Chapter 7

Numerical Solution of Partial Differential Equations

As we have done in the preceding chapter, we begin by classifying the problems. Most of the classifications that were given for ODEs apply to partial differential equations (PDEs) as well. Thus PDEs can be of first or higher order; there may be a single equation or a system of equations; they may be linear or nonlinear; and they may be homogeneous or inhomogeneous. These classifications are as important in the PDE case as they were in the ODE case. Much of what was said in the preceding chapter applies to PDEs as well as ODEs and is not repeated here.

The distinction between initial and boundary value problems that is so important in the numerical approach to the solution of ODEs is more complicated for PDEs. The reason is that there are now two or more independent variables, so it is possible for a problem to be an initial value problem with respect to one variable and, simultaneously, a boundary value problem with respect to another variable. It is possible to have pure initial value problems (Cauchy problems), pure boundary value problems, and mixed initial-boundary value problems. The question of what appropriate initial and/or boundary conditions are for a given equation is intimately tied to the type of equation, so a short discussion of the classification of PDEs is given.

First order PDEs occur only occasionally in physics and engineering problems. Systems of first order equations are more common, but they can be discussed along with higher order equations. Many PDEs occurring in applications are second order in at least one of the independent variables. Fourth order equations arise in mechanics of solid bodies (and elsewhere).

The classification scheme for PDEs depends on the nature of their characteristics. In the case of two independent variables, characteristics are lines in the plane of the independent variables (in higher dimensions the characteristics become surfaces) along which “signals” can propagate, and they are also the locations of possible discontinuities in the solution of the equation.

A hyperbolic equation possesses only families of real characteristics. Physical systems that are governed by hyperbolic equations are ones in which signals propagate at finite speed or over a finite region. The model hyperbolic equation we study is the first order wave equation

\[
\frac{\partial \phi}{\partial t} + c \frac{\partial \phi}{\partial x} = 0.
\]

We use the terms “space” and “time” coordinates even though there are cases of interest in which all the coordinates are in fact spatial. Hyperbolic equations are always posed in domains that extend to infinity in the timelike coordinate and are thus open in this direction. The spatial coordinate may or may not be bounded. In either case one typically specifies one condition at \( t = 0 \). If the spatial region is bounded, there may also be boundary conditions (one at each boundary); otherwise, we have a pure initial value problem. Note that since our model equation is first order in the spacelike coordinate, only one boundary condition can be applied at most, so that the spacelike coordinate is also open.

Parabolic equations have only one independent family of real characteristics. The model parabolic
equation we shall study is the diffusion equation

\[
\frac{\partial \phi}{\partial t} - \alpha \frac{\partial^2 \phi}{\partial x^2} = 0.
\]

The initial and boundary conditions that are applied to parabolic equations are typically similar to those for hyperbolic equations. The domain of solution is open in the time dimension and the spatial domain may be open or closed. In the case of closed spatial domain we require two boundary conditions for our model equation.

Finally, we come to the elliptic case, in which there are no real characteristics. The model equation we shall study is the Poisson equation

\[
\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = f(x, y).
\]

In elliptic problems every point in the solution domain is affected by disturbances at every other point. This fact makes elliptic problems particularly difficult to solve. The proper domain for the solution of an elliptic equation is normally a closed region. Furthermore, it can be shown that necessary and sufficient boundary conditions are provided by giving one datum (the value of the dependent variable, its normal derivative, or some linear combination of the two) at each point on the boundary.

In this chapter parabolic and hyperbolic equations are treated first because the methods used for solving these are closely related to methods used for ODEs. Elliptic equations are discussed later since they are usually (but not always) solved by iterative procedures that are closely related to the methods used for boundary value problems and parabolic equations.

### 7.1 Excursion into Banded Matrices

We designate a banded matrix by the notation \( B[\ldots, b_{-2}, b_{-1}, b_0, b_1, b_2, \ldots] \), where \( B \) stands for a banded matrix and \( b_i \) are vector elements representing entries along the diagonals below, on, and above the central diagonal. The number of arguments is always odd and the central argument refers to entries in the main diagonal.

A general \( N \times N \) tridiagonal banded matrix is written as

\[
B[a, b, c] = \begin{bmatrix}
  b_1 & c_1 & & & \\
  a_2 & b_2 & c_2 & & \\
  & \ddots & \ddots & \ddots & \\
  \end{bmatrix}.
\]

If the vector designation is omitted from \( b_i \), then \( b_i \) is a scalar that spreads along the entire diagonal. The notation \( I \) is used for the unitary or identity matrix and the notation \( D[b] \) for a diagonal matrix. Notice that

\[
\]

A convention is also used to express the block structure of a two-dimensional system. For this we introduce the definition of a banded block matrix, \( B[\text{arguments}] \), each argument of which is a matrix itself. The role of the arguments is the same as before. For example

\[
B[B[a, b, c], B[d, e, f], B[g, h, i]]
\]

refers to a tridiagonal group of block matrices in which all of the blocks are themselves tridiagonals of scalars.
7.2 Difference Schemes as Banded Matrices

A difference scheme is usually written as a point operator. The three-point central difference formula for a second derivative is given by

\[
\left( \frac{\partial^2 \phi}{\partial x^2} \right)_i \approx (\delta_x x \delta_x) \phi_i = \frac{1}{\Delta x^2} \left( \phi_{i-1} - 2\phi_i + \phi_{i+1} \right),
\]

where index “i” refers to the location where the dependent variable is evaluated in the uniformly spaced \( x \)-direction. This difference scheme can be expressed as a matrix operator once the boundary conditions are specified. If \( \phi \) is specified at an endpoint, the boundary condition is said to be Dirichlet; a Neumann boundary condition corresponds to the specification of \( \partial \phi / \partial x \) at an endpoint; finally, a periodic boundary condition can be specified at the endpoints. As we shall see shortly, the matrix representation for parabolic and hyperbolic equations with the three kinds of boundary conditions can now be written as the system of first order ODEs

\[
\frac{d\Phi}{dt} = A\Phi + f.
\]

The ability to express finite difference schemes in vector-matrix forms is absolutely essential to our subsequent analysis. In their matrix forms, finite difference, finite volume, and finite element schemes all appear similar and much of their evaluation follows the same pattern.

Periodic banded matrices come from the use of periodic boundary conditions. It can be easily shown that for the model diffusion equation we have

\[
A = \frac{1}{\Delta x^2} B_p[1, -2, 1] = \frac{1}{\Delta x^2} \begin{bmatrix}
-2 & 1 & 1 & 1 \\
1 & -2 & 1 & \ddots \\
& \ddots & \ddots & \ddots \\
1 & -2 & 1 & -2 \\
\end{bmatrix}.
\]

The 1’s at the top right and bottom left corners enter because of the “wrap around” properties of periodicity. Periodic banded matrices are designated by the subscript “\( p \)” for this reason.

When the derivative of a function is specified at an endpoint, another matrix form occurs. Consider an approximation of \( \partial^2 \phi / \partial x^2 \) at point 4 shown below:

\[
\begin{array}{l}
\begin{array}{llllllll}
(\phi)_0 \\
\downarrow \quad \bullet \\
0 & 1 & 2 & 3 & 4 \\
\end{array}
\end{array}
\quad \begin{array}{l}
\begin{array}{llllllll}
(\phi_4)_{9/2} \\
\downarrow \quad \bullet \\
0 & 1 & 2 & 3 & 4 \\
\end{array}
\end{array}
\]

Notice that (in our example) \( \partial \phi / \partial x \) is given at \( x = 9/2\Delta x \). We can approximate \( \partial^2 \phi / \partial x^2 \) at \( x_4 \) by

\[
\left( \frac{\partial^2 \phi}{\partial x^2} \right)_4 \approx \delta_x \left( \frac{\partial \phi}{\partial x} \right)_4 = \frac{1}{\Delta x} \left[ \left( \frac{\partial \phi}{\partial x} \right)_{9/2} - \left( \frac{\partial \phi}{\partial x} \right)_{7/2} \right].
\]

Now \( \partial \phi / \partial x \) at \( x = 9/2 \) is given and \( \partial \phi / \partial x \) at \( x = 7/2 \) is \( \delta_x (\phi_{7/2}) = (\phi_4 - \phi_3) / \Delta x \). This gives

\[
\left( \frac{\partial^2 \phi}{\partial x^2} \right)_4 \approx \frac{1}{\Delta x} \left( \frac{\partial \phi}{\partial x} \right)_{9/2} - \frac{1}{\Delta x^2} (\phi_4 - \phi_3).
\]

In this case, the above expression leads to a matrix \( A \) that is still banded, but the elements along the main diagonal are not uniform. For such a case we have the tridiagonal matrix

\[
A = \frac{1}{\Delta x^2} B[1, b, 1],
\]

where

\[
b = [-2, -2, -2, -1]^T.
\]
7.3 Semi-Discretization and Matrix Stability Analysis

PDEs can now be readily converted to a system of ODEs by using finite difference approximations for derivatives in all but one of the independent variables, and by using the matrix formulation given above. Consider the following diffusion equation:

$$\frac{\partial \phi}{\partial t} = \alpha \frac{\partial^2 \phi}{\partial x^2}$$  \hspace{1cm} (7.1)

with initial and boundary conditions

$$\phi(x,0) = g(x), \quad \text{and} \quad \phi(0,t) = \phi(L,t) = 0.$$  

We discretize the coordinate \( x \) with \( N + 1 \) uniformly spaced grid points

$$x_j = x_{j-1} + \Delta x, \quad x_0 = 0, \quad x_N = L, \quad j = 1, 2, \ldots, N.$$  

Note that \( \Delta x = L/N \) and we can also write \( x_j = j \Delta x \). Using the second order central difference approximation for the second derivative in \( x \) results in

$$\frac{d\phi_j}{dt} = \alpha \frac{\phi_{j+1} - 2\phi_j + \phi_{j-1}}{\Delta x^2}, \quad j = 1, 2, \ldots, N - 1,$$

where \( \phi_j(t) = \phi(x_j,t) \). This is a system of ordinary differential equations which can be written in matrix form as

$$\frac{d\Phi}{dt} = A\Phi,$$

where \( \phi_j \)'s are the elements of the vector \( \Phi \), and \( A \) is the \((N-1) \times (N-1)\) tridiagonal matrix

$$A = \frac{\alpha}{\Delta x^2} \begin{bmatrix}
-2 & 1 & & & & \\
1 & -2 & 1 & & & \\
& \ddots & \ddots & \ddots & & \\
& & 1 & -2 & 1 & \\
& & & 1 & -2 & \\
\end{bmatrix}.$$  

Note that higher order space-derivative approximations can be formed by increasing the bandwidth of the matrix.

Since \( A \) is a banded matrix, it can now be denoted using the compact notation

$$A = \frac{\alpha}{\Delta x^2} B[1,-2,1].$$

We have completed the semi-discretization of the PDE (7.1). The result is a system of ordinary differential equations and can be solved using any of the numerical methods such as Runge-Kutta formulas or multi-step methods. However, when dealing with systems, we have to be concerned about stiffness. To investigate stiffness, we have to study the eigenvalue structure of the matrix \( A \). Eigenvalues of \( A \) can be obtained from a known formula for the eigenvalues of a tridiagonal matrix with constant entries. Let \( B[a,b,c] \) be an \((N-1) \times (N-1)\) tridiagonal matrix. It can be shown that the eigenvalues of \( B \) are

$$\lambda_j = b + 2\sqrt{ac} \cos \alpha_j, \quad \alpha_j = \frac{j\pi}{N}, \quad j = 1, 2, \ldots, N - 1.$$  \hspace{1cm} (7.2)

Therefore, the eigenvalues of \( A \) are

$$\lambda_j = -\frac{2\alpha}{\Delta x^2} \left[ 1 - \cos \left( \frac{j\pi}{N} \right) \right].$$
The eigenvalue with the smallest magnitude is
\[ \lambda_1 = -\frac{2\alpha}{\Delta x^2} \left[ 1 - \cos \left( \frac{\pi}{N} \right) \right]. \]

For large \( N \), the series expansion for \( \cos(\pi/N) \) converges rapidly
\[ \cos \left( \frac{\pi}{N} \right) = 1 - \frac{1}{2!} \left( \frac{\pi}{N} \right)^2 + \frac{1}{4!} \left( \frac{\pi}{N} \right)^4 - \ldots. \]

Retaining the first two terms in the expansion results in
\[ \lambda_1 \approx -\frac{\pi^2 \alpha}{N^2 \Delta x^2}. \]

Also, for large \( N \), we have
\[ \lambda_{N-1} \approx -\frac{4\alpha}{\Delta x^2}. \] (7.3)

Therefore, the ratio of the eigenvalue with the largest modulus to the eigenvalue with the smallest modulus is
\[ \frac{|\lambda_{N-1}|}{|\lambda_1|} \approx \left( \frac{2N}{\pi} \right)^2. \]

Clearly, for large \( N \) the system is stiff.

Notice that all the eigenvalues of \( A \) are real and negative. Standard decoupling of the equations using the matrix of eigenvectors \( S \) yields
\[ \frac{d\Psi}{dt} = \Lambda \Psi, \]
where \( \Lambda = S^{-1}AS \) is a diagonal matrix with the eigenvalues of \( A \) on the diagonal and \( \Psi = S^{-1}\Phi \). Since the equations are uncoupled, the solutions can be obtained readily
\[ \Psi_j(t) = e^{\lambda_j t} \Psi_j(0). \]

Thus, negative eigenvalues result in the decay of the solution which is expected for the heat equation.

In passing we note that the eigenvalues of \( B_p[a,b,c] \) of order \( N \times N \) are given by
\[ \lambda_j = b + (a + c) \cos \alpha_j - i(a - c) \sin \alpha_j, \quad \alpha_j = \frac{j2\pi}{N}, \quad j = 0, 1, \ldots, N - 1. \]

Now, consider the model hyperbolic partial differential equation
\[ \frac{\partial \phi}{\partial t} + c \frac{\partial \phi}{\partial x} = 0 \] (7.4)
with initial and boundary condition
\[ \phi(x, 0) = g(x), \quad \text{and} \quad \phi(0, t) = 0. \]

Semi-discretization with the second order central difference formula leads to
\[ \frac{d\phi_j}{dt} + c \frac{\phi_{j+1} - \phi_{j-1}}{2\Delta x} = 0. \]

In matrix notation we have
\[ \frac{d\Phi}{dt} = -\frac{c}{2\Delta x} B[-1, 0, 1]\Phi. \]
Using equation (7.2), the eigenvalues of this matrix are
\[ \lambda_j = -\frac{c}{\Delta x} \left[ i \cos \left( \frac{j\pi}{N+1} \right) \right] = i\omega_j. \]
Thus, the eigenvalues of the matrix resulting from semi-discretization of the PDE (7.4) are purely imaginary. The temporal part of the solution is of the form \( e^{i\omega_j t} \) with
\[ \omega_j = -\frac{c}{\Delta x} \cos \left( \frac{j\pi}{N+1} \right), \]
which has an oscillatory (non-decaying) character.

Discretization of equations (7.1) and (7.4) led to additional insights into the behavior of the solutions. These two equations are examples of two extreme cases, one with a decaying solution (negative eigenvalues) and the other with oscillatory behavior (imaginary eigenvalues). These two examples and the associated discretizations provide support for the use of the model equation
\[ y' = \lambda y \]
for partial differential equations. The case with \( \lambda \) real and negative is a model for the PDE (7.1), and the case with \( \lambda \) purely imaginary is a model for equation (7.4). Thus, in application of numerical methods to these partial differential equations, the same issues discussed for ODE’s should be considered. For example, application of the explicit Euler scheme to equation (7.4) will lead to unconditionally unstable numerical solutions, and application of the same scheme to equation (7.1) is conditionally stable. In the latter case, the time maximum step is obtained from the requirement that
\[ |\sigma_j| = |1 + \Delta t \lambda_j| \leq 1, \quad j = 1, 2, \ldots, N - 1, \]
which leads to
\[ \Delta t \leq \frac{2}{|\lambda|_{\text{max}}}. \]
where \( |\lambda|_{\text{max}} \) is the magnitude of the eigenvalue with the largest modulus of the matrix obtained from semi-discretization of equation (7.1). Substituting from (7.3) leads to
\[ \Delta t \leq \frac{\Delta x^2}{2\alpha}. \quad (7.5) \]
This is a rather severe restriction on the time step. It implies that increasing the spatial accuracy (reducing \( \Delta x \)) must be accompanied by a significant reduction in time step.

The stability analysis performed above uses the eigenvalues of the matrix obtained from semi-discretization of the PDE at hand. We shall refer to this type of analysis as matrix stability analysis. Since boundary conditions are implemented in the semi-discretization, their effects are accounted for in the matrix stability analysis.

### 7.4 von Neumann Stability Analysis

In virtually all cases, numerical stability problems arise solely from (full) discretization of the PDEs and not from the boundary conditions. von Neumann’s stability analysis is a readily implementable analytical procedure for determination of the stability properties of a numerical method applied to a PDE which does not take into account the effects of boundary conditions. In fact, it is assumed that the spatial domain is infinite or for finite domains the boundary conditions are periodic. The technique works for linear constant coefficient differential equations.

Let’s demonstrate von Neumann’s technique by applying it to the fully discrete equation
\[ \phi_j^{(n+1)} = \phi_j^{(n)} + \frac{\alpha \Delta t}{\Delta x^2} \left( \phi_{j+1}^{(n)} - 2\phi_j^{(n)} + \phi_{j-1}^{(n)} \right). \quad (7.6) \]
7.5. **MODIFIED WAVE NUMBER ANALYSIS**

Equation (7.6) results from approximating the spatial derivative in (7.1) with second order central difference and using explicit Euler time advancement. The key part of von Neumann’s analysis is assuming a solution of the form

\[ \phi_j^{(n)} \sim \sigma^n e^{ikx_j}. \] (7.7)

Substituting in (7.6) results in

\[ \sigma^{n+1} e^{ikx_j} = \sigma^n e^{ikx_j} + \frac{\alpha \Delta t}{\Delta x^2} \sigma^n (e^{ikx_{j+1}} - 2e^{ikx_j} + e^{ikx_{j-1}}). \]

Noting that

\[ x_{j\pm1} = x_j \pm \Delta x, \]

and dividing both sides by \( \sigma^n e^{ikx_j} \) leads to

\[ \sigma = 1 + \frac{2\alpha \Delta t}{\Delta x^2} [\cos(k \Delta x) - 1] = 1 - \frac{4\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right). \]

For stability, we must have \(|\sigma| \leq 1\), that is

\[ \left| 1 - \frac{4\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right) \right| \leq 1. \]

In other words, we must have

\[ -1 \leq 1 - \frac{4\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right) \leq 1. \]

The right-hand inequality is always satisfied. The left-hand inequality can be recast in the following forms:

\[ -\frac{4\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right) \geq -2, \]

or

\[ \Delta t \leq \frac{\Delta x^2}{2\alpha \sin^2(k \Delta x/2)}. \]

The worst (or most restrictive) case occurs when \( \sin^2(k \Delta x/2) = 1 \). Thus, the maximum time step is given by

\[ \Delta t \leq \frac{\Delta x^2}{2\alpha}. \]

This is identical to equation (7.5) which was obtained using matrix stability analysis.

### 7.5 Modified Wave Number Analysis

Modified wave number analysis is very similar to von Neumann analysis; in many ways it is more straightforward. It is intended to expand the range of applicability of what we have learned about stability properties of a numerical method applied to ODEs, to the application of the same numerical method to PDEs.

Consider equation (7.1). Assuming a solution of the form

\[ \phi(x, t) = \psi(t)e^{ikx}, \]
and substituting in (7.1) leads to

$$\frac{d\psi}{dt} = -\alpha k^2 \psi. \quad (7.8)$$

In the assumed form of the solution, \(k\) is the wave number. In practice, one uses a finite difference scheme to approximate the spatial derivative. For example, using second order central difference we have

$$\frac{d\phi_j}{dt} = \frac{\phi_{j+1} - 2\phi_j + \phi_{j-1}}{\Delta x^2}, \quad j = 1, 2, \ldots, N - 1.$$ 

Let

$$\phi(x_j, t) = \phi_j(t) = \psi(t)e^{ikx_j}.$$ 

Substitution and division by \(e^{ikx_j}\) leads to

$$\frac{d\psi}{dt} = -\frac{4\alpha}{\Delta x^2} \sin^2 \left(\frac{k\Delta x}{2}\right) \psi,$$

or

$$\frac{d\psi}{dt} = -\alpha k'^2 \psi, \quad (7.9)$$

where

$$k'^2 = \frac{4}{\Delta x^2} \sin^2 \left(\frac{k\Delta x}{2}\right) = \frac{4}{\Delta x^2} \left[ \left(\frac{k\Delta x}{2}\right)^2 - \frac{1}{3} \left(\frac{k\Delta x}{2}\right)^4 + \frac{2}{45} \left(\frac{k\Delta x}{2}\right)^6 - \cdots \right].$$

By analogy to equation (7.8), \(k'\) is called the modified wave number. Application of any other finite-difference scheme, instead of the second order scheme used here, would have also led to the same form as equation (7.9) but with a different modified wave number. Each finite-difference scheme has a distinct modified wave number associated with it.

Equation (7.9) is identical to the model equation \(y' = \lambda y\), with \(\lambda = -\alpha k'^2\). We studied the stability properties of various numerical methods with respect to this model equation. Using the modified wave number analysis, we can readily obtain stability properties of a time advancement method applied to a PDE. All we have to do is to replace \(\lambda\) with \(-\alpha k'^2\) in our ODE analysis. For example, the stability restriction on the time step that can be used when the explicit Euler scheme is applied to equation (7.1) in conjunction with the second order central difference for the spatial derivative is given by

$$\Delta t \leq \frac{\Delta x^2}{2\alpha \sin^2 (k\Delta x/2)}.$$ 

The worst case scenario occurs when \(\sin^2(k\Delta x/2) = 1\), which leads to equation (7.5). If instead of the explicit Euler scheme we had used the fourth order Runge-Kutta scheme, the stability limit would have been

$$\Delta t \leq \frac{2.81 \Delta x^2}{4\alpha},$$

which is obtained directly from the intersection of the stability diagram for the 4th order Runge-Kutta scheme with the real axis. Similarly, since \(-\alpha k'^2\) is real and negative, application of the leapfrog scheme to equation (7.1) would lead to numerical instability.

As a further illustration of the modified wave number analysis, consider equation (7.4) with the central difference approximation for the spatial derivative. It can be easily verified that the substitution \(\phi_j = \psi(t)e^{ikx_j}\) results in

$$\frac{d\psi}{dt} = -ik' c \psi,$$
where

\[ k' = \frac{\sin(k \Delta x)}{\Delta x} \]

is the modified wave number. In this case, the corresponding \( \lambda \) in the model equation is purely imaginary. Thus, we would know, for example, that application of the explicit Euler or second order Runge-Kutta scheme would lead to numerical instabilities. On the other hand, if the leapfrog scheme is used, the time step has to be such that

\[ \Delta t \leq \frac{1}{|k'|c} = \frac{\Delta x}{c|\sin(k \Delta x)|} \]

Again, we will consider the worst case which leads to

\[ \Delta t_{\text{max}} = \frac{\Delta x}{c}, \]

and

\[ \frac{c \Delta t}{\Delta x} \leq 1. \]

The quantity \( c \Delta t/\Delta x \) is called the CFL number and is named after Courant, Friedrich, and Lewy.

### 7.6 Higher Dimensions

Consider the two dimensional heat equation

\[
\frac{\partial \phi}{\partial t} = \alpha \left( \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} \right). \tag{7.10}
\]

For numerical solution, one introduces a grid in the \((x, y)\) plane. Let \( \phi^{(n)}_{j,k} \) denote the value of \( \phi \) at the grid point \((j, k)\) at time step \(n\). We consider \(J + 1\) grid points in \(x\) and \(K + 1\) points in \(y\). The boundary points are at \(j = 0, J\) and \(k = 0, K\). Application of any explicit numerical method is very straightforward. For example, consider explicit Euler in conjunction with second order finite-difference approximation for the spatial derivatives

\[
\frac{\phi^{(n+1)}_{j,k} - \phi^{(n)}_{j,k}}{\Delta t} = \alpha \left[ \frac{\phi^{(n)}_{j+1,k} - 2\phi^{(n)}_{j,k} + \phi^{(n)}_{j-1,k}}{\Delta x^2} + \frac{\phi^{(n)}_{j,k+1} - 2\phi^{(n)}_{j,k} + \phi^{(n)}_{j,k-1}}{\Delta y^2} \right].
\]

Given an initial condition on the grid points \( \phi^{(0)}_{j,k} \), for each \(j\) and \(k\) one simply marches forward in time to obtain the solution at subsequent time steps. When \(j = 1, J - 1\), or \(k = 1, K - 1\), boundary values are required, and their value from prescribed boundary conditions are used.

The stability properties of this scheme can be analyzed in the same manner as in the one-dimensional case. Considering solutions of the form \( \phi(x_j, y_k, t) = \psi(t)e^{ik_1x_j+ik_2y_k} \), the heat equation transforms to

\[
\frac{d\psi}{dt} = -\alpha \left( k'_1^2 + k'_2^2 \right) \psi,
\]

where \(k'_1\) and \(k'_2\) are the modified wave numbers

\[ k'_1 = \frac{4}{\Delta x^2} \sin^2 \left( \frac{k_1 \Delta x}{2} \right), \quad k'_2 = \frac{4}{\Delta y^2} \sin^2 \left( \frac{k_2 \Delta y}{2} \right) . \]

Therefore, for stability we must have

\[ \Delta t \leq \frac{2}{\alpha} \left\{ \frac{4}{\Delta x^2} \sin^2 \left( \frac{k_1 \Delta x}{2} \right) + \frac{4}{\Delta y^2} \sin^2 \left( \frac{k_2 \Delta y}{2} \right) \right\}^{-1} . \]
The worst case is when \( \sin^2(k_1 \Delta x/2) = 1 \) and \( \sin^2(k_2 \Delta y/2) = 1 \). Thus,

\[
\Delta t = \frac{1}{2\alpha} \left( \frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right)^{-1}.
\]

In the special case where \( \Delta x = \Delta y = h \), we obtain

\[
\Delta t \leq \frac{h^2}{4\alpha},
\]

which is twice more restrictive compared to the one-dimensional case. Similarly, in three dimensions one obtains

\[
\Delta t \leq \frac{h^2}{6\alpha}.
\]

### 7.7 Accuracy via Modified Equation

Since the numerical solution is an approximation of the exact solution, it does not exactly satisfy the continuous partial differential equation at hand, but it satisfies a modified equation. Consider the heat equation (7.16). Let \( \tilde{\phi} \) be the exact solution and \( \phi \) the numerical solution obtained from application of the explicit Euler and second order spatial differencing

\[
\phi_j^{(n+1)} = \phi_j^{(n)} + \Delta t \frac{\partial \phi_j^{(n)}}{\partial t} + \frac{1}{2} \Delta t^2 \frac{\partial^2 \phi_j^{(n)}}{\partial t^2} + \ldots.
\]

Thus,

\[
\frac{\phi_j^{(n+1)} - \phi_j^{(n)}}{\Delta t} = \frac{\partial \phi_j^{(n)}}{\partial t} + \frac{1}{2} \Delta t \frac{\partial^2 \phi_j^{(n)}}{\partial t^2} + \ldots.
\]

Similarly,

\[
\frac{\phi_j^{(n)} - 2\phi_j^{(n)} + \phi_{j-1}^{(n)}}{\Delta x^2} = \frac{\partial^2 \phi_j^{(n)}}{\partial x^2} + \frac{1}{12} \Delta x^2 \frac{\partial^4 \phi_j^{(n)}}{\partial x^4} + \ldots.
\]

Substitution in (7.12) leads to

\[
L[\phi_j^{(n)}] = \left( \frac{\partial \phi_j^{(n)}}{\partial t} - \alpha \frac{\partial^2 \phi_j^{(n)}}{\partial x^2} \right) - \frac{1}{12} \Delta x^2 \frac{\partial^4 \phi_j^{(n)}}{\partial x^4} + \frac{1}{2} \Delta t \frac{\partial^2 \phi_j^{(n)}}{\partial t^2} + \ldots.
\]

This equation was derived without reference to a specific set of space-time grid points. In other words, the indices \( j \) and \( n \) are generic, and equation (7.13) applies to any point in space and time. That is,

\[
L[\phi] = \left( \frac{\partial \phi}{\partial t} - \alpha \frac{\partial^2 \phi}{\partial x^2} \right) - \frac{1}{12} \Delta x^2 \frac{\partial^4 \phi}{\partial x^4} + \frac{1}{2} \Delta t \frac{\partial^2 \phi}{\partial t^2} + \ldots.
\]
Let $\phi$ be the solution of the discrete equation (7.11). Let $L[\phi] = 0$, and we can see that the numerical solution actually satisfies the following modified differential equation

\[
\frac{\partial \phi}{\partial t} - \alpha \frac{\partial^2 \phi}{\partial x^2} = \frac{1}{12} \Delta x^2 \frac{\partial^2 \phi}{\partial x^4} + \frac{1}{2} \Delta t \frac{\partial^2 \phi}{\partial t^2} + \ldots,
\]

instead of equation (7.16). As $\Delta t$ and $\Delta x$ independently approach zero, the modified equation approaches the exact PDE and the scheme is said to be consistent. The second form of the modified differential equation is obtained by eliminating the higher-order time derivatives and the mixed time and space derivatives from the above form by repeated differentiation of the modified equation itself. After doing this we obtain

\[
\frac{\partial \phi}{\partial t} - \alpha \frac{\partial^2 \phi}{\partial x^2} = \left( \frac{1}{12} \Delta x^2 - \frac{1}{2} \alpha^2 \Delta t \right) \frac{\partial^2 \phi}{\partial x^4} + \left( \frac{1}{360} \Delta x^4 - \frac{1}{12} \alpha^2 \Delta t \Delta x^2 + \frac{1}{3} \alpha^3 \Delta t^2 \right) \frac{\partial^6 \phi}{\partial x^6} + \ldots
\]

Thus, we can increase the accuracy of the numerical solution by setting

\[
\frac{\alpha \Delta x^2}{12} = \frac{\alpha^2 \Delta t}{2},
\]

or selecting the space and time increments such that

\[
\frac{\alpha \Delta t}{\Delta x^2} = \frac{1}{6}.
\]

This constraint is within the stability limit derived earlier but is rather restrictive.

### 7.8 Dufort-Frankel Method

An interesting application of the modified equation analysis arises in examination of a numerical scheme due to Dufort and Frankel. The method is derived in two steps. First consider the combination of leapfrog time advancement and second order spatial differencing:

\[
\phi^{(n+1)}_j - \phi^{(n-1)}_j = \frac{\alpha}{\Delta x^2} \left[ \phi^{(n)}_{j+1} - 2 \phi^{(n)}_j + \phi^{(n)}_{j-1} \right] + O(\Delta x^2, \Delta t^2).
\]

This scheme is formally second order accurate in space and time. However, it is \textit{unconditionally unstable}. The Dufort-Frankel scheme is obtained by substituting for $\phi^{(n)}_j$ in the right hand side the following second order approximation:

\[
\phi^{(n)}_j = \frac{\phi^{(n+1)}_j + \phi^{(n-1)}_j}{2} + O(\Delta t^2).
\]

With this substitution, the formal order of accuracy does not change. Rearranging terms results in

\[
(1 + 2\beta)\phi^{(n+1)}_j = (1 - 2\beta)\phi^{(n-1)}_j + 2\beta \phi^{(n)}_{j+1} + 2\beta \phi^{(n)}_{j-1},
\]

where $\beta = \alpha \Delta t / \Delta x^2$. It turns out that this method is \textit{unconditionally stable}. It has the same stability property of the implicit methods but with a lot less work per time step. Recall that application of implicit methods requires matrix inversions at each time step.

We shall now derive the modified equation for the Dufort-Frankel scheme. Substitution of Taylor series expansions for $\phi^{(n)}_j$, $\phi^{(n)}_{j+1}$, $\phi^{(n+1)}_j$, and $\phi^{(n-1)}_j$ into (7.15) and after some algebra leads to

\[
\frac{\partial \phi}{\partial t} - \alpha \frac{\partial^2 \phi}{\partial x^2} = \frac{\Delta t^2}{6} \frac{\partial^2 \phi}{\partial x^4} + \frac{\alpha \Delta x^2}{12} \frac{\partial^4 \phi}{\partial x^4} - \frac{\alpha \Delta t^4}{12 \Delta x^2} \frac{\partial^6 \phi}{\partial x^6} + \ldots
\]

The difficulty arises from the third term on the right hand side. For a given time step, if we refine the spatial mesh, the error actually increases. Thus, one cannot increase the accuracy of the numerical solution by arbitrarily letting $\Delta x \to 0$ and $\Delta t \to 0$. The third term approaches zero only if $\Delta t \to 0$ faster than $\Delta x$. This is an example of an \textit{inconsistent} numerical method.
7.9 Crank-Nicolson Method

We have established that semi-discretization of the heat equation leads to a stiff system of ODEs. Moreover, the stability limit for application of explicit schemes is too stringent. For these reasons implicit methods are typically used in solving parabolic equations. The most popular scheme is the trapezoidal method which is often referred to as the Crank-Nicolson method when applied to the heat equation

$$\frac{\partial \phi}{\partial t} = \alpha \frac{\partial^2 \phi}{\partial x^2}. \quad (7.16)$$

Application of the trapezoidal method leads to

$$\phi^{(n+1)} - \phi^{(n)} = \frac{\Delta t}{2} \left[ \frac{\partial^2 \phi^{(n+1)}}{\partial x^2} + \frac{\partial^2 \phi^{(n)}}{\partial x^2} \right].$$

Now, we approximate the spatial derivative with the second order finite-difference scheme

$$\phi_j^{(n+1)} - \phi_j^{(n)} = \frac{\alpha \Delta t}{2} \left[ \frac{\phi_{j+1}^{(n+1)} - 2\phi_j^{(n+1)} + \phi_{j-1}^{(n+1)}}{\Delta x^2} + \frac{\phi_j^{(n)} - 2\phi_j^{(n)} + \phi_{j-1}^{(n)}}{\Delta x^2} \right].$$

Let $$\beta = \frac{\alpha \Delta t}{2 \Delta x^2}$$. Then, collecting the unknowns (terms with superscript $$(n+1)$$) on the left hand side, results in the following tridiagonal system of equations

$$-\beta \phi_{j+1}^{(n+1)} + (1 + 2\beta) \phi_j^{(n+1)} - \beta \phi_{j-1}^{(n+1)} = \beta \phi_{j+1}^{(n)} + (1 - 2\beta) \phi_j^{(n)} + \beta \phi_{j-1}^{(n)}, \quad j = 1, 2, \ldots, J - 1. \quad (7.17)$$

Thus, at every time step a tridiagonal system of equations must be solved. In one dimension, this does not cause any difficulty and requires of the order of $$J$$ arithmetic operations. Application of implicit methods to partial differential equations always requires solving a system of algebraic equations.

We can investigate the stability properties of this scheme using the von Neumann analysis or the equivalent modified wave number analysis. Recall that when applied to the model equation $$y' = \lambda y$$, the amplification factor for the trapezoidal method is

$$\sigma = \frac{1 + \lambda \Delta t/2}{1 - \lambda \Delta t/2}.$$ 

Using the modified wave number analysis, the amplification factor for the trapezoidal method applied to the heat equation is obtained by substituting $$-\alpha k^2$$ for $$\lambda$$ in this equation. Here, $$k'$$ is the modified wave number which was derived in equation (7.9):

$$k'^2 = \frac{4}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right).$$ 

Thus,

$$\sigma = \frac{1 - 2 \frac{\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right)}{1 + 2 \frac{\alpha \Delta t}{\Delta x^2} \sin^2 \left( \frac{k \Delta x}{2} \right)}.$$ 

Since $$|\sigma| \leq 1$$, and the method is unconditionally stable. Notice that for large $$\alpha \Delta t/\Delta x^2$$, $$\sigma \rightarrow -1$$, which leads to temporal oscillations in the solution. However the solution will always remain bounded.

7.10 Crank-Nicolson Scheme in Higher Dimensions

Severe time step restrictions with explicit schemes provide an inducement for the use of implicit methods. Applying the Crank-Nicolson scheme to the two-dimensional heat equation results in

$$\frac{\phi^{(n+1)} - \phi^{(n)}}{\Delta t} = \frac{\alpha}{2} \left[ \frac{\partial^2 \phi^{(n+1)}}{\partial x^2} + \frac{\partial^2 \phi^{(n+1)}}{\partial y^2} + \frac{\partial^2 \phi^{(n)}}{\partial x^2} + \frac{\partial^2 \phi^{(n)}}{\partial y^2} \right].$$
Using second order finite-difference in space and assuming \( \Delta x = \Delta y = h \) yields
\[
\phi^{(n+1)}_{j,k} - \phi^{(n)}_{j,k} = \frac{\alpha \Delta t}{2h^2} \left[ \phi^{(n+1)}_{j+1,k} - 2\phi^{(n+1)}_{j,k} + \phi^{(n+1)}_{j-1,k} + \phi^{(n+1)}_{j,k+1} - 2\phi^{(n+1)}_{j,k} + \phi^{(n+1)}_{j,k-1} \right] + \frac{\alpha \Delta t}{2h^2} \left[ \phi^{(n)}_{j+1,k} - 2\phi^{(n)}_{j,k} + \phi^{(n)}_{j-1,k} + \phi^{(n)}_{j,k+1} - 2\phi^{(n)}_{j,k} + \phi^{(n)}_{j,k-1} \right].
\]

Let \( \beta = \alpha \Delta t / 2h^2 \), and collecting unknowns in the left hand side, yields
\[
-\beta \phi^{(n+1)}_{j+1,k} + (1 + 4\beta) \phi^{(n+1)}_{j,k} - \beta \phi^{(n+1)}_{j-1,k} - \beta \phi^{(n+1)}_{j,k+1} + \beta \phi^{(n+1)}_{j,k-1} = \\
\beta \phi^{(n)}_{j+1,k} + (1 - 4\beta) \phi^{(n)}_{j,k} + \beta \phi^{(n)}_{j-1,k} + \beta \phi^{(n)}_{j,k+1} + \beta \phi^{(n)}_{j,k-1} = F^{(n)}_{j,k}.
\]

This is a system of algebraic equations for \( \phi^{(n+1)}_{j,k} \). The resulting \((J-1)^2 \times (K-1)^2\) matrix is of block tridiagonal form
\[
B[A, B, C] \Phi^{(n+1)} = F^{(n)},
\]
where
\[
\Phi = (\Phi_1, \ldots, \Phi_{J-1})^T, \quad \Phi_j = (\phi_{j,1}, \ldots, \phi_{j,K-1})^T, \quad F = (F_1, \ldots, F_{J-1})^T, \quad F_j = (F_{j,1}, \ldots, F_{j,K-1})^T,
\]
and
\[
B[A, B, C] = \begin{bmatrix}
B & C \\
A & B & C \\
\vdots & \vdots & \ddots & \ddots \\
A & B & C \\
A & B
\end{bmatrix},
\]
where \( A, B, \) and \( C \) are \((K-1) \times (K-1)\) matrices. In the present case, \( A \) and \( C \) are diagonal matrices whereas \( B \) is tridiagonal. Clearly, \( B \) is very large. For example, for \( J = 101 \) and \( K = 101 \), \( B \) has \( 10^{8} \) elements. Noting that \( B \) is banded, in the Gauss elimination process one needs to store only \( 2(J-1)^2 K \) elements, which for the above example, amounts to \( 2 \times 10^6 \) words of memory. This is still too demanding.

### 7.11 Relaxation Methods

The word relaxation is used to mean the process of finding the iterative solution of a coupled set of algebraic equations. In our applications, the algebraic equations represent a set of difference equations usually derived from the differencing of partial differential equations. The number of iterations required to bring about the solution depends upon both the rate of convergence and the convergence path. Since the exact solution is generally unknown, and since the process is iterative, some test of a converged solution must be supplied. The quantity tested is called the residual. The degree to which the residual must be reduced to zero before a solution is acceptable is quite arbitrary and generally problem dependent. In the best of worlds, the solution is independent of the initial guess and is said to have converged when it is invariant with further iteration.

These methods are to be contrasted with direct methods, of which Gaussian elimination is the prototype. With direct methods the solution is determined exactly (up to machine precision) in a finite number of arithmetic steps. For systems that arise from a two-dimensional elliptic equation (such as we will discuss shortly), very efficient direct methods have been developed. They are usually based on the fast Fourier transform (FFT) or the method of cyclic reduction. When applied to problems on an \( N \times N \) grid, these methods require \( O(N^2 \log N) \) arithmetic operations. Therefore, since they approach the minimum operation count of \( O(N^2) \) operations, these methods are nearly optimal. However, they are also rather specialized and can be applied primarily to systems which arise from separable self-adjoint
boundary value problems. Classical relaxation methods are easy to implement and may be successfully applied to more general systems than the direct methods. However, as we will see, relaxation schemes also suffer from some disabling limitations.

We start by considering the two-dimensional, second-order partial differential equation known as Poisson’s equation

\[ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y), \]  

(7.19)

in the unit square \((0 < x < 1, \ 0 < y < 1)\) subject to the condition that \(u = 0\) on the boundary.

This problem may be cast in a discrete form by defining the grid points \((x_i, y_j) = (i\Delta x, j\Delta y)\), where \(\Delta x = 1/I\) and \(\Delta y = 1/J\). Replacing the derivatives in (7.19) by central second-order finite differences leads to the system of linear equations

\[ \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{\Delta x^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{\Delta y^2} = f_{i,j}, \quad 1 \leq i \leq I - 1, \ 1 \leq j \leq J - 1, \]  

(7.20)

where \(u_{i,j}\) is an approximation to the exact solution \(u(x_i, y_j)\) and \(f_{i,j} = f(x_i, y_j)\). There are now \((I - 1) \times (J - 1)\) interior grid points and the same number of unknowns in the problem. Many different ways to order the unknowns exist. Consider the lexicographic ordering by lines of constant \(i\). The unknowns of the \(i\)th row of the grid may be collected in the vector \(u_i = (u_{i,1}, \ldots, u_{i,J-1})^T\) for \(1 \leq i \leq I - 1\). Similarly, let \(f_i = h^2(f_{i,1}, \ldots, f_{i,J-1})^T\). The system (7.20) may then be given in matrix form as

\[ Au \equiv B[I, B[1, -4, 1], I]u = f, \]

where \(u = (u_1, \ldots, u_{I-1})^T\) and \(f = (f_1, \ldots, f_{I-1})^T\), or more explicitly

\[ \begin{bmatrix} B & I \\ I & B & I \\ \vdots & \ddots & \ddots \\ I & B & I \\ I & B \\ \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{I-2} \\ u_{I-1} \\ \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{I-2} \\ f_{I-1} \\ \end{bmatrix}, \]

(7.21)

where for simplicity, we have taken \(\Delta x = \Delta y = h\). This system is symmetric, block tridiagonal and sparse (the matrix entries are predominantly zero). It has dimension of \((I - 1) \times (I - 1)\) blocks. Each diagonal block is a \((J - 1) \times (J - 1)\) tridiagonal matrix and each off-diagonal block is a \((J - 1) \times (J - 1)\) identity matrix \(I\).

We now consider how our model problem might be treated using conventional relaxation methods. Let us first establish the notation. Let

\[ Au = f \]  

(7.22)

denote the system of linear equations (7.21). We will always use \(u\) to denote the exact solution of (7.22) and \(v\) to denote an approximation to the exact solution, which is generated by some iterative method. Bold symbols, such as \(u\) and \(v\), represent vectors, while the \(i\)th components of these vectors are denoted by \(u_i\) and \(v_i\). Note that \(u\) is not exact solution of the Poisson equation, but is an \(O(h^2)\) approximation to it.

Clearly the exact solution of the problem (7.22) is unknown, while an approximation \(v\), which we assume has been computed, is known. There are two important measures of \(v\) as an approximation to \(u\). One is the error and is given simply by

\[ e = u - v. \]
7.11. RELAXATION METHODS

The error is also a vector and its size may be measured by any of the standard vector norms. The most commonly used norms for this purpose are the $l^\infty$ and $l^2$ norms defined by

$$ ||e||_\infty = \max_{1 \leq i \leq I-1, 1 \leq j \leq J-1} |e_{i,j}| \quad \text{and} \quad ||e||_2 = \left( \sum_{i=1}^{I-1} \sum_{j=1}^{J-1} e_{i,j}^2 \right)^{1/2}. $$

Unfortunately, the error is just as unobtainable as the exact solution itself. However, a computable measure of how well $v$ approximates $u$ is the residual, given by

$$ r = f - Av. \quad (7.23) $$

The residual is simply the amount by which the approximation $v$ fails to satisfy the original problem (7.22). It is also a vector and its size may be measured by the same norm used for the error. Notice that by the uniqueness of the solution $u$, $r = 0$ if and only if $e = 0$. However, it may not be true that, when $r$ is small in norm, $e$ is also small in norm.

If we rearrange (7.23) as

$$ Av = f - r \quad (7.24) $$

and then subtract it from (7.22), we find an extremely important relationship between the error and the residual

$$ Ae = r \quad (7.25) $$

called the residual equation. It says that the error satisfies the same equation as the unknown $u$ when $f$ is replaced by the residual $r$.

We can now anticipate, in an imprecise way, how the residual equation (7.25) can be used to great advantage. Assume that an approximation $v$ has been computed by some unspecified method. It is easy to compute the residual from (7.23). To improve the approximation $v$, we can solve the residual equation (7.25) for $e$ and then compute a new approximation using the definition of the error

$$ u = v + e. $$

In practice, this method is not as straightforward as stated and must be applied more carefully than has been indicated. Nevertheless, this idea of residual correction will be very important.

An iterative method applied to equation (7.22) is a procedure of the type

$$ A_1v^{(p+1)} = A_2v^{(p)} + f, \quad (7.26) $$

where $A_1$ and $A_2$ are matrices (from the nontrivial decomposition $A = A_1 - A_2$, where neither $A_1$ nor $A_2$ are equal to zero). Given some initial guess $v^{(0)}$ for the solution, we use equation (7.26) to find $v^{(1)}$, $v^{(2)}$, and so on. The iterates are given by

$$ v^{(p+1)} = A_1^{-1}A_2v^{(p)} + A_1^{-1}f = Pv^{(p)} + b. \quad (7.27) $$

If this is to be a satisfactory means of solving the original equation, it should have the following properties:

1. It should converge to the exact solution of the original equation (7.22). That is, we should have

$$ \lim_{p \to \infty} v^{(p)} = u. $$

2. The convergence should be rapid if the method is to be efficient; that is, the limit should be reached quickly.

3. So that each iteration does not require much computation, the matrix $A_1$ should be easy to invert (the best possibility for this is that $A_1$ is diagonal; other good ones are that $A_1$ is tridiagonal or triangular) and the matrix $A_2$ should be as simple as possible to facilitate the computation of $A_2v^{(p)}$. Generally, this means that $A_2$ should be sparse, which is possible only if $A$ is also sparse. Iterative methods are thus best for sparse matrices.
It is not difficult to construct an iterative method, but constructing one with all of the above desired properties is another story.

Stability and convergence proofs are quite tedious. One therefore seeks a few simple guidelines to determine the suitability of a method without going to a full stability or convergence proof. The following rules derive from ordinary differential equations and relations between eigenvalues and the main diagonal of a matrix. At the same time we also establish a connection between the iterative process and the time-marching process. We start by considering the set of linear ODE’s

\[
\frac{du}{dt} = Au - f, \tag{7.28}
\]

where \(A\) and \(f\) are independent of \(t\). This has both a particular and a complementary solution. In the following, the particular solution is called the steady-state solution and the complementary solution is called the transient solution. Then it follows that

\[
u(t) = u'(t) + A^{-1}f, \tag{7.29}\]

where \(u'(t) = e^{At}u(0)\) represents the transient solution. If we use a forward Euler time integration, equation (7.28) becomes

\[
u^{(n+1)} = (I + \alpha A)u^{(n)} - \alpha f,
\]

where we have suggestively used the symbol \(\alpha\) for the time step. Now note that if we choose \(A_1 = -1/\alpha\) and \(A_2 = -(I + \alpha A)/\alpha\), then the iterative scheme (7.27) represents a time-accurate solution of the ODE (7.28). For general \(\alpha\), the steady-state solution does not necessarily follow as the limit of a time-accurate solution when \(t \to \infty\).

If \(A\) in (7.28) can be diagonalized (and we optimistically assume that it can always be diagonalized), then via eigenvector transformation one obtains uncoupled equations of the form

\[
\frac{dw}{dt} = \lambda w - b
\]

where \(\lambda\) is any one of the eigenvalues of \(A\). Then the general solution is

\[
w(t) = e^{Mt}w(0) + \frac{b}{\lambda}.
\]

To avoid exponential growth, all of the eigenvalues of \(A\) must have negative real parts. If not, the scheme is unstable or divergent.

The first rule then is: avoid operators that cause eigenvalues with positive real parts to be formed. Since we are not particularly interested in a time-accurate solution, one approach is to pre-condition the right hand side of equation (7.28) by multiplying it by some non-singular matrix \(M\) and to study the solution of the system

\[
\frac{du}{dt} = M [Au - f]. \tag{7.30}
\]

The procedure is to choose \(M\) such that the eigenvalues of \(MA\) have negative real parts, and then compute (using as little arithmetic and as few “time” steps as possible) the steady-state solution of equation (7.30). Obviously, the general solution of (7.30) is

\[
u(t) = u''(t) + (MA)^{-1}Mf,
\]

or since

\[(MA)^{-1}Mf = A^{-1}M^{-1}Mf = A^{-1}f,
\]

we have the same solution (7.29). Since we have (somehow) insured that the matrix \(MA\) is inherently stable, the converged steady-state solution is

\[
u(\infty) = A^{-1}f,
\]
which is the desired result. Although the transient solution has lost its physical meaning, it still exists in the solution process.

To make the first rule useful, one needs to have some feeling for eigenvalue behavior. The following are some of their properties:

1. The sum of the eigenvalues equals the trace (the sum of the diagonal elements).
2. A lower or upper triangular matrix has as its eigenvalues the diagonal elements.
3. A symmetric matrix has real eigenvalues.
4. A skew-symmetric matrix has imaginary eigenvalues.
5. The eigenvalues of $\alpha A$, where $\alpha$ is a scalar multiplier, are $\alpha$ times the eigenvalues of $A$, i.e., $\lambda(\alpha A) = \alpha \lambda(A)$.
6. The eigenvalues of $\alpha I + A$ are $\alpha + \lambda(A)$.
7. Gerschgorin theorem: All the eigenvalues of $A$ lie in the union of disks

$$|\lambda - a_{i,i}| \leq \sum_{j \neq i} |a_{i,j}|.$$ 

Thus, if each diagonal element is negative and greater or equal in absolute value than the sum of the absolute values of each element in its row (diagonal dominance), then the eigenvalues will have negative real parts.

One sees from these relations that the diagonal is important. From this, one has a second rule: avoid adding positive coefficients to the diagonal of $A$.

Another method for investigating the convergence of iterative methods is to look at the error vector

$$e^{(p)} = u - v^{(p)}.$$ 

If the method converges, then as $p \to \infty$, $v^{(p)} = v^{(p+1)} = u$, so that from equation (7.27) we can write

$$u = Pu + b.$$ 

Subtracting equation (7.27) from this equation we get a homogeneous equation for the error vector

$$e^{(p+1)} = Pe^{(p)},$$ 

leading to

$$e^{(p)} = P^p e^{(0)}.$$ 

If we now choose a particular vector norm and its associated matrix norm, it is possible to bound the error after $p$ iterations by

$$\|e^{(p)}\| \leq \|P\|^p \|e^{(0)}\|.$$ 

This leads to the conclusion that if $\|P\| < 1$, then the error is forced to zero as the iteration proceeds. Thus, as $p \to \infty$, the necessary and sufficient condition for the initial error to go to zero is that all of the eigenvalues of the matrix $P$ must be less than unity in magnitude. The rate of convergence of the method is set by the rate of the slowest decaying eigenmode (the factor with the largest modulus). The modulus of the largest eigenvalue is called the spectral radius

$$\rho(P) = \max \{|\lambda_i(P)|\},$$ 

where $\lambda_i(P)$ denotes any of the eigenvalues of $P$. 

\[87\]
The term \( \rho(P) \) is also called the \textit{convergence factor} when it appears in the context of iterative methods. It is roughly the worst factor by which the error is reduced by each relaxation sweep. It also tells us how many iterations are required to reduce the error by a factor of \( 10^{-d} \). Let \( N \) be the smallest number of iterations required to reduce the overall error by a factor of \( 10^{-d} \). Then
\[
\|e^{(N)}\| \leq 10^{-d}.
\]
This condition is approximately satisfied if
\[
[\rho(P)]^N \leq 10^{-d}.
\]
Solving for \( N \), we have
\[
N \geq -d \log_{10}[\rho(P)] = -d \frac{\ln 10}{\ln \rho(P)}.
\]
The quantity \(-d \log_{10}[\rho(P)]\) is called the \textit{convergence rate}. Its reciprocal gives the number of iterations required to reduce the error by one decimal digit. We see that as \( \rho(P) \) approaches 1, the convergence rate decreases. Small values of \( \rho(P) \) (that is, \( \rho(P) \) positive and near zero) give a high convergence rate.

The eigenvectors of the matrix \( A \) will also be important in later discussions. These eigenvectors correspond very closely to the eigenfunctions of the continuous model problem. Just as we can expand fairly arbitrary functions using a set of eigenfunctions, it is also possible to expand arbitrary vectors in terms of a set of eigenvectors. Let \( e^{(0)} \) be the error in an initial guess. Then it is possible to represent \( e^{(0)} \) using the eigenvectors of \( A \) in the form
\[
e^{(0)} = \sum_{k=1}^{l-1} c_k w_k,
\]
where the coefficients \( c_k \) are real. We have seen that the error after \( p \) iterations is given by
\[
e^{(p)} = P^p e^{(0)}.
\]
Using the eigenvector expansion for \( e^{(0)} \) we see that if the eigenvectors of \( A \) and \( P \) are the same (and therefore \( Pw_k = \lambda_k(P)w_k \)) then
\[
e^{(p)} = \sum_{k=1}^{l-1} c_k P^p w_k = \sum_{k=1}^{l-1} c_k [\lambda_k(P)]^p w_k.
\]
This provides an eigenvector expansion for \( e^{(p)} \). We see that after \( N \) iterations, the \( k \)th mode of the initial error has been reduced by a factor of \( [\lambda_k(P)]^N \). When applied to a single mode, the iteration can change at most the amplitude of that mode; however, it cannot convert that mode into different modes. As we will see later, this property is not shared by all stationary iterations.

7.12 Jacobi Method

We begin with one of the simplest possible iterative schemes and systematically improve on it. As already observed, it is important to express the relaxation schemes in matrix form. We write the matrix \( A \) in the form
\[
A = D - L - U,
\]
where \( D \) is the diagonal of \( A \) with elements \( a_{i,i} \) which is assumed to be nonzero, and \(-L\) and \(-U\) are the strictly lower and upper triangular parts of \( A \) with elements \(-a_{i,j} \) (\( i > j \)) and \(-a_{i,j} \) (\( i < j \)), respectively. Then equation (7.22) becomes
\[
(D - L - U)u = f.
\]
This may be rewritten as
\[ Du = (L + U)u + f \]
or
\[ u = D^{-1}(L + U)u + D^{-1}f. \]

In component form the iteration scheme expressed by this last equation is the Jacobi method and is written as
\[ v_{i}^{(p+1)} = \frac{1}{a_{i,i}} \left[ \sum_{j=1}^{I-1} -a_{i,j}v_{j}^{(p)} + f_{i} \right]. \]

This corresponds precisely to solving the \( i \)th equation for the \( i \)th unknown \( v_{i} \) for \( 1 \leq i \leq I - 1 \). If \( A \) is strictly diagonally dominant, then the Jacobi method will converge. If we substitute the appropriate values of the matrices corresponding to equation (7.20) the last equation is simply
\[ v_{i,j}^{(p+1)} = \frac{1}{4} \left[ v_{i-1,j}^{(p)} + v_{i+1,j}^{(p)} + v_{i,j-1}^{(p)} + v_{i,j+1}^{(p)} \right] - \frac{1}{4} f_{i,j}. \]

If we define the Jacobi iteration matrix by
\[ P_{J} = D^{-1}(L + U), \]
the Jacobi method in matrix form is
\[ v^{(p+1)} = P_{J}v^{(p)} + D^{-1}f. \]

There is a simple but important modification which can be made to the Jacobi iteration. As before, we compute the new Jacobi iterates using
\[ v^{(\star)} = P_{J}v^{(p)} + D^{-1}f. \]

However, \( v^{(\star)} \) is now only an intermediate value. The new approximation is given by the weighted average
\[ v^{(p+1)} = (1 - \omega)v^{(p)} + \omega v^{(\star)}, \]
where \( 0 < \omega \leq 1 \) is a weighting factor which may be chosen. This generates an entire family of iterations called the weighted or damped Jacobi method. Notice that with \( \omega = 1 \) we have the original Jacobi iteration. The combined method is given by
\[ v^{(p+1)} = [(1 - \omega)I + \omega P_{J}]v^{(p)} + \omega D^{-1}f. \]

If we define the weighted Jacobi iteration matrix by
\[ P_{\omega} = (1 - \omega)I + \omega P_{J}, \]
then the method may be expressed as
\[ v^{(p+1)} = P_{\omega}v^{(p)} + \omega D^{-1}f. \]

Note that by using equation (7.24) the weighted Jacobi iteration can also be written in the form
\[ v^{(p+1)} = v^{(p)} + \omega D^{-1}r^{(p)}. \]

This says that the new approximation is obtained from the current one by adding an appropriate weighting of the residual. This is just one example of a stationary linear iteration. In general, recalling that \( e = u - v \) and \( Ae = r \), we have
\[ u = v + A^{-1}r. \]
From this expression, an iteration may be formed by taking
\[ v^{(p+1)} = v^{(p)} + Bv^{(p)}, \]
where \( B \) is some approximation to \( A^{-1} \). If \( B \) can be chosen “close” to \( A^{-1} \), the iteration should be effective.

The Jacobi and weighted Jacobi methods wait until all components of the new approximation have been computed before using them. This requires \( 2(I-1)(J-1) \) storage locations for the approximation vector. It also means that new information is not used as soon as it is available.

### 7.13 Gauss-Seidel Method

The Gauss-Seidel method incorporates a simple change on the Jacobi method: components of the new approximation are used as soon as they computed. This means that components of the approximation vector \( v \) are over-written as soon as they are updated. This small change reduces the storage requirements for the approximation vector to only \((I-1)(J-1)\) locations. It can be shown that the Gauss-Seidel method is equivalent to successively setting each component of the residual vector to zero and solving for the corresponding component of the solution.

As in the Jacobi method, splitting the matrix \( A \) in the form \( A = D - L - U \), we can write the system of equations (7.22) as
\[
(D - U)u = Lu + f
\]
or
\[
u = (D - U)^{-1}Lu + (D - U)^{-1}f.
\]

This corresponds precisely to solving the \( i \)th equation for \( u_i \) and using the new approximations for components \( 1, 2, \ldots, i-1 \). If we now define the Gauss-Seidel iteration matrix by
\[
P_G = (D - U)^{-1}L,
\]
we can express the method as
\[v ← P_Gv + (D - U)^{-1}f.\]

For our discrete Poisson equation the method is
\[v^{(p+1)}_{i,j} = \frac{1}{4} \left( v^{(p+1)}_{i-1,j} + v^{(p)}_{i+1,j} + v^{(p+1)}_{i,j-1} + v^{(p)}_{i,j+1} \right) - \frac{1}{4}f_{i,j}\]

So far we have assumed that the components of \( v \) are updated in ascending order. For the Jacobi and weighted Jacobi methods, the order is immaterial, since components are never over-written. However, for the Gauss-Seidel method, the order of updating is significant. Instead of sweeping through the grid points in ascending order, we could sweep through the unknowns in descending order or alternate between ascending and descending orders. The latter procedure is called the symmetric Gauss-Seidel method.

Another effective alternative is to update all the \((i + j) = even\) components first, and then return and update all the \((i + j) = odd\) components. This strategy leads to the red-black Gauss-Seidel method which is illustrated in Figure 7.1 for our two-dimensional grid. The advantages of red-black versus regular Gauss-Seidel are not immediately apparent; the issue is often problem-dependent. However, red-black Gauss-Seidel does have a clear advantage in terms of implementation on a parallel computer. The red points need only black points for their updating and therefore may be updated in any order. This work represents \((I-1)(J-1)/2\) independent tasks that can be distributed among several independent processors. In a similar way, the black sweep can also be done by several independent processors.
7.14 Convergence Properties

Before continuing with more iterative methods, it is important to gain some understanding of how these basic iterations perform. We shall proceed both by analysis and by experimentation.

When studying stationary linear iterations, it is sufficient to work with the homogeneous linear system $Au = 0$ and use arbitrary initial guesses to start the relaxation scheme. One reason for doing this is that the exact solution ($u = 0$) is known and the error in an approximation $v$ is simply $-v$. Furthermore, for illustrative purposes, we use the one-dimensional form of the model problem with $f = 0$. Then we have

$$u_{j-1} - 2u_j + u_{j+1} = 0, \quad 1 \leq j \leq J - 1,$$

$$u_0 = u_J = 0.$$

We obtain some insight by applying various iterations to this system of equations with an initial guess consisting of the vectors (of Fourier modes)

$$v_j = \sin\left(\frac{jk\pi}{J}\right),$$

where $0 \leq j \leq J$ and $1 \leq k \leq J - 1$. Recall that $j$ denotes the component (or associated grid point) of the vector $v$. The integer $k$ is the wavenumber and it indicates the number of half sine waves which constitute $v$ on the domain of the problem. When it becomes necessary to designate the entire vector $v$ with wavenumber $k$, we shall use $v_k$. Figure 7.2 illustrates initial guesses $v_1$, $v_3$, and $v_6$.

Notice that small values of $k$ correspond to long, smooth waves, while large values of $k$ correspond to highly oscillatory waves.

We begin by applying the weighted Jacobi iteration with $\omega = 2/3$ on a grid with $J = 64$. Beginning with initial guesses of $v_1$, $v_3$, and $v_6$, the iteration is applied 100 times. After each sweep, we compute the maximum norm of the error (which is $-v$). Figure 7.3(a) shows a plot of the error norm versus the iteration number.

For the moment, only qualitative behavior of the iteration is important. The error clearly decreases with each relaxation sweep and the rate of decrease is larger for the higher wavenumbers. Figures 7.3(b, c) show analogous plots for the regular and red-black Gauss-Seidel iterations. We see a similar relationship among the error, the number of iterations, and the wavenumber.

In general, most initial guesses (or equivalently, most right-hand side vectors $f$) would not consist of a single mode. Figure 7.4 shows a slightly more realistic situation in which the initial guess consists of three modes: a low-frequency wave ($k = 1$), a medium-frequency wave ($k = 6$), and a high-frequency
CHAPTER 7. NUMERICAL SOLUTION OF PARTIAL DIFFERENTIAL EQUATIONS

Figure 7.2: The modes $v_j = \sin \left( \frac{jk\pi}{J} \right)$, $0 \leq j \leq J$, with wavenumbers $k = 1, 3, 6.$

wave ($k = 32$) on a grid with $J = 64$. As before, the maximum norm of the error is plotted against the number of iterations. The error decreases rapidly within the first five iterations, after which it decreases much more slowly. As we will see shortly, the initial decrease corresponds to the quick elimination of the high-frequency mode. The slow decrease is due to the presence of the low-frequency modes. The important observation is that the standard iterations converge very quickly as long as the error has high-frequency components. However, the slower elimination of the low-frequency components degrades the performance of these methods.

To study the convergence of the schemes we will look at their spectral radii. For the weighted Jacobi method we have that

$$ P_\omega = (1 - \omega)I + \omega P_J = I - \omega D^{-1}A = I + \frac{\omega}{2}A = I + \frac{\omega}{2}B[1, -2, 1]. $$

It follows that the eigenvalues of $P_\omega$ and $A$ are related by

$$ \lambda(P_\omega) = 1 + \frac{\omega}{2}\lambda(A), $$

and the problem becomes one of finding the eigenvalues of the original matrix $A$. The result is that the eigenvalues of $A$ are

$$ \lambda_k(A) = -2 \left[ 1 - \cos \left( \frac{k\pi}{J} \right) \right] = -4 \sin^2 \left( \frac{k\pi}{2J} \right), \quad 1 \leq k \leq J - 1. $$

Also of interest are the corresponding eigenvectors of $A$. In all that follows we will use the notation that $w_{k,j}$ is the $j$th component of the $k$th eigenvector. The eigenvectors of $A$ are then given by

$$ w_{k,j} = \sin \left( \frac{jk\pi}{J} \right), \quad 1 \leq k \leq J - 1, \quad 0 \leq j \leq J. $$

We see that the eigenvectors of $A$ are the Fourier modes which we discussed earlier.

With these results, we find the eigenvalues of $P_\omega$ are

$$ \lambda_k(P_\omega) = 1 - 2\omega \sin^2 \left( \frac{k\pi}{2J} \right), \quad 1 \leq k \leq J - 1, $$

while the eigenvectors of $P_\omega$ are the same as the eigenvectors of $A$. Note that if $0 < \omega \leq 1$, then $|\lambda_k(P_\omega)| < 1$ and the weighted Jacobi iteration converges.

We need some qualitative terms for the various Fourier modes. The modes in the lower half of the spectrum, with wavenumbers in the range $1 \leq k < J/2$, will be called low-frequency or smooth modes. The modes in the upper half of the spectrum, with $J/2 \leq k \leq J - 1$, will be called high-frequency or oscillatory modes.
Figure 7.3: The (a) weighted Jacobi iteration with $\omega = 2/3$, (b) regular Gauss-Seidel iteration, and (c) red-black Gauss-Seidel iteration applied to the one-dimensional model problem with $J = 64$ and with initial guesses consisting of $v_1$, $v_3$, and $v_6$. The maximum norm of the error, $\|e\|_\infty$, is plotted against the iteration number for 100 iterations.
CHAPTER 7. NUMERICAL SOLUTION OF PARTIAL DIFFERENTIAL EQUATIONS

Figure 7.4: The weighted Jacobi method with $\omega = 2/3$ applied to the one-dimensional model problem with $J = 64$ and with initial guess $v_j = \frac{1}{3} \left[ \sin \left( \frac{j\pi}{J} \right) + \sin \left( \frac{6j\pi}{J} \right) + \sin \left( \frac{32j\pi}{J} \right) \right]$. The maximum norm of the error, $\|e\|_\infty$, is plotted against the iteration number for 100 iterations.

Figure 7.5: The eigenvalues of the iteration matrix $P_\omega$ for $\omega = 1/3, 1/2, 2/3, 1$. The eigenvalues $\lambda_k$ are plotted as if $k$ were a continuous variable on the interval $0 \leq k \leq J$. In fact, $k$ takes only integer values on the interval $1 \leq k \leq J - 1$. 
Now one may ask what is the “best” choice of $\omega$. That is, we would like to find the value of $\omega$ which makes $|\lambda_k(P_\omega)|$ as small as possible for all $1 \leq k \leq J - 1$. Figure 7.5 is a plot of the eigenvalues $\lambda_k$ for different values of $\omega$. Notice that for all values of $\omega$ satisfying $0 < \omega \leq 1$, the spectral radius is given by

$$
\rho_\omega = \lambda_1 = 1 - 2\omega \sin^2 \left( \frac{\pi}{2J} \right) = 1 - 2\omega \sin^2 \left( \frac{\pi h}{2} \right) \approx 1 - \frac{\omega \pi^2 h^2}{2},
$$

which implies that $\lambda_1$, the eigenvalue associated with the smoothest mode, will always be close to 1. Therefore, no value of $\omega$ will reduce the smooth components of the error effectively. Furthermore, the smaller the grid spacing $h$, the closer $\lambda_1$ is to 1. Any attempt to improve the accuracy of the solution (by decreasing the grid spacing) will only worsen the convergence on the smooth components of the error. Most basic relaxation schemes share this ironic limitation. The smallest number of iterations $N_\omega$ required to reduce the overall error by a factor $10^{-d}$ is approximately given by

$$
N_\omega \geq \frac{d}{2\omega h^2}.
$$

Note that this last expression is consistent with Figure 7.3(a).

Having accepted the fact that no value of $\omega$ will damp the smooth components satisfactorily, we ask what value of $\omega$ provides the best damping of the oscillatory components (those with $J/2 \leq k \leq J - 1$). We could impose this condition by requiring that

$$
\lambda_{J/2}(P_\omega) = -\lambda_J(P_\omega).
$$

Solving this equation for $\omega$ leads to an optimal value of $\omega = 2/3$.

It can also be shown that with $\omega = 2/3$, $|\lambda_k| \leq 1/3$ for all $J/2 \leq k \leq J - 1$. This says that the oscillatory components are reduced by a factor of at least $1/3$ with each relaxation. This damping rate for the oscillatory modes, which is independent of the grid spacing $h$, is an important property of any relaxation scheme and is called the smoothing factor of the scheme.

Figures 7.6(a,b) show how the Jacobi method performs in terms of different wavenumbers. Specifically, the wavenumber of the initial error is plotted against the number of iterations required to reduce the norm of the initial error by a factor of 100. This experiment is done for weighting factors of $\omega = 1, 2/3$.

With $\omega = 1$, both the high- and low-frequency components of the error are damped very slowly. Components with wavenumbers near $J/2$ are damped rapidly. This behavior is consistent with the eigenvalue curves of Figure 7.5. We see a quite different behavior in Figure 7.6(b) with $\omega = 2/3$. Recall that $\omega = 2/3$ was chosen to give preferential damping to the oscillatory components. Indeed, the smooth waves are damped very slowly, while the upper half of the spectrum ($k \geq J/2$) shows a rapid convergence. Again, this is consistent with Figure 7.5.

Another perspective of these convergence properties is provided in Figures 7.7(a-c). This time the actual approximations are plotted. Figure 7.7(a) shows the error with wavenumber $k = 3$ after one relaxation sweep in the left-hand plot and the error after ten relaxation sweeps in the right-hand plot. This long, smooth component is damped very slowly. Figure 7.7(b) shows a more oscillatory wave ($k = 16$) after one and after ten iterations. The damping is now much more dramatic. Notice also, as mentioned before, that the weighted Jacobi method preserves modes: once a $k = 3$ mode, always a $k = 3$ mode. Figure 7.7(c) illustrates the selectivity of the damping property. This experiment uses an initial guess consisting of two modes with $k = 2$ and $k = 16$. After ten relaxation sweeps, the high-frequency modulation on the long wave has been nearly eliminated. However, the original smooth component persists.

We have belabored the discussion of the weighted Jacobi method because it is easy to analyze and because it shares many properties with other basic relaxation schemes. In much less detail, let us look at the Gauss-Seidel iteration. It can be shown that the Gauss-Seidel iteration matrix for the one-dimensional model problem has eigenvalues

$$
\lambda_k(P_G) = \cos^2 \left( \frac{k\pi}{J} \right), \quad 1 \leq k \leq J - 1.
$$
Figure 7.6: The weighted Jacobi method with (a) \( \omega = 1 \) and (b) \( \omega = 2/3 \). The initial guesses consist of the modes \( w_k \) for \( 1 \leq k \leq 63 \). The graph shows the number of iterations required to reduce the norm of the initial error by a factor of 100 for each \( w_k \). Note that for \( \omega = 2/3 \), the damping is strongest for the oscillatory modes \( (32 \leq k \leq 63) \).
Figure 7.7: The weighted Jacobi method with $\omega = 2/3$ with an initial guess consisting of (a) $w_3$, (b) $w_{16}$, and (c) a combination of $w_2$ and $w_{16}$. The figures show the approximation after one iteration (on the left) and after ten iterations (on the right).
These eigenvalues are plotted in Figure 7.8. Note that \( \lambda_k(P_G) = [\lambda_k(P_J)]^2 \), so that the spectral radius for large \( J \) is given by

\[
\rho_G = \lambda_1 = \cos^2 \left( \frac{\pi}{J} \right) = \cos^2 (\pi h) \approx 1 - \pi^2 h^2,
\]
and the convergence rate can be obtained from

\[
N_G \geq \frac{d}{4h^2}.
\]

We see that when \( k \) is close to 1 or \( J \), the corresponding eigenvalues are close to 1 and the convergence is slow. However, the eigenvectors of \( P_G \) are given by

\[
w_{k,j} = [\lambda_k(P_G)]^{j/2} \sin \left( \frac{jk\pi}{J} \right) = \left[ \cos \left( \frac{k\pi}{J} \right) \right]^j \sin \left( \frac{jk\pi}{J} \right),
\]
where \( 0 \leq j \leq J \) and \( 1 \leq k \leq J - 1 \). These eigenvectors do not coincide with the eigenvectors of \( A \). Therefore, \( \lambda_k \) gives the convergence rate, not for the \( k \)th mode of \( A \), but for the \( k \)th eigenvector of \( P_G \). This distinction is illustrated in Figures 7.9(a, b).

In Figure 7.9(a), the initial guess (and error) consists of the \( k \)th eigenvector of \( P_G \). The graph looks similar to the eigenvalue graph of Figure 7.8. In Figure 7.9(b), the initial guess consists of the \( k \)th eigenvector of the original matrix \( A \). The structure of this graph would be much more difficult to anticipate analytically. We see that when convergence of the Gauss-Seidel method is described in terms of the modes of \( A \), then once again the smooth modes are damped slowly, while the oscillatory modes show a rapid decay.

We have looked in detail at the convergence properties of some basic relaxation schemes. The experiments that were presented reflect the experience of many practitioners. These schemes work very well for the first few iterations. Inevitably, however, the convergence slows and the entire scheme appears to stall. We have found a simple explanation for this phenomenon: the rapid decrease in error during the early iterations is due to the efficient elimination of the oscillatory modes of that error. Once the oscillatory modes have been removed, the iteration is much less effective in reducing the remaining smooth components.

Many relaxation schemes possess this property of eliminating the oscillatory modes and leaving the smooth modes. We call this property the smoothing property. It is a serious limitation of these methods.

### 7.15 Successive Overrelaxation (SOR) Method

Since in the iteration process the solution moves monotonically toward the exact solution, we can almost guess the result of the next iteration without computing it. In other words, the past history of the calculation contains important information that can be used to increase the convergence of a method.
Figure 7.9: The figure shows the number of iterations required to reduce the norm of the initial error by a factor of 100 for each initial guess. The initial guesses consist of (a) the eigenvectors of the iteration matrix $P_G$ with wavenumbers $1 \leq k \leq 63$ and (b) the eigenvectors of $A$ with wavenumbers $1 \leq k \leq 63$. 
Instead of using $v^{(p)}$ as the input for the next iteration, we can extrapolate the preceding results. This leads to a procedure known as extrapolation, acceleration, or overrelaxation.

To introduce the idea, suppose that we use one of the previously discussed methods to compute $\tilde{v}^{(p+1)}$:

$$\tilde{v}^{(p+1)} = P v^{(p)} + b.$$  

In the preceding methods $\tilde{v}^{(p+1)}$ would be used as the input guess for the next iteration, that is, $v^{(p+1)} = \tilde{v}^{(p+1)}$. Introducing the idea of extrapolation, instead of using $\tilde{v}^{(p+1)}$ as the input guess, we use an extrapolated value

$$v^{(p+1)} = \tilde{v}^{(p+1)} + \alpha \left( \tilde{v}^{(p+1)} - v^{(p)} \right) = (1 + \alpha) \tilde{v}^{(p+1)} - \alpha v^{(p)} = \omega \tilde{v}^{(p+1)} + (1 - \omega) v^{(p)}. \quad (7.31)$$

When $\alpha = 0$ we recover the original method. Thus, for this to represent an extrapolation we must have $\alpha > 0$. In addition, so that we don’t extrapolate too far, we should have $\alpha < 1$. Then $\omega$, which is called the overrelaxation factor, should be such that $1 < \omega < 2$.

The two steps can be combined to give

$$v^{(p+1)} = P_{OR} v^{(p)} + \omega b, \quad (7.32)$$

where

$$P_{OR} = \omega P + (1 - \omega) I.$$  

So the use of the overrelaxation method is equivalent to replacing the iteration matrix $P$ by $P_{OR}$. The simple relationship between the matrix governing the accelerated method and that of the original method makes it simple to compute the eigenvalues of the new matrix. In fact, any eigenvector of $P$ with eigenvalue $\lambda$ is also an eigenvector of $P_{OR}$ with eigenvalue

$$\lambda_{OR} = \omega \lambda + (1 - \omega).$$

Further improvement can be obtained by accelerating or extrapolating the solution at each point as it is calculated rather than waiting for the entire iteration to be completed. For the Jacobi method, which only uses old data, there is no difference, and furthermore it is not improved by acceleration. For the Gauss-Seidel method, however, a surprising improvement occurs.

The method is quite simple. At each point, a new value is computed using the Gauss-Seidel method, which is then extrapolated using the point version of equation (7.32). These operations can be combined into the single equation

$$v^{(p+1)} = P_{SOR} v^{(p)} + d,$$  

where

$$P_{SOR} = (D - \omega U)^{-1}[(1 - \omega) D + \omega L].$$

For our model problem this is expressed as

$$v^{(p+1)}_{i,j} = \frac{\omega}{4} \left( v^{(p+1)}_{i-1,j} + v^{(p)}_{i+1,j} + v^{(p+1)}_{i,j-1} + v^{(p)}_{i,j+1} - f_{i,j} \right) + (1 - \omega) v^{(p)}_{i,j}.$$

The problem is now one of finding the value of the acceleration parameter that makes the maximum eigenvalue of $P_{SOR}$ as small as possible. This question is rather difficult and useful results are known only for particular cases. The following theorems can be proved:

- The method is convergent only for $0 < \omega < 2$. If $0 < \omega < 1$, we speak of underrelaxation.
- Under certain mathematical restrictions, generally satisfied by matrices arising from finite differencing, only overrelaxation ($1 < \omega < 2$) can give faster convergence than the Gauss-Seidel method.
If $\rho_J$ is the spectral radius of the Jacobi iteration (so that the square of it is the spectral radius of the Gauss-Seidel iteration, $\rho_G$), then the optimal choice for $\omega$ is given by

$$\omega = \frac{2}{1 + \sqrt{1 - \rho_J^2}}.$$  

For this optimal choice, the spectral radius is

$$\rho_{SOR} = \left( \frac{\rho_J}{1 + \sqrt{1 - \rho_J^2}} \right)^2.$$  

For our model problem, the above results give (for large $J$)

$$\omega \approx \frac{2}{1 + \pi h},$$

$$\rho_{SOR} \approx 1 - 2\pi h,$$

and

$$N_{SOR} \geq \frac{d}{3h}.$$  

7.16 Method of Weighted Residuals

The method of weighted residuals (MWR) is an approximate method which seeks a solution that is a good approximation to the exact solution over the whole domain. This may be contrasted to a Taylor expansion which seeks an accurate solution in the small region surrounding a single point in the domain. Thus, the error between the exact and the approximate solutions may be thought of as being distributed, hopefully evenly, throughout the whole domain.

A solution of

$$L(u) = 0$$ (7.33)

is sought where $L$ is a differential operator and equation (7.33) may be an ordinary differential equation or a partial differential equation of elliptic, parabolic, or hyperbolic type. The solution $u = u(x, t)$ is sought within some domain $D$ and boundary conditions are specified on $S$, the boundary of $D$, namely

$$N(u) = f(x, t),$$ (7.34)

where $N$ is a differential operator of lower order than $L$.

The first step in the application of MWR is to introduce a (linear) trial solution

$$v(x, t) = v_0(x, t) + \sum_{j=1}^{n} a_j(t)\phi_j(x),$$ (7.35)

which, it is hoped, is close to $u$ in some sense or can be made so if $n$ is large enough. The trial solution is chosen to satisfy the boundary conditions exactly. This is often done by making $v_0$ satisfy the boundary conditions exactly; then the analytic functions $\phi_j(x)$ satisfy homogeneous boundary conditions. Since the approximate solution should be capable of converging to the exact solution as $n$ approaches infinity, it is important that the $\phi_j(x)$ are linearly independent and chosen from a set of functions which is complete in the domain of interest. That is, the complete set of functions $\phi_j(x)$, $j = 1, \ldots, \infty$ is defined such that there exists no exact solution that cannot be expanded in terms of $\phi_j(x)$. In equation (7.35), $\phi_j(x)$ are prescribed analytic functions (e.g., polynomials or trigonometric functions) whose form may be suggested by the symmetry of the problem, or a boundary condition, or the exact solution of a related problem.
Thus an initial value problem might suggest the use of exponential functions. The solution of Laplace’s equation might suggest the use of circular and hyperbolic functions.

In equation (7.35), $a_j(t)$ are coefficients to be determined. The form of equation (7.35) is typical of initial value problems in which the governing equation (7.33) is to be reduced to a system of ordinary differential equations for $a_j(t)$. For a steady state problem $\phi_j$ might depend on all the independent variables and the $a_j$’s would be constants to be determined from algebraic relations.

The method proceeds by substituting equation (7.35) into equation (7.33). In general, equation (7.33) is no longer satisfied and a residual $r$ results, i.e.

$$L(v) = r(x, t, a_j).$$

(7.36)

It is to be expected that $r$ will be small, in some sense, but non-zero throughout the domain in which a solution is sought. Since $r = 0$ throughout the domain when the exact solution is obtained, $r$ may be considered a measure of the error. Since the exact solution is not available in general, the size and distribution of $r$ in the domain can be used to assess the accuracy of the solution. Thus, if a solution for a particular value of $n$ has been obtained, $r(x, t, a_j)$ can be evaluated. The effect of obtaining a new solution with increased $n$ should cause a reduction in $r$ in some average sense.

The method proceeds by imposing the condition

$$\int_D r(x, t, a_j) W_k(x) dx = 0, \quad k = 1, \ldots, n,$$

(7.37)

where $W_k(x)$ represents one member of a family of weighting functions which depend on the same independent variables as the integrating variables and the analytic functions $\phi_j(x)$ in equation (7.35). Repeated application of equation (7.37) for the different values of $k$ produces enough equations to permit solution for the $a_j$’s. Since each equation must be independent, then each $W_k(x)$ must be an independent function. As $n$, the number of unknown coefficients in equation (7.35), is increased, the approximate solution is expected to approach the exact solution. Consequently, the residual $r$ given by equation (7.36), is expected to reduce as $n$ increases. In the limit of $n$ approaching infinity, the approximate solution will converge to the exact solution, $r$ will be zero, and equation (7.35) will be satisfied for any choice of the weighting function.

Different choices of the weighting function give rise to the different methods which collectively are known as methods of weighted residuals.

- **Subdomain Method**
  
  The domain $D$ is split up into $n$ subdomains, $D_k$, which may overlap. Then
  
  $$W_k = \begin{cases} 
  1 & \text{in } D_k, \\
  0 & \text{outside } D_k.
  \end{cases}$$

  (7.38)

- **Collocation Method**
  
  $$W_k(x) = \delta(x - x_k),$$

  (7.39)

  where $\delta$ is the Dirac delta function. This choice of $W_k(x)$ reduces equation (7.37) to evaluating $r(x, t, a_j)$ at $x = x_k$.

- **Least Squares**
  
  $$W_k(x) = \frac{\partial r}{\partial a_k}.$$ 

  (7.40)

  This is equivalent to replacing equation (7.37) with the requirement that

  $$\int_D r^2 dx$$

  is a minimum.
7.16. METHOD OF WEIGHTED RESIDUALS

- Method of Moments

$$W_k(x) = \sum_{m=1}^{k} b_m x^{m-1}. \quad (7.41)$$

Often just $x^k$ is used for the weighting function (hence the name “method of moments”).

- Galerkin Method

$$W_k(x) = \phi_k(x), \quad (7.42)$$

i.e., the weighting functions are chosen from the same family as the trial functions in equation (7.35). Since the trial functions are chosen from a linearly independent set of functions, complete in the domain of interest, equation (7.37) and equation (7.42) may be interpreted as requiring that the residual must be orthogonal to every member of the complete set up to order \( n \). Thus in the limit of \( n \) tending to infinity the residual must tend to zero. If the weighting function used in the method of moments is chosen from a complete set then the above remarks hold for the method of moments. One of the unique features of the Galerkin method is its equivalence, under certain conditions, to a variational formulation.

The Galerkin method using infinitely differentiable functions \( \phi_j(x) \) over the whole domain (usually taken from a set of orthogonal functions such as trigonometric functions or Chebyshev polynomials) is also referred as the “spectral method”, and the collocation method for the same type functions as the “pseudo-spectral method”.

**Example of the Galerkin Method**

The purpose here is to lay out the step-by-step procedure required to implement the method. Consider a problem governed by the unsteady heat equation

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}, \quad (7.43)$$

subject to the boundary conditions

$$T(x, 0) = T_a + T_0(x),$$
$$T(0, t) = T_a,$$
$$T(1, t) = T_a. \quad (7.44)$$

An analytic representation is postulated

$$T - T_a = T_0(x) + \sum_{j=1}^{n} a_j(t)\phi_j(x). \quad (7.45)$$

This representation is chosen deliberately so that \( \phi_j(x) \) satisfy homogeneous boundary conditions. Also \( \phi_j(x) \) are chosen from a complete set of functions, e.g. \( x^j \). Substitution of equation (7.45) into equation (7.43) produces some residual, \( r \), given by

$$r(x, t) = \sum_{j=1}^{n} \frac{da_j}{dt} \phi_j(x) - \frac{d^2 T_0}{dx^2} - \sum_{j=1}^{n} \phi_j \frac{d^2 \phi_j(x)}{dx^2}. \quad (7.46)$$

Imposing the Galerkin condition (i.e. equations (7.37) and (7.42)) gives

$$\sum_{j=1}^{n} \left[ \int_0^1 \phi_j(x)\phi_k(x) dx \right] \frac{da_j}{dt} - \int_0^1 \frac{d^2 T_0}{dx^2} \phi_k(x) dx - \sum_{j=1}^{n} \left[ \int_0^1 \frac{d^2 \phi_j}{dx^2} \phi_k(x) dx \right] a_j = 0, \quad (7.47)$$

$$k = 1, \ldots, n.$$
Equation (7.47) can be rewritten as

$$P \frac{da}{dt} = Qa + r,$$

(7.48)

where \(a\) is a vector with elements \(a_j\), \(P\) is a matrix with elements \(p_{kj}\) given by

$$p_{kj} = \int_0^1 \phi_j(x)\phi_k(x)dx,$$

(7.49)

the matrix \(Q\) has elements \(q_{kj}\) given by

$$q_{kj} = \int_0^1 \frac{d^2\phi_j(x)}{dx^2}\phi_k(x)dx,$$

(7.50)

and the vector \(r\) with elements \(r_k\) given by

$$r_k = \int_0^1 \frac{d^2T_0(x)}{dx^2}\phi_k(x)dx.$$

(7.51)

Clearly \(P\), \(Q\), and \(r\) can be evaluated once and for all and equation (7.48) can be manipulated to give

$$\frac{da}{dt} = P^{-1}Qa + P^{-1}r.$$

(7.52)

Equation (7.52) represents a system of ordinary differential equations for the \(a_j\)’s which can be solved by any of the standard means.

At this point it is worth re-emphasizing that the following conditions are required in applying the traditional Galerkin method:

1. The weighting functions, \(W_k\), used in equation (7.37) are chosen from the same family as the analytic functions, \(\phi_j\), used in equation (7.35).

2. The analytic functions (and hence the weighting functions) must be linearly independent.

3. The analytic functions should be chosen from a complete set of functions.

4. The analytic functions should satisfy the boundary conditions exactly.

As can be seen from the example above, the analytic representation is often manipulated so that condition 4 can be satisfied. Of the above conditions, 1 defines the Galerkin method and 2 follows from the need to generate independent equations to determine the unknown coefficients \(a_j\). Condition 3 is imposed as a necessary condition for ensuring convergence to the exact solution. Violation of conditions (2) and (3) would not necessarily be disastrous rather it would lead to solutions that were less accurate, for a given number of unknown coefficients \(a_j\), than solutions which satisfied conditions (2) and (3).

### 7.17 Finite Element Method

As with the traditional Galerkin method, the starting point is a differential equation, e.g.,

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f,$$

(7.53)

for which a solution is sought within some domain \(D\) subject to appropriate boundary conditions on \(S\), the boundary of \(D\). An approximate solution, defined by

$$v = \sum_{j=1}^n N_j(x, y)v_j,$$

(7.54)
is introduced. In equation (7.54), \( v_j \) are the nodal values of \( v \) and they become the unknowns of the problem, that is they play the same role as the unknown coefficients \( a_j \) in equation (7.35). \( N_j(x, y) \), which play the role of \( \phi_j(x) \) in equation (7.35), are referred to as the shape functions and under certain circumstances they are the same as the Lagrange interpolation coefficients. The shape functions have the property

\[
N_i = \begin{cases} 
1 & \text{at the } i\text{th node,} \\
0 & \text{at all other nodes.} 
\end{cases} \tag{7.55}
\]

In figure 7.10 \( N_1 \) equals one at node 1, equals zero at nodes 2 to 9, takes values between zero and one in elements \( W, X, Y, \) and \( Z \), and is zero outside \( W, X, Y, \) and \( Z \).

If element coordinates are defined as in figure 7.11, then a linear shape function \( N_1 \) would have the following form in element \( Y \)

\[
N_1(\xi, \eta) = \frac{1}{4}(1 - \xi)(1 - \eta). \tag{7.56}
\]

Comparable formulae are available for the variation of \( N_1 \) in elements \( W, X, \) and \( Z \). It is apparent from equation (7.54) that \( N_1 \) demonstrates the correct behavior at nodes 1, 2, 3, and 4. The nature of equation (7.54) makes it inappropriate to choose trial functions \( N_j(x, y) \) that satisfy the boundary conditions directly.

The finite element method proceeds by substituting equation (7.54) into equation (7.53), which results in a residual given by

\[
r = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} - f. \tag{7.57}
\]
Application of the weighted residual integral gives
\[ \int_D rN_i \, dx \, dy = 0, \quad i = 1, \ldots, n, \quad (7.58) \]
or
\[ \int_D \left[ N_i \left( \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) - N_i f \right] \, dx \, dy = 0, \quad i = 1, \ldots, n. \quad (7.59) \]

Because of inter-element continuity requirements, it is desirable to reduce the maximum order of the derivatives appearing in equation (7.59). This can be done by applying Green’s theorem to the Laplacian in equation (7.59). The result is
\[ \int_S N_i \left( \frac{\partial v}{\partial x} l_x + \frac{\partial v}{\partial y} l_y \right) ds - \int_D \left( \frac{\partial N_i}{\partial x} \frac{\partial v}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial v}{\partial y} \right) \, dx \, dy - \int_D N_i f \, dx \, dy = 0, \quad i = 1, \ldots, n. \quad (7.60) \]

The first integral only makes a contribution on the boundary \( S \) and only the \( i \)th node is on the boundary. Substituting for \( v \) from equation (7.54) into equation (7.60) produces the result
\[ \sum_{j=1}^n \left[ \int_S N_i \left( \frac{\partial N_j}{\partial x} l_x + \frac{\partial N_j}{\partial y} l_y \right) \, ds - \int_D \left( \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right) \, dx \, dy \right] v_j = \int_D N_i f \, dx \, dy, \quad i = 1, \ldots, n. \quad (7.61) \]

Equation (7.61) is just an algebraic equation which can be written as
\[ \sum_{j=1}^n c_{ij} v_j = b_i, \quad i = 1, \ldots, n. \quad (7.62) \]

The whole system of equations can be written as
\[ C \mathbf{v} = \mathbf{b}, \quad (7.63) \]
where an element of the matrix \( C \) is \( c_{ij} \), defined by
\[ c_{ij} = \int_S N_i \left( \frac{\partial N_j}{\partial x} l_x + \frac{\partial N_j}{\partial y} l_y \right) ds - \int_D \left( \frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right) \, dx \, dy, \quad (7.64) \]
and an element of the vector \( \mathbf{b} \) is \( b_i \), defined by
\[ b_i = \int_D N_i f \, dx \, dy. \quad (7.65) \]

The form of equation (7.63) is precisely the same as would be obtained from application of the Galerkin method. However, the evaluation of the coefficients \( c_{ij} \) demonstrates a significant difference. Because of the local nature of \( N_i \), the only contributions to \( c_{ij} \) come from the elements surrounding the \( i \)th node. This is because \( N_i \) is zero outside those elements.

If the linear shape functions are used and only one dependent variable appears in the governing equation, then the \( i \)th row in the \( C \) matrix can only have a maximum of nine non-zero entries, irrespective of how many nodes are in the total domain. With a sensible numbering system the non-zero elements are clustered around the main diagonal. Since \( C \) is sparse, equation (7.63) can be solved economically if the sparse character of \( C \) is taken advantage of. This advantage can be traced back directly to the local nature of the shape function \( N_i \) in equation (7.54).

Another advantage of the finite element method comes from reducing the integration over the whole domain to the sum of the integrations over individual elements. Although the elements shown here have
been rectangular, this is not generally necessary. Using an isoparametric formulation the elements can be of any reasonable shape. Consequently by distorting the elements within the domain it is possible to handle arbitrary boundary contours and the corresponding boundary conditions without the need for special procedures. This situation may be contrasted with the need for boundaries to follow coordinate lines in the Galerkin method. The integration over the distorted elements is usually done numerically after mapping the distorted element in the \((\xi, \eta)\) plane.

Thus, the finite element method formulation has two advantages over the Galerkin formulation:

- it leads to a sparse solution matrix \(C\) that can be accurately and economically factorized, and

- by reducing the integration over the whole domain to the numerical integration over discrete, distortable elements, arbitrary boundaries are easily handled.