier components with some wave number ξ will be amplified (and, as expected, it is the largest wave numbers that become unstable first as k is increased).

Example 9.6. The fact that the Crank–Nicolson method is stable for all k and h can also be shown using von Neumann analysis. Substituting (9.25) and (9.26) into the difference equations (9.7) and canceling the common factor of $e^{ijh\xi}$ gives the following relation for $g \equiv g(\xi)$:

$$g = 1 + \frac{k}{2h^2}\left(e^{-i\xi h} - 2 + e^{i\xi h}\right)(1 + g),$$

and hence

$$g = \frac{1 + \frac{1}{2}z}{1 - \frac{1}{2}z}, \tag{9.27}$$

where

$$\begin{aligned} z &= \frac{k}{h^2}(e^{-i\xi h} - 2 + e^{i\xi h}) \\ &= \frac{2k}{h^2}(\cos(\xi h) - 1). \end{aligned} \tag{9.28}$$

Since $z \le 0$ for all ξ, we see that $|g| \le 1$ and the method is stable for any choice of k and h.

Note that (9.27) agrees with the root ζ_1 found for the trapezoidal method in Example 7.6, while the z determined in (9.28), for certain values of ξ, is simply k times an eigenvalue λ_p from (9.13), the eigenvalues of the MOL matrix (9.11). In general there is a close connection between the von Neumann approach and the MOL reduction of a periodic problem to a system of ODEs.

9.7 Multidimensional problems

In two space dimensions the heat equation takes the form

$$u_t = u_{xx} + u_{yy} \tag{9.29}$$

with initial conditions $u(x, y, 0) = \eta(x, y)$ and boundary conditions all along the boundary of our spatial domain Ω. We can discretize in space using a discrete Laplacian of the form considered in Chapter 3, say, the 5-point Laplacian from Section 3.2:

$$\nabla_h^2 U_{ij} = \frac{1}{h^2}(U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1} - 4U_{ij}). \tag{9.30}$$

If we then discretize in time using the trapezoidal method, we will obtain the two-dimensional version of the Crank–Nicolson method,

$$U_{ij}^{n+1} = U_{ij}^n + \frac{k}{2}\left[\nabla_h^2 U_{ij}^n + \nabla_h^2 U_{ij}^{n+1}\right]. \tag{9.31}$$

Since this method is implicit, we must solve a system of equations for all the U_{ij} where the matrix has the same nonzero structure as for the elliptic systems considered in Chapters 3 and 4. This matrix is large and sparse, and we generally do not want to solve the system by a direct method such as Gaussian elimination. This is even more true for the systems we are now considering than for the elliptic equation, because of the slightly different nature of this system, which makes other approaches even more efficient relative to direct methods. It is also extremely important now that we use the most efficient method possible, because we must now solve a linear system of this form *in every time step*, and we may need to take thousands of time steps to solve the time-dependent problem.

We can rewrite the equations (9.31) as

$$\left(I - \frac{k}{2}\nabla_h^2\right) U_{ij}^{n+1} = \left(I + \frac{k}{2}\nabla_h^2\right) U_{ij}^n. \tag{9.32}$$

The matrix for this linear system has the same pattern of nonzeros as the matrix for ∇_h^2 (see Chapter 3), but the values are scaled by $k/2$ and then subtracted from the identity matrix, so that the diagonal elements are fundamentally different. If we call this matrix A,

$$A = I - \frac{k}{2}\nabla_h^2,$$

then we find that the eigenvalues of A are

$$\lambda_{p,q} = 1 - \frac{k}{h^2}\big[(\cos(p\pi h) - 1) + (\cos(q\pi h) - 1)\big]$$

for $p,\ q = 1,\ 2,\ \ldots,\ m$, where we have used the expression for the eigenvalues of ∇_h^2 from Section 3.4. Now the largest eigenvalue of the matrix A thus has magnitude $O(k/h^2)$ while the ones closest to the origin are at $1 + O(k)$. As a result the condition number of A is $O(k/h^2)$. By contrast, the discrete Laplacian ∇_h^2 alone has condition number $O(1/h^2)$ as we found in Section 3.4. The smaller condition number in the present case can be expected to lead to faster convergence of iterative methods.

Moreover, we have an excellent starting guess for the solution U^{n+1} to (9.31), a fact that we can use to good advantage with iterative methods but not with direct methods. Since $U_{ij}^{n+1} = U_{ij}^n + O(k)$, we can use U_{ij}^n, the values from the previous time step, as initial values $U_{ij}^{[0]}$ for an iterative method. We might do even better by extrapolating forward in time, using, say, $U_{ij}^{[0]} = 2U_{ij}^n - U_{ij}^{n-1}$, or by using an explicit method, say,

$$U_{ij}^{[0]} = (I + k\nabla_h^2)U_{ij}^n.$$

This explicit method (forward Euler) would probably be unstable as a time-marching procedure if we used only this with the value of k we have in mind, but it can sometimes be used successfully as a way to generate initial data for an iterative procedure.

Because of the combination of a reasonably well-conditioned system and very good initial guess, we can often get away with taking only one or two iterations in each time step, and still get global second order accuracy.

9.8 The locally one-dimensional method

Rather than solving the coupled sparse matrix equation for all the unknowns on the grid simultaneously as in (9.32), an alternative approach is to replace this fully coupled single time step with a sequence of steps, each of which is coupled in only one space direction, resulting in a set of tridiagonal systems which can be solved much more easily. One example is the *locally one-dimensional (LOD)* method:

$$U_{ij}^* = U_{ij}^n + \frac{k}{2}(D_x^2 U_{ij}^n + D_x^2 U_{ij}^*), \tag{9.33}$$

$$U_{ij}^{n+1} = U_{ij}^* + \frac{k}{2}(D_y^2 U_{ij}^* + D_y^2 U_{ij}^{n+1}), \tag{9.34}$$

or, in matrix form,

$$\left(I - \frac{k}{2}D_x^2\right)U^* = \left(I + \frac{k}{2}D_x^2\right)U^n, \tag{9.35}$$

$$\left(I - \frac{k}{2}D_y^2\right)U^{n+1} = \left(I + \frac{k}{2}D_y^2\right)U^*. \tag{9.36}$$

In (9.33) we apply Crank–Nicolson in the x-direction only, solving $u_t = u_{xx}$ alone over time k, and we call the result U^*. Then in (9.34) we take this result and apply Crank–Nicolson in the y-direction to it, solving $u_t = u_{yy}$ alone, again over time k. Physically this corresponds to modeling diffusion in the x- and y-directions over time k as a decoupled process in which we first allow u to diffuse only in the x-direction and then only in the y-direction. If the time steps are very short, then this might be expected to give similar physical behavior and hence convergence to the correct behavior as $k \to 0$. In fact, for the constant coefficient diffusion problem, it can even be shown that (in the absence of boundaries at least) this alternating diffusion approach gives *exactly* the same behavior as the original two-dimensional diffusion. Diffusing first in x alone over time k and then in y alone over time k gives the same result as if the diffusion occurs simultaneously in both directions. (This is because the differential operators ∂_x^2 and ∂_y^2 commute, as discussed further in Example 11.1.)

Numerically there is a great advantage in using (9.35) and (9.36) rather than the fully coupled (9.32). In (9.35) the unknowns U_{ij}^* are coupled together only across each row of the grid. For any fixed value of j we have a *tridiagonal system* of equations to solve for $U_{ij}^*(i = 1, 2, \ldots, m)$. The system obtained for each value of j is completely decoupled from the system obtained for other values of j. Hence we have a set of $m + 2$ tridiagonal systems to solve (for $j = 0, 1, \ldots, m + 1$), each of dimension m, rather than a single coupled system with m^2 unknowns as in (9.32). Since each of these systems is tridiagonal, it is easily solved in $O(m)$ operations by Gaussian elimination and there is no need for iterative methods. (In the next section we will see why we need to solve these for $j = 0$ and $j = m + 1$ as well as at the interior grid points.)

Similarly, (9.34) decouples into a set of m tridiagonal systems in the y-direction for $i = 1, 2, \ldots, m$. Hence taking a single time step requires solving $2m + 2$ tridiagonal systems of size m, and thus $O(m^2)$ work. Since there are m^2 grid points, this is the optimal order and no worse than an explicit method, except for a constant factor.

9.8.1 Boundary conditions

In solving the second set of systems (9.34), we need boundary values U^*_{i0} and U^{n+1}_{i0} along the bottom boundary and $U^*_{i,m+1}$ and $U^{n+1}_{i,m+1}$ along the top boundary, for terms that go on the right-hand side of each tridiagonal system. The values at level $n+1$ are available from the given boundary data for the heat equation, by evaluating the boundary conditions at time t_{n+1} (assuming Dirichlet boundary conditions are given). To obtain the values U^*_{i0} we solve (9.33) for $j = 0$ and $j = m+1$ (along the boundaries) in addition to the systems along each row interior to the grid.

To solve the first set of systems (9.33), we need boundary values U^n_{0j} and U^*_{0j} along the left boundary and values $U^n_{m+1,j}$ and $U^*_{m+1,j}$ along the right boundary. The values at level n come from the given boundary conditions, but we must determine the intermediate boundary conditions at level $*$ along these boundaries. It is not immediately clear what values should be used. One might be tempted to think of level $*$ as being halfway between t_n and t_{n+1}, since U^* is generated in the middle of the two-step procedure used to obtain U^{n+1} from U^n. If this were valid, then evaluating the given boundary data at time $t_{n+1/2} = t_n + k/2$ might provide values for U^* on the boundary. This is not a good idea, however, and would lead to a degradation of accuracy. The problem is that in the first step, (9.33) does not model the full heat equation over time $k/2$ but rather models part of the equation (diffusion in x alone) over the full time step k. The values along the boundary will in general evolve quite differently in the two different cases.

To determine proper values for U^*_{0j} and $U^*_{m+1,j}$, we can use (9.34) along the left and right boundaries. At $i = 0$, for example, this equation gives a system of equations along the left boundary that can be viewed as a tridiagonal linear system or the unknowns U^*_{0j} in terms of the values U^{n+1}_{0j}, which are already known from the boundary conditions at time t_{n+1}. Note that we are solving this equation backward from the way it will be used in the second step of the LOD process on the interior of the grid, and this works only because we already know U^{n+1}_{0j} from boundary data.

Since we are solving this equation backward, we can view this as solving the diffusion equation $u_t = u_{yy}$ over a time step of length $-k$, backward in time. This makes sense physically—the intermediate solution U^* represents what is obtained from U^n by doing diffusion in x alone, with no diffusion yet in y. There are in principle two ways to get this, either by starting with U^n and diffusing in x or by starting with U^{n+1} and "undiffusing" in y. We are using the latter approach along the boundaries to generate data for U^*.

Equivalently we can view this as solving the *backward heat equation* $u_t = -u_{yy}$ over time k. This may be cause for concern, since the backward heat equation is ill posed (see Section E.3.4). However, since we are doing this only over one time step starting with given values U^{n+1}_{0j} in each time step, this turns out to be a stable procedure.

There is still a difficulty at the corners. To solve (9.34) for U^*_{0j}, $j = 1, 2, \ldots, m$, we need to know the values of U^*_{00} and $U^*_{0,m+1}$ that are the boundary values for this system. These can be approximated using some sort of explicit and uncentered approximation to either $u_t = u_{xx}$ starting with U^n, or to $u_t = -u_{yy}$ starting with U^{n+1}. For example, we might use

$$U_{00}^* = U_{00}^{n+1} - \frac{k}{h^2}(U_{00}^{n+1} - 2U_{01}^{n+1} + U_{02}^{n+1}),$$

which uses the approximation to u_{yy} centered at (x_0, y_1).

Alternatively, rather than solving the tridiagonal systems obtained from (9.34) for U_{0j}^*, we could simply use an explicit approximation to the backward heat equation along this boundary,

$$U_{0j}^* = U_{0j}^{n+1} - \frac{k}{h^2}(U_{0,j-1}^{n+1} - 2U_{0j}^{n+1} + U_{0,j+1}^{n+1}) \tag{9.37}$$

for $j = 1, 2, \ldots, m$. This eliminates the need for values of U^* in the corners. Again, since this is not iterated but done only starting with given (and presumably smooth) boundary data U^{n+1} in each time step, this yields a stable procedure.

With proper treatment of the boundary conditions, it can be shown that the LOD method is second order accurate (see Example 11.1). It can also be shown that this method, like full Crank–Nicolson, is unconditionally stable for any time step.

9.8.2 The alternating direction implicit method

A modification of the LOD method is also often used, in which the two steps each involve discretization in only one spatial direction at the advanced time level (giving decoupled tridiagonal systems again) but coupled with discretization in the *opposite* direction at the old time level. The classical method of this form is

$$U_{ij}^* = U_{ij}^n + \frac{k}{2}(D_y^2 U_{ij}^n + D_x^2 U_{ij}^*), \tag{9.38}$$

$$U_{ij}^{n+1} = U_{ij}^* + \frac{k}{2}(D_x^2 U_{ij}^* + D_y^2 U_{ij}^{n+1}). \tag{9.39}$$

This is called the *alternating direction implicit (ADI)* method and was first introduced by Douglas and Rachford [26]. This again gives decoupled tridiagonal systems to solve in each step:

$$\left(I - \frac{k}{2}D_x^2\right)U^* = \left(I + \frac{k}{2}D_y^2\right)U^n, \tag{9.40}$$

$$\left(I - \frac{k}{2}D_y^2\right)U^{n+1} = \left(I + \frac{k}{2}D_x^2\right)U^*. \tag{9.41}$$

With this method, each of the two steps involves diffusion in both the x- and the y-direction. In the first step the diffusion in x is modeled implicitly, while diffusion in y is modeled explicitly, with the roles reversed in the second step. In this case each of the two steps can be shown to give a *first order accurate* approximation to the full heat equation over time $k/2$, so that U^* represents a first order accurate approximation to the solution at time $t_{n+1/2}$. Because of the symmetry of the two steps, however, the local error introduced in the second step almost exactly cancels the local error introduced in the first step, so that the combined method is in fact *second order accurate* over the full time step.

Because U^* does approximate the solution at time $t_{n+1/2}$ in this case, it is possible to simply evaluate the given boundary conditions at time $t_{n+1/2}$ to generate the necessary boundary values for U^*. This will maintain second order accuracy. A better error constant

can be achieved by using slightly modified boundary data which introduces the expected error in U^* into the boundary data that should be canceled out by the second step.

9.9 Other discretizations

For illustration purposes we have considered only the classic Crank–Nicolson method consisting of second order centered approximation to u_{xx} coupled with the trapezoidal method for time stepping. However, an infinite array of other combinations of spatial approximation and time stepping methods could be considered, some of which may be preferable. The following are a few possibilities:

- The second order accurate spatial difference operator could be replaced by a higher order method, such as the fourth order accurate approximations of Section 2.20.1 in one dimension of Section 3.5 in more dimensions.

- A spectral method could be used in the spatial dimension(s), as discussed in Section 2.21. Note that in this case the linear system that must be solved in each time step will be dense. On the other hand, for many problems it is possible to use a much coarser grid for spectral methods, leading to relatively small linear algebra problems.

- The time-stepping procedure could be replaced by a different implicit method suitable for stiff equations, of the sort discussed in Chapter 8. In particular, for some problems it is desirable to use an L-stable method. While the trapezoidal method is stable, it does not handle underresolved transients well (recall Figure 8.4). For some problems where diffusion is coupled with other processes there are constantly high-frequency oscillations or discontinuities introduced that should be smoothed by diffusion, and Crank–Nicolson can suffer from oscillations in time.

- The time stepping could be done by using a method such as the Runge–Kutta–Chebyshev method described in Section 8.6. This is an explicit method that works for mildly stiff problems with real eigenvalues, such as the heat equation.

- The time stepping could be done using the exponential time differencing (ETD) methods described in Section 11.6. The heat equation with constant coefficients and time-varying boundary conditions leads to a MOL discretization of the form (9.11), where A is a constant matrix. If the centered difference approximation is used in one dimension, then (9.12) holds, but even with other discretizations, or in more dimensions, the semidiscrete system still has the form $U'(t) = AU(t) + g(t)$. The exact solution can be written in terms of the matrix exponential e^{At} and this form is used in the ETD methods. The manner in which this is computed depends on whether A is large and sparse (the typical case with a finite difference discretization) or small and dense (as it might be if a spectral discretization is used in space). See Section 11.6.1 for more discussion of this.