Chapter 5

Theory

The numerical simulation of PDE's requires careful considerations of properties of the approximate solution. A necessary condition for the scheme used to model a physical problem is the consistency of the employed scheme with the actual set of differential equations. Consistency requires that the error for algebraic equations is small such that to significant order the algebraic equations are a fair representation of the actual PDE's. In other words the the actual set of PDE's can be recovered from the set of algebraic equations.

A second important property of the algebraic equations is stability. It has already been indicated that a numerical scheme is not necessarily stable. Numerical stability is absolutely necessary for the numerical method.

Third the approximate solution is required to actually converge to the exact solution in the limit of $\Delta x \rightarrow 0$ and $\Delta t \rightarrow 0$. However, this convergence is not obvious. In the following it will be demonstrated that for some cases consistency and stability are necessary for the convergence of the approximate solution to the actual solution.

The relation between consistency, stability, and convergence is illustrated further in Figure (5.1).

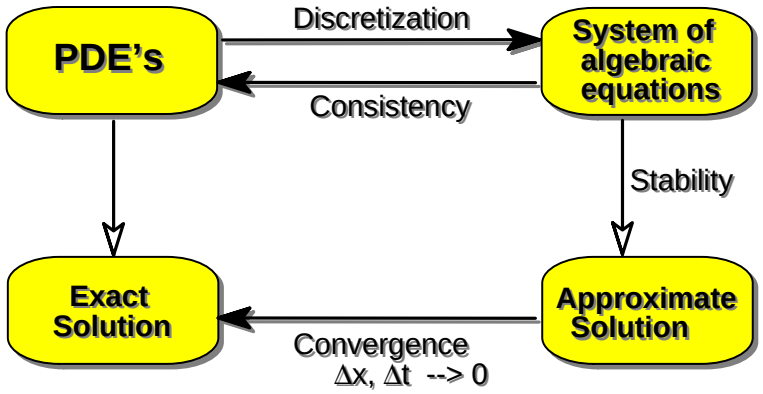

Figure 5.1: Relation between consistency, stability, and convergence.

Two other important considerations for the numerical solution method are accuracy and efficiency. If at all possible means to achieve a particular accuracy are highly desirable and important. For

the choice of the actual approximation method efficiency is also important. The following section address these basic concepts of numerical modeling and simulation.

5.1 Convergence

A solution of a set of algebraic equations is convergent if the approximate solution approaches the exact solution of the set of PDE's for each value of the independent variable as the grid spacing and time step goes to 0.

$$\lim_{\Delta x, \Delta t \to 0} T_i^n = T_{exact}(x_i, t_n) \tag{5.1}$$

This implies for the error at x_i and t_n

$$e_i^n = T_{exact}(x_i, t_n) - T_i^n \to 0 \quad for \quad \Delta x, \Delta t \to 0 \tag{5.2}$$

The proof for this is in general difficult, however, for a special case the following theorem applies.

Lax Equivalence Theorem: "Given a properly posed linear initial value problem and a finite difference approximation to it that satisfies the consistency condition, stability is the necessary and sufficient condition for convergence"

Notes:

- the theorem is applicable to any discretization
- real fluid dynamics is usually nonlinear
- typical problem are usually boundary value or mixed initial and boundary value problems

In conclusion: For most problems the Lax Equivalence Theorem is only a necessary but not necessarily sufficient condition.

5.1.1 Numerical convergence

In cases where an analytic solution is known it can be used to verify convergence for a numerical solution. In the case of the diffusion equation the error for different grid resolution is given in Table 5.1 for a diffusion coefficient $\alpha = 10^{-5}$.

Table 5.1: Solution error for sim1.f for different values of s and Δx.

$s = \alpha \Delta t / \Delta x^2$	$\Delta x = 0.2$	0.1	0.05	0.025
0.5	1.66	0.35	0.086	0.021
0.3	0.6	0.19	0.048	0.012

The result demonstrates that the error decreases with Δx^2 indicating that the numerical solution indeed converges to the exact solution. Figure (5.2) provides a graphic representation of the error as a function of Δx. Note that the convergence of sim1.f for $s = 1/6$ is a special case and of order Δx^4.

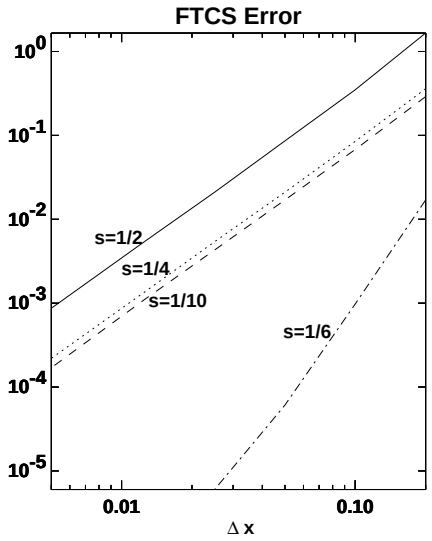

Figure 5.2: Error for the diffusion equation approximated by the FTCS scheme as a function of Δx.

In cases where where an analytic solution is not know one can test convergence using the difference of two approximate solutions, i.e.,

$$e_{\Delta x} = T(\Delta x, \Delta t)_i^n - T(\Delta x/2, \Delta t/2)_{2i}^{2n} \tag{5.3}$$

for the limit $\Delta x, \Delta t \hookrightarrow 0$.

5.2 Consistency

Consistency requires that the original equations can be recovered from the algebraic equations. Obviously this is a minimum requirement for any discretization. in the following we will illustrate how this can be done in terms of a Taylor expansion of the discretized equations.

FTCS scheme

The FTCS scheme used for sim1.f uses the approximation

$$T_i^{n+1} = T_i^n + s\left(T_{i-1}^n + T_{i+1}^n - 2T_i^n\right)$$

A Taylor expansion of the individual terms in this equation yields

$$
\begin{aligned}
T_i^{n+1} &= T_i^n + \Delta t \left[\frac{\partial T}{\partial t}\right]_i^n + \frac{\Delta t^2}{2}\left[\frac{\partial^2 T}{\partial t^2}\right]_i^n + \frac{\Delta t^3}{6}\left[\frac{\partial^3 T}{\partial t^3}\right]_i^n + O\left(\Delta t^4\right) \\
T_{i+1}^n &= T_i^n + \Delta x \left[\frac{\partial T}{\partial x}\right]_i^n + \frac{\Delta x^2}{2}\left[\frac{\partial^2 T}{\partial x^2}\right]_i^n + \frac{\Delta x^3}{6}\left[\frac{\partial^3 T}{\partial x^3}\right]_i^n + \frac{\Delta x^4}{24}\left[\frac{\partial^4 T}{\partial x^4}\right]_i^n \\
&\quad + \frac{\Delta x^5}{120}\left[\frac{\partial^5 T}{\partial x^5}\right]_i^n + \frac{\Delta x^6}{720}\left[\frac{\partial^6 T}{\partial x^6}\right]_i^n + O\left(\Delta x^7\right) \\
T_{i-1}^n &= T_i^n - \Delta x \left[\frac{\partial T}{\partial x}\right]_i^n + \frac{\Delta x^2}{2}\left[\frac{\partial^2 T}{\partial x^2}\right]_i^n - \frac{\Delta x^3}{6}\left[\frac{\partial^3 T}{\partial x^3}\right]_i^n + \frac{\Delta x^4}{24}\left[\frac{\partial^4 T}{\partial x^4}\right]_i^n \\
&\quad - \frac{\Delta x^5}{120}\left[\frac{\partial^5 T}{\partial x^5}\right]_i^n + \frac{\Delta x^6}{720}\left[\frac{\partial^6 T}{\partial x^6}\right]_i^n + O\left(\Delta x^7\right)
\end{aligned}
$$

which yields

$$
T_t + \frac{\Delta t}{2}T_{tt} + \frac{\Delta t^2}{6}T_{ttt} = \frac{s\Delta x^2}{\Delta t}\left(T_{xx} + \frac{\Delta x^2}{12}T_{xxxx} + O\left(\Delta x^4\right)\right)
$$

with $s = \alpha\Delta t/\Delta x^2$ and $T_{tt} = \alpha^2 T_{xxxx}$ and some rearranging one obtains

$$
T_t + \alpha T_{xx} + \frac{\alpha\Delta x^2}{2}\left(s - \frac{1}{6}\right)T_{xxxx} + O\left(\Delta x^4, \Delta t^2\right) = 0 \tag{5.4}
$$

Thus the error for this scheme is

$$
E_i^n = \frac{\alpha\Delta x^2}{2}\left(s - \frac{1}{6}\right)\left[\frac{\partial^4 T}{\partial x^4}\right]_i^n + O\left(\Delta x^4, \Delta t^2\right) \tag{5.5}
$$

This result also explains the prior note on 4th order accuracy of the FTCS scheme for $s = 1/6$. In this case the second order error terms are zero such that the lowest order error becomes 4th order. It is also worth noting that a refinement of Δx implies an actually smaller time step because the integration parameter is s such that the temporal integration step for some fixed value of s is $\Delta t = s\Delta x^2/\alpha$. In other words with a fixed value of s a smaller grid spacing implies a smaller time step.

Fully implicit scheme.

A fully implicit scheme can be expressed as

$$
-sT_{i-1}^{n+1} + (1+2s)T_i^{n+1} - sT_{i+1}^{n+1} = T_i^n \tag{5.6}
$$

Usually a faster way to recover the original equations is achieved by casting the algebraic equations into a form that is more suitable to apply the Taylor expansion. Here this is

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} = \alpha \frac{T_{i-1}^{n+1} - 2T_i^{n+1} + T_{i+1}^{n+1}}{\Delta x^2} \tag{5.7}$$

Let us first substitute the expansion in Δx.

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} - \alpha \left\{ T_{xx}|_i^{n+1} + \frac{\Delta x^2}{12} T_{xxxx}|_i^{n+1} + \frac{\Delta x^4}{360} T_{xxxxxx}|_i^{n+1} ... \right\} = 0$$

In the next step we apply the expansion in time.

$$\begin{aligned} T_t + \frac{\Delta t}{2} T_{tt} + \frac{\Delta t^2}{6} T_{ttt} + ... - \alpha \Big\{ & T_{xx} + \Delta t T_{xxt} + \frac{\Delta t^2}{2} T_{xxtt} + ... \\ & + \frac{\Delta x^2}{12} \left[T_{xxxx} + \Delta t T_{xxxxt} + ... \right] + \frac{\Delta x^4}{360} T_{xxxxxx} + ... \Big\} \quad = \quad 0 \end{aligned}$$

Substituting $\Delta t = s\Delta x^2/\alpha$ and $T_{tt} = \alpha^2 T_{xxxx}$ yields

$$\begin{aligned} T_t + s\frac{\Delta x^2}{2} \alpha T_{xxxx} + s^2 \frac{\Delta x^4}{6} \alpha T_{xxxxxx} + ... - \alpha \Big\{ & T_{xx} + s\Delta x^2 T_{xxxx} + s^2 \frac{\Delta x^4}{2} T_{xxxxxx} + ... \\ & + \frac{\Delta x^2}{12} \left[T_{xxxx} + s\Delta x^2 T_{xxxxxx} + ... \right] + \frac{\Delta x^4}{360} T_{xxxxxx} \Big\} \quad = \quad 0 \end{aligned}$$

Re-ordering terms with all terms retained up the 4th order in Δx gives

$$T_t - \alpha T_{xx} - \frac{\alpha \Delta x^2}{2} \left(s + \frac{1}{6} \right) T_{xxxx} - \frac{\alpha \Delta x^4}{3} \left(s^2 + \frac{s}{4} + \frac{1}{120} \right) T_{xxxxxx} = 0 \tag{5.8}$$

which shows an error of

$$E_i^n = -\frac{\alpha \Delta x^2}{2} \left(s + \frac{1}{6} \right) T_{xxxx}|_i^n - \frac{\alpha \Delta x^4}{3} \left(s^2 + \frac{s}{4} + \frac{1}{120} \right) T_{xxxxxx}|_i^n + O\left(\Delta x^6\right). \tag{5.9}$$

Similar to the FTCS scheme one recovers a second order accuracy. However in order to achieve a fourth order scheme one would require $s = -1/6$ corresponding to a negative time step in the integration such that it is not possible to construct a fourth order scheme in this case.

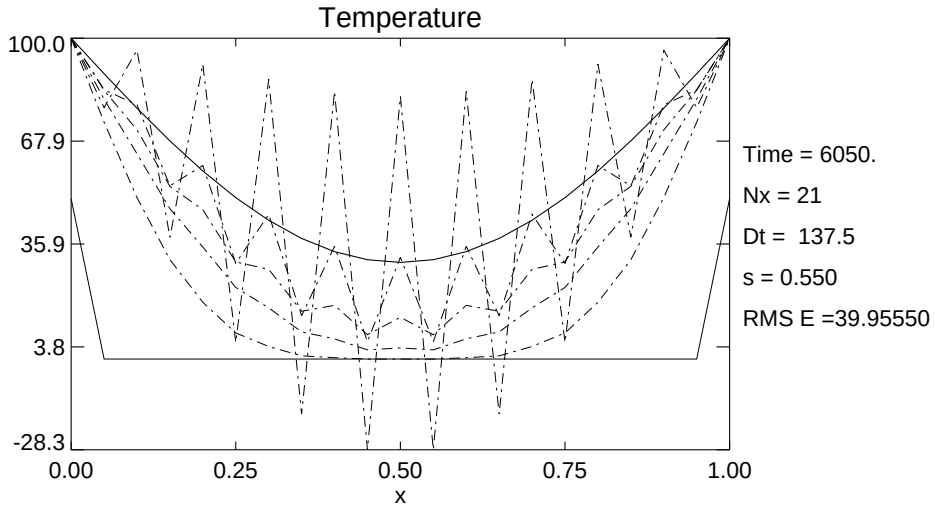

Figure 5.3: Grid oscillations increasing with time are typical for numerical instabilities.

5.3 Stability

For any numerical model small perturbations (such as truncation errors) should decay in time. However, for instance the FTCS scheme for the diffusion equation (sim1.f) will generate increasing errors for $s = 0.55$ as illustrated in Figure 5.3. The cause for the instability is an amplification of errors which occur and accumulate during a simulation.

Round-off errors occur at any stage in a computation such that the computational solution is not actually T_i^n but $\widetilde{T}_i^n$, i.e., a value which contains some newly generated round-off errors and the entire accumulation of prior errors. The error between the computed solution and the expected solution to the algebraic equation can be expressed as

$$\xi_i^n = T_i^n - \widetilde{T}_i^n \tag{5.10}$$

Similar to T_i^n, the solution $\widetilde{T}_i^n$ which includes the errors also satisfies the algebraic equations. For instance for the FTCS scheme applied to the diffusion equation

$$\widetilde{T}_i^{n+1} = (1-2s)\widetilde{T}_i^n + s\left(\widetilde{T}_{i-1}^n + \widetilde{T}_{i+1}^n\right)$$

If the algebraic equations are homogeneous and linear the error also satisfies the same equation which can be demonstrated by subtracting the equation for $\widetilde{T}_i^n$ from the one for T_i^n.

$$\xi_i^{n+1} = \xi_i^n + s\left(\xi_{i-1}^n + \xi_{i+1}^n - 2\xi_i^n\right) \tag{5.11}$$

For given initial conditions all initial errors ξ_i^0 are zero for all grid points i. Similarly Dirichlet boundary conditions imply that the error at the boundary points ξ_1^n and $\xi_{N_x}^n$ are zero for all times t_n.

The two most common methods to analyze stability are the matrix method and the von Neumann method.

5.3.1 Matrix method

FTCS scheme

Consider the FTCS scheme for the diffusion equation with Dirichlet boundary conditions, i.e., $\xi_1^n = 0$ and $\xi_{N_x}^n = 0$.

The equations for the solution are

$$\begin{aligned}
\xi_2^{n+1} &= (1-2s)\,\xi_2^n + s\xi_3^n \\
\xi_3^{n+1} &= s\xi_1^n + (1-2s)\,\xi_2^n + s\xi_3^n \\
\xi_i^{n+1} &= s\xi_{i-1}^n + (1-2s)\,\xi_i^n + s\xi_{i+1}^n \\
\xi_{N_x}^{n+1} &= s\xi_{N_x-1}^n + (1-2s)\,\xi_{N_x}^n
\end{aligned}$$

Thus the solution equations can be written as

$$\underline{\xi}^{n+1} = \underline{\underline{\mathbf{A}}} \cdot \underline{\xi}^n \tag{5.12}$$

with

$$\underline{\xi} = \begin{pmatrix} \xi_2^n \\ . \\ \xi_i^n \\ . \\ . \\ \xi_{N_x-1}^n \end{pmatrix}, \qquad \underline{\underline{\mathbf{A}}} = \begin{pmatrix} (1-2s) & s & 0 & & & \\ s & (1-2s) & s & & & \\ 0 & s & (1-2s) & s & & \\ & & . & . & . & \\ & & & . & . & . \\ & & & . & s & (1-2s) \end{pmatrix} \tag{5.13}$$

It can be shown that all errors are bounded if all Eigenvalues are distinct and the absolute values are smaller than 1. If the Eigenvectors are given by

$$\underline{\xi}_k = \underline{\underline{\mathbf{T}}}^{-1} \cdot \underline{\eta}_k \tag{5.14}$$

where $\underline{\eta}_k$ is the unit vector in the kth direction the equation can be re-written as

$$\underline{\eta}^{n+1} = \left(\underline{\underline{\mathbf{T}}}\,\underline{\underline{\mathbf{A}}}\,\underline{\underline{\mathbf{T}}}^{-1}\right)\underline{\eta}^n \tag{5.15}$$

where the matrix $\underline{\underline{\widetilde{\mathbf{A}}}} = \underline{\underline{\mathbf{T}}}\,\underline{\underline{\mathbf{A}}}\,\underline{\underline{\mathbf{T}}}^{-1}$ is a diagonal matrix with the Eigenvalues on the diagonal. Thus for all Eigenvectors $\left|\underline{\eta}^{n+1}\right| < \left|\underline{\eta}^n\right|$ if all Eigenvalue are ≤ 1.

The Eigenvalues of a tridiagonal $(N_x - 2) \times (N_x - 2)$ matrix

$$\begin{pmatrix} b & c & 0 & 0 & & & \\ a & b & c & 0 & & & \\ 0 & a & b & c & . & & \\ & & & . & . & & \\ & & & . & . & & \\ & & & & a & b & c \\ & & & & 0 & a & b \end{pmatrix}$$

are given by

$$\lambda_k = b + 2\left(ac\right)^{1/2} \cos \frac{k\pi}{N_x - 1} \tag{5.16}$$

for $k = 2, ..., N_x - 2$. Thus the Eigenvalues of the FTCS algorithm are

$$\lambda_k = 1 - 2s + 2s \cos \frac{k\pi}{N_x - 1} = 1 - 4s \sin^2 \frac{k\pi}{2\left(N_x - 1\right)} \tag{5.17}$$

Thus the condition for stability is

$$-1 \leq 1 - 4s \sin^2 \frac{k\pi}{2\left(N_x - 1\right)} \leq 1 \tag{5.18}$$

or $s \leq 1/2$. Therefore convergence of the approximate solution requires $s \leq 1/2$ and in this case all perturbations decrease in amplitude over time.

General two level scheme

For the diffusion equation this scheme is given by

$$\frac{T_i^{n+1} - T_i^n}{\Delta t} - \alpha\beta L_{xx} T_i^{n+1} - \alpha\left(1 - \beta\right) L_{xx} T_i^n = 0 \tag{5.19}$$

where the operator L_{xx} is defined by

$$L_{xx} T_i^n = \frac{1}{\Delta x^2}\left(T_{i-1} - 2T_i + T_{i+1}\right) \tag{5.20}$$

Substituting this operator and using the definition $s = \alpha \Delta t / \Delta x^2$ leads to the error equation

$$-s\beta\xi_{i-1}^{n+1} + \left(1 + 2s\beta\right)\xi_i^{n+1} - s\beta\xi_{i+1}^{n+1} = s\left(1 - \beta\right)\xi_{i-1}^n + \left[1 - 2s\left(1 - \beta\right)\right]\xi_i^n + s\left(1 - \beta\right)\xi_{i+1}^n \tag{5.21}$$

such that the set of equations (for Dirichlet conditions) can be written as

$$\underline{\underline{\mathbf{A}}} \cdot \underline{\xi}^{n+1} = \underline{\underline{\mathbf{B}}} \cdot \underline{\xi}^{n} \tag{5.22}$$

with

$$\underline{\underline{\mathbf{A}}} = \begin{pmatrix} (1+2s\beta) & -s\beta & 0 & & & & \\ -s\beta & (1+2s\beta) & -s\beta & & & & \\ 0 & -s\beta & (1+2s\beta) & & & & \\ & & . & . & . & & \\ & & & . & . & . & \\ & & & . & -s\beta & (1+2s\beta) & \end{pmatrix} \tag{5.23}$$

and

$$\underline{\underline{B}} = \begin{pmatrix} 1-2s(1-\beta) & s(1-\beta) & 0 & & & & \\ s(1-\beta) & 1-2s(1-\beta) & s(1-\beta) & & & & \\ 0 & s(1-\beta) & 1-2s(1-\beta) & s & & & \\ & & . & . & . & & \\ & & & . & . & . & \\ & & & . & s(1-\beta) & 1-2s(1-\beta) & \end{pmatrix} \tag{5.24}$$

Clearly this algorithm is stable if the absolute value of all eigenvalues of $\underline{\underline{\mathbf{A}}}^{-1}\underline{\underline{\mathbf{B}}}$ are smaller than unity. In this case the magnitude of the eigenvalues is

$$\left| \frac{(\lambda_B)_i}{(\lambda_A)_i} \right| \leq 1 \tag{5.25}$$

For the above matrices the Eigenvalues are

$$\lambda_A = 1 + 2s - 2s\beta \cos\frac{k\pi}{N_x - 1} = 1 + 4s\beta \sin^2 \frac{k\pi}{2(N_x - 1)} \tag{5.26}$$

$$\lambda_B = 1 + 2s(1-\beta) - 2s(1-\beta)\cos\frac{k\pi}{N_x - 1} = 1 - 4s(1-\beta)\sin^2 \frac{k\pi}{2(N_x - 1)} \tag{5.27}$$

and for the stability requirement

$$|\lambda_k| = \left| \frac{1 - 4s(1-\beta)\sin^2 \frac{k\pi}{2(N_x-1)}}{1 + 4s\beta \sin^2 \frac{k\pi}{2(N_x-1)}} \right| \leq 1 \tag{5.28}$$

a) for $\sin^2 \frac{k\pi}{2(N_x-1)} = 0$ with $|\lambda_k| = 1$

b) for $\sin^2 \frac{k\pi}{2(N_x-1)} = 1$ one obtains two conditions:

i. $1 - 4s(1-\beta) \leq 1 + 4s\beta$ which is always satisfied.

ii. $1 - 4s(1-\beta) \geq -1 - 4s\beta$ which implies $2 \geq 4s(1-\beta)$ or $s \leq 0.5/(1-2\beta)$ for $\beta < 1/2$.

Derivative boundary conditions

Assuming Neumann boundary conditions of the form

$$\frac{\partial T}{\partial t} = \delta, \quad at \quad x = 0$$

which can be implemented by

$$\frac{T_2 - T_0}{2\Delta x} = \delta$$

or $T_0 = T_2 - 2\delta\Delta x$.

Substitution into the equation for T_2

$$(1 + 2s\beta)\,\xi_1^{n+1} - 2s\beta\xi_2^{n+1} = [1 - 2s\,(1 - \beta)]\,\xi_1^n + 2s\,(1 - \beta)\,\xi_2^n - 2s\delta\Delta x \tag{5.29}$$

Thus compared to the case of Dirichlet conditions the first equation (or terms in the matrices) are altered and an inhomogeneity is introduced into the equations, i.e.,

$$\underline{\underline{\mathbf{A}}} \cdot \underline{\xi}^{n+1} = \underline{\underline{\mathbf{B}}} \cdot \underline{\xi}^n + \underline{\underline{\mathbf{C}}} \tag{5.30}$$

or

$$\underline{\xi}^{n+1} = \underline{\underline{\mathbf{A}}}^{-1}\underline{\underline{\mathbf{B}}} \cdot \underline{\xi}^n + \underline{\underline{\mathbf{A}}}^{-1}\underline{\underline{\mathbf{C}}}$$

Solutions to the eigenvectors can be found through iteration of

$$\underline{X}^{n+1} = \underline{\underline{\mathbf{D}}} \cdot \underline{X}^n \tag{5.31}$$

with $\underline{\underline{\mathbf{D}}} = \underline{\underline{\mathbf{A}}}^{-1}\underline{\underline{\mathbf{B}}}$. The iteration is repeated until $\underline{X}^{n+1}$ coincides with $\underline{X}^n$. The modification of the Eigenvalues by Neumann boundary conditions is usually relatively small.

5.3.2 Von Neumann method

The von Neumann method is usually easy to apply, straightforward, and dependable. However, similar to the matrix method the von Neumann method is applicable strictly only to linear equations with constant coefficients (which is already assumed in the definition for ξ). Nonlinear equations can be linearized to attempt to apply the method. However, in cases of larger sets of equations also the von Neumann methods can be challenging.

The basic idea is the following: Assume that the error can be expanded in a Fourier series

$$\xi_j^0 = \sum_{l=1}^{N_x} a_l \exp\left(i\Theta_l j\right) \tag{5.32}$$

with $\Theta_l = l\pi\Delta x$. Considering an initial error from one particular mode with amplitude 1 one can express

$$\xi_j^n = g^n \exp\left(i\Theta_l j\right) \tag{5.33}$$

where g is an amplification factor for this mode. **Stability requires** $|g| \leq 1$. Equivalent and maybe more intuitive we can assume that an error is given by a

$$\xi_j^n = \xi_0 \exp\left[i\left(kx_j + qt_n\right)\right], \tag{5.34}$$

i.e., a Fourier mode with amplitude ξ_0, wave number k, and frequency q.

$$\xi_j^{n+1} = \xi_0 \exp\left[i\left(kx_j + qt_n\right) + iq\Delta t\right]$$

(Again a general error is expanded by an appropriate superposition of all modes as in (5.32)). The amplification (or damping) of this mode is

$$\frac{\xi_j^{n+1}}{\xi_j^n} = \exp\left[iq\Delta t\right] = g \tag{5.35}$$

Similarly we can express

$$\xi_{j+1}^n = \xi_0 \exp\left[i\left(kx_j + qt_n\right) + ik\Delta x\right]$$

and

$$\frac{\xi_{j+1}^n}{\xi_j^n} = \exp\left[ik\Delta x\right] \tag{5.36}$$

The task now becomes to derive g for all modes from the discretized equations and to insure that $|g| \leq 1$ for all modes (all values of k).

FTCS scheme

The basic equation for the FTCS scheme is

$$\xi_i^{n+1} = \left(1 - 2s\right)\xi_i^n + s\left(\xi_{i-1}^n + \xi_{i+1}^n\right)$$

By substituting the single Fourier mode and division by ξ_j^n we obtain

$$\exp\left[iq\Delta t\right] = \left(1 - 2s\right) + s\left[\exp\left(ik\Delta x\right) + \exp\left(-ik\Delta x\right)\right]$$

Substituting the amplification factor g and applying the definition for the cosine function yields

$$g = 1 - 2s + 2s\cos(k\Delta x) = 1 - 4s\sin^2(k\Delta x/2) \tag{5.37}$$

where $1 - \cos(x) = 2\sin^2(x/2)$ has been used. Thus the stability condition reads

$$-1 \leq 1 - 4s\sin^2(k\Delta x/2) \leq 1 \tag{5.38}$$

Note that **stability requires that the above condition is satisfied for all possible wave numbers** k**.** The condition on the right is always satisfied and the left inequality requires

$$s \leq \frac{1}{2} \tag{5.39}$$

General two-level scheme

The general two level scheme can be expressed by

$$\frac{\xi_i^{n+1} - \xi_i^n}{\Delta t} - \alpha\beta L_{xx}\xi_i^{n+1} - \alpha(1-\beta)L_{xx}\xi_i^n = 0$$

with the second derivative operator $L_{xx}\xi_i = (\xi_{i-1} + \xi_{i+1} - 2\xi_i)/\Delta x^2$. Substitution of $s = \alpha\Delta t/\Delta x^2$ leads to the equation

$$\xi_i^{n+1} - \xi_i^n - s\beta\Delta x^2 L_{xx}\xi_i^{n+1} - s(1-\beta)\Delta x^2 L_{xx}\xi_i^n = 0 \tag{5.40}$$

With

$$\begin{aligned}\frac{\Delta x^2}{\xi_i^n}L_{xx}\xi_i^n &= \exp(ik\Delta x) + \exp(-ik\Delta x) - 2 \\ &= 2\cos(k\Delta x) - 2 = -4\sin^2(k\Delta x/2)\end{aligned}$$

we obtain

$$g - 1 + 4gs\beta\sin^2(k\Delta x/2) + 4s(1-\beta)\sin^2(k\Delta x/2) = 0 \tag{5.41}$$

or

$$g = \frac{1 - 4s(1-\beta)\sin^2(k\Delta x/2)}{1 + 4s\beta\sin^2(k\Delta x/2)} \tag{5.42}$$

The condition for stability is $|g| \leq 1$ such that

$$-1 \leq \frac{1 - 4s\,(1-\beta)\sin^2\left(k\Delta x/2\right)}{1 + 4s\beta\sin^2\left(k\Delta x/2\right)} \leq 1 \tag{5.43}$$

a) The inequality on the right yields

$$1 - 4s\sin^2\left(k\Delta x/2\right) \leq 1$$

which is always satisfied.

b) The condition on the left yields

$$-1 - 4s\beta\sin^2\left(k\Delta x/2\right) \leq 1 - 4s\,(1-\beta)\sin^2\left(k\Delta x/2\right)$$

or

$$4s(1-2\beta)\sin^2\left(k\Delta x/2\right) \leq 2$$

i. For $\beta \geq 1/2$ this is always satisfied.

ii. For $\beta < 1/2$

$$s \leq \frac{1}{2(1-2\beta)\sin^2\left(k\Delta x/2\right)}$$

This condition must be satisfied for all values of k or equivalently all values of $\sin^2\left(k\Delta x/2\right)$. The right side minimizes for $\sin^2\left(k\Delta x/2\right) = 1$ implying

$$s \leq \frac{1}{2(1-2\beta)} \tag{5.44}$$

In conclusion the general two level scheme is unconditionally stable (i.e., there is no stability limit for s) for $\beta \geq 1/2$ and has the stability limit $s \leq 0.5/(1-2\beta)$ for $\beta < 1/2$.

The prior examples demonstrate the stability analysis of numerical approximations for a single equation (dependent variable) and cases where only two time levels are involved in the analysis. In cases with more time levels the equation for g becomes a higher order polynomial which will have several solutions for g. Each of the roots of the polynomial must satisfy $|g| \leq 1$ for stability. The solutions in this case are in general complex solutions such that the amplification is best measured by $|g|^2 = g \cdot g^*$ where g^* is the complex conjugate of g and the condition for stability becomes $g \cdot g^* \leq 1$. It may be necessary to identify the solutions numerically for a range of parameters such as s and β. In cases of systems of equations the amplification factor becomes a vector where each component has to satisfy the stability condition.

Strictly the demonstrated approach is valid only for linear equations. In a general nonlinear case it may be a rather complex task to identify the amplification factor. The first step in such cases is the linearization of the underlying equations. A second complication in a more general case is the occurrence of physical instabilities which have naturally growing solutions. In that case it is allowed that the magnitude of the amplification factor is $|g| \leq 1 + O(\Delta t)$.

5.4 Solution accuracy

In cases where both consistency and stability are satisfied it can usually be expected that the solution error will be approximately the truncation error. This is particularly the case if

- the grid spacing and time step are sufficiently small,
- boundary conditions are without discontinuities, and
- the solution is analytic, i.e., involve on finite order derivatives.

Even though the numerical solution may converge to the accurate solution it is usually not necessary to approximate the exact solution with arbitrary accurateness. In many cases it expected to derive qualitative or quantitative results with sufficient accurateness for the particular purpose of the model.

To obtain confidence in more complex nonlinear models where analytic solution are in general not available a variety of confidence building measure can be applied. For many applications it is not only helpful but necessary to test simplified related problems first. In fact industrial/commercial applications of numerical models have usually a long history of various applications. In terms of simplified problems there is usually a variety of simplifications which yield analytic solutions such as

- equilibrium configurations,
- steady state systems,
- and linear modes (waves).

In addition consistency test should use

- refining the spatial grid and/or the time step,
- establish that gradients are resolved by the grid scale , and
- test conservation law such as mass, momentum, energy, and magnetic flux conservation.

The following presents a rather specific example on how grid refinement can be used to increase accuracy

Richardson extrapolation

Consider two solution T_a and T_b with different resolution Δx_a and Δx_b of the diffusion equation obtained with the FTCS scheme sim1.f. The goal here is to combine the solutions

$$T_c = aT_a + bT_b \tag{5.45}$$

where a and b are constants such that the combined errors of the solutions partially cancel each other with a net solution of higher accuracy.

$$\left.\frac{\partial T}{\partial t}\right|_i^n - \alpha \left.\frac{\partial^2 T}{\partial x^2}\right|_i^n + a\ E_a|_i^n + b\ E_b|_i^n = 0 \tag{5.46}$$

with

$$\begin{aligned} E_a|_i^n &= \frac{\alpha \Delta x_a^2}{2}\left(s - \frac{1}{6}\right)\left[\frac{\partial^4 T}{\partial x^4}\right]_i^n + O\left(\Delta x^4\right) \\ E_b|_i^n &= \frac{\alpha \Delta x_b^2}{2}\left(s - \frac{1}{6}\right)\left[\frac{\partial^4 T}{\partial x^4}\right]_i^n + O\left(\Delta x^4\right) \end{aligned}$$

For $\Delta x_b = \Delta x_a/2$ one obtains the condition

$$\begin{aligned} a + \frac{1}{4}b &= 0 \\ a + b &= 1 \end{aligned}$$

with the solution

$$a = -\frac{1}{3}, \qquad b = \frac{4}{3} \tag{5.47}$$

This linear combination of two solutions reduces the Error to $O\left(^4\right)$ but requires two computations of solutions.

The time to reach a particular solution is proportional to $1/\Delta x$ and $1/\Delta t$ (or $\sim N_x N_t$). Thus we can write the computer time need for a particular result with fixed t_{max}

$$c_t = \frac{K}{\Delta x \Delta t} = \frac{\alpha K}{s \Delta x^3} \tag{5.48}$$

where K is a constant. For $s = 1/2$ the computer times needed for the two runs (with Δx and $2\Delta x$) are

Table 5.2: RMS error obtained by regular FTCS, Richardson Extrapolation, and $s = 1/6$ for the diffusion equation.

Scheme	$\Delta x = 0.1$	$\Delta x = 0.05$	$\Delta x = 0.025$
ftcs, $s = 1/2$	0.4915	0.0869	0.0153
ftcs, $s = 1/2$, RE	0.9744	0.0104	0.00063
ftcs, $s = 1/6$	0.0017	0.000054	0.0000034

$$c_{t,\Delta x} = \frac{2\alpha K}{\Delta x^3} \quad , \quad c_{t,2\Delta x} = \frac{0.25\alpha K}{\Delta x^3}$$

such that the combined time for the Richardson extrapolation is

$$c_{t,\Delta x,RE} = \frac{2.25\alpha K}{\Delta x^3} \tag{5.49}$$

In comparison the time needed for the special case $s = 1/6$ is

$$c_{t,\Delta x,s=1/6} = \frac{6\alpha K}{\Delta x^3} \tag{5.50}$$

In conclusion the Richardson extrapolation is by more than a factor of 2 faster to achieve the same order of accuracy. Note, however, that the fourth order error has a term $O\left((2\Delta x)^4\right)$ in the Richardson extrapolation such that the accuracy is in factor a factor of about 16 less than in the corresponding special case with $s = 1/6$.

5.5 Efficiency

Computational efficiency is a concept that is not entirely well defined. However in general a method has large efficiency if it produces results with small errors ϵ (or large accuracy) in a short period of computer time c_t. So conceptual one can express computational efficiency as

$$CE = \frac{K}{\epsilon c_t} \tag{5.51}$$

Note, however, that this is in a way subjective and it is not alway clear what definition should should be applied to the error ϵ. For instance if ϵ were merely the truncation error then the product of ϵc_t might converge to zero for $\epsilon \hookrightarrow 0$ even though $c_t \hookrightarrow \infty$. This, however, is of no use because it would suggest to use infinite computer time because in this limit the efficiency is largest. A better approach is to fix a required quality of the result and then to compare the execution time for different methods and realizations including aspects such as vectorization, parallelization, and the actual style of programming in addition to the particular method for the numerical approximation.

Operation counts:

Basic computational operations have a typical execution time on any computer. The principle categories of computer operations are

- floating point operations (fl)
- fixed point operations (fx)
- assignments (=)
- logical operations, e.g., if, and, or (l)
- mathematical library functions, e.g., sin, exp, etc (m)

To derive an estimate for the actual computation time it is necessary to derive times for each of the typical operations in a computer program. An overview for the relative execution times of various typical operations is provided in Table 5.3. In the table the microcomputer is one of the earliest versions of an Intel based PC. The supermicro represents an early SUN workstation.

Table 5.3: Relative execution times for basic operations.

Operation	Microcomputer NEC-APC IV	Supermicro SUN Sparc St1	Mainframe IBM-3090
add (fl)	1.0	1.0	1.0
subtr (fl)	1.0	1.0	0.8
multipl (fl)	1.1	1.0	0.8
div (fl)	1.4	5.8	3.1
assign (=)	0.1	1.0	0.1
if (l)	0.1	0.9	0.2
add (fx)	0.05	0.6	0.2
subtr (fx)	0.05	0.6	0.2
multipl (fx)	0.4	0.6	0.6
div (fx)	0.5	5.4	3.2
power	2.7	20	16
sqrt	2.0	29	16
sin	10.0	29	17.6
exp	6.7	33	15

Inspecting the table one can conclude a number of ground rules for programming fast code. If possible one should avoid divisions in particular on RISC (reduced instruction set) processors. Also library calls to mathematical functions are usually expansive and should be carefully chosen or avoided.

An example program to evaluate execution times for different operations is provide on the web page. It should be noted that this evaluation is not always unique for present computer generations.

First compilers typically optimize computer programs and this optimization can be sophisticated. For instance if the operations within a loop are identical it is not necessary to execute the loop n times and some compilers detect this and cut the loop short by executing the required commands only once. Other optimizations exist of which we may not be aware for any particular part of a program. Also modern computer processors are usually 64 bit wide allowing more than a single operation per processor cycle. Thus the time for a particular operation can depend in part on the program context.

As an example for an operation count consider the FTCS equation

$$T_i^{n+1} = (1 - 2s)T_i^n + s\left(T_{i-1}^n + T_{i+1}^n\right)$$

which comprises 3 multiplications, two additions, one subtraction, and integer add, and an assignment or in total: 6 fl, 1fx, and 1 "=". This provides us with the means to carry out an operation count for the important (most frequently used) subroutines of a program.

A last remark: It is of large importance to determine always estimates on the execution time for a simulation/modeling program. For real applications This is helpful not only to determine possible inefficiencies due to inefficient programming but also for planning of larger size programs and to gain an understanding of the potential for a program, i.e., whether it can be scaled up to treat a compelling application.