9 Numerical Solution of Partial Differential Equations

9.0 Classification of Partial Differential Equations

We now consider general second-order partial differential equations (PDEs) of the form

$$Lu = au_{tt} + bu_{xt} + cu_{xx} + f = 0, \tag{1}$$

where u is an unknown function of x and t, and a, b, c, and f are given functions. If these functions depend only on x and t, then the PDE (1) is called *linear*. If a, b, c, or f depend also on u, u_x, or u_t, then the PDE is called *quasi-linear*.

Remarks:

1. The notation used in (1) suggests that we think of one of the variables, t, as time, and the other, x, as space.
2. In principle, we could also have second-order PDEs involving more than one space dimension. However, we limit the discussion here to PDEs with a total of two independent variables.
3. Of course, a second-order PDE can also be independent of time, and contain two space variables only (such as Laplace's equation).

There are three fundamentally different types of second-order quasi-linear PDEs:

- If $b^2 - 4ac > 0$, then L is hyperbolic.
- If $b^2 - 4ac = 0$, then L is parabolic.
- If $b^2 - 4ac < 0$, then L is elliptic.

Examples:

1. The *wave equation*
$$u_{tt} = \alpha^2 u_{xx} + f(x,t)$$
is a second-order linear hyperbolic PDE since $a \equiv 1$, $b \equiv 0$, and $c \equiv -\alpha^2$, so that
$$b^2 - 4ac = 4\alpha^2 > 0.$$
2. The *heat* or *diffusion equation*
$$u_t = ku_{xx}$$
is a second-order quasi-linear parabolic PDE since $a = b \equiv 0$, and $c \equiv -k$, so that
$$b^2 - 4ac = 0.$$
3. *Poisson's equation* (or *Laplace's equation* in case $f \equiv 0$)
$$u_{xx} + u_{yy} = f(x,y)$$
is a second-order linear elliptic PDE since $a = c \equiv 1$ and $b \equiv 0$, so that
$$b^2 - 4ac = -4 < 0.$$

Remarks: Since a, b, and c may depend on x, t, u, u_x, and u_t the classification of the PDE may even vary from point to point.

9.1 Forward Differences for Parabolic PDEs

We consider the following model problem

$$\begin{aligned} u_{xx} &= u_t, & t \geq 0,\ 0 \leq x \leq 1, \\ u(x,0) &= g(x), & 0 \leq x \leq 1, \\ u(0,t) &= \alpha(t), & t \geq 0, \\ u(1,t) &= \beta(t), & t \geq 0, \end{aligned} \tag{2}$$

which can be interpreted as the model for heat conduction in a one-dimensional rod of length 1 with prescribed end temperatures, α and β, and given initial temperature distribution g.

Finite Differences

We will proceed in a way very similar to our solution for two-point boundary value problems. However, now we are dealing with an initial-boundary value problem, and need to find the solution on the two-dimensional semi-infinite strip $D = \{(x,t)\ :\ 0 \leq x \leq 1,\ t \geq 0\}$.

As before, we use a symmetric difference approximation for the second-order space derivative, i.e., for any given location x and fixed time t

$$u_{xx}(x,t) \approx \frac{1}{h^2}\left[u(x+h,t) - 2u(x,t) + u(x-h,t)\right]. \tag{3}$$

For the first-order time derivative we use a forward difference approximation

$$u_t(x,t) \approx \frac{1}{k}\left[u(x,t+k) - u(x,t)\right]. \tag{4}$$

Next, we introduce a partition of the spatial domain

$$x_i = ih, \quad i = 0, \ldots, n+1,$$

with *mesh size* $h = \frac{1}{n+1}$. Another partition is used for the time domain, i.e.,

$$t_j = jk, \quad j \geq 0,$$

where k is the *time step*.

With the more compact notation

$$v_{i,j} = u(x_i, t_j), \quad v_{i+1,j} = u(x_i + h, t_j), \quad v_{i-1,j} = u(x_i - h, t_j), \quad v_{i,j+1} = u(x_i, t_j + k).$$

the heat equation (2) turns into the difference equation

$$\frac{1}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] = \frac{1}{k}\left[v_{i,j+1} - v_{i,j}\right], \quad i = 1, \ldots, n,\ j = 0, 1, \ldots. \tag{5}$$

For the initial condition we have

$$v_{i,0} = g(x_i), \qquad i = 0, 1, \ldots, n+1,$$

whereas the boundary conditions become

$$v_{0,j} = \alpha(t_j), \qquad v_{n+1,j} = \beta(t_j), \qquad j = 0, 1, \ldots .$$

Remark: The discrete values $v_{i,j}$ will be our representation of the solution of the PDE. The values at the points along the "boundaries" of the domain are specified by the initial and boundary conditions, whereas the those at interior points are computed one "row" at a time using a *stencil* involving values at four mesh points as stated in (5). This is illustrated schematically in Figure 1.

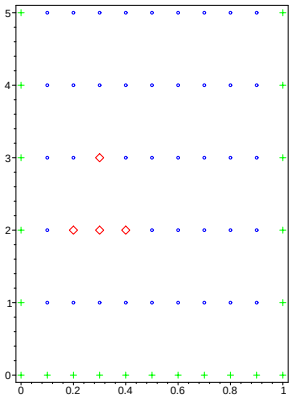

Figure 1: Schematic illustration of the forward difference method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

Figure 1 shows that the values of the approximate solution v are computed one row at a time, i.e., the "new" value $v_{i,j+1}$ at time level $j+1$ is computed from three "old" values at time level j. Therefore, the method we just described is a *forward* (in time) *difference method*, or an *explicit time-stepping method*.

In order to implement formula (5) in an algorithm we solve (5) for the "new" value, and obtain

$$v_{i,j+1} = \frac{k}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] + v_{i,j}$$

or

$$v_{i,j+1} = sv_{i+1,j} + (1-2s)v_{i,j} + sv_{i-1,j}, \quad i = 1, \ldots, n, \ j = 0, 1, \ldots, \tag{6}$$

where $s = \frac{k}{h^2}$ is called the *mesh ratio*.

Remarks:

1. Note that this method is very simple since no linear systems have to be solved.
2. Moreover, *all* values at the new time level can be computed simultaneously.
3. An algorithm for this method is listed in the textbook on pages 618–619.

The combined choice of mesh size h and time step k is critical for the performance of the algorithm. According to the finite difference approximations (3) and (4) of the partial derivatives we expect an accuracy of order $\mathcal{O}(k + h^2)$.

We now investigate the effects of h and k on the stability of the given method.

Stability Analysis

There are two standard techniques used for the stability analysis of PDEs. We now illustrate them both.

First we use the matrix form. We simplify the following discussion by introducing homogeneous boundary conditions, i.e., $\alpha(t) = \beta(t) \equiv 0$. Otherwise we would have more complicated first and last rows in the matrix below.

We begin by writing (6) in matrix form, i.e.,

$$V_{j+1} = AV_j, \tag{7}$$

where $V_j = [v_{1,j}, v_{2,j}, \ldots, v_{n,j}]^T$ is a vector that contains the values of the approximate solution at all of the interior mesh points at any given time level j. The matrix A is tridiagonal, and of the form

$$A = \begin{bmatrix} 1-2s & s & 0 & \ldots & 0 \\ s & 1-2s & s & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & s & 1-2s & s \\ 0 & \ldots & 0 & s & 1-2s \end{bmatrix}.$$

Note, however, that A appears on the right-hand side of (7) so that no system is solved, only a matrix-vector product is formed.

To investigate the stability of this method we assume that the initial data V_0 are represented with some error E_0, i.e.,

$$V_0 = \widetilde{V}_0 + E_0,$$

where $\widetilde{V}_0$ denotes the exact initial data. Then

$$V_1 = AV_0 = A(\widetilde{V}_0 + E_0) = A\widetilde{V}_0 + AE_0.$$

Advancing the solution further in time, i.e., iterating this formula, we get

$$V_j = AV_{j-1} = A^j\widetilde{V}_0 + A^jE_0.$$

Therefore, the initial error E_0 is magnified by A^j at the j-th time level. In order to have a stable and convergent method we need $E_j = A^jE_0 \to 0$ as $j \to \infty$.

Taking norms we get

$$\|E_j\| = \|A^jE_0\| \le \|A^j\|\|E_0\|,$$

so that we know that stability will be ensured if the spectral radius

$$\rho(A) < 1. \tag{8}$$

One can show (see textbook pages 620–621) that the eigenvalues of A are given by

$$\lambda_i = 1 - 2s(1 - \cos\theta_i), \quad \theta_i = \frac{i\pi}{n+1}, \quad i = 1, \ldots, n. \tag{9}$$

Combining (8) and (9) we require

$$\rho(A) = \max_{i=1,\ldots,n} |\lambda_i| = \max_{i=1,\ldots,n} |1 - 2s(1 - \cos\theta_i)| < 1$$

or

$$0 < s < \frac{1}{1 - \cos\theta_i}$$

for each $i = 1, \ldots, n$.

Now, the greatest restriction on s occurs if $\cos\theta_i = -1$, in which case $s < \frac{1}{2}$. Indeed, for $i = n$, $\theta_n = \frac{n\pi}{n+1} \to \pi$ for $n \to \infty$, and so

$$\frac{1}{1 - \cos\theta_i} \to \frac{1}{2}.$$

Therefore, we have the stability requirement $s \leq 1/2$. This is summarized in

Theorem 9.1 *The forward difference method (6) is stable provided* $k \leq \dfrac{h^2}{2}$.

Remarks:

1. The special choice of boundary conditions makes the preceding analysis simpler, but has no effect on the stability of the method.

2. The stability requirement shows that a small mesh size h forces the time step k to be *very* small. This makes the forward difference method potentially very slow. Consider, e.g., a spatial domain $[0, 1]$ with mesh size $h = 0.01$. Then the time step k can be chosen no larger than 5×10^{-5}. This means that in order to advance the solution from $t = 0$ to $t = 10$ we require 200000 time steps, or 20 million mesh points.

The second technique used for stability analyses was introduced by John von Neumann around 1940 and is known as the *von Neumann method* or *Fourier method*.

It is possible to show (using separation of variables) that solutions to the finite difference equation (6) are of the form

$$v_{i,j} = e^{\sqrt{-1}\sigma ih} Q^j, \tag{10}$$

where σ could be determined by the boundary conditions. Using the principle of superposition it is clear that all solutions of (6) will contain terms with the factor Q^j, for different values of j. This term will decay for large values of j (and thus indicate stability) only if $|Q| < 1$.

We can insert the solution (10) into (6) and obtain

$$e^{\sqrt{-1}\sigma ih} Q^{j+1} = se^{\sqrt{-1}\sigma(i+1)h} Q^j + (1 - 2s)e^{\sqrt{-1}\sigma ih} Q^j + se^{\sqrt{-1}\sigma(i-1)h} Q^j,$$

where, as before, $s = \frac{k}{h^2}$ is the mesh ratio. Now we remove the factor $e^{\sqrt{-1}\sigma ih} Q^j$ from the equation above which results in

$$Q = se^{\sqrt{-1}\sigma h} + 1 - 2s + se^{-\sqrt{-1}\sigma h}.$$

With the help of Euler's formula

$$e^{\sqrt{-1}\phi} = \cos\phi + \sqrt{-1}\sin\phi$$

and the trigonometric formula

$$\sin^2\phi = \frac{1}{2}\left(1 - \cos 2\phi\right)$$

the expression for Q can be simplified as follows:

$$\begin{aligned} Q &= se^{\sqrt{-1}\sigma h} + 1 - 2s + se^{-\sqrt{-1}\sigma h} \\ &= 1 - 2s + 2s\cos\sigma h \\ &= 1 - 2s\left(1 - \cos\sigma h\right) \\ &= 1 - 4s\sin^2\frac{\sigma h}{2}. \end{aligned}$$

From this it follows immediately that $Q < 1$ (since both h and k are positive, and thus also s). To ensure also $Q > -1$ we need

$$4s\sin^2\frac{\sigma h}{2} \leq 2$$

or

$$s\sin^2\frac{\sigma h}{2} \leq \frac{1}{2}.$$

Since the sine term can approach 1, we need to impose the stability condition

$$s \leq \frac{1}{2}$$

which is the same condition obtained earlier via the matrix method.

Remarks:

1. Once stability is ensured, the forward difference method converges with order $\mathcal{O}(k + h^2)$ as stated earlier.

2. Computer Problem 9.1.1 illustrates the effect the choice of mesh size h and time step k have on the stability of the forward difference method.

9.2 Implicit Methods for Parabolic PDEs

As we learned in the context of multistep methods for initial value problems, one can expect a better stability behavior from an implicit method.

Therefore, we discretize the first-order time derivative as

$$u_t(x,t) \approx \frac{1}{k}\left[u(x,t) - u(x,t-k)\right], \tag{11}$$

and so the formula corresponding to (5) now becomes

$$\frac{1}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] = \frac{1}{k}\left[v_{i,j} - v_{i,j-1}\right]. \tag{12}$$

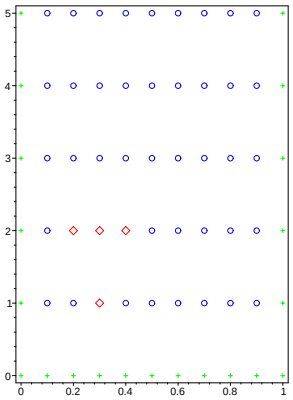

Figure 2: Schematic illustration of the backward difference method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

A typical stencil along with the values given by initial and boundary data is shown in Figure 2. Clearly, we now have an *implicit* or *backward difference method*.

The "new" values of the approximate solution (at time level j) are no longer so easily found. Using the mesh ratio $s = \frac{k}{h^2}$ as before, we can re-write (12) as

$$-sv_{i-1,j} + (1+2s)v_{i,j} - sv_{i+1,j} = v_{i,j-1}, \quad i = 1, \ldots, n, \ j = 0, 1, \ldots. \tag{13}$$

If we make the simplification to homogeneous boundary conditions, i.e., $\alpha(t) = \beta(t) \equiv 0$, then we can write (13) in matrix form as

$$AV_j = V_{j-1}, \tag{14}$$

where $V_j = [v_{1,j}, v_{2,j}, \ldots, v_{n,j}]^T$ as before, and

$$A = \begin{bmatrix} 1+2s & -s & 0 & \ldots & 0 \\ -s & 1+2s & -s & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -s & 1+2s & -s \\ 0 & \ldots & 0 & -s & 1+2s \end{bmatrix}.$$

Since the matrix A now appears on the left-hand side in (14) we need to solve a tridiagonal linear system for each time step. However, since the mesh ratio $s > 0$, it is clear that A is diagonally dominant.

Remarks:

1. An algorithm for the backward difference method with a call to a tridiagonal solver is in the textbook on page 625.

2. Note that, even though the matrix A is constant for all time steps, it is more efficient to solve the triangular system anew for each time step instead of computing a factorization and using forward and back substitution.

Stability Analysis

Under the same assumptions as for the forward difference method, the error now propagates according to

$$V_j = A^{-1}V_{j-1} = \left(A^{-1}\right)^j \widetilde{V}_0 + \left(A^{-1}\right)^j E_0,$$

with E_0 the error present in the initial data. Therefore, this time we want

$$\rho\left(A^{-1}\right) < 1$$

for stability. The eigenvalues of A are given by

$$\lambda_i = 1 + 2s(1 - \cos\theta_i), \quad \theta_i = \frac{i\pi}{n+1}, \quad i = 1, \ldots, n.$$

Since $0 < \theta_i < \pi$ we know that $|\cos\theta_i| < 1$, and therefore $\lambda_i > 1$ for all i. Thus, $\rho(A) > 1$.

Finally, since the eigenvalues of A^{-1} are given by the reciprocals of the eigenvalues of A we know that

$$\rho\left(A^{-1}\right) < 1,$$

and there is no stability restriction.

Theorem 9.2 *The backward difference method (13) is unconditionally stable.*

Remarks:

1. Again, having non-homogeneous boundary conditions does not affect the stability of the method.

2. The accuracy of the backward difference method is $\mathcal{O}(k+h^2)$, which implies that the time step needs to be chosen small for high accuracy.

It has been suggested to use the symmetric difference approximation

$$u_t(x,t) \approx \frac{1}{2k}\left[u(x,t+k) - u(x,t-k)\right], \tag{15}$$

for the time derivative. This choice gives the following difference scheme an accuracy of order $\mathcal{O}(k^2+h^2)$.

Using the above discretization of the time derivative, the difference version of the heat equation is

$$\frac{1}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] = \frac{1}{2k}\left[v_{i,j+1} - v_{i,j-1}\right].$$

or

$$v_{i,j+1} = \frac{2k}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] + v_{i,j-1}. \tag{16}$$

This method is known as *Richardson's method*. Richardson's method can be viewed as an explicit method that requires two rows of initial values.

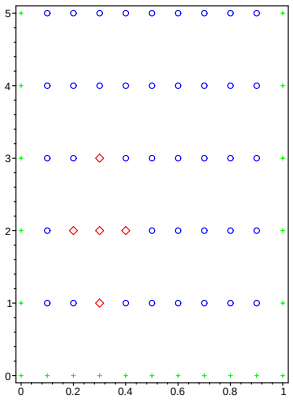

Figure 3: Schematic illustration of Richardson's method. Basic mesh points are indicated with blue ∘, green + correspond to given values, and points marked with red ♢ form a typical stencil.

The stencil for Richardson's method is shown in Figure 3.

Remark: Even though Richardson's method seem like a very sensible approach, and is known to have convergence order $\mathcal{O}(k^2+h^2)$ it is prone to serious stability problems. In fact, it turns out that this method is unstable *for all values of the mesh ratio* $s = \frac{k}{h^2}$.

Crank-Nicolson Method

The most successful difference method for parabolic equations is due to John Crank and Phyllis Nicolson (1947). It is based on a convex combination of the spatial terms of the forward and backward difference methods, i.e.,

$$\frac{\theta}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right] + \frac{1-\theta}{h^2}\left[v_{i+1,j-1} - 2v_{i,j-1} + v_{i-1,j-1}\right] = \frac{1}{k}\left[v_{i,j} - v_{i,j-1}\right]. \tag{17}$$

This method is known as *theta method* or, in the case $\theta = \frac{1}{2}$, as the *Crank-Nicolson method*.

For $\theta = 0$ we recover the forward difference method (with a shift in the time index j), and for $\theta = 1$ (17) yields the backward difference method. For the general case, the stencil contains six points at the two time levels $j-1$ and j as shown in Figure 4.

To get a better understanding of the Crank-Nicolson method we rewrite (17) for the case $\theta = \frac{1}{2}$. This leads to

$$-\frac{k}{2h^2}v_{i-1,j} + \left(1 + \frac{k}{h^2}\right)v_{i,j} - \frac{k}{2h^2}v_{i+1,j} = \frac{k}{2h^2}v_{i-1,j-1} + \left(1 - \frac{k}{h^2}\right)v_{i,j-1} + \frac{k}{h^2}v_{i+1,j-1}.$$

If we multiply by 2 and introduce the mesh ratio $s = \frac{k}{h^2}$ we get

$$-sv_{i-1,j} + (2+2s)\,v_{i,j} - sv_{i+1,j} = sv_{i-1,j-1} + (2-2s)\,v_{i,j-1} + sv_{i+1,j-1}. \tag{18}$$

By restricting the discussion to the case of homogeneous boundary conditions we get a streamlined matrix form since, as before, $\alpha(t) = \beta(t) \equiv 0$ results in $v_{0,j} = v_{n+1,j} = 0$ for all j. The resulting matrix form of (18) is

$$AV_j = BV_{j-1}$$

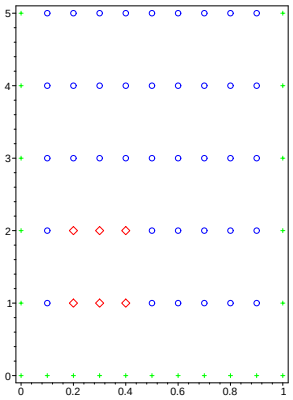

Figure 4: Schematic illustration of the Crank-Nicolson method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

with $V_j = [v_{1,j}, v_{2,j}, \ldots, v_{n,j}]^T$,

$$A = \begin{bmatrix} 2+2s & -s & 0 & \ldots & 0 \\ -s & 2+2s & -s & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -s & 2+2s & -s \\ 0 & \ldots & 0 & -s & 2+2s \end{bmatrix},$$

and

$$B = \begin{bmatrix} 2-2s & s & 0 & \ldots & 0 \\ s & 2-2s & s & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & s & 2-2s & s \\ 0 & \ldots & 0 & s & 2-2s \end{bmatrix}.$$

It is clear that A is a diagonally dominant tridiagonal matrix. Therefore, the implementation of the Crank-Nicolson method can be broken down into two phases:

1. Compute $\widetilde{V}_{j-1} = BV_{j-1}$. This can be done with an algorithm analogous to the forward difference method.

2. Solve $AV_j = \widetilde{V}_{j-1}$. For this we can basically use the backward difference algorithm.

Remarks:

1. Using a stability analysis as above, one can show that the Crank-Nicolson method is unconditionally stable. An outline of the argument is given in the textbook on page 626, and a detailed discussion can be found in the classical book "Analysis of Numerical Methods" by Isaacson and Keller (1966).

2. Since the averaging of the forward and backward difference methods used for the Crank-Nicolson can be interpreted as an application of the (implicit) trapezoid rule for the time discretization, it follows that the order of accuracy of the Crank-Nicolson method is $\mathcal{O}(k^2 + h^2)$. Therefore, this method is much more efficient (for the same accuracy) than the forward or backward difference methods by themselves.

3. If $\theta \neq \frac{1}{2}$ then the accuracy deteriorates to $\mathcal{O}(k + h^2)$.

It is possible to "fix" the Richardson method (16). To this end we replace the value $v_{i,j}$ by the average of $v_{i,j+1}$ and $v_{i,j-1}$. We then end up with

$$v_{i,j+1} = \frac{2k}{h^2 + 2k}\left[v_{i+1,j} + v_{i-1,j}\right] + \frac{h^2 - 2k}{h^2 + 2k} v_{i,j-1}. \tag{19}$$

This method is known as the *Dufort-Frankel method*. It is stable for all values of s, and converges with order $\mathcal{O}(k^2 + h^2)$ as long as $k/h \to 0$.

The stencil for the Dufort-Frankel method consists of four points, and is depicted in Figure 5.

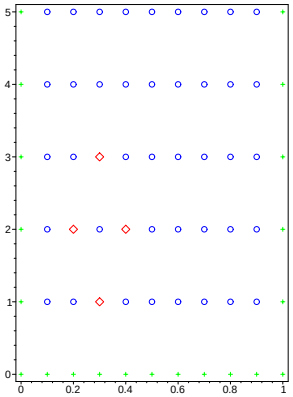

Figure 5: Schematic illustration of the Dufort-Frankel method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

Remarks:

1. The Dufort-Frankel method is considered an explicit method, and therefore fast and simple to implement. There are no linear systems to solve. However, the condition $k/h \to 0$ is essential for convergence.

2. Since the Dufort-Frankel method involves mesh points at three time levels, it requires a separate (possibly lower-order) method to get started.

9.3 Finite Differences for Elliptic PDEs

We use Laplace's equation in the unit square as our model problem, i.e.,

$$\begin{aligned} u_{xx} + u_{yy} &= 0, \qquad (x,y) \in \Omega = (0,1)^2, \\ u(x,y) &= g(x,y), \qquad (x,y) \text{ on } \partial\Omega. \end{aligned} \tag{20}$$

This problem arises, e.g., when we want to determine the steady-state temperature distribution u in a square region with prescribed boundary temperature g. Of course, this simple problem can be solved analytically using Fourier series.

However, we are interested in numerical methods. Therefore, in this section, we use the usual finite difference discretization of the partial derivatives, i.e.,

$$u_{xx}(x,y) = \frac{1}{h^2}\left[u(x+h,y) - 2u(x,y) + u(x-h,y)\right] + \mathcal{O}(h^2) \tag{21}$$

and

$$u_{yy}(x,y) = \frac{1}{h^2}\left[u(x,y+h) - 2u(x,y) + u(x,y-h)\right] + \mathcal{O}(h^2). \tag{22}$$

The computational grid introduced in the domain $\overline{\Omega} = [0,1]^2$ is now

$$(x_i, y_j) = (ih, jh), \quad i,j = 0,\ldots,n+1,$$

with *mesh size* $h = \frac{1}{n+1}$.

Using again the compact notation

$$v_{i,j} = u(x_i, y_j), \quad v_{i+1,j} = u(x_i + h, y_j), \quad \text{etc.},$$

Laplace's equation (20) turns into the difference equation

$$\frac{1}{h^2}\left[v_{i-1,j} - 2v_{i,j} + v_{i+1,j}\right] + \frac{1}{h^2}\left[v_{i,j-1} - 2v_{i,j} + v_{i,j+1}\right] = 0. \tag{23}$$

This equation can be rewritten as

$$4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1} = 0. \tag{24}$$

This is the method referred to several times last semester.

Example: Let's consider a computational mesh of 5×5 points, i.e., $h = \frac{1}{4}$, or $n = 3$. Discretizing the boundary conditions in (20), the values of the approximate solution around the boundary

$$\begin{aligned} &v_{0,j},\ v_{4,j} \quad j = 0,\ldots,4, \\ &v_{i,0},\ v_{i,4} \quad i = 0,\ldots,4, \end{aligned}$$

are determined by the appropriate values of g. There remain 9 points in the interior that have to be determined using the stencil (24). Figure 6 illustrates one instance of this task. By applying the stencil to each of the interior points, we obtain 9 conditions for the 9 undetermined values.

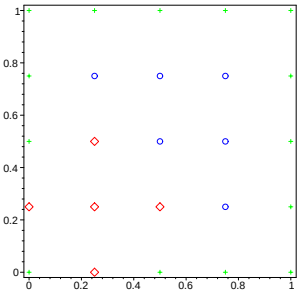

Figure 6: Illustration for finite difference method for Laplace's equation on 5×5 grid. Interior mesh points are indicated with blue $\circ$, green $+$ correspond to given boundary values, and points marked with red $\diamondsuit$ form a typical stencil.

Thus, we obtain the following 9 equations

$$\begin{aligned}
4v_{1,1} - v_{2,1} - v_{1,2} &= v_{0,1} + v_{1,0} \\
4v_{2,1} - v_{1,1} - v_{3,1} - v_{2,2} &= v_{2,0} \\
4v_{3,1} - v_{2,1} - v_{3,2} &= v_{4,1} + v_{3,0} \\
4v_{1,2} - v_{2,2} - v_{1,1} - v_{1,3} &= v_{0,2} \\
4v_{2,2} - v_{1,2} - v_{3,2} - v_{2,1} - v_{2,3} &= 0 \\
4v_{3,2} - v_{2,2} - v_{3,1} - v_{3,3} &= v_{4,2} \\
4v_{1,3} - v_{2,3} - v_{1,2} &= v_{1,4} + v_{0,3} \\
4v_{2,3} - v_{1,3} - v_{3,3} - v_{2,2} &= v_{2,4} \\
4v_{3,3} - v_{2,3} - v_{3,2} &= v_{4,3} + v_{3,4}.
\end{aligned}$$

The first equation corresponds to the stencil shown in Figure 6. The other equations are obtained by moving the stencil row-by-row across the grid from left to right.

We can also write the above equations in matrix form. To this end we introduce the vector

$$V = [v_{1,1}, v_{2,1}, v_{3,1}, v_{1,2}, v_{2,2}, v_{3,2}, v_{1,3}, v_{2,3}, v_{3,3}]^T$$

of unknowns. Here we have used the *natural* (row-by-row) ordering of the mesh points. Then we get

$$AV = B$$

with

$$A = \begin{bmatrix}
\begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix} & \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
\begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} & \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix} & \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \\
\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} & \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}
\end{bmatrix}$$

and

$$B = \begin{bmatrix} v_{0,1} + v_{1,0} \\ v_{2,0} \\ v_{4,1} + v_{3,0} \\ v_{0,2} \\ 0 \\ v_{4,2} \\ v_{1,4} + v_{0,3} \\ v_{2,4} \\ v_{4,3} + v_{3,4} \end{bmatrix}.$$

We can see that A is a block-tridiagonal matrix of the form

$$A = \begin{bmatrix} T & -I & O \\ -I & T & -I \\ O & -I & T \end{bmatrix}.$$

In general, for problems with $n \times n$ interior mesh points, A will be of size $n^2 \times n^2$ (since there are n^2 unknown values at interior mesh points), but contain no more than $5n^2$ nonzero entries (since equation (24) involves at most 5 points at one time). Thus, A is a classical example of a sparse matrix. Moreover, A still has a block-tridiagonal structure

$$A = \begin{bmatrix} T & -I & O & \dots & O \\ -I & T & -I & & \vdots \\ O & \ddots & \ddots & \ddots & O \\ \vdots & & -I & T & -I \\ O & \dots & O & -I & T \end{bmatrix}$$

with $n \times n$ blocks

$$T = \begin{bmatrix} 4 & -1 & 0 & \dots & 0 \\ -1 & 4 & -1 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & -1 & 4 & -1 \\ 0 & \dots & 0 & -1 & 4 \end{bmatrix}$$

as well as $n \times n$ identity matrices I, and zero matrices O.

Remarks:

1. Since A is sparse (and symmetric positive definite) it lends itself to an application of Gauss-Seidel iteration (see Chapter 4.6 of the previous semester). After initializing the values at all mesh points (including those along the boundary) to some appropriate value (in many cases zero will work), we can simply iterate with formula (24), i.e., we obtain the algorithm fragment for M steps of Gauss-Seidel iteration

for $k = 1$ to M do

for $i = 1$ to n do

for $j = 1$ to n do

$v_{i,j} = (v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1})/4$

end

end

end

Note that the matrix A never has to be fully formed or stored during the computation.

2. State-of-the-art algorithms for the Laplace (or nonhomogeneous Poisson) equation are so-called *fast Poisson solvers* based on the fast Fourier transform, or multigrid methods.

9.4 Galerkin and Ritz Methods for Elliptic PDEs

Galerkin Method

We begin by introducing a generalization of the collocation method we saw earlier for two-point boundary value problems. Consider the PDE

$$Lu(\boldsymbol{x}) = f(\boldsymbol{x}), \tag{25}$$

where L is a linear elliptic partial differential operator such as the Laplacian

$$L = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}, \quad \boldsymbol{x} = (x, y, z) \in \mathbb{R}^3 .$$

At this point we will not worry about the boundary conditions that should be posed with (25).

As with the collocation method we will obtain the approximate solution in the form of a function (instead of as a collection of discrete values). Therefore, we need an approximation space $\mathcal{U} = \text{span}\{u_1, \ldots, u_n\}$, so that we are able to represent the approximate solution as

$$u = \sum_{j=1}^{n} c_j u_j, \quad u_j \in \mathcal{U}. \tag{26}$$

Using the linearity of L we have

$$Lu = \sum_{j=1}^{n} c_j L u_j.$$

We now need to come up with n (linearly independent) conditions to determine the n unknown coefficients c_j in (26). If $\{\Phi_1, \ldots, \Phi_n\}$ is a linearly independent set of linear functionals, then

$$\Phi_i \left[\sum_{j=1}^{n} c_j L u_j - f \right] = 0, \quad i = 1, \ldots, n, \tag{27}$$

is an appropriate set of conditions. In fact, this leads to a system of linear equations

$$Ac = b$$

with matrix

$$A = \begin{bmatrix} \Phi_1 Lu_1 & \Phi_1 Lu_2 & \dots & \Phi_1 Lu_n \\ \Phi_2 Lu_1 & \Phi_2 Lu_2 & \dots & \Phi_2 Lu_n \\ \vdots & \vdots & & \vdots \\ \Phi_n Lu_1 & \Phi_n Lu_2 & \dots & \Phi_n Lu_n \end{bmatrix},$$

coefficient vector $c = [c_1, \dots, c_n]^T$, and right-hand side vector

$$b = \begin{bmatrix} \Phi_1 f \\ \Phi_2 f \\ \vdots \\ \Phi_n f \end{bmatrix}.$$

Two popular choices are

1. Point evaluation functionals, i.e., $\Phi_i(v) = v(\boldsymbol{x}_i)$, where $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$ is a set of points chosen such that the resulting conditions are linearly independent, and v is some function with appropriate smoothness. With this choice (27) becomes

$$\sum_{j=1}^{n} c_j Lu_j(\boldsymbol{x}_i) = f(\boldsymbol{x}_i), \quad i = 1, \dots, n,$$

   and we now have a extension of the *collocation method* discussed in Section 8.11 to elliptic PDEs is the multi-dimensional setting.

2. If we let $\Phi_i(v) = \langle u_i, v \rangle$, an inner product of the basis function u_i with an appropriate *test function* v, then (27) becomes

$$\sum_{j=1}^{n} c_j \langle u_i, Lu_j \rangle = \langle u_i, f \rangle, \quad i = 1, \dots, n.$$

   This is the classical *Galerkin method*.

Ritz-Galerkin Method

For the following discussion we pick as a model problem a multi-dimensional Poisson equation with homogeneous boundary conditions, i.e.,

$$\begin{aligned} -\nabla^2 u &= f \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{28}$$

with domain $\Omega \subset \mathbb{R}^d$. This problem describes, e.g., the steady-state solution of a vibrating membrane (in the case $d = 2$ with shape Ω) fixed at the boundary, and subjected to a vertical force f.

The first step for the Ritz-Galerkin method is to obtain the *weak form* of (28). This is accomplished by choosing a function v from a space $\mathcal{U}$ of smooth functions, and then forming the inner product of both sides of (28) with v, i.e.,

$$-\langle \nabla^2 u, v\rangle = \langle f, v\rangle. \tag{29}$$

To be more specific, we let $d = 2$ and take the inner product

$$\langle u, v\rangle = \iint_\Omega u(x,y)v(x,y)dxdy.$$

Then (29) becomes

$$-\iint_\Omega (u_{xx}(x,y) + u_{yy}(x,y))v(x,y)dxdy = \iint_\Omega f(x,y)v(x,y)dxdy. \tag{30}$$

In order to be able to complete the derivation of the weak form we now assume that the space $\mathcal{U}$ of test functions is of the form

$$\mathcal{U} = \{v \ : \ v \in C^2(\Omega),\ v = 0 \text{ on } \partial\Omega\},$$

i.e., besides having the necessary smoothness to be a solution of (28), the functions also satisfy the boundary conditions.

Now we rewrite the left-hand side of (30):

$$\begin{aligned}\iint_\Omega (u_{xx} + u_{yy})\, v dxdy &= \iint_\Omega \left[(u_x v)_x + (u_y v)_y - u_x v_x - u_y v_y\right] dxdy \\ &= \iint_\Omega \left[(u_x v)_x + (u_y v)_y\right] dxdy - \iint_\Omega \left[u_x v_x - u_y v_y\right] dxdy. \end{aligned} \tag{31}$$

By using Green's Theorem (integration by parts)

$$\iint_\Omega (P_x + Q_y)dxdy = \int_{\partial\Omega} (Pdy - Qdx)$$

the first integral on the right-hand side of (31) turns into

$$\iint_\Omega \left[(u_x v)_x + (u_y v)_y\right] dxdy = \int_{\partial\Omega} \left(u_x v dy - u_y v dx\right).$$

Now the special choice of $\mathcal{U}$, i.e., the fact that v satisfies the boundary conditions, ensures that this term vanishes. Therefore, the weak form of (28) is given by

$$\iint_\Omega \left[u_x v_x + u_y v_y\right] dxdy = \iint_\Omega f v dxdy.$$

Another way of writing the previous formula is of course

$$\iint_\Omega \nabla u \cdot \nabla v dxdy = \iint_\Omega f v dxdy. \tag{32}$$

To obtain a numerical method we now need to require $\mathcal{U}$ to be finite-dimensional with basis $\{u_1, \ldots, u_n\}$. Then we can represent the approximate solution u^h of (28) as

$$u^h = \sum_{j=1}^{n} c_j u_j. \tag{33}$$

The superscript h indicates that the approximate solution is obtained on some underlying discretization of Ω with mesh size h.

Remarks:

1. In practice there are many ways of discretizing Ω and selecting $\mathcal{U}$.
   (a) For example, regular (tensor product) grids can be used. Then $\mathcal{U}$ can consist of tensor products of piecewise polynomials or B-spline functions that satisfy the boundary conditions of the PDE.
   (b) It is also possible to use irregular (triangulated) meshes, and again define piecewise (total degree) polynomials or splines on triangulations satisfying the boundary conditions.
   (c) More recently, meshfree approximation methods have been introduced as possible choices for $\mathcal{U}$.
2. In the literature the piecewise polynomial approach is usually referred to as the *finite element method*.
3. The discretization of Ω will almost always result in a computational domain that has piecewise linear (Lipschitz-continuous) boundary.

We now return to the discussion of the general numerical method. Once we have chosen a basis for the approximation space $\mathcal{U}$, then it becomes our goal to determine the coefficients c_j in (33). By inserting u^h into the weak form (32), and selecting as trial functions v the basis functions of $\mathcal{U}$ we obtain a system of equations

$$\iint_\Omega \nabla u^h \cdot \nabla u_i dxdy = \iint_\Omega f u_i dxdy, \quad i = 1, \ldots, n.$$

Using the representation (33) of u^h we get

$$\iint_\Omega \nabla \left[\sum_{j=1}^{n} c_j u_j \right] \cdot \nabla u_i dxdy = \iint_\Omega f u_i dxdy, \quad i = 1, \ldots, n,$$

or by linearity

$$\sum_{j=1}^{n} c_j \iint_\Omega \nabla u_j \cdot \nabla u_i dxdy = \iint_\Omega f u_i dxdy, \quad i = 1, \ldots, n. \tag{34}$$

This last set of equations is known as the *Ritz-Galerkin method* and can be written in matrix form

$$Ac = b,$$

where the *stiffness matrix* A has entries

$$A_{i,j} = \iint_{\Omega} \nabla u_j \cdot \nabla u_i dxdy.$$

Remarks:

1. The stiffness matrix is usually assembled element by element, i.e., the contribution to the integral over Ω is split into contributions for each *element* (e.g., rectangle or triangle) of the underlying mesh.

2. Depending on the choice of the (finite-dimensional) approximation space $\mathcal{U}$ and underlying discretization, the matrix will have a well-defined structure. This is one of the most important applications driving the design of efficient linear system solvers.

How does solving the Ritz-Galerkin equations (34) relate to the solution of the strong form (28) of the PDE? First, we remark that the left-hand side of (32) can be interpreted as a new inner product

$$[u, v] = \iint_{\Omega} \nabla u \cdot \nabla v dxdy \tag{35}$$

on the space of functions whose first derivatives are square integrable and that vanish on $\partial\Omega$. This space is a *Sobolev space*, usually denoted by $H_0^1(\Omega)$.

The inner product $[\cdot, \cdot]$ induces a norm $\|v\| = [v, v]^{1/2}$ on $H_0^1(\Omega)$. Now, using this norm, the best approximation to u from $H_0^1(\Omega)$ is given by the function u^h that minimizes $\|u - u^h\|$. Since we define our numerical method via the finite-dimensional subspace $\mathcal{U}$ of $H_0^1(\Omega)$, we need to find u^h such that

$$u - u^h \perp \mathcal{U}$$

or, using the basis of $\mathcal{U}$,

$$\Big[u - u^h, u_i\Big] = 0, \quad i = 1, \ldots, n.$$

Replacing u^h with its expansion in terms of the basis of $\mathcal{U}$ we have

$$\left[u - \sum_{j=1}^{n} c_j u_j, u_i\right] = 0, \quad i = 1, \ldots, n,$$

or

$$\sum_{j=1}^{n} c_j \left[u_j, u_i\right] = [u, u_i], \quad i = 1, \ldots, n. \tag{36}$$

The right-hand side of this formula contains the exact solution u, and therefore is not useful for a numerical scheme. However, by (35) and the weak form (32) we have

$$[u, u_i] \;\; = \;\; \iint_{\Omega} \nabla u \cdot \nabla u_i dxdy$$

$$= \iint_\Omega f u_i dxdy.$$

Since the last expression corresponds to the inner product $\langle f, u_i \rangle$, (36) can be viewed as

$$\sum_{j=1}^{n} c_j [u_j, u_i] = \langle f, u_i \rangle, \quad i = 1, \ldots, n,$$

which is nothing but the Ritz-Galerkin method (34).

The best approximation property in the Sobolev space $H_0^1(\Omega)$ can also be interpreted as an energy minimization principle. In fact, a smooth solution of the Poisson problem (28) minimizes the energy functional

$$E(u) = \frac{1}{2} \iint_\Omega \nabla^2 u dxdy - \iint_\Omega f u dxdy$$

over all smooth functions that vanish on the boundary of Ω. By considering the energy of nearby solutions $u + \lambda v$, with arbitrary real λ we see that

$$\begin{aligned} E(u+\lambda v) &= \frac{1}{2} \iint_\Omega \nabla(u+\lambda v) \cdot \nabla(u+\lambda v) dxdy - \iint_\Omega f(u+\lambda v) dxdy \\ &= \frac{1}{2} \iint_\Omega \nabla u \cdot \nabla u dxdy + \lambda \iint_\Omega \nabla u \cdot \nabla v dxdy + \frac{\lambda^2}{2} \iint_\Omega \nabla v \cdot \nabla v dxdy \\ &\quad - \iint_\Omega f u dxdy - \lambda \iint_\Omega f v dxdy \\ &= E(u) + \lambda \iint_\Omega \left[\nabla u \cdot \nabla v - f v \right] dxdy + \frac{\lambda^2}{2} \iint_\Omega \nabla^2 v dxdy \end{aligned}$$

The right-hand side is a quadratic polynomial in λ, so that for a minimum, the term

$$\iint_\Omega \left[\nabla u \cdot \nabla v - f v \right] dxdy$$

must vanish for all v. This is again the weak formulation (32).

A discrete "energy norm" is then given by the quadratic form

$$E(u^h) = \frac{1}{2} c^T A c - bc$$

where A is the stiffness matrix, and c is such that the Ritz-Galerkin system (34)

$$Ac = b$$

is satisfied.

Example: One of the most popular finite element versions is based on the use of piecewise linear C^0 polynomials (built either on a regular grid, or on a triangular partition of Ω). The basis functions u_i are "hat functions", i.e., functions that are piecewise linear, have value one at one of the vertices, and zero at all of its neighbors. This choice makes it very easy to satisfy the homogeneous Dirichlet boundary conditions of the model problem exactly (along a polygonal boundary).

Since the gradients of piecewise linear functions are constant, the entries of the stiffness matrix essentially boil down to the areas of the underlying mesh elements.

Therefore, in this case, the Ritz-Galerkin method is very easily implemented. We generate some examples with Matlab's PDE toolbox `pdetool`.

It is not difficult to verify that the stiffness matrix for our example is symmetric and positive definite. Since the matrix is also very sparse due to the fact that the "hat" basis functions have a very localized support, efficient iterative solvers can be applied. Moreover, it is known that the piecewise linear FEM converges with order $\mathcal{O}(h^2)$.

Remarks:

1. The Ritz-Galerkin method was independently introduced by Walther Ritz (1908) and Boris Galerkin (1915).

2. In the textbook this method is referred to as the Rayleigh-Ritz method. Many variations in terminology seem to exist indicating only slight (or no) differences in methodology.

3. The finite element method is one of the most-thoroughly studied numerical methods. Many textbooks on the subject exist, e.g., "The Mathematical Theory of Finite Element Methods" by Brenner and Scott (1994), "An Analysis of the Finite Element Method" by Strang and Fix (1973), or "The Finite Element Method" by Zienkiewicz and Taylor (2000).

4. Even though the use of piecewise linear finite elements is very simple, there are many problems that arise with higher-order methods, mesh refinement, or with the generation of finite element meshes in higher space dimensions. This is where meshfree approximation methods are trying to establish themselves as a viable alternative.

9.5 Characteristics

We start with the simple *advection equation*

$$u_t(x,t) = -cu_x(x,t), \quad -\infty < x < \infty, t \geq 0, \tag{37}$$

where c is some nonzero constant. This problem has a unique solution as soon as we specify an initial condition

$$u(x,0) = f(x), \quad -\infty < x < \infty.$$

A pure initial value problem of this type is called a *Cauchy problem*. It is easily verified (using the chain rule) that the solution to this problem is given by

$$u(x,t) = f(x - ct).$$

This shows us that the solution u is simply a shifted version of the initial profile f. In fact, the initial profile "moves to the right with velocity c". Therefore, the PDE (37) is also referred to as the *transport equation* or *one-way wave equation*.

A *characteristic curve* or simply *characteristic* is a curve in the (x,t)-plane on which the solution u of a PDE is constant. For the advection equation we have

$$\frac{d}{dt}u(x(t),t) = u_x\frac{dx(t)}{dt} + u_t. \tag{38}$$

Since $u_t = -cu_x$ we see that the solution u is constant, i.e.,

$$\frac{d}{dt}u(x(t),t) = 0$$

if

$$\frac{dx(t)}{dt} = c. \tag{39}$$

This is an ODE that determines the characteristic $x = x(t)$ of the advection equation. From (39) it is clear that the characteristics of the advection equation are the lines

$$x(t) = ct + x_0.$$

Remark: The left-hand side of equation (38) can be interpreted as the change as experienced by a moving observer, whereas the two parts on the right-hand side represent the change due to the fact that the the observer is moving into a region of possibly different u, and the change of the solution at a fixed point. Our discussion of characteristics shows that an observer moving along with speed c will see no change in the solution.

Characteristics give us fundamental insight into the solution of a partial differential equation. Assume, e.g., that we have specified initial values for the advection equation (37) only on the finite interval $0 \le x \le 1$. If we consider the domain $\{(x,t) \ : \ 0 \le x \le 1, \ t \ge 0\}$, then the given initial values will determine the solution only in part of this domain. The left part of Figure 7 illustrates this fact for a positive value of c.

The right part of Figure 7 shows how the solution "propagates along characteristics". This is also clear when we recall that a characteristic curve is the location of points in the (x,t)-plane along which the solution of the PDE is constant.

In order to obtain the solution on the remainder (unshaded part) of the domain we need to specify boundary values along the line $x = 0$. However, specifying values along the other boundary, $x = 1$, would possibly overdetermine the problem.

If we want to study characteristics for general second-order quasi-linear PDEs then the situation is more complicated.

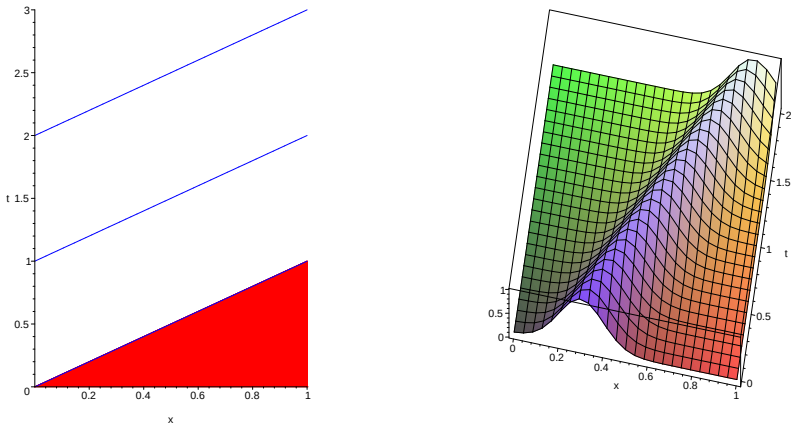

Figure 7: Left: Characteristics (blue lines) and domain of influence (red area) for the advection equation. Right: Typical propagation of solution of the advection equation.

If we consider the wave equation

$$u_{tt}(x,t) = \alpha^2 u_{xx}(x,t), \quad 0 < x < 1, \ t \geq 0,$$

then we can factor the differential operator

$$\frac{\partial^2}{\partial t^2} - \alpha^2 \frac{\partial^2}{\partial x^2} = \left(\frac{\partial}{\partial t} - \alpha \frac{\partial}{\partial x}\right)\left(\frac{\partial}{\partial t} + \alpha \frac{\partial}{\partial x}\right).$$

Therefore, the wave equation will be satisfied if either of the two advection equations

$$u_t(x,t) = -\alpha u_x(x,t) \quad \text{or} \quad u_t(x,t) = \alpha u_x(x,t)$$

is satisfied.

Remark: This is the traveling wave solution of the wave equation found already by d'Alembert in the 18th century. We illustrate this in the Maple worksheet `578_TravelingWaves.mws`.

Characteristics for more general second-order quasi-linear equations are discussed in Section 9.6 of the textbook. We just mention that if the PDE is of the form

$$au_{tt} + bu_{xt} + cu_{xx} + e = 0$$

then the characteristic curves are determined by the nonlinear first-order ordinary differential equation

$$a\left(\frac{dx(t)}{dt}\right)^2 - b\frac{dx(t)}{dt} + c = 0.$$

Remark: It is the discriminant of this quadratic equation that gives rise to the classification of second-order quasi-linear PDEs into hyperbolic, elliptic, and parabolic types as discussed in Section 9.0.

Elliptic problems, that do not depend on time, i.e., describe steady-state problems, have no real characteristics. Parabolic equations, that are time-dependent (and generally evolve toward a steady-state) have one family of characteristics. And hyperbolic

PDEs, that are time-dependent (but do not evolve toward a steady-state) have two independent families of characteristics. If these characteristics cross, then the solution can develop discontinuities or shocks (even for smooth initial data). This is one of the major difficulties associated with the solution of hyperbolic PDEs.

Since hyperbolic PDEs are *conservative*, i.e., the "energy" of the system remains constant over time (such as in the perpetual vibrations for the wave equation), it is important to ensure that numerical schemes are also conservative. We will follow up on this idea in Section 9.7.

Remark: A numerical method, called the *method of characteristics*, asks the user to determine the ordinary differential equation that defines the characteristics of a given partial differential equations. Then a numerical ODE solver is used to solve this ODE, resulting in level curves for the solution. Once these level curves are found, it is possible to use another ODE solver to integrate the PDE (which is now an ODE) along the characteristic (see (39)).

9.6 Finite Differences for Hyperbolic PDEs

Before we study specialized numerical methods for conservation laws, we take a look at a straightforward finite difference scheme. This material is touched on in Section 9.7 of the textbook. We now consider as a model problem the one-dimensional wave equation

$$\begin{aligned} u_{tt}(x,t) &= \alpha^2 u_{xx}(x,t), \quad 0 < x < 1,\ t \geq 0, \\ u(x,0) &= f(x), \quad 0 \leq x \leq 1, \\ u_t(x,0) &= g(x), \quad 0 \leq x \leq 1, \\ u(0,t) &= u(1,t) = 0, \quad t \geq 0. \end{aligned}$$

This model is used to describe the vibrations of a string of unit length fixed at both ends subject to an initial displacement $f(x)$ and initial velocity $g(x)$.

In order to be able to formulate finite differences we need a discrete approximation of the second derivatives. We use the symmetric formulas

$$u_{tt} = \frac{1}{k^2}\left[u(x,t+k) - 2u(x,t) + u(x,t-k)\right] + \mathcal{O}(k^2)$$

and

$$u_{xx} = \frac{1}{h^2}\left[u(x+h,t) - 2u(x,t) + u(x-h,t)\right] + \mathcal{O}(h^2).$$

Here we have discretized the computational domain with

$$\begin{aligned} x_i &= ih, \quad i = 0, \ldots, n+1, \\ t_j &= jk, \quad j \geq 0. \end{aligned}$$

As always, $h = \frac{1}{n+1}$ is the mesh size and k is the time step. Also, we use the notation $v_{i,j} = u(x_i, t_j)$, etc..

Inserting the finite difference approximations into the wave equation we get

$$\frac{1}{k^2}\left[v_{i,j+1} - 2v_{i,j} + v_{i,j-1}\right] = \frac{\alpha^2}{h^2}\left[v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right]. \tag{40}$$

Finally, we can introduce the mesh ratio $s = \left(\frac{\alpha k}{h}\right)^2$, and solve (40) for the value at time level $j+1$, i.e.,

$$v_{i,j+1} = 2(1-s)v_{i,j} + s(v_{i-1,j} + v_{i+1,j}) - v_{i,j-1}.$$

The stencil for this method is shown in Figure 8.

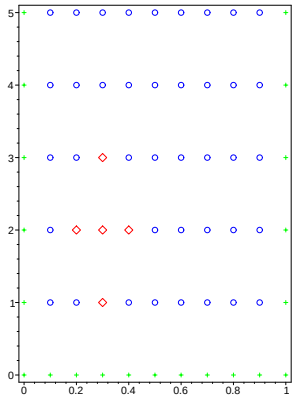

Figure 8: Finite difference stencil for wave equation. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

Remarks:

1. From the finite difference approximations employed it is clear that our method has accuracy $\mathcal{O}(k^2 + h^2)$.

2. As for Richardson's method and the Dufort-Frankel method studied earlier, there is a problem starting this method. Some additional procedure has to be applied to generate the second row of starting values. In the context of the Dufort-Frankel method we suggested that this was most likely accomplished using a lower-order method. However, now we have the advantage that not only the values of u are specified at the $j = 0$ level, but also values of the time derivative. We should be able to use this to our advantage.

We can attempt to use the backward Euler discretization for the first-order time derivative, i.e.,

$$u_t(x_i, 0) \approx \frac{u(x_i, t_1) - u(x_i, 0)}{k},$$

or

$$u(x_i, t_1) \approx u(x_i, 0) + k u_t(x_i, 0).$$

Using the given initial velocity this becomes

$$u(x_i, t_1) \approx u(x_i, 0) + k g(x_i)$$

which gives us the numerical scheme

$$v_{i,1} = v_{i,0} + k g(x_i), \quad i = 1, \ldots, n.$$

This lets us advance the information given at the $j = 0$ level to the first row of the computational mesh at $j = 1$. However, the backward Euler discretization is order $\mathcal{O}(k)$ only, so that the entire numerical method will be $\mathcal{O}(k + h^2)$ only.

In order to get a second-order accurate formula for this first step we employ a Taylor expansion of u with respect to the time variable, i.e.,

$$u(x_i, t_1) \approx u(x_i, 0) + ku_t(x_i, 0) + \frac{k^2}{2} u_{tt}(x_i, 0). \tag{41}$$

Now, the wave equation tells us that

$$u_{tt}(x_i, 0) = \alpha^2 u_{xx}(x_i, 0).$$

Moreover, since $u = f$ for time $t = 0$ we get (provided f is smooth enough)

$$u_{tt}(x_i, 0) = \alpha^2 f''(x_i),$$

and so the Taylor expansion (41) becomes

$$u(x_i, t_1) \approx u(x_i, 0) + ku_t(x_i, 0) + \frac{k^2\alpha^2}{2} f''(x_i).$$

This allows us to express the values at the $j = 1$ level as

$$v_{i,1} = v_{i,0} + kg(x_i) + \frac{k^2\alpha^2}{2} f''(x_i).$$

Using again a symmetric finite difference approximation for f'' we have

$$v_{i,1} = v_{i,0} + kg(x_i) + \frac{k^2\alpha^2}{2} \frac{1}{h^2} \left[f(x_{i+1}) - 2f(x_i) + f(x_{i-1}) \right].$$

Since $f(x_i) = u(x_i, 0) = v_{i,0}$ we arrive at

$$v_{i,1} = (1 - s)v_{i,0} + \frac{s}{2} \left[v_{i-1,0} + v_{i+1,0} \right] + kg(x_i), \tag{42}$$

where we have used the mesh ratio $s = \left(\frac{\alpha k}{h}\right)^2$.

Remarks:

1. Formula (42) is second-order accurate in time. Therefore, the finite difference method consisting of (42) for the first step, and (40) for subsequent steps is of order $\mathcal{O}(k^2 + h^2)$.

2. Once the first two rows of values have be obtained, the method proceeds in an *explicit* way, i.e., no linear systems have to be solved.

3. However, as with all explicit methods there are restrictions on the mesh ratio in order to maintain stability. For this method we have $s \leq 1$.

9.7 Flux Conservation for Hyperbolic PDEs

We mentioned earlier that hyperbolic partial differential equations are conservative. This was clearly illustrated by the simple advection equation (37). It is customary to re-write hyperbolic PDEs in the so-called *flux conservative form*. We illustrate this with the wave equation

$$u_{tt}(x,t) = \alpha^2 u_{xx}(x,t), \quad 0 < x < 1, \ t \geq 0.$$

If we introduce the new variables

$$v = \alpha u_x, \quad \text{and} \quad w = u_t$$

then

$$\begin{aligned} v_t &= \alpha u_{xt} \\ w_x &= u_{tx} \end{aligned}$$

so that $v_t = \alpha w_x$. Since we also have

$$\begin{aligned} v_x &= \alpha u_{xx} \\ w_t &= u_{tt} \end{aligned}$$

we see that the wave equation is equivalent to a system of two coupled first-order partial differential equations

$$\begin{aligned} v_t &= \alpha w_x \\ w_t &= \alpha v_x. \end{aligned}$$

Remark: In physics these equations arise in the study of one-dimensional electromagnetic waves, and are known as *Maxwell's equations*.

To get the standard flux conservative form of a *hyperbolic conservation law* we introduce matrix-vector notation

$$\mathbf{u} = \begin{bmatrix} v \\ w \end{bmatrix}, \quad F(\mathbf{u}) = \begin{bmatrix} 0 & -\alpha \\ -\alpha & 0 \end{bmatrix} \mathbf{u},$$

so that

$$\mathbf{u}_t = -F(\mathbf{u})_x.$$

The function $F(\mathbf{u})$ is called the *flux* of the problem.

Remarks:

1. If the system consists of only one equation, then we recover the advection equation (37).

2. An excellent book on hyperbolic conservation laws (covering theory and numerics) is "Numerical Methods for Conservation Laws" by Randy LeVeque (1992).

We now investigate numerical methods for the simple conservation law

$$\begin{aligned} u_t(x,t) &= -cu_x(x,t), \quad 0 < x < 1, t \geq 0, \\ u(x,0) &= f(x), \quad 0 \leq x \leq 1. \end{aligned}$$

The most natural finite difference approach is based on the derivative approximations

$$\begin{aligned} u_t(x,t) &= \frac{1}{k}\left[u(x,t+k) - u(x,t)\right] + \mathcal{O}(k) \\ u_x(x,t) &= \frac{1}{2h}\left[u(x+h,t) - u(x-h,t)\right] + \mathcal{O}(h^2). \end{aligned}$$

Using the standard discretization

$$\begin{aligned} x_i &= ih, \quad i = 0, \ldots, n+1, \\ t_j &= jk, \quad j \geq 0, \end{aligned}$$

with mesh size $h = \frac{1}{n+1}$ and time step k, as well as the notation $v_{i,j} = u(x_i, t_j)$, we obtain

$$\frac{1}{k}\left[v_{i,j+1} - v_{i,j}\right] = -\frac{c}{2h}\left[v_{i+1,j} - v_{i-1,j}\right]$$

or

$$v_{i,j+1} = -\frac{kc}{2h}\left[v_{i+1,j} - v_{i-1,j}\right] + v_{i,j}. \tag{43}$$

Remarks:

1. The method (43) is an explicit, and therefore fast method.
2. However, as we will show next, this method is *completely useless* since it is *unconditionally unstable*.
3. We use the advection equation with a smooth step function as initial condition to illustrate the performance of this method. The (slightly modified) Matlab program `transport.m` (with menu option "forward Euler in time / centered differences in space") originally written by Stefan Hüeber, University of Stuttgart, Germany, serves this purpose.

We now perform the von Neumann stability analysis to verify this claim. Let's assume that the solution of the model problem is of the form

$$\begin{aligned} v_{i,j} &= e^{\sqrt{-1}i\beta h} e^{j\lambda k} \\ &= e^{\sqrt{-1}i\beta h} \left(e^{\lambda k}\right)^j \end{aligned} \tag{44}$$

obtained via separation of variables (the so-called *plane wave solution*). The question is what the long time behavior of this solution is, i.e., we are interested in

$$\lim_{j \to \infty} v_{i,j}.$$

Since (44) contains an oscillatory as well as an exponentially growing component, we will need $|e^{\lambda k}| \leq 1$ for stability.

In order to compute the amplification factor $e^{\lambda k}$ we substitute (44) into our numerical method (43). This yields

$$e^{\sqrt{-1}i\beta h}e^{(j+1)\lambda k} = -\frac{kc}{2h}\left[e^{\sqrt{-1}(i+1)\beta h} - e^{\sqrt{-1}(i-1)\beta h}\right]e^{j\lambda k} + e^{\sqrt{-1}i\beta h}e^{j\lambda k}.$$

Division by $e^{\sqrt{-1}i\beta h}e^{j\lambda k}$ gives us

$$\begin{aligned} e^{\lambda k} &= -\frac{kc}{2h}\left[e^{\sqrt{-1}\beta h} - e^{-\sqrt{-1}\beta h}\right] + 1 \\ &= 1 - \sqrt{-1}\frac{kc}{h}\sin\beta h. \end{aligned} \tag{45}$$

Here we have used the complex trigonometric identity

$$\sin z = \frac{e^{\sqrt{-1}z} - e^{-\sqrt{-1}z}}{2\sqrt{-1}}.$$

From (45) it is clear that

$$|e^{\lambda k}| \geq 1$$

for all positive time steps k since $e^{\lambda k}$ lies somewhere on the line $\Re(z) = 1$ in the complex plane. Therefore the method is – as claimed – unconditionally unstable.

Lax-Friedrichs Method

Earlier, we were able to "fix" Richardson's method for parabolic PDEs by replacing $v_{i,j}$ by an average of neighboring points (in time). This led to the Dufort-Frankel method.

Now we replace $v_{i,j}$ in (43) by an average of neighboring values in space, i.e., we insert

$$v_{i,j} = \frac{v_{i+1,j} + v_{i-1,j}}{2}$$

into (43). That results in

$$v_{i,j+1} = \frac{v_{i+1,j} + v_{i-1,j}}{2} - \frac{kc}{2h}\left[v_{i+1,j} - v_{i-1,j}\right]. \tag{46}$$

This is known as the *Lax-Friedrichs method* due to Peter Lax and Kurt Friedrichs. Introducing, as always, the mesh ratio $s = \frac{kc}{2h}$ (46) becomes

$$v_{i,j+1} = \left(\frac{1}{2} - s\right)v_{i+1,j} + \left(\frac{1}{2} + s\right)v_{i-1,j}$$

The Lax-Friedrich method is clearly an explicit method, and its three point stencil is shown in Figure 9.

The first part of the stability analysis for this method is as above. However, now we get an amplification factor of

$$e^{\lambda k} = \cos\beta h - \sqrt{-1}\frac{kc}{h}\sin\beta h,$$

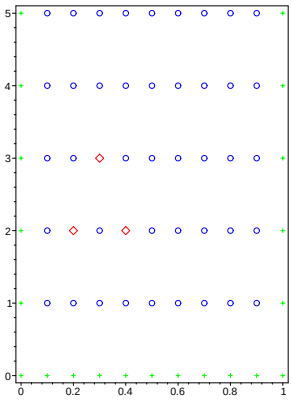

Figure 9: Schematic illustration of the Lax-Friedrichs method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

and stability will be ensured if

$$|e^{\lambda k}|^2 \leq 1.$$

This condition is equivalent to

$$\cos^2 \beta h + \left(\frac{kc}{h}\right)^2 \sin^2 \beta h \leq 1$$

or

$$\left(\frac{kc}{h}\right)^2 \leq \frac{1 - \cos^2 \beta h}{\sin^2 \beta h} = 1.$$

Therefore, stability of the Lax-Friedrichs method is ensured if

$$\frac{k|c|}{h} \leq 1,$$

or (assuming positive c),

$$s \leq \frac{1}{2}.$$

This condition is known in the literature as the *CFL condition* (or Courant-Friedrichs-Levy condition).

Remarks:

1. As indicated in Section 9.6, the characteristics of a hyperbolic PDE define a domain of dependence, i.e., that part of the domain that influences the solution at a given point. The CFL condition says that for each mesh point the domain of dependence of the numerical scheme must contain the domain of dependence of the PDE.

2. The accuracy of the Lax-Friedrichs method is $\mathcal{O}(k + h^2)$, i.e., only first-order accurate in time.

3. We use the Matlab program `transport.m` (with menu option "Lax-Friedrichs") on the advection equation with a smooth step function as initial condition to illustrate the performance of this method.

We now present an alternate interpretation of the Lax-Friedrichs method. To this end we rewrite (46). By subtracting the term $v_{i,j}$ from both sides of (46) we get

$$\frac{v_{i,j+1} - v_{i,j}}{k} = -c\left[\frac{v_{i+1,j} - v_{i-1,j}}{2h}\right] + \frac{1}{2}\left[\frac{v_{i+1,j} - 2v_{i,j} + v_{i-1,j}}{k}\right]. \tag{47}$$

We can now interpret (47) as a finite difference discretization of the PDE

$$u_t(x,t) = -cu_x(x,t) + \frac{h^2}{2k}u_x x(x,t)$$

using symmetric approximation for the space derivatives, and a forward difference in time.

According to this interpretation we have implicitly added a diffusion term to the advection equation. This process is referred to as adding *numerical viscosity* (or artificial viscosity) to a hyperbolic problem. By doing so we introduce stability in the numerical method, but destroy its conservation properties. The effect is that sharp discontinuites in the solution (shocks) are smeared out.

The diffusive effect of the Lax-Friedrichs method can also be observed by looking at the amplification factor. Since

$$|e^{\lambda k}|^2 \begin{cases} = 1, & |c|k = h, \\ < 1, & |c|k < h, \end{cases}$$

the amplitude of the wave decreases, i.e., energy is lost, if the time step is taken too small.

Remark: For certain hyperbolic problems the diffusive effect is not much of an issue since for those problems one is interested in the case when $\beta h \ll 1$, so that $|e^{\lambda k}| \approx 1$.

Lax-Wendroff Method

We now present another finite difference scheme for conservation laws named after Peter Lax and Burt Wendroff. We start with the Taylor expansion

$$u(x,t+k) = u(x,t) + ku_t(x,t) + \frac{k^2}{2}u_t t(x,t) + \ldots.$$

Using the conservation equation $u_t = -cu_x$ this becomes

$$u(x,t+k) = u(x,t) - cku_x(x,t) + \frac{(ck)^2}{2}u_x x(x,t) + \ldots.$$

If we keep the terms explicitly listed, then the truncation error in the time component obviously is $\mathcal{O}(k^2)$.

With the standard notation $v_{i,j} = u(x_i, t_j)$ this expansion gives rise to the numerical scheme

$$v_{i,j+1} = v_{i,j} - ck\frac{v_{i+1,j} - v_{i-1,j}}{2h} + \frac{(ck)^2}{2}\frac{v_{i+1,j} - 2v_{i,j} + v_{i-1,j}}{h^2},$$

where we have used $\mathcal{O}(h^2)$ approximations for both the first-order and second-order spatial derivatives. Therefore, introducing the mesh ratio $s = \frac{ck}{2h}$ we get the *Lax-Wendroff method*

$$v_{i,j+1} = \left(2s^2 - s\right) v_{i+1,j} + \left(1 - 4s^2\right) v_{i,j} + \left(2s^2 + s\right) v_{i-1,j}. \tag{48}$$

This method enjoys $\mathcal{O}(k^2 + h^2)$ accuracy, is an explicit method, and involves a stencil as shown in Figure 10.

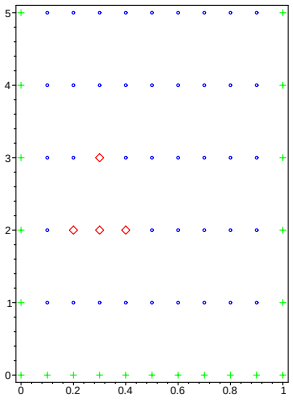

Figure 10: Schematic illustration of the Lax-Wendroff method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

For the stability analysis we again begin with a plane wave solution of the form

$$v_{i,j} = e^{\sqrt{-1}i\beta h} \left(e^{\lambda k}\right)^j.$$

Inserting this into the Lax-Wendroff scheme yields

$$e^{\sqrt{-1}i\beta h} e^{(j+1)\lambda k} = \left(2s^2 - s\right) e^{\sqrt{-1}(i+1)\beta h} e^{j\lambda k} + \left(1 - 4s^2\right) e^{\sqrt{-1}i\beta h} e^{j\lambda k} + \left(2s^2 + s\right) e^{\sqrt{-1}(i-1)\beta h} e^{j\lambda k}.$$

Next, we divide by $e^{\sqrt{-1}i\beta h} e^{(j+1)\lambda k}$ so that

$$\begin{aligned} e^{\lambda k} &= \left(2s^2 - s\right) e^{\sqrt{-1}\beta h} + \left(1 - 4s^2\right) + \left(2s^2 + s\right) e^{-\sqrt{-1}\beta h} \\ &= 2s^2 \left(e^{\sqrt{-1}\beta h} + e^{-\sqrt{-1}\beta h}\right) - s \left(e^{\sqrt{-1}\beta h} - e^{-\sqrt{-1}\beta h}\right) + 1 - 4s^2 \\ &= 4s^2 \cos \beta h - 2\sqrt{-1} s \sin \beta h + 1 - 4s^2, \end{aligned}$$

due to the trigonometric identities

$$\cos z = \frac{e^{\sqrt{-1}z} + e^{-\sqrt{-1}z}}{2}, \quad \sin z = \frac{e^{\sqrt{-1}z} - e^{-\sqrt{-1}z}}{2\sqrt{-1}}.$$

Now, a straightforward (but lengthy) calculation yields

$$|e^{\lambda k}|^2 = 1 - 16s^2 \sin^2 \frac{\beta h}{2} \left(1 - 4s^2 \sin^2 \frac{\beta h}{2} + \cos^2 \frac{\beta h}{2}\right).$$

Stability will be ensured if the term in parentheses is non-negative, i.e.,

$$1 - 4s^2 \sin^2 \frac{\beta h}{2} + \cos^2 \frac{\beta h}{2} \geq 0.$$

It is easy to show that the expression on the left-hand side takes its minimum for $\beta h = \frac{\pi}{2}$. The minimum value is then $1 - 4s^2$, and it will be nonnegative as long as

$$|s| \leq \frac{1}{2}.$$

This is the CFL condition encountered earlier.

Remark: We use the Matlab program `transport.m` (with menu option "Lax-Wendroff") on the advection equation with a smooth step function as initial condition to illustrate the performance of this method.

Leapfrog Method

Another method that is second-order accurate in time and space for the advection equation can be found by using symmetric approximations to the partial derivatives, i.e.,

$$\frac{v_{i,j+1} - v_{i,j-1}}{2k} = -c \frac{v_{i+1,j} - v_{i-1,j}}{2h}$$

or

$$v_{i,j+1} = -\frac{ck}{h} \left[v_{i+1,j} - v_{i-1,j} \right] + v_{i,j-1}. \tag{49}$$

This method has its name from the fact that the values used in the stencil (see Figure 11) leapfrog over the missing central point $v_{i,j}$.

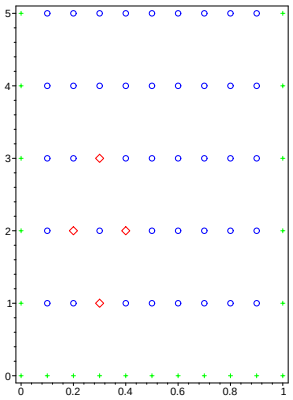

Figure 11: Schematic illustration of the leapfrog method. Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

Remarks:

1. The leapfrog method requires more storage than the previous methods since the two most recent rows of values need to be kept in memory.

2. Also, this method requires a special start-up step (which may be of lower order).

The stability analysis of the leapfrog method involves the solution of a quadratic equation, and leads to the amplification factor

$$e^{\lambda k} = -\sqrt{-1}\frac{ck}{h}\sin\beta h \pm \sqrt{1-\left(\frac{ck}{h}\right)^2 \sin^2\beta h}$$

so that

$$\begin{aligned} |e^{\lambda k}|^2 &= \left(\frac{ck}{h}\right)^2 \sin^2\beta h + 1 - \left(\frac{ck}{h}\right)^2 \sin^2\beta h \\ &= 1. \end{aligned}$$

Therefore, the leapfrog method is stable for our model problem for all mesh ratios $s = \frac{ck}{2h}$, and does not involve any numerical diffusivity.

However, for more complicated (nonlinear) problems this method becomes unstable. This instability is due to the fact that the stencil isolates the "black" from the "white" mesh points. This leads to so-called *mesh drifting*.

Remark: By combining the leapfrog method with the Lax-Friedrichs method it is possible to get a two-step Lax-Wendroff type method which has accuracy $\mathcal{O}(k^2 + h^2)$ and is stable (i.e., does not suffer from mesh drifting).

As the examples viewed earlier with the help of the Matlab program `transport.m` have shown, most of the methods presented so far are not capable of advecting a sharp transition accurately. The numerical solution of the test problem

$$u_t(x,t) = -2u_x(x,t), \quad 0 \le x \le 1,\ t \ge 0,$$

with initial profile

$$u(x,0) = \arctan\left(\frac{(0.5-x)}{0.002}\right), \quad 0 \le x \le 1,$$

as displayed in Figure 12

- was not handled at all by the Euler/centered difference method (43),
- was significantly smeared out by the Lax-Friedrichs method (46) (see left part of Figure 13),
- and suffered from Gibbs-like oscillations with the Lax-Wendroff method (48) (see right part of Figure 13).

Remark: Use of the leapfrog method (49) will result in oscillations similar to the Lax-Wendroff method.

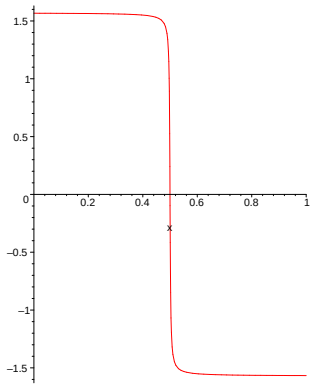

Figure 12: A smoothed out step function as initial condition for the advection equation.

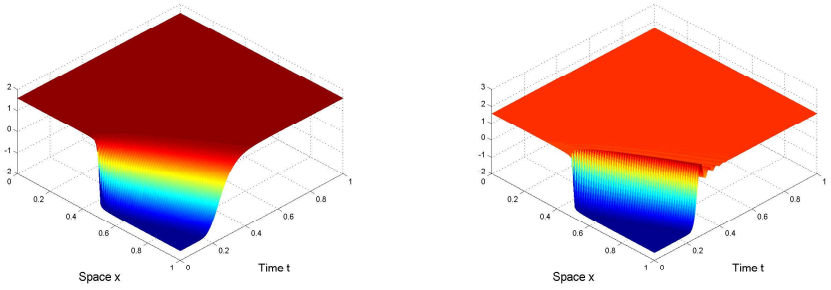

Figure 13: Advection of initial profile shown in Figure 12 by the Lax-Friedrichs (left) and Lax-Wendroff (right) methods.

Upwind Differencing

A slight modification of the Lax-Friedrichs method lets us obtain a more accurate solution for the test problem just discussed. By considering the direction in which the solution will propagate, i.e., using information from the characteristics, we use the difference scheme

$$v_{i,j+1} = v_{i,j} - \begin{cases} \dfrac{kc}{h}\left[v_{i,j} - v_{i-1,j}\right], & c > 0, \\ \dfrac{kc}{h}\left[v_{i+1,j} - v_{i,j}\right], & c < 0. \end{cases} \tag{50}$$

The stencils for this so-called *upwind differencing method* are shown in Figure 14.

Remarks:

1. The accuracy of the upwind differencing method (50) is only $\mathcal{O}(k+h)$. However, it deals with sharp discontinuities more effectively.

2. We use the Matlab program `transport.m` (with menu option "upwind") on the advection equation with a smooth step function as initial condition to illustrate the performance of this method. A plot of this solution is presented in Figure 15.

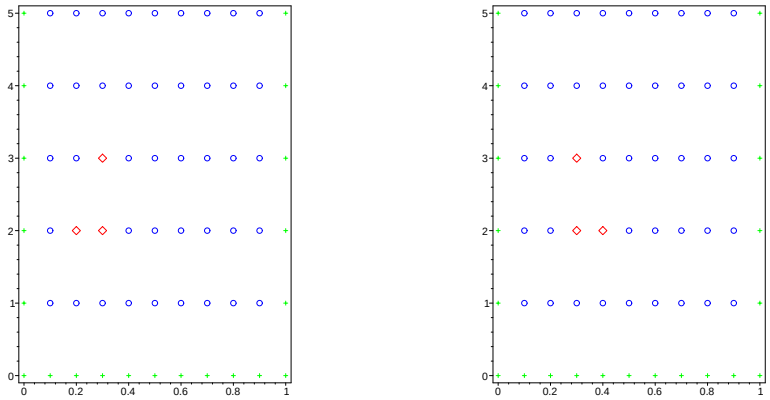

Figure 14: Schematic illustration of upwind differencing (left: $c > 0$, right: $c < 0$). Basic mesh points are indicated with blue $\circ$, green $+$ correspond to given values, and points marked with red $\diamondsuit$ form a typical stencil.

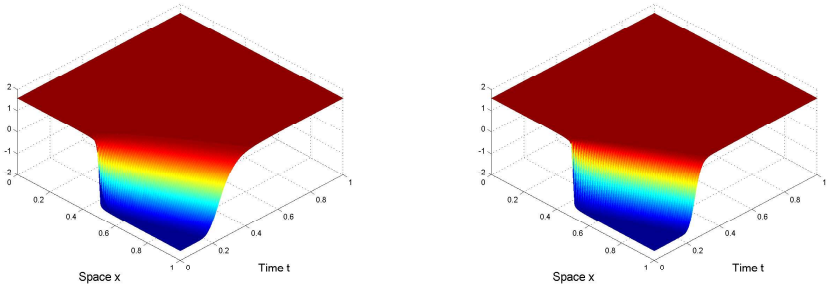

Figure 15: Advection of initial profile shown in Figure 12 using upwind differencing (right) compared to Lax-Friedrichs solution (left).

A Nonlinear Model Problem

Matters start to become really interesting once we consider nonlinear conservation equations. The *inviscid Burgers' equation*

$$u_t(x,t) = -u(x,t)u_x(x,t) \tag{51}$$

is a nonlinear advection equation whose speed depends on the solution u. This equation will produce *shocks* even for smooth initial data.

The solution of the inviscid Burgers' equation (51) with initial profile $u(x,0) = \cos(x)$ is illustrated in the (slightly modified) Matlab program `burgers.m` originally written by Mary Pugh (University of Toronto). The program uses explicit time-stepping and a spectral (FFT-based) differencing method in space. The initial profile and shock at time $t = 1$ are shown in Figure 16.

We end our discussion of hyperbolic conservation laws with a discussion of flux conservation. The idea is to introduce a *numerical flux* into the numerical scheme.

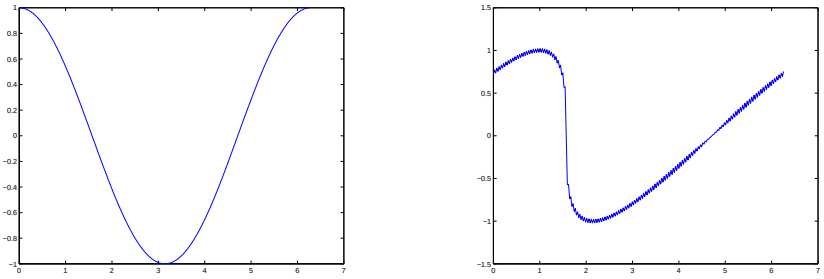

Figure 16: Initial profile $u(x,0) = \cos(x)$ (left) and shock at $t = 1$ for Burgers' equation generated with `burgers.m`.

Let's consider again the upwind differencing scheme. Applied to the inviscid Burgers' equation in the case of $v_{i,j} > 0$ we get

$$v_{i,j+1} = v_{i,j} - \frac{k}{h} v_{i,j} \left[v_{i,j} - v_{i-1,j} \right]. \tag{52}$$

Now we introduce the numerical flux f so that (52) becomes

$$v_{i,j+1} = v_{i,j} - \frac{k}{h} \left[f(v_{i,j}, v_{i+1,j}) - f(v_{i-1,j}, v_{i,j}) \right]. \tag{53}$$

For example, with $f(v,w) = \frac{1}{2}v^2$ this leads to

$$v_{i,j+1} = v_{i,j} - \frac{k}{2h} \left[(v_{i,j})^2 - (v_{i-1,j})^2 \right], \tag{54}$$

which is the conservation form of (52).

To understand this formula better, recall the flux conservation form of the standard hyperbolic PDE

$$u_t(x,t) = -F(u)_x(x,t),$$

where $F(u)$ is the flux. From a physical point of view it is natural to consider the integral form

$$\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_{j+1})dx = \int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_j)dx - \left[\int_{t_j}^{t_{j+1}} F(u(x_{i+1/2},t))dt - \int_{t_j}^{t_{j+1}} F(u(x_{i-1/2},t))dt \right], \tag{55}$$

where $x_{i\pm1/2} = x_i \pm \frac{h}{2}$. In order to simplify this relation we introduce the notion of a *cell average*

$$\overline{v}_{i,j} = \frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_j)dx.$$

If we divide (55) by h and use the cell averages just introduced, then we get

$$\overline{v}_{i,j+1} = \overline{v}_{i,j} - \frac{1}{h} \left[\int_{t_j}^{t_{j+1}} F(u(x_{i+1/2},t))dt - \int_{t_j}^{t_{j+1}} F(u(x_{i-1/2},t))dt \right].$$

Finally, using a numerical flux defined as a time-average of the true flux, i.e.,

$$f(v_{i,j}, v_{i+1,j}) = \frac{1}{k} \int_{t_j}^{t_{j+1}} F(u(x_{i+1/2}, t)dt,$$

we get

$$\overline{v}_{i,j+1} = \overline{v}_{i,j} - \frac{k}{h} \left[f(v_{i,j}, v_{i+1,j}) - f(v_{i-1,j}, v_{i,j}) \right].$$

Therefore, $v_{i,j}$ in (53) can be interpreted as an approximation to the cell average

$$\overline{v}_{i,j} = \frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t_j)dx.$$

Remarks:

1. State-of-the-art methods for solving hyperbolic conservation laws are the finite volume methods described in the book "Finite Volume Methods for Hyperbolic Problems" by Randy LeVeque (2002), and also featured in the software library CLAW.

2. An alternative are so-called essentially non-oscillating (ENO) methods introduced by Stan Osher, Chi-Wang Shu, and others. These methods are based on the integral form of the conservation law, and use polynomial interpolation on small *stencils* that are chosen so that the developing shock is preserved.

3. The full (viscous) Burgers' equation

   $$u_t(x, t) = -u(x, t)u_x(x, t) + \frac{1}{R} u_{xx}(x, t)$$

   with *Reynolds number* R is much easier to solve numerically than the inviscid equation (51). This is due to the smoothing effect of the diffusive term u_{xx}. Only for large Reynolds numbers does this equation resemble the inviscid case, and becomes a challenge for numerical methods.

4. All of the methods discussed above can be generalized to systems of conservation laws.

9.8 Other Methods for Time-Dependent PDEs

Method of Lines

The method of lines is a semi-discrete method. The main idea of this method is to convert the given PDE to a system of ODEs. This is done by discretizing in space, and leaving the time variable continuous. Thus, we will end up with a system of ODE initial value problems that can be attacked with standard solvers.

If, e.g., we want to solve the heat equation

$$u_t(x, t) = u_{xx}(x, t)$$

with zero boundary temperature and initial temperature $u(x,0) = f(x)$, then we can discretize this problem in space only (using a second-order accurate symmetric difference approximation). This means that on each line through a spatial mesh point in $\{x_0, x_1, \ldots, x_n\}$ we have the ODE

$$\frac{du_i}{dt}(t) = \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2}, \quad i = 1, \ldots, n-1, \tag{56}$$

with boundary and initial conditions

$$u_0(t) = u_n(t) = 0, \quad u_i(0) = f(x_i).$$

In matrix form we have

$$\mathbf{u}'(t) = A\mathbf{u}(t),$$

where, for this example, the matrix A is given by

$$A = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & \ldots & 0 \\ 1 & -2 & 1 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & 1 & -2 & 1 \\ 0 & \ldots & 0 & 1 & -2 \end{bmatrix},$$

and $\mathbf{u} = [u_1, \ldots, u_n - 1]^T$. Any standard IVP solver can be used to solve this system. However, for small h this matrix is very stiff, and so some care has to be taken in choosing the solver.

Remark: From (56) it is apparent that we are obtaining the solution of the wave equation along lines with constant x-value (parallel to the time-axis). This interpretation has given the method its name.

(Pseudo-)Spectral Methods

Another semi-discrete approach involves an expansion in terms of basis functions for the spatial part of the PDE, and an IVP solver for the time component.

Assume that the numerical solution is given in the form

$$u_h(x,t) = \sum_{j=1}^{n} c_j(t)\phi_j(x),$$

where the ϕ_j form a basis for an approximation space used for the space component. Ideally, these basis functions will satisfy the boundary conditions of the PDE. The time-dependent coefficients c_j can be determined via collocation at a discrete set of points $\{x_1, \ldots, x_n\}$. If we again use the heat equation

$$u_t(x,t) = u_{xx}(x,t)$$

to illustrate this method, then the collocation conditions are

$$\frac{\partial u_h}{\partial t}(x_i,t) = \frac{\partial^2 u_h}{\partial x^2}(x_i,t), \quad i = 1, \ldots, n,$$

or

$$\sum_{j=1}^{n} c_j'(t)\phi_j(x_i) = \sum_{j=1}^{n} c_j(t)\phi_j''(x_i), \quad i = 1, \ldots, n.$$

This is a system of n first-order ODEs for the coefficients c_j. In matrix form we have

$$A\mathbf{c}'(t) = B\mathbf{c}(t), \tag{57}$$

where $\mathbf{c} = [c_1, \ldots, c_n]^T$ and the matrices A and B have entries

$$A_{ij} = \phi_j(x_i), \quad B_{ij} = \phi_j''(x_i).$$

The initial conditions for (57) can be obtained by interpolation

$$\sum_{j=1}^{n} c_j(0)\phi_j(x_i) = f(x_i),$$

where f specifies the initial condition of the PDE.

Remarks:

1. Standard IVP solvers can be used to solve the system (57). Of course, the basis functions should be such that the matrix A is invertible.

2. All standard interpolation or approximation methods can be used to generate the basis $\{\phi_1, \ldots, \phi_n\}$ (e.g., polynomials, splines, radial basis functions).

3. Again, the ODE systems (57) are often rather stiff, which means that the IVP solvers need to be chosen carefully.

4. The program `burgers.m` used earlier employs a trigonometric basis coupled with the fast Fourier transform, and leapfrog time-stepping for ODEs (initialized with one forward Euler step).

Time-Dependent Galerkin Methods

It is also possible to combine the basis function expansion with a least squares best approximation criterion instead of collocation as indicated in Section 9.4. This leads to *time-dependent Galerkin methods*.

Assume that the numerical solution is given in the form

$$u_h(x,t) = \sum_{j=1}^{n} c_j(t)\phi_j(x), \tag{58}$$

where the ϕ_j form a basis for an approximation space used for the space component. To keep the discussion simple, we will assume that the basis functions satisfy the boundary conditions of the PDE. We will once more use the heat equation

$$u_t(x,t) = u_{xx}(x,t)$$

with boundary conditions $u(0,t) = u(1,t) = 0$ and initial condition $u(x,0) = f(x)$ to illustrate this method. Inserting the expansion (58) into the heat equation results in

$$\sum_{j=1}^{n} c_j'(t)\phi_j(x) = \sum_{j=1}^{n} c_j(t)\phi_j''(x). \tag{59}$$

The conditions for the time-dependent coefficients c_j are generated by taking inner products of (59) with the basis functions ϕ_i, $i = 1, \ldots, n$. This results in a system of n first-order ODEs for the coefficients c_j of the form

$$\sum_{j=1}^{n} c_j'(t)\langle \phi_j, \phi_i \rangle = \sum_{j=1}^{n} c_j(t)\langle \phi_j'', \phi_i \rangle,$$

where the inner products are standard L_2-inner products over the spatial domain (assumed to be $[0,1]$ here)

$$\langle v, w \rangle = \int_0^1 v(x)w(x)dx.$$

In matrix form we have

$$A\mathbf{c}'(t) = B\mathbf{c}(t), \tag{60}$$

where $\mathbf{c} = [c_1, \ldots, c_n]^T$ and the matrices A and B have entries

$$A_{ij} = \langle \phi_j, \phi_i \rangle, \quad B_{ij} = \langle \phi_j'', \phi_i \rangle.$$

If the basis functions are orthogonal on $[0,1]$ then the matrix A is diagonal, and the system (60) is particularly simple to solve.

As for the spectral methods above, the initial condition becomes

$$\sum_{j=1}^{n} c_j(0)\phi_j(x) = f(x),$$

and the initial coefficients $c_j(0)$, $j = 1, \ldots, n$, can be obtained by interpolation or least squares approximation. In the case of least squares approximation (with orthogonal basis functions) the coefficients are given by the generalized Fourier coefficients $c_j(0) = \langle f, \phi_i \rangle$.

By applying integration by parts to the entries of B (remember that the basis functions are assumed to satisfy the boundary conditions) we get an equivalent system

$$A\mathbf{c}'(t) = -C\mathbf{c}(t)$$

with "stiffness matrix" C with entries

$$C_{ij} = [\phi_j, \phi_i] = \int_0^1 \phi_j'(x)\phi_i'(x)dx.$$

This is analogous to the inner product in the Ritz-Galerkin method of Section 9.4.

Remarks:

1. Higher-dimensional time-dependent PDEs can be attacked by using higher-dimensional spatial methods, such as finite elements, or higher-dimensional collocation schemes.

2. Matlab's `pdetool.m` uses piecewise linear finite elements in a Galerkin method to solve both (linear) parabolic and hyperbolic PDEs.