ES_APPM 446-1 Notes
Numerical Methods for Partial Differential Equations

Prof. David Chopp

Last update: November 4, 2011

Contents

1 Review of Numerical Methods for ODE's

Lec. 1 We will start with a review of analyzing numerical methods for ordinary differential equations. Many of the techniques we will be using this quarter are easier to understand in the framework of ordinary differential equations. We will then carry these ideas over to partial differential equations when possible.

Definition 1 An Ordinary differential equation is a problem of the form

$$\begin{aligned} y'(t) &= f(y(t), t) \\ y(0) &= y_0 \end{aligned} \tag{1}$$

where $y(0) = y_0$ is the initial condition.

Suppose we wish to solve (1) on the interval $0 \le t \le T$ numerically.

Definition 2 Let $0 = t_0 < t_1 < t_2 < \cdots < t_N = T$ be a finite sequence of points in the interval $[0, T]$. A *numerical method* for solving (1) is a mapping $A(f, y_0)$ from t_i to y_i for each i. The value y_i is an approximation to the true solution $y(t_i)$ of (1).

Example 1.1:

[Euler's Method] We start with $y_0 = y(0)$, the initial condition of (1). Successive values of y_i are given by the definition:

$$y_{i+1} = y_i + f(y_i, t_i)(t_{i+1} - t_i)$$

Definition 3 An algorithm *converges* if given $\epsilon > 0$, there is an $h > 0$ such that if $\max_i |t_{i+1} - t_i| < h$, then

$$|y(T) - y_N| < \epsilon.$$

The definition of convergence is nice, but unwieldy. What we need is conditions on a method which will ensure convergence. The main result we will obtain is one which applies equally well to linear partial differential equations as well, namely **stability plus consistency implies convergence**.

To start, we make a simple definition:

Definition 4 A function $f(x)$ is *Lipschitz continuous* in x if there is a constant L such that for any x, and y, the following inequality holds:

$$|f(x) - f(y)| \le L\,|x - y|\,.$$

Note that if f is differentiable, then taking $L = \max(f'(x))$ would suffice, so any differentiable function is Lipschitz continuous. This form of continuity is stronger than simple continuity, but weaker than differentiability. For example, $f(x) = |x|$ is not differentiable, but is Lipschitz continuous.

Assume that the function f in (1) is continuous in t and Lipschitz continuous in y for $-\infty < y < \infty$. Suppose further that we try to solve this equation using a one-step method of the form

$$y_{i+1} = y_i + h\psi(y_i, t_i, h) \tag{2}$$

where we will assume $t_{i+1} - t_i = h$ is constant.

Definition 5 A one-step method of the form (2) is *stable* if there is an $h_0 > 0$ and $K > 0$ such that if y_0 and $\tilde{y}_0$ are two different starting values, then

$$|y_N - \tilde{y}_N| \le K|y_0 - \tilde{y}_0|$$

for all $0 \le h \le h_0$.

Put more simply, this definition means that for a stable method, a bounded change in the initial conditions results in a bounded change in the computed solution.

Example 1.2:

A very simple example of a stable method is $\psi(y, t, h) \equiv 0$. Of course this example is exceedingly stable, but it is also exceedingly useless.

Clearly, stability is not enough to ensure convergence. In particular, this definition makes no mention of the differential equation that is being solved. We will come back to this point later.

Theorem 1 If $\psi(y,t,h)$ is Lipschitz in y with Lipschitz constant K, then the one-step method (2) is stable.

Proof 1
Consider two solutions y_N, $\tilde{y}_N$, derived from (2).

$$\begin{aligned}
|y_N - \tilde{y}_N| &= |y_{N-1} + h\psi(y_{N-1}, t_{N-1}, h) - \tilde{y}_{N-1} - h\psi(\tilde{y}_{N-1}, t_{N-1}, h)| \\
&\leq |y_{N-1} - \tilde{y}_{N-1}| + h|\psi(y_{N-1}, t_{N-1}, h) - \psi(\tilde{y}_{N-1}, t_{N-1}, h)| \\
&\leq |y_{N-1} - \tilde{y}_{N-1}| + hK|y_{N-1} - \tilde{y}_{N-1}| \\
&= (1 + hK)|y_{N-1} - \tilde{y}_{N-1}| \\
&\vdots \\
&\leq (1 + hK)^N |y_0 - \tilde{y}_0| \\
&\leq e^{hNK}|y_0 - \tilde{y}_0| = e^{TK}|y_0 - \tilde{y}_0|
\end{aligned}$$

Therefore, the method is stable.

Note here that we needed the last equation because the constant must be independent of the step size h.

Example 1.3:

Euler's method is stable. For Euler's method, $\psi(y,t,h) = f(y,t)$, and if we assume f itself is Lipschitz continuous in y then so is ψ, hence by the theorem, Euler's method is stable.

Definition 6 A method of the form (2) is consistent if $\psi(y,t,0) = f(y,t)$ for all y, t in the domain.

Suppose that $y(t)$ is the exact solution, and define an error term τ by the equation

$$y(t_{n+1}) = y(t_n) + h\psi(y(t_n), t_n, h) + h\tau.$$

Solving for τ produces

$$\begin{aligned}
h\tau &= y(t_{n+1}) - y(t_n) - h\psi(y(t_n), t_n, h) \\
&= y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n + \theta h) - y(t_n) - h\psi(y(t_n), t_n, h) \\
&= h[y'(t_n) - \psi(y(t_n), t_n, h)] + \frac{h^2}{2}y''(t_n + \theta h) \\
\tau &= y'(t_n) - \psi(y(t_n), t_n, h) + \frac{h}{2}y''(t_n + \theta h) \\
&= f(y(t_n), t_n) - \psi(y(t_n), t_n, h) + \frac{h}{2}y''(t_n + \theta h)
\end{aligned}$$

As $h \to 0$, we want $\tau \to 0$, so this implies that we must have $|f(y(t_n), t_n) - \psi(y(t_n), t_n, h)| \to 0$ as $h \to 0$. So consistency is a useful requirement, since it is connected to the size of the local truncation error τ.

Theorem 2 If $\psi(y,t,h)$ is Lipschitz in y with constant L, continuous in t and h, for all $0 \leq t \leq T$, $-\infty < y < \infty$, $0 \leq h \leq h_0$, then the method (2) is consistent iff it is convergent.

Proof 2
Let $\psi(y,t,0) = g(y,t)$ and consider the initial value problem

$$\begin{aligned}
z'(t) &= g(z,t) \\
z(0) &= y_0
\end{aligned}$$

We will prove that $y_n \to z(t_n)$ as $h \to 0$ where y_n is given by (2).

Let $e_n = y_n - z(t_n)$, then

$$\begin{aligned} e_{n+1} &= y_{n+1} - z(t_{n+1}) \\ &= y_n + h\psi(y_n, t_n, h) - z(t_n) - hz'(t_n + \theta h) \\ &= e_n + h[\psi(y_n, t_n, h) - g(z(t_n + \theta h), t_n + \theta h)] \\ &= e_n + h[\psi(y_n, t_n, h) - \psi(z(t_n + \theta h), t_n + \theta h, 0)] \end{aligned}$$

and hence

$$|e_{n+1}| \le |e_n| + h|\psi(y_n, t_n, h) - \psi(z(t_n + \theta h), t_n + \theta h, 0)|$$

Thus,

$$\begin{aligned} &\le |e_n| + h|\psi(y_n, t_n, h) - \psi(z(t_n), t_n, h) \\ &\qquad + \psi(z(t_n), t_n, h) - \psi(z(t_n), t_n, 0) \\ &\qquad + \psi(z(t_n), t_n, 0) - \psi(z(t_n + \theta h), t_n + \theta h, 0)| \\ &\le |e_n| + h|\psi(y_n, t_n, h) - \psi(z(t_n), t_n, h)| \\ &\qquad + h|\psi(z(t_n), t_n, h) - \psi(z(t_n), t_n, 0)| \\ &\qquad + h + |\psi(z(t_n), t_n, 0) - \psi(z(t_n + \theta h), t_n + \theta h, 0)| \\ &\le |e_n| + hL|e_n| + hD \end{aligned}$$

where

$$D = \max_{\substack{0 \le t \le T \\ 0 \le h \le h_0}} |\psi(z(t), t, h) - \psi(z(t), t, 0)| + L \max_{\substack{0 \le t \le T \\ 0 \le h \le h_0}} |z'(t) - z'(t + \theta h)|$$

Since ψ, (and hence g and z') is continuous as a function of t, h on the compact set $[0, T] \times [0, h_0]$, then D is finite. Also, ψ is uniformly continuous, so $D \to 0$ as $h \to 0$.

Therefore, $|e_{n+1}| \le (1 + hL)|e_n| + hD$.

Lemma 1 If $|e_{n+1}| \le (1 + hL)|e_n| + hD$ and $0 \le nh \le T$, then

$$|e_n| \le D\left(\frac{(1 + hL)^n - 1}{L}\right) + (1 + hL)^n |e_0| \le \frac{D}{L}(e^{LT} - 1) + e^{LT}|e_0|.$$

Proof 3

The proof is by induction. It is trivially true for $n = 0$. Assume it is true for $k = 0, \ldots, n$, then

$$\begin{aligned} |e_{n+1}| &\le (1 + hL)|e_n| + hD \\ &\le (1 + hL)\left[D\left(\frac{(1 + hL)^n - 1}{L}\right) + (1 + hL)^n |e_0|\right] + hD \\ &= D\left(\frac{(1 + hL)^n - 1}{L}\right)(1 + hL) + hD + (1 + hL)^{n+1}|e_0| \\ &= D\left[(1 + hL)\left(\frac{(1 + hL)^n - 1}{L}\right) + h\right] + (1 + hL)^{n+1}|e_0| \\ &= D\left(\frac{(1 + hL)^{n+1} - (1 + hL)}{L} + h\right) + (1 + hL)^{n+1}|e_0| \\ &= D\left(\frac{(1 + hL)^{n+1} - 1}{L}\right) + (1 + hL)^{n+1}|e_0| \end{aligned}$$

which proves the first inequality by induction. The second inequality follows from the fact $1 + hL \le e^{hL}$, so $(1 + hL)^n \le e^{hnL} = e^{TL}$.

Since $|e_0| = y_0 - z(0)$, then we now have

$$|e_n| \leq \frac{D}{L}(e^{LT} - 1)$$

Finally, we have as $h \to 0$, $D \to 0$, so $e_n \to 0$ and therefore $y_n \to z(t_n)$ as $h \to 0$.

If ψ is consistent, then $\psi(y, t, 0) = f(y, t) = g(y, t)$, so $y'(t) = z'(t)$ and $y(0) = z(0)$, hence by uniqueness of solutions for system (1), it follows that $y(t) = z(t)$ which proves convergence.

If the method converges, then $y_n \to y(t_n)$ and $y_n \to z(t_n)$, hence $y(t_n) = z(t_n)$. Since this is true for every $t \in [0, T]$, then it must be that $z(t) = y(t)$ on $[0, T]$ and hence $f(y, t) = y'(t) = z'(t) = g(y, t) = \psi(y, t, 0)$. Therefore, ψ is consistent.

We have now shown that a one-step explicit numerical method for solving ordinary differential equations of the form (1) will converge provided it is both stable and consistent.

Example 1.4:

Euler's method, Runge-Kutta methods are both stable and consistent, hence they are convergent.

This proof also shows that the order of the error is one less than the order of the error made at the local level. For Euler's method, if we plug the true solution of (1) into the method, we make a small error τ:

$$\begin{aligned} y_{n+1} &= y_n + hf(y_n, t_n) \\ y(t_{n+1}) &= y(t_n) + hf(y(t_n), t_n) + \tau h \end{aligned}$$

Solve for τ:

$$\begin{aligned} \tau h &= y(t_{n+1}) - y(t_n) - hf(y(t_n), t_n) \\ &= y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n + \theta h) - y(t_n) - hf(y(t_n), t_n) \\ &= \frac{h^2}{2}y''(t_n + \theta h) \end{aligned}$$

This is called the *truncation error*.

Again, write $e_n = y(t_n) - y_n$, then we have

$$\begin{aligned} |e_{n+1}| &\leq |e_n| + h|f(y(t_n), t_n) - f(y_n, t_n)| + h|\tau| \\ &\leq (1 + hL)|e_n| + h|\tau|. \end{aligned}$$

Let $D = \frac{h}{2}\max_{0 \leq t \leq T}|y''(t)| = \frac{1}{2}hK$, then we again get

$$|e_{n+1}| \leq |e_n|(1 + hL) + hD$$

and hence

$$\begin{aligned} |e_n| &\leq \frac{D}{L}(e^{LT} - 1) \\ &= \frac{1}{2}\frac{hK}{L}(e^{LT} - 1) \to 0 \text{ as } h \to 0. \end{aligned}$$

In particular, $D = O(h)$, so Euler's method is a *first order method*.

Lastly, we will show how the order of a method can be demonstrated *a posteriori* by experiment using a technique similar to Richardson's extrapolation. Suppose we are solving a differential equation on the interval $t_0 \leq t \leq T$, then we have $Nh = T$ where N is the number of time steps, and h is the step size.

Assume the error at time T using N steps is given by $e_{N,h} \approx Ah^p$ where A is some uknown constant and p is the order of convergence of the numerical method we are using. (Note that this is a rather bold assumption, but for small h is not as far off from reality as one might think.) By the same assumption, if we use $2N$ steps to reach T we must use a time step $h/2$ and we have the error $e_{2N,h/2} \approx A\left(\frac{h}{2}\right)^p$. We can now isolate the desired value of p by writing

$$\frac{e_{N,h}}{e_{2N,h/2}} \approx \frac{Ah^p}{A\left(\frac{h}{2}\right)^p} = 2^p,$$

and consequently,

$$p \approx \log_2 \frac{e_{N,h}}{e_{2N,h/2}}.$$

The quantities $e_{N,h}$, $e_{2N,h/2}$ can be computed by solving an ODE with a known exact solution.

We often use this type of technique to verify the order of a method and as a check that the method has been programmed correctly. In practice, it is not often that an exact solution for the equation is known (otherwise, why are we computing it?).

2 Partial Differential Equations

We move on now to the approximation of partial differential equations. We will begin with some general information about partial differential equations. There are three main classifications for partial differential equations.

- Hyperbolic: $u_t = u_x$, or $u_{tt} = u_{xx}$.
- Parabolic: $u_t = u_{xx}$.
- Elliptic: $u_{xx} + u_{yy} = 0$.

We will only deal with hyperbolic and parabolic equations in this course. These are time dependent problems and will generally be formulated as initial value problems. Elliptic equations are not time dependent, and are normally handled using techniques from numerical linear algebra.

2.1 Hyperbolic Equations

The simplest hyperbolic equation is the "baby wave equation"

$$u_t = u_x, \quad u(x,0) = f(x).$$

The solution of this problem is $u(x,t) = f(x+t)$ and is easily verified. The solution shows that the initial data is simply translated at speed 1 to the left as depicted in the figure below.

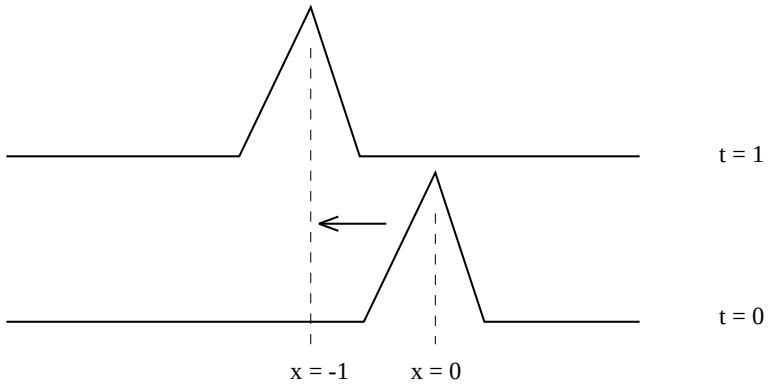


Illustration of "baby wave equation" solution

Hyperbolic differential equations are typified by finite time transmission of data.

Similarly, the second order wave equation is given by

$$u_{tt} = u_{xx}, \quad u(x,0) = f(x), \quad u_t(x,0) = 0.$$

This system has the solution

$$u(x,t) = \frac{1}{2}f(x+t) + \frac{1}{2}f(x-t).$$

Thus, information travels at a finite rate, but in both directions.

Hyperbolic equations are often referred to as convective, non-dissipative, or dispersive. This is also true for equations which contain an odd number of derivatives of x. Even the second order wave equation is convective because it can be written as a first order system that looks like the baby wave equation with only one spatial derivative.

Consider a linear hyperbolic system of equations

$$u_t = Au_x, \quad -\infty < x < \infty, \quad u(x,0) = u_0(x)$$

where $u \in R^n$ and A is an $n \times n$ matrix. This problem is hyperbolic if A has n distinct real eigenvalues.

In two dimensions,

$$u_t = Au_x + Bu_y$$

is hyperbolic if for any α, β not both zero, $\alpha A + \beta B$ has n distinct real eigenvalues. The matrices A and B may depend on x, y, and t in which case the n distinct eigenvalues must hold for all x, y, and t.

If we add a forcing term $f(x,t)$,

$$u_t = Au_x + f(x,t)$$

hyperbolicity is still determined by the eigenvalues of A independent of the forcing term.

2.2 Parabolic Equations

Parabolic equations are typical of dissipative phenomena such as energy loss (friction, tension, or heat). The canonical parabolic equation is the "heat equation"

$$u_t = u_{xx}, \quad u(x,0) = f(x).$$

We will not give the full solution to this problem here. However, we will give the solution for a particular wave mode

$$u_t = u_{xx}, \quad u(x,0) = e^{ikx}.$$

To solve, we assume that $u(x,t) = e^{\omega t}e^{ikx}$. Then,

$$\omega e^{\omega t}e^{ikx} = -k^2 e^{\omega t}e^{ikx}$$
$$\omega = -k^2$$

Thus,

$$u(x,t) = e^{-k^2 t}e^{ikx}$$

The number k is called the wave number with corresponding wave length $\lambda = 2\pi/k$. Notice that $u(x,t)$ decays in time (dissipation). Another key characteristic of parabolic equations is that information travels infinitely fast.

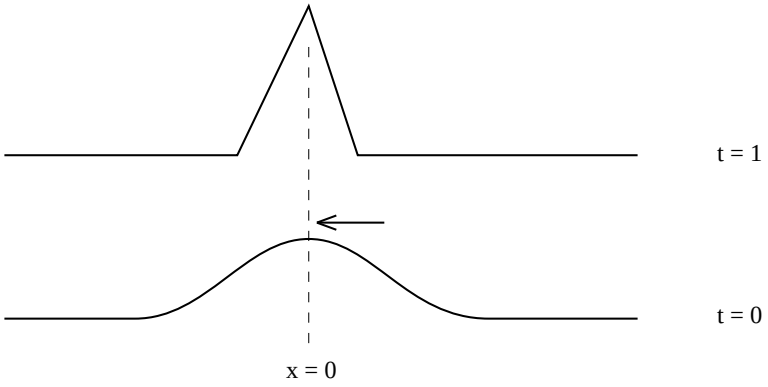


Illustration of "heat equation" solution

The heat equation and other equations with an even number of spatial derivatives are called often called diffusive or dissipative. Also, note that the sign is important for parabolic equations (unlike for hyperbolic equations).

2.3 Well Posedness

In general, we say a time dependent partial differential equation is *well posed* if solutions grow no worse than exponentially. Thus, a Cauchy problem of the form

$$u_t = Au_x, \quad u(x,0) = g(x)$$

is well posed if there are constants C, α such that

$$||u(x,t)|| \leq Ce^{\alpha t}||u(x,0)|| = Ce^{\alpha t}||g(x)||.$$

Note that C, α are independent of the initial data $g(x)$.

Note also that we are using functional norms which we will be using often in this course. The typical norms we will use are:

$$||f(x)||_2 = \left(\int_{-\infty}^{\infty} ||f(x)||^2 \, dx\right)^{\frac{1}{2}} \qquad \text{2-norm:}$$

$$||f(x)||_1 = \int_{-\infty}^{\infty} ||f(x)|| \, dx \qquad \text{1-norm:}$$

$$||f(x)||_\infty = \max_{-\infty<x<\infty} ||f(x)|| \qquad \infty\text{-norm:}$$

Example 2.1:

1. The "backward heat equation", $u_t = -u_{xx}$, is ill-posed. Suppose we look for periodic solutions of the form $u(x,t) = e^{\omega t}e^{ikx}$. Plugging this into the equation, we see that $\omega = k^2$, and hence

   $$u(x,t) = e^{k^2 t}e^{ikx}$$

   Since bounded growth cannot be established for arbitrary k, this equation is ill-posed.

2. Let

   $$u_t = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} u_x$$

and suppose we look for periodic solutions corresponding to the k^{th} Fourier mode. We assume that $u(x,t) = e^{\omega t}e^{ikx}z$ where z is an unspecified constant vector. Plugging it in we get

$$\omega e^{\omega t}e^{ikx}z = ik\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}e^{\omega t}e^{ikx}z$$

or

$$\omega z = ik\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}z$$
$$\frac{\omega}{ik}z = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}z$$

Thus, $\frac{\omega}{ik}$ must be an eigenvalue of the matrix. This matrix has eigenvalues ± 1, so we have

$$\frac{\omega}{ik} = \pm 1 \quad \Rightarrow \quad \omega = \pm ik.$$

Therefore, $u(x,t) = e^{\pm ikt}e^{ikx}z$ and we have an oscillating, non-growing solution.

3. Consider the equation

$$u_t = u_{xxxx}.$$

Again, we look for a solution of the form $u(x,t) = e^{\omega t}e^{ikx}$. We get

$$\omega e^{\omega t}e^{ikx} = k^4 e^{\omega t}e^{ikx}.$$

So $\omega = k^4$, and hence $u(x,t) = e^{k^4 t}e^{ikx}$. Since k can be taken to be arbitrarily large, we cannot bound the solution by $Ce^{\alpha t}$.

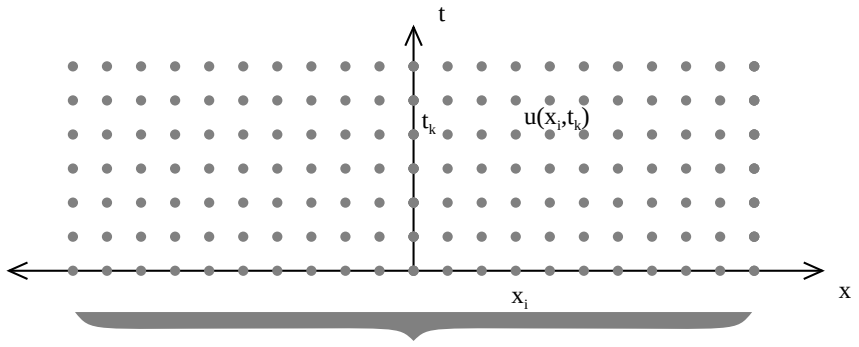
---

3 Solution strategy for partial differential equations

Consider the following initial value problem

$$u_t = u_x, \quad u(x,0) = g(x), \quad -\infty < x < \infty. \tag{3}$$

Let's take this apart and see how we can solve this numerically.



Numerical domain of the solution

We will advance in time just like in ordinary differential equations, and the time derivative is computed by computing the discrete spatial derivatives on the right hand side.

We present here a simple algorithm for solving the initial value problem (3) as a basic method from which all other methods evolve. If we let $u_{ik} \approx u(x_i, t_k)$, then we are given the initial data u_{i0}, and we must find u_{ik} for $k > 0$. The algorithm for computing the u_{ik} would go something like

1. Initialize $u_{i0} = g(x_i)$.
2. Given u_{ik} for all i and a given k, compute an approximation for $u_x(x_i, t_k)$.
3. Compute $u_{i,k+1}$ using a single or multi-step, implicit or explicit ordinary differential equation method.
4. Go to step 2.

4 Spatial Derivatives

We begin by focusing on step 2 above, where we need an approximation for the spatial derivative u_x. If we treat u_x as an ordinary differential equation in x, then Euler's method would become

$$\begin{aligned} u(x+h,t) &\approx u(x,t) + hu_x(x,t) \\ u_x(x,t) &\approx \frac{1}{h}(u(x+h,t) - u(x,t)) \end{aligned}$$

The discretized version would then be

$$u_x(x_i, t_k) \approx \frac{1}{h}(u_{i+1,k} - u_{ik}) \equiv D_+ u_{ik}$$

Alternatively, we could just have easily used the backward Euler approximation to obtain

$$u_x(x_i, t_k) \approx \frac{1}{h}(u_{ik} - u_{i-1,k}) \equiv D_- u_{ik}.$$

We saw that these approximations to the derivative are only first order accurate, can we do better? We will try to find the highest order approximation for $u_x(x_i, t_k)$ that only uses the values $u_{i-1,k}$, u_{ik}, and $u_{i+1,k}$. At this point we are only interested in spatial derivatives, so every approximation will be taken at time t_k. To reduce clutter, let us drop the time increment k for the time being.

Assume $u_x(x_i) \approx \alpha_1 u_{i+1} + \alpha_0 u_i + \alpha_{-1} u_{i-1}$. What values of α_{-1}, α_0, and α_1 will produce the smallest truncation error? The local truncation error is

$$\begin{aligned} \tau &= \alpha_1 u(x+h) + \alpha_0 u(x) + \alpha_{-1} u(x-h) - u_x(x) \\ &= \alpha_1 \left(u(x) + hu_x(x) + \frac{h^2}{2} u_{xx}(x) + \frac{h^3}{6} u_{xxx}(x) + O(h^4) \right) \\ &\quad + \alpha_0 u(x) \\ &\quad + \alpha_{-1} \left(u(x) - hu_x(x) + \frac{h^2}{2} u_{xx}(x) - \frac{h^3}{6} u_{xxx}(x) + O(h^4) \right) \\ &\quad - u_x(x) \\ &= (\alpha_{-1} + \alpha_0 + \alpha_1) u(x) + h \left(\alpha_1 - \alpha_{-1} - \frac{1}{h} \right) u_x(x) + \frac{h^2}{2} (\alpha_1 + \alpha_{-1}) u_{xx}(x) \\ &\quad + \frac{h^3}{6} (\alpha_1 - \alpha_{-1}) u_{xxx}(x) + O(h^4) \end{aligned}$$

We need to knock off terms, thus we get the system of equations

$$\begin{aligned}\alpha_1 + \alpha_0 + \alpha_{-1} &= 0\\ h\alpha_1 - h\alpha_{-1} &= 1\\ \alpha_1 + \alpha_{-1} &= 0\end{aligned}$$

which has solution $\alpha_{-1} = -\frac{1}{2h}$, $\alpha_0 = 0$, $\alpha_1 = \frac{1}{2h}$. Plugging these value back in, we get the truncation error

$$\begin{aligned}\tau &= \frac{h^3}{6}\frac{1}{h}u_{xxx}(x) + O(h^4)\\ &= \frac{h^2}{6}u_{xxx}(x) + O(h^4)\end{aligned}$$

This means the approximation

$$u_x(x_i, t_k) \approx \frac{1}{2h}(u_{i+1,k} - u_{i-1,k}) \equiv D_0 u_{ik}$$

is a second order approximation.

The above analysis relied upon Taylor series approximations to analyze the spatial derivatives D_+, D_-, D_0. We can also use Fourier analysis as well.

Assume $u(x) = e^{ikx}$, then we are approximating $u_x = ike^{ikx}$. Plugging this into the difference $D_0 u_{ik}$, we get

$$\begin{aligned}\frac{1}{2h}(u_{i+1,k} - u_{i-1,k}) &= \frac{1}{2h}(u(x+h) - u(x-h))\\ &= \frac{1}{2h}(e^{ik(x+h)} - e^{ik(x-h)})\\ &= e^{ikx}\frac{1}{2h}(e^{ikh} - e^{-ikh})\\ &= e^{ikx}ik\left(\frac{e^{ikh} - e^{-ikh}}{2ikh}\right)\\ &= ike^{ikx}\frac{\sin(kh)}{kh}\end{aligned}$$

The error in Fourier space is then

$$\text{error} = ike^{ikx} - \frac{\sin(kh)}{kh}ike^{ikx} = \left(1 - \frac{\sin(kh)}{kh}\right)ike^{ikx}$$

Let $e(kh) = 1 - \frac{\sin(kh)}{kh}$ be the relative error term in Fourier space. There are some important conclusions we can draw from this error term.

1. For fixed k, $\frac{\sin(kh)}{kh} \to 1$ as $h \to 0$, so for fixed k, $e(kh) \to 0$ as $h \to 0$. This is the analog of the consistency requirement we have seen for ordinary differential equations.

2. $$\begin{aligned}1 - \frac{\sin(kh)}{kh} &= 1 - \frac{kh - \frac{(kh)^3}{3!} + \frac{(kh)^5}{5!} + O((kh)^7)}{kh}\\ &= 1 - 1 + \frac{(kh)^2}{6} + O((kh)^4)\\ &= \frac{(kh)^2}{6} + O((kh)^4)\end{aligned}$$

   This shows that the approximation is second order.

3. The error is only a function of kh which is non-dimensional and can be related to the number of grid points in a wavelength.

Let λ be a wavelength, then

$$e^{ik(x+\lambda)} = e^{ikx}$$

and hence $\lambda = \frac{2\pi}{|k|}$. Let N be the number of grid points in a wavelength, then $N = \frac{\lambda}{h}$, and therefore

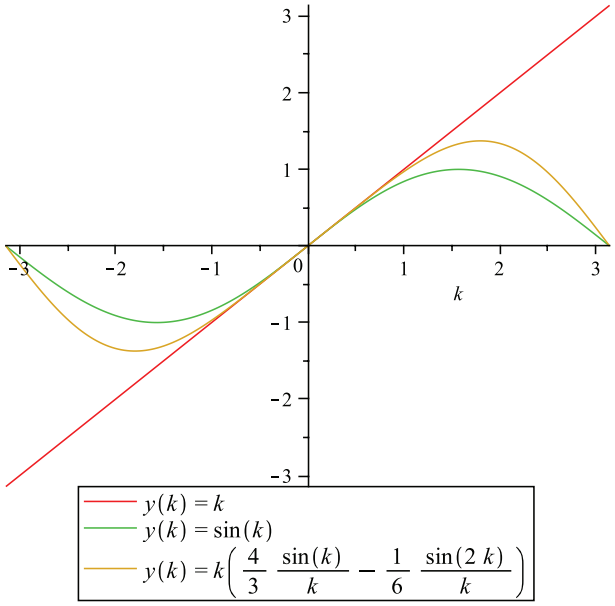
$$hN = \frac{2\pi}{|k|} \text{ or } |hk| = \frac{2\pi}{N}.$$

In fact, we have

$$e(kh) = \frac{(kh)^2}{6} + O((kh)^4) \approx \frac{(2\pi)^2}{6N^2}.$$

So the local relative error can be determined directly by the number of grid points in a wavelength.

We can graph the approximations to u_x in Fourier space. Assume h is fixed, say $h = 1$, then plot the numerical and exact multipliers of e^{ikx} as a function of k. The exact solution is $y(k) = k$. Central differencing has the graph of $y(k) = k\frac{\sin(k)}{k}$, and the 5-point stencil is $y(k) = k\left(\frac{4}{3}\frac{\sin(k)}{k} - \frac{1}{3}\frac{\sin(2k)}{2k}\right)$.



Plot of approximations of u_x in Fourier space

1. Notice that as $k \to 0$, all the plots pass through the origin. This is required in order to claim the method is consistent.

2. Notice that the errors are largest for $kh = \pi$. In terms of grid points per wavelength, this corresponds to $N = 2$, which is the fewest number of points needed in order to see the wave.

3. On a related note, higher wave numbers in a problem beyond $|k| = \frac{2\pi}{Nh}$ can manifest itself on the grid masquerading as a low wave number. This can be the source of serious numerical errors as we will see when we discuss spectral methods.

Higher order derivatives for u_x are also possible. However, it comes at a price. We have already seen that the best we could do was second order if we restrict our viewpoint to the points u_{i-1}, u_i, u_{i+1}. To obtain higher order approximations, we will need more grid points. We follow the same procedure as before: find constants α_{-n}, α_{-n+1}, $\ldots$, α_{-1}, α_0, α_1, $\ldots$, α_n such that

$$u_x \approx \sum_{j=-n}^{n} \alpha_{i+j} u_{i+j}.$$

While balanced formulae, i.e. $|\alpha_{-j}| = |\alpha_j|$, most often produce the best results, unbalanced formulae with certain properties have virtues as well as we will see later.

Next, let's compute a stencil for approximating u_{xx}. Again, we will use only the nearest neighbor grid points.

$$\begin{aligned}
\tau &= \alpha_1 u_{i+1} + \alpha_0 u_i + \alpha_{i-1} u_{i-1} - u_{xx} \\
&= \alpha_1 u(x+h) + \alpha_0 u(x) + \alpha_{-1} u(x-h) - u_{xx}(x) \\
&= \alpha_1 \left(u(x) + h u_x(x) + \frac{h^2}{2} u_{xx}(x) + \frac{h^3}{6} u_{xxx}(x) + \frac{h^4}{24} u_{xxxx}(x) + O(h^5) \right) \\
&\quad + \alpha_0 u(x) \\
&\quad + \alpha_{-1} \left(u(x) - h u_x(x) + \frac{h^2}{2} u_{xx}(x) - \frac{h^3}{6} u_{xxx}(x) + \frac{h^4}{24} u_{xxxx}(x) + O(h^5) \right) \\
&\quad - u_{xx}(x)
\end{aligned}$$

Again, we knock off terms to get the equations

$$\begin{aligned}
\alpha_1 + \alpha_0 + \alpha_{-1} &= 0 \\
h\alpha_1 - h\alpha_{-1} &= 0 \\
\frac{h^2}{2}\alpha_1 + \frac{h^2}{2}\alpha_{-1} &= 1
\end{aligned}$$

which has the solution $\alpha_1 = \alpha_{-1} = \frac{1}{h^2}$, $\alpha_0 = \frac{-2}{h^2}$. Therefore, the approximation is

$$u_{xx} \approx \frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) \equiv D_+ D_- u_{ik}$$

and the truncation error is

$$\tau = \frac{h^2}{12} u_{xxxx} + O(h^4)$$

This approximation is also second order.

Let us employ Fourier analysis as before. Let $u = e^{ikx}$, then $u_{xx} = -k^2 e^{ikx}$. We have

$$\begin{aligned}
D_+ D_- u_i &= \frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) \\
&= \frac{1}{h^2}(e^{ik(x+h)} - 2e^{ikx} + e^{ik(x-h)}) \\
&= \frac{1}{h^2}(e^{ikh} + e^{-ikh} - 2)e^{ikx} \\
&= \frac{2}{(kh)^2}(1 - \cos(kh))(-k^2 e^{ikx})
\end{aligned}$$

Thus, the relative error is

$$\begin{aligned} e(kh) &= 1 - \frac{2}{(kh)^2}(1-\cos(kh)) \\ &= 1 - \frac{2}{(kh)^2}\left(1 - \sum_{n=0}^{\infty}\frac{(-1)^n(kh)^{2n}}{(2n)!}\right) \\ &= 1 - \frac{2}{(kh)^2}\left(1 - \left(1 - \frac{(kh)^2}{2} + \frac{(kh)^4}{24} + O((kh)^6)\right)\right) \\ &= 1 - 2\left(\frac{1}{2} - \frac{(kh)^2}{24} + O((kh)^4)\right) \\ &= \frac{(kh)^2}{12} + O((kh)^4) \end{aligned}$$

Clearly, for fixed k, $e(kh) \to 0$ as $h \to 0$. Using the relation $|kh| = \frac{2\pi}{N}$, we see that

$$e(N) \approx \frac{(2\pi)^2}{12N^2}.$$

5 Temporal derivatives

Now that we have studied spatial derivatives, let's combine the spatial derivative with time derivatives as in ordinary differential equations. We will see that including time derivatives will add new wrinkles beyond the simple ODE problems. For example:

1. A perfectly stable spatial derivative coupled with a perfectly stable temporal derivative does not guarantee a stable numerical method.

2. Stability is not an independent quality of a discretization, but depends on the relative size of the spatial and temporal time steps, as well as the actual differential equation being solved.

Numerical errors fall into two categories: dispersive and dissipative. Dispersive error is where different wave numbers have different phase velocities. Dissipative error is artificial loss of energy.

We will start by analyzing dispersive errors. Suppose you have an initial value problem with solution of the form

$$u(x,t) = e^{i\omega(k)t}e^{ikx}z \tag{4}$$

so that $u(x,t)$ oscillates with frequency $\omega(k)$. The quantity $\frac{d\omega}{dk}$ is a velocity and is sometimes called the *group velocity*. A solution with no dispersion has $\frac{d\omega}{dk} =$ constant.

Example 5.1:

Consider the equation $u_t = cu_x$, and plug in the solution (4) to get

$$\begin{aligned} i\omega(k)e^{i\omega(k)t}e^{ikx} &= icke^{i\omega(k)t}e^{ikx} \\ \omega(k) &= ck \end{aligned}$$

Then $\frac{d\omega}{dk} = c$, a constant, and so this equation is non-dispersive.

Example 5.2:

Consider the equation $u_t = u_x + cu_{xxx}$, and plug in equation (4) to get

$$i\omega(k)e^{i\omega(k)t}e^{ikx} = ike^{i\omega(k)t}e^{ikx} - ik^3ce^{i\omega(k)t}e^{ikx}$$
$$\omega(k) = k - ck^3$$

and hence $\frac{d\omega}{dk} = 1 - 3ck^2$ which is not constant. Therefore this equation is dispersive.

Notice that this also indicates that higher wave numbers oscillate faster and faster without bound which is not normally encountered in physical problems. However, numerical methods can introduce numerical dispersion giving interesting, real, and completely wrong results. Stability alone does not guarantee a reliable solution.

How does dispersion appear in numerical methods solving non-dispersive problems? Consider again the equation $u_t = u_x$, $u(x,0) = e^{ikx}$. Assume that $u(x,t) = \phi(t)e^{ikx}$, then

$$e^{ikx}\dot{\phi} = ik\phi e^{ikx}$$

and hence $\phi(t) = e^{ikt}$ so that $\omega(k) = k$, $\frac{d\omega}{dk} = 1$. So this problem is non-dispersive.

Next, let us apply D_0 on the right hand side. Assume $v(x,t) = \psi(t)e^{ikx}$ where

$$v_t = D_0 v.$$

Plugging in v yields

$$\dot{\psi}e^{ikx} = ik\frac{\sin(kh)}{kh}e^{ikx}$$

as we saw earlier. Hence $\dot{\psi} = ik\frac{\sin(kh)}{kh}\psi$ and

$$\psi(t) = e^{i\frac{\sin(kh)}{h}t}$$

so that $\frac{d\omega}{dk} = \cos(kh) \neq$ constant.

We shouldn't be shocked by this. Recall that when we approximated u_x by D_0 we got the local truncation error of

$$\tau = \frac{h^2}{6}u_{xxx} + O(h^4).$$

Thus, when we approximate $u_t = u_x$ by $u_t = D_0 u$, we are really solving the *modified equation*

$$u_t = u_x + \frac{h^2}{6}u_{xxx} + O(h^4) \approx u_x + \frac{h^2}{6}u_{xxx}.$$

But we saw earlier that this equation is dispersive.

Let us look at the two solutions again to study how to manage dispersive errors. The exact solution is

$$u(x,t) = e^{ikt}e^{ikx} = e^{ik(x+t)}$$

and the computed solution is

$$v(x,t) = e^{ik\left(\frac{\sin(kh)}{kh}t + x\right)}.$$

Clearly, the errors are in the phase, and they will grow linearly in t. Let $p(x,t) = ik(x+t)$ and $q(x,t) = ik\left(\frac{\sin(kh)}{kh}t + x\right)$ be the phases respectively. The phase error is then

$$|p(x,t) - q(x,t)| = \left|k\left(1 - \frac{\sin(kh)}{kh}\right)t\right|$$
$$\approx \left|kt\frac{(kh)^2}{6}\right|$$

Now to control the amount of error in the computed solution we need to know the number of periods in time the solution will be calculated as well as the spatial wavelength of the problem. Suppose you wish to solve for J periods in time, where a period is given by $T = 2\pi/k$, then $t = JT = J2\pi/k$ or $kt = 2\pi J$. At the same time, we have the number of grid points per spatial wavelength is given by $N = 2\pi/kh$ or $kh = 2\pi/N$. Plugging these into the error expression gives

$$e(J, N) \approx \left| 2\pi J \frac{(2\pi)^2}{6N^2} \right| = \left| \frac{(2\pi)^3}{6} \frac{J}{N^2} \right|.$$

We can use this to control the size of phase errors by increasing the number of grid points per wavelength.

Next we turn to dissipative error. We continue to solve $u_t = u_x$, $u(x, 0) = e^{ikx}$ but now we'll take a different approximation for u_x, namely

$$u_t = D_+ u$$

Recall that the modified equation in this case is

$$u_t = u_x + \frac{h}{2} u_{xx}$$

The even derivative is indicative of a dissipative term.

let us again apply Fourier analysis to the problem. Let $w(x, t) = \rho(t) e^{ikx}$, then

$$\begin{aligned} \dot{\rho} e^{ikx} &= \rho \frac{e^{ik(x+h)} - e^{ikx}}{h} \\ &= \rho e^{ikx} \frac{e^{ikh} - 1}{h} \\ &= \rho e^{ikx} \frac{1}{h} (\cos(kh) - 1 + i \sin(kh)) \end{aligned}$$

so

$$\begin{aligned} \rho(t) &= e^{\frac{\cos(kh) - 1 + i \sin(kh)}{h} t} \\ &= e^{i \frac{\sin(kh)}{h} t} e^{\frac{\cos(kh) - 1}{h} t} \end{aligned}$$

Thus, $w(x, t) = e^{i \frac{\sin(kh)}{h} t} e^{\frac{\cos(kh) - 1}{h} t} e^{ikx}$. Notice that the second term decays with time. This is a purely artificial result and is not from the original equation.

The decay rate is then

$$\frac{\cos(kh) - 1}{h} \approx \frac{1 - \frac{(kh)^2}{2} - 1}{h} = \frac{-k^2 h}{2}$$

so the decay term becomes $e^{-\frac{(kt)(kh)}{2}}$. Using $kt = 2\pi J$ and $kh = 2\pi/N$, we get a rate of decay of

$$e^{-\frac{J(2\pi)^2}{2N} t}.$$

Again, by increasing the number of grid points per wavelength and computing over a fixed number of periods J in time, we can control the amount of dissipation in the computed solution.

Finally, let us try using D_- for the approximation. In this case we get

$$\begin{aligned} \dot{\rho} e^{ikx} &= \rho \frac{e^{ikx} - e^{ik(x-h)}}{h} \\ &= \rho e^{ikx} \frac{1 - e^{-ikh}}{h} \\ &= \rho e^{ikx} \frac{1 - (\cos(kh) - i \sin(kh))}{h} \end{aligned}$$

and hence,

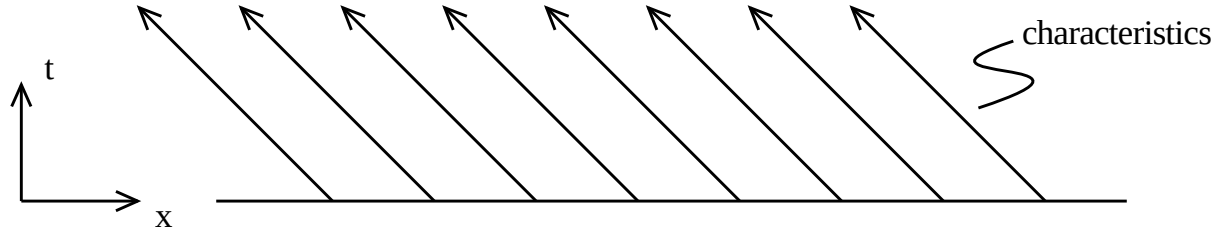
$$\begin{aligned}\rho(t) &= e^{-\frac{i\sin(kh)}{h}t}e^{\frac{1-\cos(kh)}{h}t} \\ &\approx e^{-\frac{i\sin(kh)}{h}t}e^{\frac{1-1+\frac{(kh)^2}{2}}{h}t} \\ &= e^{-\frac{i\sin(kh)}{h}t}e^{\frac{k^2h}{2}t}\end{aligned}$$

The second exponential this time is a growth term which grows faster with increasing k. That means this method blows up.

There is another explanation for why D_+ is stable while D_- is not. For this, let us look at the differential equation again. Recall that the exact solution of the initial value problem

$$u_t = u_x, \quad u(x,0) = g(x)$$

is $u(x,t) = g(x+t)$. Thus, $u(x,t)$ is constant along the lines $x+t =$constant as depicted in the figure below.



Plot of approximations of characteristics in $x-t$ space

Thus, information is traveling from right to left in this case. Now look what each approximation does in the figure below.

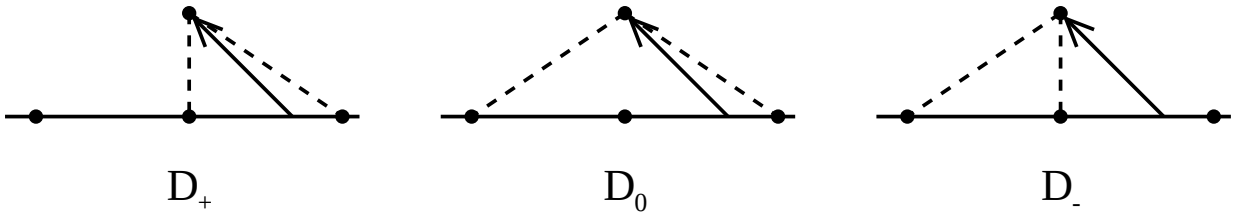


Illustration of stencils compared to characteristics

Given a point (x,t), the *domain of dependence* is the set of points y such that a change in the initial data at the point y results in a change in the value of the solution $u(x,t)$. For this problem, the domain of dependence is the single point $(x+t,0)$.

A general rule of thumb is that the numerical domain of dependence must contain the theoretical domain of dependence. Thus, we see that D_+, D_0 satisfy this condition provided that $\Delta t \leq \Delta x$. On the other hand, D_- cannot satisfy this condition for any value of Δt.

6 Explicit Methods

We are now ready to start discussing the analysis of fully discretized numerical methods. The first method we will analyze is one we have done in the homework. We wish to solve the initial value problem $u_t = cu_x$, $u(x,0) = g(x)$. We also assume that the space step size is h, and the time step size is k.

6.1 Forward Euler in time, Central Difference in Space

The first method we will try is

$$\begin{aligned} D^+u_i^n &= cD_0u_i^n \\ \frac{1}{k}(u_i^{n+1} - u_i^n) &= \frac{c}{2h}(u_{i+1}^n - u_{i-1}^n) \\ u_i^{n+1} &= u_i^n + \frac{ck}{2h}(u_{i+1}^n - u_{i-1}^n) \\ u_i^{n+1} &= u_i^n + \frac{\lambda}{2}(u_{i+1}^n - u_{i-1}^n) \end{aligned}$$

where $\lambda = \frac{ck}{h}$. This is an *explicit one-step* scheme. Explicit because the value at time $n+1$ is defined explicitly in terms of earlier time steps. It is a one-step method because the value of $n+1$ depends only on data at time n.

Let's compute the truncation error for this method. Let $u(x,t)$ be the exact solution, then

$$\begin{aligned} k\tau &= u_i^n + \frac{ck}{2h}(u_{i+1}^n - u_{i-1}^n) - u_i^{n+1} \\ &= u(x,t) + \frac{ck}{2h}(u(x+h,t) - u(x-h,t)) - u(x,t+k) \\ &= u + \frac{ck}{2h}(u + hu_x + \frac{h^2}{2}u_{xx} + \frac{h^3}{6}u_{xxx} + \frac{h^4}{24}u_{xxxx} + O(h^5)) \\ &\quad - (uu - hu_x + \frac{h^2}{2}u_{xx} - \frac{h^3}{6}u_{xxx} + \frac{h^4}{24}u_{xxxx} + O(h^5)) \\ &\quad - (u + ku_t + \frac{k^2}{2}u_{tt} + O(k^3)) \\ &= cku_x + ck\frac{h^2}{6}u_{xxx} + O(kh^4) - ku_t - \frac{k^2}{2}u_{tt} + O(k^3) \\ &= ck\frac{h^2}{6}u_{xxx} - \frac{k^2}{2}u_{tt} + O(kh^4 + k^3) \\ \tau &= \frac{ch^2}{6}u_{xxx} - \frac{k}{2}u_{tt} + O(h^4 + k^2) \\ &= O(h^2 + k) \end{aligned}$$

Thus, this is a $(1,2)$ method, i.e. a first order in time, second order in space method.

Next, let us do the analog of Fourier analysis called von Neuman analysis. Again, we assume the initial data is $e^{i\ell x}$. In this case, we also assume $x_j = jh$. At the same time, we let z represent the growth term in time. Thus, we assume

$$u_j^n = z^n e^{i\ell x_j} = z^n e^{i\ell jh} = z^n e^{ij\xi}$$

where $\xi = \ell h$.

As before, we plug this into the discretization to get

$$z^{n+1}e^{ij\xi} = z^n e^{ij\xi} + \frac{\lambda}{2}(z^n e^{i(j+1)\xi} - z^n e^{i(j-1)\xi}).$$

Dividing through by $z^n e^{ij\xi}$, we get

$$\begin{aligned} z &= 1 + \frac{\lambda}{2}(e^{i\xi} - e^{-i\xi}) \\ &= 1 + i\lambda\sin(\xi) \end{aligned}$$

For stability, we need $|z| \leq 1 + Ck$ for some constant C. We have

$$\begin{aligned}
|z|^2 &= 1 + \lambda^2 \sin^2(\xi) \\
&= 1 + \frac{c^2k^2}{h^2} \sin^2(\xi) \\
&\leq 1 + \frac{c^2k}{h^2} k \\
&= 1 + Ck \qquad \text{where we assume } \frac{c^2k}{h^2} \leq C \\
&\leq (1 + Ck)^2
\end{aligned}$$

so,

$$\begin{aligned}
|z| &\leq 1 + Ck \\
&\leq e^{Ck}
\end{aligned}$$

and hence,

$$|z|^n \leq e^{Cnk} = e^{CT}$$

for some fixed time T. This argument follows the same argument we used for showing stability of methods for ordinary differential equations. Therefore, the method is stable, though not very desirable because it grows in size while the exact solution does not.

6.2 Lax-Friedrichs Method

The next method we'll discuss is the Lax-Friedrichs method. It is very similar to the forward Euler method.

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) + \frac{\lambda}{2}(u_{j+1}^n - u_{j-1}^n)$$

Von Neuman analysis gives

$$\begin{aligned}
z^{n+1}e^{ij\xi} &= \frac{1}{2}(z^n e^{i(j+1)\xi} + z^n e^{i(j-1)\xi}) + \frac{\lambda}{2}(z^n e^{i(j+1)\xi} - z^n e^{i(j-1)\xi}) \\
z &= \frac{1}{2}(e^{i\xi} + e^{-i\xi}) + \frac{\lambda}{2}(e^{i\xi} - e^{-i\xi}) \\
&= \cos(\xi) + i\lambda \sin(\xi) \\
|z|^2 &= \cos^2(\xi) + \lambda^2 \sin^2(\xi) \\
&= \cos^2(\xi) + \sin^2(\xi) + (\lambda^2 - 1)\sin^2(\xi) \\
&= 1 + (\lambda^2 - 1)\sin^2(\xi)
\end{aligned}$$

Thus, this method is stable if $|\lambda| \leq 1$, i.e. $\frac{ck}{h} \leq 1$. This is called the *Courant-Friedrichs-Levy condition*, or more briefly the CFL condition. It is a condition which essentially forces the numerical method's domain of dependence to contain the theoretic domain of dependence.

The truncation error for this method is

$$\begin{aligned}
k\tau &= \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) + \frac{ck}{2h}(u_{j+1}^n - u_{j-1}^n) - u_j^{n+1} \\
&= \frac{1}{2}(u + hu_x + \frac{h^2}{2}u_{xx} + \frac{h^3}{6}u_{xxx} + O(h^4) \\
&\quad + u - hu_x + \frac{h^2}{2}u_{xx} - \frac{h^3}{6}u_{xxx} + O(h^4)) \\
&\quad + \frac{ck}{2h}(u + hu_x + \frac{h^2}{2}u_{xx} + \frac{h^3}{6}u_{xxx} + O(h^4) \\
&\quad\quad - u + hu_x - \frac{h^2}{2}u_{xx} + \frac{h^3}{6}u_{xxx} + O(h^4)) \\
&\quad - (u + ku_t + \frac{k^2}{2}u_{tt} + O(k^3)) \\
&= u + \frac{h^2}{2}u_{xx} + cku_x + \frac{ckh^2}{6}u_{xxx} + O(h^4) - u - ku_t - \frac{k^2}{2}u_{tt} + O(k^3) \\
\tau &= \frac{h^2}{2k}u_{xx} + \frac{ch^2}{6}u_{xxx} - \frac{k}{2}u_{tt}
\end{aligned}$$

Clearly, the averaging of u_j^n changed the dispersive error into a dissipative error. This means that this method tends to smooth things out.

The modified equation can be written down easily by taking the original equation and tacking on the correction terms. Care must be taken with the sign of the truncation error, which will depend on which side of the equation the truncation error term was appended. In the above derivation, the truncation error term was added to the left side, so it should be added to the left side, so the corrections terms should be added to the left side of the original equation. Thus, we have a modified equation of

$$u_t - \frac{k}{2}u_{tt} = u_x - \frac{h^2}{2k}u_{xx},$$

where only the leading order terms from the truncation error have been used. Solving this equation analytically is beyond the scope of this class, but there is one noticeable observation to make here: as $k \to 0$ for a fixed h, the last term on the right can get quite large. This can be observed by noting that the use of the averaging term in place of u_j^n means that the more iterations that averaging is used, the faster the solution will diffuse away any sharp corners. This is exactly what one will observe in practice, which is not very desirable. Thus, for this method, you want to keep k as close to the stability limit as possible.

In general, explicit schemes are easier to program, but are less robust and always have restrictions on the size of time steps to ensure stability. On the other hand, implicit schemes are very robust and stable, but they are expensive to compute, harder to program, and in the case of non-linear problems be very difficult to solve for the update. Which method to use is very much dependent on the problem.

6.3 Leap-Frog Method

The next method we will consider is the Leap-Frog method so called because to get from time step $n-1$ to $n+1$, one leaps over time n. The scheme is

$$\begin{aligned}
D^0 u_j^n &= D_0 u_j^n \\
\frac{1}{2k}(u_j^{n+1} - u_j^{n-1}) &= \frac{1}{2h}(u_{j+1}^n - u_{j-1}^n) \\
u_j^{n+1} &= u_j^{n-1} + \frac{k}{h}(u_{j+1}^n - u_{j-1}^n) \\
&= u_j^{n-1} + \lambda(u_{j+1}^n - u_{j-1}^n)
\end{aligned}$$

Von Neuman analysis then proceeds as before.

$$\begin{aligned}
z^{n+1}e^{ij\xi} &= z^{n-1}e^{ij\xi} + \lambda(z^n e^{i(j+1)\xi} - z^n e^{i(j-1)\xi}) \\
z &= \frac{1}{z} + \lambda(e^{i\xi} - e^{-i\xi}) \\
0 &= z^2 - z\lambda i 2\sin(\xi) - 1 \\
z &= \frac{i2\lambda\sin(\xi) \pm \sqrt{-4\lambda^2\sin^2(\xi) + 4}}{2} \\
&= i\lambda\sin(\xi) \pm \sqrt{1 - \lambda^2\sin^2(\xi)} \\
|z|^2 &= 1 - \lambda^2\sin^2(\xi) + \lambda^2\sin^2(\xi) \\
&= 1
\end{aligned}$$

provided that $1 - \lambda^2\sin^2(\xi) \geq 0$, or $\lambda \leq 1$.

This method is not very robust, its error is purely dispersive, $\tau = O(k^2 + h^2)$. Note also that z has two roots $\approx \pm 1$. The $z = -1$ mode is not physical and represents the fact that the even and odd time steps are decoupled. With no damping, the $z = -1$ mode will persist.

The decoupling can be visualized by looking at the figure below where the dependency of data is illustrated. The Leap-Frog scheme leads to the fact that the grid points with squares depend only on the previous squares and never on grid points with circles, and *vice versa*. Note that the coupling of the two sets of data is *only* through the initial data. It is this decoupling which causes oscillations to propogate and persist in computations.

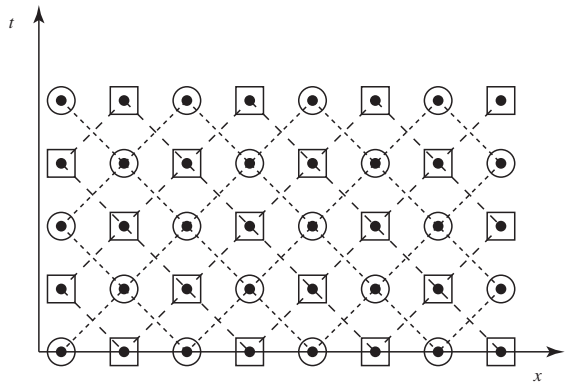


Illustration of decoupling by Leap-Frog $(2, 2)$.

The Leap-Frog method can be extended to higher order in space by using higher order approximations on the right hand side. The Leap-Frog $(2, 4)$ method is

$$u_j^{n+1} = u_j^{n-1} + \lambda\left(\frac{4}{3}(u_{j+1}^n - u_{j-1}^n) - \frac{1}{6}(u_{j+2}^n - u_{j-2}^n)\right)$$

If we apply von Neuman analysis to this method we get

$$\begin{aligned}
z^2 &= 1 + \lambda z \left(\frac{4}{3}(e^{i\xi} - e^{-i\xi}) - \frac{1}{6}(e^{i2\xi} - e^{-i2\xi}) \right) \\
z^2 &= 1 + 2i\lambda z \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right) \\
0 &= z^2 - 2i\lambda \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right) z - 1 \\
z &= \frac{2i\lambda \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right) \pm \sqrt{4 - 4\lambda^2 \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right)^2}}{2} \\
&= i\lambda \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right) \pm \sqrt{1 - \lambda^2 \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right)^2} \\
|z|^2 &= \lambda^2 \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right)^2 + 1 - \lambda^2 \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right)^2 \\
&= 1
\end{aligned}$$

provided $1 - \lambda^2 \left(\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi) \right)^2 \geq 0$. This happens when

$$\lambda \leq \frac{1}{\frac{4}{3}\sin(\xi) - \frac{1}{6}\sin(2\xi)} < 0.728$$

Again, this method has purely dispersive error and the odd/even steps still decouple.

Note that if $k = \theta h$ for some θ, then the truncation error, $\tau = O(k^2) + O(h^4) = O(h^2 + h^4) = O(h^2)$. Thus, if the time step k is taken to be on the order of h, then in fact, the fourth order accuracy in space is drown out by the second order accuracy in time. For that reason, Leap-Frog (2,4) is better when smaller time steps are taken. This is in contrast to Leap-Frog (2,2) where times steps closer to the stability limit is better.

There are three things about Leap-Frog in the present state that need to be discussed: how to start, how to recouple the odd/even steps, and how to damp some of the high order oscillations due to dispersion.

The starting procedure is not difficult, but does need to be done with some care. Errors in the starting procedure will not be damped because of the dispersive error. The simplest starting procedure is to use a one-step method for the first step. A better startup is by bootstrapping from a very small first one-step method. For example, use a one-step method to go from 0 to $k/8$. Then use Leap-Frog to go from 0 to $k/4$, then $k/2$ using 0 and $k/4$, and finally out to k by using 0 and $k/2$. From there, it can be run at a uniform time step. This idea is illustrated below:

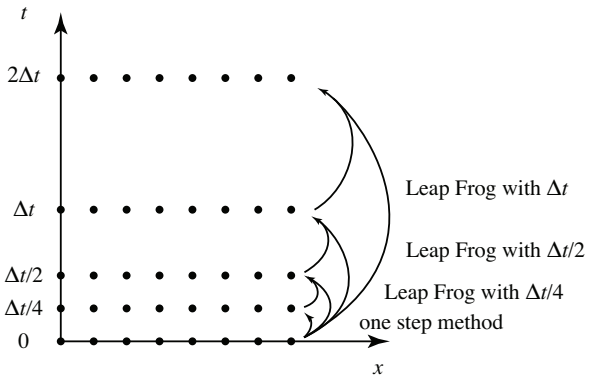


Illustration of Bootstrap technique for startup

The next problem is how to recouple the odd/even time steps. There are a couple different approaches to this problem. One way to recouple the alternate time levels is to use a one-step method at regular intervals. The frequency of the one-step applications is problem dependent where they are frequent enough to prevent decoupling and not so often as to compromise the second order in time accuracy. Second order accuracy in time becomes a problem if the number of times the one-step method is applied scales with the number of time steps.

An alternative method of recoupling is to periodically average in time. Suppose that the computation has proceeded to time step $n+1$. Define $u_j^{n+1/2} = (u_j^n + u_j^{n+1})/2$ and $u_j^{n-1/2} = (u_j^n + u_j^{n-1})/2$. Now compute on the half-steps until the next averaging puts the computation back on the whole-steps.

Finally, we need to discuss what to do with methods which have purely dispersive error and how to damp high oscillation errors in a computation. To analyze the addition of dissipative errors, let us model dissipation by the equation

$$y' = -ay$$

If we take a straightforward Leap-Frog approach, we get

$$\frac{1}{2k}(u^{n+1} - u^{n-1}) = -au^n$$

Assume $u^n = z^n u^0$, then we get

$$\begin{aligned} z^2 - 1 &= -2kaz \\ z^2 + 2kaz - 1 &= 0 \\ z = \frac{-2ka \pm \sqrt{4 + 4k^2a^2}}{2} &= -ka \pm \sqrt{1 + a^2k^2} \end{aligned}$$

Let $z_1 = \sqrt{1 + a^2k^2} - ak$, and $z_2 = -\sqrt{1 + a^2k^2} - ak$. Then

$$\begin{aligned} z_1 = \sqrt{1 + a^2k^2} - ak &< \sqrt{1 + 2ak + a^2k^2} - ak \\ &= \sqrt{(1 + ak)^2} - ak = 1 + ak - ak = 1 \end{aligned}$$

On the other hand,

$$|z_2| = ak + \sqrt{1 + a^2k^2} > 1 + ak$$

So z_2 represents an exponentially growing solution. While this is stable growth, it is a growth that is not desirable.

There are two remedies for this problem. We can either lag the dissipation or average in time. A lag in dissipation would look like

$$\begin{aligned} \frac{1}{2k}(u^{n+1} - u^{n-1}) &= -au^{n-1} \\ z^2 - 1 &= -2ka \\ z^2 &= 1 - 2ka \end{aligned}$$

Thus, $|z| = \sqrt{1 - 2ka} < 1$. It has no growth modes.

Averaging in time gives

$$\begin{aligned} \frac{1}{2k}(u^{n+1} - u^{n-1}) &= -a(u^{n+1} + u^{n-1})/2 \\ z^2 - 1 &= -ak(z^2 + 1) \\ (1 + ak)z^2 &= 1 - ak \\ z^2 = \frac{1 - ak}{1 + ak} &< 1 \end{aligned}$$

Carrying this over to the Leap-Frog (2,2) method, we modify $u_t = cu_x$ by

$$u_t = cu_x - \epsilon h^4 u_{xxxx}(x, t-k).$$

For the Leap-Frog (2,4) method the equation becomes

$$u_t = cu_x + \epsilon h^6 u_{xxxxxx}(x, t-k)$$

where the new spatial derivatives are computed using central difference approximations and ϵ is optimized by experiment. Note also that the lag in time simply means that the dissipation terms are computed at time $n-1$ instead of n.

6.4 Lax-Wendroff Method

The next method we will consider will have second order accuracy in time, but also has some built-in higher order diffusion terms. If we take a higher order Taylor expansion of $u(x, t+k)$, we get

$$u(x, t+k) = u + ku_t + \frac{k^2}{2}u_{tt} + O(k^3)$$

Now, recall that $u_t = cu_x$, and hence $u_{tt} = c^2 u_{xx}$, so

$$u(x, t+k) = u + kcu_x + \frac{k^2c^2}{2}u_{xx} + O(k^3)$$

If we take central differences we get

$$u_j^{n+1} = u_j^n + \frac{kc}{2h}(u_{j+1}^n - u_{j-1}^n) + \frac{k^2c^2}{2h^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$$

Taking the Taylor expansion shows that the truncation error is

$$\tau = \frac{k^2}{6}u_{ttt} + \frac{h^2c}{6}u_{xxx} + \frac{k^3}{24}u_{tttt} + \frac{kh^2}{24}u_{xxxx}.$$

This shows that while the leading order error is still dispersive, there is a higher order dissipative term that is present in the system.

Von Neuman analysis gives

$$\begin{aligned}
z &= 1 + \frac{\lambda}{2}(e^{i\xi} - e^{-i\xi}) + \frac{\lambda^2}{2}(e^{i\xi} - 2 + e^{-i\xi}) \\
&= 1 + i\lambda\sin(\xi) + \lambda^2(\cos(\xi) - 1) \\
&= 1 - 2\lambda^2\sin^2(\xi/2) + i\lambda\sin(\xi) \\
|z|^2 &= (1 - 2\lambda^2\sin^2(\xi/2))^2 + \lambda^2\sin^2(\xi) \\
&= 1 - 4\lambda^2\sin^2(\xi/2) + 4\lambda^4\sin^4(\xi/2) + 4\lambda^2\sin^2(\xi/2)\cos^2(\xi/2) \\
&= 1 - 4\lambda^2\sin^2(\xi/2) + 4\lambda^4\sin^4(\xi/2) + 4\lambda^2\sin^2(\xi/2)(1 - \sin^2(\xi/2)) \\
&= 1 + 4\lambda^4\sin^4(\xi/2) - 4\lambda^2\sin^4(\xi/2) \\
&= 1 + 4\lambda^2(\lambda^2 - 1)\sin^4(\xi/2)
\end{aligned}$$

Therefore, if $\lambda \leq 1$, then for every mode $-\pi \leq \xi \leq \pi$, $\xi \neq 0$, $|z| < 1$. This means that Lax-Wendroff is dissipative for all modes except $\xi = 0$.

The next thing to note is that for fixed λ, $|z| \leq 1 - \delta\xi^4$ for δ depending only upon λ. This indicates that the Lax-Wendroff method has fourth order dissipation.

6.5 MacCormack's Method

MacCormack's method is a variation of Lax-Wendroff which is easier to adapt to more complicated equations. Lax-Wendroff is broken into a predictor corrector pair. Thus,

$$\hat{u}_j = u_j^n + \lambda D_+ u_j^n \tag{F}$$

$$u_j^{n+1} = \frac{1}{2}(\hat{u}_j + u_j^n + \lambda D_- \hat{u}_j) \tag{B}$$

This is called the FB step where F stands for forward differencing and B stands for backward differencing. If we interchange the D_+ and D_- then we would have a BF step. It is easy to show that for $u_t = cu_x$ this method and Lax-Wendroff are the same. The advantage for this method comes when it is applied to more complicated equations. For example, consider the hyperbolic conservation law

$$u_t = [f(u)]_x$$

To compute u_{tt} as in Lax Wendroff would result in a complicated nonlinear expression. MacCormack's scheme makes it easier:

$$\hat{u}_j = u_j^n + \lambda D_+ f(u_j^n) \tag{F}$$

$$u_j^{n+1} = \frac{1}{2}(\hat{u}_j + u_j^n + \lambda D_- f(\hat{u}_j)) \tag{B}$$

In practice, FB and BF steps are used alternating every time step. A stability bound for this method applied to conservation laws is similar to that for Lax-Wendroff, namely

$$k \leq \frac{h}{\max_j |f'(u_j^n)|}$$

There are extensions of MacCormack's method to obtain (2,4) and (2,6) versions. In this case alternation of FB and BF is required and these steps are modified from the above description. In each case, the time step must be taken much smaller to reap the benefits of the higher spatial accuracy similar to what happened with Leap-Frog.

6.6 Runge-Kutta Time Stepping

We can also use Runge-Kutta methods for advancing in time. The fourth order Runge-Kutta method is

$$K_1 = kf(u^n, t_n), \quad \hat{u}_1 = u^n + \frac{K_1}{2}$$

$$K_2 = kf\left(\hat{u}_1, t_n + \frac{k}{2}\right), \quad \hat{u}_2 = u^n + \frac{K_2}{2}$$

$$K_3 = kf\left(\hat{u}_2, t_n + \frac{k}{2}\right), \quad \hat{u}_3 = u^n + K_3$$

$$K_4 = kf(\hat{u}_3, t_n + k), \quad u^{n+1} = u^n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

Let us analyze this for its growth modes. We will solve the equation

$$\frac{du}{dt} = au, \quad u(0) = u^0$$

and assume that $u^n = z^n u^0$. Then,

$$\begin{aligned}
K_1 &= kaz^n u^0 \\
\hat{u}_1 &= z^n u^0 + \frac{kaz^n u^0}{2} \\
K_2 &= ka\left(z^n u^0 + \frac{kaz^n u^0}{2}\right) \\
\hat{u}_2 &= z^n u^0 + \frac{1}{2}\left(ka + \frac{(ka)^2}{2}\right) z^n u^0 \\
&= \left(1 + \frac{1}{2}ka + \frac{1}{4}(ka)^2\right) z^n u^0 \\
K_3 &= ka\left(1 + \frac{1}{2}ka + \frac{1}{4}(ka)^2\right) z^n u^0 \\
\hat{u}_3 &= z^n u^0 + ka\left(1 + \frac{1}{2}ka + \frac{1}{4}(ka)^2\right) z^n u^0 \\
&= \left(1 + ka + \frac{1}{2}(ka)^2 + \frac{1}{4}(ka)^3\right) z^n u^0 \\
K_4 &= ka\left(1 + ka + \frac{1}{2}(ka)^2 + \frac{1}{4}(ka)^3\right) z^n u^0 \\
u^{n+1} &= z^n u^0 + \frac{1}{6}\Bigg(ka + 2ka + (ka)^2 + 2ka + (ka)^2 + \frac{1}{2}(ka)^3 + ka + (ka)^2 \\
&\qquad + \frac{1}{2}(ka)^3 + \frac{1}{4}(ka)^4\Bigg) z^n u^0
\end{aligned}$$

and since $u^{n+1} = z^{n+1} u^0$, we get

$$z = 1 + (ka) + \frac{1}{2}(ka)^2 + \frac{1}{6}(ka)^3 + \frac{1}{24}(ka)^4$$

Of course, we should have been able to predict this result since we know that this method is fourth order, and that the exact solution would be e^{ak}. For purposes of analyzing the wave equation, we now suppose that $a = i\alpha$, then we get

$$\begin{aligned}
z &= 1 + ik\alpha - k^2\alpha^2 - ik^3\alpha^3 + k^4\alpha^4 \\
&= 1 - k^2\alpha^2 + k^4\alpha^4 + i(k\alpha - k^3\alpha^3)
\end{aligned}$$

From this we find that $|z| \leq 1$ provided $|ka| \leq 2.8$. Let D_4 be the fourth order central difference scheme, and let $U^n = \begin{bmatrix} \vdots \\ u_j^n \\ \vdots \end{bmatrix}$, then we apply Runge-Kutta to solve

$$\frac{dU^n}{dt} = D_4 U^n. \tag{5}$$

Writing it this way will assist with the von Neuman analysis.

Assume $U^n = a(t)e^{i\ell hj}U^0$. Plugging this into (5), we get

$$\begin{aligned}
\frac{da}{dt} &= i\left(\frac{4}{3}\frac{\sin(\xi)}{h} - \frac{1}{3}\frac{\sin(2\xi)}{2h}\right) \\
&= i\frac{G(\xi)}{h}
\end{aligned}$$

From our earlier analysis, we know that we get stability if $|k\frac{G(\xi)}{h}| \leq 2.8$ or

$$k \leq 2.8\frac{h}{|G(\xi)}| \leq 2.8\frac{h}{\max_{-\pi\leq\xi\leq\pi}|G(\xi)|}$$

We came across $G(\xi)$ before with Leap-Frog (2,4) and so we get

$$k \leq h(2.8)(.728)$$

Note that using Runge-Kutta in time means more intermediate storage. A low storage version for time independent equations is given for the equation $\frac{du}{dt} = f(u)$:

$$\begin{aligned} K_1 &= kf(u^n), \quad \hat{u}_1 = u^n + \frac{K_1}{4} \\ K_2 &= kf(\hat{u}_1), \quad \hat{u}_2 = u^n + \frac{K_2}{3} \\ K_3 &= kf(\hat{u}_2), \quad \hat{u}_3 = u^n + \frac{K_3}{2} \\ K_4 &= kf(\hat{u}_3), \quad u^{n+1} = u^n + K_4 \end{aligned}$$

This method has the same stability criterion as the original Runge-Kutta fourth order method.

This is then a (4,4) method. It has dispersive error, so artificial dissipation is often added similar to Leap-Frog, but for this method the time lagging of the dissipation is not needed.

7 Systems of Equations

Every one of the methods we have can also be applied to systems of equations such as

$$\vec{u}_t = A(\vec{u}, x, t)\vec{u}_x, \quad \vec{u}(x, 0) = \vec{u}_0(x)$$

where $\vec{u}$ is an m-vector and A is an $m \times m$ matrix. Recall that the requirement for hyperbolicity is that A have m distinct real eigenvalues, call them $\mu_1,\ldots,\mu_m$. These eigenvalues are called the *characteristic speeds* of the equation.

The von Neuman analysis for methods also still works for systems. For example, Leap-Frog (2,2) is

$$\vec{u}_j^{n+1} = \vec{u}_j^{n-1} + \lambda A(\vec{u}_{j+1}^n - \vec{u}_{j-1}^n)$$

Now assume $\vec{u}_j^n = z^n e^{ij\ell h}\vec{u}^0$, then

$$\begin{aligned} z^2\vec{u}_0 - \vec{u}_0 - \lambda Az(e^{i\xi} - e^{-i\xi})\vec{u}^0 = 0 \\ (Iz^2 - \lambda 2iAz\sin(\xi) - I)\vec{u}^0 = 0 \end{aligned}$$

Now $A = T\Lambda T^{-1}$ where Λ is a diagonal matrix with diagonal entries $\mu_1, \ldots, \mu_m$, so we can write

$$\begin{aligned} T(Iz^2 - \lambda 2i\Lambda z\sin(\xi) - I)T^{-1}\vec{u}^0 = 0 \\ (Iz^2 - 2\lambda i\Lambda z\sin(\xi) - I)T^{-1}\vec{u}^0 = 0 \end{aligned}$$

since T has a trivial null space, then for this equation to hold, we must have at least one of the equations

$$z^2 - 2\lambda i\mu_k z\sin(\xi) - 1 = 0.$$

This is the same equation we saw for Leap-Frog where we had the stability limit $\lambda \leq \frac{1}{|\mu_k|}$. This must hold for any $\vec{u}$; and hence we get the stability restriction of

$$\lambda \leq \frac{1}{\max_{k=1,\ldots,m}|\mu_k|}.$$

This argument will apply to any of the methods we have analyzed so far.

This method of decoupling will also be useful for when we deal with boundary conditions, so we will discuss it further. If we have the original equation

$$\vec{u} = A\vec{u}_x$$

Then,

$$\begin{aligned} \vec{u}_t &= T\Lambda T^{-1}\vec{u}_x \\ T^{-1}\vec{u}_t &= \Lambda T^{-1}\vec{u}_x \end{aligned}$$

Let $\vec{w} = T^{-1}\vec{u}$, then we have the equation

$$\vec{w}_t = \Lambda\vec{w}_x$$

This is the decoupled equation which can now be solved as a list of scalar equations

$$\frac{\partial w_i}{\partial t} = \mu_i \frac{\partial w_i}{\partial x}.$$

These equations will be recoupled via the boundary conditions as we will see later.

8 Implicit Methods

Implicit methods are one way to overcome the strict time-step requirements imposed by the CFL condition. The simplest of these methods is Backward Euler.

8.1 Backward Euler

Next, let us analyze the *implicit* one-step method:

$$\begin{aligned} D^- u_j^n &= cD_0 u_j^n \\ \frac{1}{k}(u_j^n - u_j^{n-1}) &= \frac{c}{2h}(u_{j+1}^n - u_{j-1}^n) \\ u_j^n &= u_j^{n-1} + \frac{ck}{2h}(u_{j+1}^n - u_{j-1}^n) \\ \frac{\lambda}{2}u_{j-1}^n + u_j^n - \frac{\lambda}{2}u_{j+1}^n &= u_j^{n-1} \end{aligned}$$

To solve this, we would write this as a matrix equation. Let

$$U^n = \begin{bmatrix} \vdots \\ u_j^n \\ \vdots \end{bmatrix}, \text{ and } A = \begin{bmatrix} \ddots & \ddots & & & 0 \\ \ddots & \ddots & \ddots & & \\ & \frac{\lambda}{2} & 1 & \frac{-\lambda}{2} & \\ & & \ddots & \ddots & \ddots \\ 0 & & & \ddots & \ddots \end{bmatrix}.$$

Then this equation becomes $AU^n = U^{n-1}$. Thus, $U^n = A^{-1}U^{n-1}$.

Again, let us apply von Neuman analysis to this method.

$$\begin{aligned}
\frac{\lambda}{2}z^n e^{i(j-1)\xi} + z^n e^{ij\xi} - \frac{\lambda}{2}z^n e^{i(j+1)\xi} &= z^{n-1}e^{ij\xi} \\
\frac{\lambda}{2}ze^{-i\xi} + z - \frac{\lambda}{2}ze^{i\xi} &= 1 \\
z\left(\frac{\lambda}{2}e^{-i\xi} + 1 - \frac{\lambda}{2}e^{i\xi}\right) &= 1 \\
z(1 - i\lambda\sin(\xi)) &= 1 \\
z &= \frac{1}{1 - i\lambda\sin(\xi)} \\
&= \frac{1}{1 - i\lambda\sin(\xi)}\frac{1 + i\lambda\sin(\xi)}{1 + i\lambda\sin(\xi)} \\
&= \frac{1 + i\lambda\sin(\xi)}{1 + \lambda^2\sin^2(\xi)} \\
|z|^2 &= \left(\frac{1}{1 + \lambda^2\sin^2(\xi)}\right)^2 + \left(\frac{\lambda\sin(\xi)}{1 + \lambda^2\sin^2(\xi)}\right)^2 \\
&= \frac{1 + \lambda^2\sin^2(\xi)}{(1 + \lambda^2\sin^2(\xi))^2} \\
&= \frac{1}{1 + \lambda^2\sin^2(\xi)} \\
&\leq 1
\end{aligned}$$

Therefore, this method is unconditionally stable. Note also that except for certain modes, $|z| < 1$ which means that almost all modes are damped in time.

8.2 Crank-Nicolson Method

The Crank-Nicolson method is

$$D^+u_j^n = \frac{1}{2}(D_0u_j^{n+1} + D_0u_j^n).$$

The truncation error is

$$\tau = \left(\frac{h^2}{6} + \frac{5k^2}{24}\right)u_{xxx} = O(h^2) + O(k^2)$$

so it is a (2,2) scheme with pure dispersive error.

Next, we will do the von Neuman analysis:

$$\begin{aligned}
\frac{1}{k}(z-1) &= \frac{1}{2}\left(\frac{1}{2h}z(e^{i\xi} - e^{-i\xi}) + \frac{1}{2h}(e^{i\xi} - e^{-i\xi})\right) \\
z - 1 &= \frac{i}{2}\lambda(z+1)\sin(\xi) \\
z\left(1 - \frac{i}{2}\lambda\sin(\xi)\right) &= 1 + \frac{i}{2}\lambda\sin(\xi) \\
|z|^2\left(1 + \frac{\lambda^2}{4}\sin^2(\xi)\right) &= 1 + \frac{\lambda^2}{4}\sin^2(\xi) \\
|z|^2 &= 1
\end{aligned}$$

Thus, it is unconditionally stable.

The implementation of Crank-Nicolson is similar to the Backward Euler method where a matrix equation must be solved every time step and that matrix is also tridiagonal.

This and other implicit schemes are often rewritten in a "δ-formulation" where

$$\delta_j^n = u_j^{n+1} - u_j^n$$

The Crank-Nicolson method then becomes

$$u_j^{n+1} - \frac{\lambda}{4}(u_{j+1}^{n+1} - u_{j-1}^{n+1}) = u_j^n + \frac{\lambda}{4}(u_{j+1}^n - u_{j-1}^n)$$
$$\delta_j^n - \frac{\lambda}{4}(\delta_{j+1}^n - \delta_{j-1}^n) = \frac{\lambda}{2}(u_{j+1}^n - u_{j-1}^n)$$

The method is then a two step process

$$\delta_j^n - \frac{\lambda}{4}(\delta_{j+1}^n - \delta_{j-1}^n) = \frac{\lambda}{2}(u_{j+1}^n - u_{j-1}^n)$$
$$u_j^{n+1} = u_j^n + \delta_j^n$$

While theoretically this is entirely equivalent, it is less prone to catastrophic roundoff errors.

Another thing to note is that since Crank-Nicolson has purely dispersive error, it is prone to oscillations and non-linear instability. One way to combat this is to make a slight change in the method

$$D^+ u_j^n = \alpha D_0 u_j^{n+1} + (1-\alpha) D_0 u_j^n$$

For $\alpha = 1$, it is the backward Euler method and for $\alpha = 1/2$, it is Crank-Nicolson. The method is second order in time only for $\alpha = 1/2$, but a formally second order in time method can be recovered by choosing $\alpha = \frac{1}{2} + \alpha_1 k$. In practice, α is simply a parameter to be set when the method is invoked.

8.3 Compact 4^{th} Order Approximation for u_x

Next up is a fourth order in space method that requires only three points in space. Recall that

$$D_0 u_j = u_x + \frac{h^2}{6} u_{xxx} + O(h^4)$$
$$D_+ D_- u_j = u_{xx} + O(h^2)$$

Combining these, we get $u_{xxx} = D_+ D_- u_x + O(h^2)$, so

$$\begin{aligned} D_0 u_j &= u_x + \frac{h^2}{6} D_+ D_- u_x + O(h^4) \\ &= \left(I + \frac{h^2}{6} D_+ D_- \right) u_x + O(h^4) \end{aligned}$$

and hence

$$u_x = \left(I + \frac{h^2}{6} D_+ D_- \right)^{-1} D_0 u + O(h^4)$$

this is a fourth order accurate implicit approximation of U_x where this is only a tridiagonal matrix to be inverted.

The method is typically coupled with Runge-Kutta 4 to obtain a (4,4) method with a compact stencil. Also, note that this method has a time step restriction even though it is an implicit method. That is, because it is not implicit in time.

9 Semi-implicit Schemes

Consider the equation

$$u_t = u_x + R(u)$$

where R is some given nonlinear expression of u. To solve this with a fully implicit method will involve a nonlinear solve of the equation every time step. This can be expensive if it is even possible. An equation of the form (9) commonly arises in convection-reaction equations in chemistry where $R(u)$ models the chemical reaction.

A semi-implicit method is a method where part of the equation is modeled with an implicit method while another part is pure explicit. For example, we can apply a combination of Crank-Nicolson with the explicit multi-step method Adams-Bashforth.

9.1 Adams-Bashforth Multi-step Method

The Adams-Bashforth method is a family of multi-step methods for solving ordinary differential equations. The second order Adams-Bashforth method for solving $y' = f(y,t)$ is

$$y_{n+1} = y_n + \frac{k}{2}(3f(y_n,t_n) - f(y_n,t_n))$$

So if we were simply solving $u_t = R(u)$, we would have

$$u_j^{n+1} = u_j^n + \frac{k}{2}(3R(u_j^n) - R(u_j^{n-1}))$$

Combine this with Crank-Nicolson for the convection part to get

$$D^+u_j^n = \frac{1}{2}(D_0u_j^{n+1} + D_0u_j^n) + \frac{3}{2}R(u_j^n) - \frac{1}{2}R(U_j^{n-1})$$

Thus, we still have a (2,2) method. So we still have a tridiagonal linear system for the implicit step. Also, we only have to solve that system once each time step (as opposed to more if we had used a Runge-Kutta type of method for the non-linear part or MacCormack's method.)

The δ-formulation can also be used in this type of method as well.

10 Parabolic Equations

The canonical parabolic equation is the heat equation given by

$$u_t = au_{xx}, \quad a > 0$$

Many of the schemes we have already discussed can be applied to this equation except for Leap-Frog. Recall that Leap-Fog is unstable for even derivatives evaluated at time n. We had to lag in time for the artificial dissipation.

The time step restriction is more severe for the heat equation. To see this, consider the basic (1,2) method

$$D^+u_j^n = D_+D_-u_j^n$$

which is

$$u_j^{n+1} = u_j^n + a\frac{k}{h^2}(u_{j+1}^n - 2u_j^n + u_{j=1}^n)$$

Now let us apply von Neuman analysis

$$z = 1 + \beta(e^{i\xi} - 2 + e^{-i\xi})$$

where $\beta = a\frac{k}{h^2}$

$$\begin{aligned} z &= 1 + \beta(2\cos(\xi) - 2) \\ &= 1 - 2\beta(1 - \cos(\xi)) \end{aligned}$$

To get stability, we want

$$-1 \leq 1 - 2\beta(1 - \cos(\xi)) \leq 1$$

Clearly, the second inequality is always satisfied. For the first inequality, we get

$$\begin{aligned} -1 &\leq 1 - 2\beta(1 - \cos(\xi)) \\ 1 &\geq \beta(1 - \cos(\xi)) \\ \beta &\leq \frac{1}{1 - \cos(\xi)}. \end{aligned}$$

Since this must hold for all ξ, then we must have

$$\beta \leq \min_{\xi} \frac{1}{1 - \cos(\xi)} = \frac{1}{2}.$$

Therefore, we have a time step restriction of the form

$$k \leq \frac{h^2}{2a}$$

This is a significantly more restrictive time step requirement because every time the space step is halved, the time step must be cut by a quarter.

This type of restriction on parabolic equations is typical of explicit methods. If we instead switch to a backward Euler implicit method, we get

$$\begin{aligned} D^- u_j^n &= aD_+D_- u_j^n \\ \frac{1}{k}(u_j^n - u_j^{n-1}) &= a\frac{1}{h^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) \\ (1 + 2\beta)u_j^n - \beta u_{j+1}^n - \beta u_{j-1}^n &= u_j^{n-1} \end{aligned}$$

The matrix on the left is a diagonally dominant matrix which means that it is better conditioned, and hence more accurate solutions are obtained when solving the linear system.

Applying the von Neuman analysis, we get

$$\begin{aligned} 1 &= (1 + 2\beta)z - \beta e^{i\xi} z - \beta e^{-i\xi} z \\ &= (1 + 2\beta - 2\beta\cos(\xi))z \\ z &= \frac{1}{1 + 2\beta(1 - \cos(\xi))} \leq 1 \end{aligned}$$

for all ξ. So the method is unconditionally stable as expected.

Now consider a reaction-convection-diffusion equation

$$u_t = u_x + u_{xx} + R(u)$$

to avoid the time step restriction, we apply an implicit method to the diffusion term, either implicit or explicit for the convection term, and explicit for the non-linear term.

Unconditional stability when solving $u_t = au_{xx}$ does not mean that the time step size k can be taken arbitrarily large. For example, consider the Dufort-Frankel method (a modification of Leap-Frog) for solving

$u_t = u_{xx}$, which is unconditionally stable according the the von Neuman analysis.

$$\begin{aligned}
\frac{u_j^{n+1} - u_j^{n-1}}{2k} &= \frac{1}{h^2}(u_{j+1}^n - u_j^{n+1} - u_j^{n-1} + u_{j-1}^n) \\
\frac{u_j^{n+1} - u_j^{n-1}}{2k} &= \frac{1}{h^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) - \frac{1}{h^2}(u_j^{n+1} - 2u_j^n + u_j^{n-1}) \\
\frac{u_j^{n+1} - u_j^{n-1}}{2k} &= \frac{1}{h^2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) - \frac{k^2}{h^2}\frac{u_j^{n+1} - 2u_j^n + u_j^{n+1}}{k^2}
\end{aligned}$$

Now, suppose we keep $\lambda = \frac{k}{h}$ constant and let k, $h \to 0$. Then we get

$$u_t = u_{xx} - \lambda^2 u_{tt}$$

which is the wrong equation. On the other hand, let $\beta = \frac{k}{h^2}$ be held constant, then we get as k, $h \to 0$,

$$u_t = u_{xx} - k\beta u_{tt} \to u_{xx}.$$

The moral of the story is that stability and consistency is not always enough to show convergence in the world of partial differential equations. A corollary could be, just because a computed solution is stable and converges as h, $k \to 0$ does not guarantee that the limiting solution solves the original equation.

11 Boundary Conditions for the Heat Equation

Consider the initial boundary value problem

$$u_t = u_{xx}, \quad 0 \le x \le 1$$

The heat equation requires two boundary conditions, typically one boundary condition at each endpoint. The boundary conditions can be represented by

$$\begin{aligned}
\alpha u(0,t) + \beta u_x(0,t) &= f_0(t) \\
\gamma u(1,t) + \delta u_x(1,t) &= f_1(t)
\end{aligned}$$

These are called the impedance boundary conditions. If β or $\delta = 0$, it is called a Dirichlet boundary condition. If γ or $\alpha = 0$, then it is called a Neuman boundary condition.

Modeling these numerically is trivial. Pure Dirichlet conditions simply set the value at the endpoints. For mixed boundary conditions, a ghost point is often used. For implicit methods, this simply adds an additional equation to be solved at each endpoint. For explicit methods, the ghost point is determined from the already computed points in the interior. For example, at the left end point (where $u_0^n = u(0, t_n)$) we would have

$$\begin{aligned}
\alpha u_0^{n+1} + \beta\frac{1}{2h}(u_1^{n+1} - u_{-1}^{n+1}) &= f_0(t_{n+1}) \\
\Rightarrow \quad u_{-1}^{n+1} &= \frac{2h}{\beta}(f_0(t_{n+1}) - \alpha u_0^{n+1}) - u_1^{n+1}
\end{aligned}$$

For the special case where $\alpha = 0$, often the grid points are chosen to straddle the endpoint so that $x_{-1} = -h/2$ and $x_0 = h/2$. In this case, we get

$$\frac{1}{h}(u_1^{n+1} - u_{-1}^{n+1}) = f_0(t_{n+1})$$

This reduces the error in approximating u_x at the endpoint by half.

12 Boundary Conditions for Hyperbolic Problems

The approximation of boundary conditions is as important as the approximations of the equation itself. Implemented poorly and they will pollute your solution and reduce accuracy and even convergence rates and stability.

To treat boundary conditions right, we need to understand the effect of boundary conditions on the exact solutions. Consider the equation

$$u_t = u_x, \quad -\infty < x \leq 0$$
$$u(x,0) = f(x)$$

This is not enough information to solve the problem in the region $(\infty, 0] \times [0, T]$. To see why, we return to the characteristics of the problem. In the figure above, consider the point A. In order to determine the value of u we follow the characteristic back until it hits the boundary at $x = 0$. As stated, there is insufficient information to determine the value of u at A. To complete the picture, we must know the value of u on the $x = 0$ line, i.e.

$$u(0,t) = g(t)$$

So the complete solution is

$$u(x,t) = \begin{cases} f(x+t) & x \geq t \\ g(x+t) & x < t \end{cases}$$

Notice that in order for the solution to be continuous, we must also have the additional compatibility condition $g(0) = f(0)$.

On the flip side, suppose we wish to solve the same equations on the interval $0 \leq x < +\infty$. This time, we cannot impose a boundary condition because the value of u is already determined. We know that for $\epsilon > 0$, $u(\epsilon, t) = f(\epsilon + t)$ and as $\epsilon \to 0$, $u(\epsilon, t) \to f(t)$, so for continuity at $x = 0$, we must have $u(0,t) = f(t)$ and hence $u(0,t)$ is completely determined.

12.1 Systems of Equations

Next, consider the second order wave equation

$$u_{tt} = u_{xx}, \quad 0 \leq x < +\infty$$

We can rewrite this as a system, let $v_1 = u_t$ and $v_2 = u_x$, then

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}_t = \begin{bmatrix} u_t \\ u_x \end{bmatrix}_t = \begin{bmatrix} u_{tt} \\ u_{xt} \end{bmatrix} = \begin{bmatrix} u_{xx} \\ u_{tx} \end{bmatrix} = \begin{bmatrix} u_x \\ u_t \end{bmatrix}_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u_t \\ u_x \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

So we are reduced to the first order system

$$\vec{v}_t = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \vec{v}_x$$

To identify the direction of the waves, we can decouple this system. The matrix has eigenvalues ± 1 and eigenvectors $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ respectively. We can then write

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

and hence

$$\vec{v}_t = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \vec{v}_x$$
$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \vec{v}_t = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \vec{v}_x$$

Let $\vec{w} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \vec{v}$ then we get

$$\vec{w}_t = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \vec{w}_x$$

or

$$\frac{\partial w_1}{\partial t} = \frac{\partial w_1}{\partial x} \tag{6}$$

$$\frac{\partial w_2}{\partial t} = -\frac{\partial w_2}{\partial x} \tag{7}$$

Now, we saw before that we could not specify boundary conditions for equation (6) at $x = 0$. On the other hand, the characteristics for equation (7) go in the opposite direction, and so for this part we need to specify the value at $x = 0$. We call the function w_1 the outgoing wave because the data is leaving the domain. The function w_2 is the incoming wave. The boundary conditions we specify for w_2 must then be proscribed and may depend upon the outgoing data, hence we get

$$w_2(0,t) = \alpha w_1(0,t) + g(t)$$

where $g(t)$ is an arbitrarily specified function and α is an arbitrary constant. In particular, this means that we may specify only one boundary condition at this boundary.

Example 12.1:

Suppose this equation arose as part of a vibrating string simulation. We could tie the end down to a fixed height, say $u(0,t) = 0$. How does this translate into the boundary condition formulation above? We get $u_t(0,t) = 0$, and write the boundary conditions in equation (12.1) can be written in terms of u as

$$u_t(0,t) - u_x(0,t) = \alpha(u_t(0,t) + u_x(0,t)) + g(t)$$

Plugging $u_t(0,t) = 0$ into this equation gives

$$-u_x(0,t) = \alpha u_x(0,t) + g(t)$$

Thus, we get equality if we take $\alpha = -1$ and $g(t) = 0$.

The general case follows the same argument as above. Suppose we have

$$\vec{u_t} = A\vec{u}_x, \quad 0 \leq x < +\infty$$

Suppose that A has eigenvalues $\mu_1,\ldots,\mu_r > 0$, and $\mu_{r+1},\ldots,\mu_m < 0$, then we decompose A as $T\Lambda T^{-1}$ where T is the matrix of eigenvectors $e_1,\ldots,e_m$ respectively. Then $A = T\Lambda T^{-1}$. Let B_O be the first r rows of T^{-1}, and B_I be the last $m - r$ rows of T^{-1}. Then define $\vec{v}_I = B_I\vec{u}$ and $\vec{v}_O = B_O\vec{u}_O$. The boundary conditions are now specified as

$$\vec{v}_I = M\vec{v}_O + \vec{g}(t)$$

where M is an $m - r \times r$ matrix. In particular, it tells how many boundary conditions are needed $(m - r)$.

If the domain is bounded on the right, then we would get the complement of boundary conditions at the right endpoint.

Note that the matrix A need not be constant. If we are solving on the domain $[a,b]$, the sign of the eigenvalues of A at $A(a,t)$ and $A(b,t)$ determine the number of boundary conditions necessary at each endpoint. If A varies with time, this must be checked and the number of conditions adjusted in time.

What if we don't follow the rules for boundary conditions? Suppose we are solving $u_t = u_x$, $-\infty < x \leq 0$ and assume $u(x,t) = e^{\sigma t}e^{\mu x}$ where $\mu > 0$. Then if we plug this into the equation, we see that $\mu = \sigma$ and $u(x,t)$ is an exponentially growing solution with arbitrarily large rate σ, i.e. the problem is ill-posed and the

solution is unstable. So under specified boundary conditions lead to unstable solutions. On the other hand, over-specifying the boundary conditions give incorrect results and often oscillatory computations.

Example 12.2:

As an example, consider the linearized Euler equations for one-dimensional inviscid flow. Let ρ be the density, u the $x-$velocity, and p the pressure. The Euler equations are

$$\rho_t + u\rho_x + \rho u_x = 0 \tag{8}$$

$$u_t + uu_x + \frac{1}{\rho}p_x = 0 \tag{9}$$

$$p_t + up_x + \gamma p u_x = 0 \tag{10}$$

where γ is the gas constant. These equations are conservation of mass, momentum, and energy respectively.

Suppose we wish to study the disturbance in a constant ambient state as in sound waves. The ambient state is represented as ρ_0, u_0, and p_0, and then we write the variables in terms of the disturbances:

$$\begin{aligned}\rho &= \rho_0 + \rho' \\ u &= u_0 + u' \\ p &= p_0 + p'\end{aligned}$$

where ρ', u', and p' are small relative to the ambient state. Plugging these in, we get

$$\begin{aligned}\rho'_t + (u_0 + u')\rho'_x + (\rho_0 + \rho')u'_x &= 0 \\ \rho'_t + u_0\rho'_x + \rho_0 u'_x &= 0 \qquad (8')\\ u'_t + (u_0 + u')u'_x + \frac{1}{\rho_0 + \rho'}p'_x &= 0 \\ u' + u_0 u'_x + \left(\frac{1}{\rho_0} - \frac{1}{\rho_0^2}\rho'\right)p'_x &= 0 \\ u'_t + u_0 u'_x + \frac{1}{\rho_0}p'_x &= 0 \qquad (9')\\ p'_t + (u_0 + u')p'_x + \gamma(p_0 + p')u'_x &= 0 \\ p'_t + u_0 p'_x + \gamma p_0 u'_x &= 0 \qquad (10')\end{aligned}$$

Written as a system, this becomes

$$\begin{bmatrix}\rho' \\ u' \\ p'\end{bmatrix}_t = \begin{bmatrix}-u_0 & -\rho_0 & 0 \\ 0 & -u_0 & -\frac{1}{\rho_0} \\ 0 & -\gamma p_0 & -u_0\end{bmatrix}\begin{bmatrix}\rho' \\ u' \\ p'\end{bmatrix}_x$$

This system describes the propagation of sound in an ambient medium.

Assume $u_0 \geq 0$ and that we are solving this on the finite interval $a \leq x \leq b$. The left endpoint $x = a$ is called the inflow boundary, and $x = b$ is the outflow boundary. To determine the number of boundary conditions on each end, we must find the eigenvalues of the matrix.

$$\begin{aligned}0 &= \det\begin{bmatrix}-u_0 - \lambda & -\rho_0 & 0 \\ 0 & -u_0 - \lambda & -\frac{1}{\rho_0} \\ 0 & -\gamma p_0 & -u_0 - \lambda\end{bmatrix} \\ &= -(u_0 + \lambda)\left[(u_0 + \lambda)^2 - \gamma p_0 \frac{1}{\rho_0}\right]\end{aligned}$$

Thus, we get the eigenvalues

$$\lambda_1 = -u_0$$
$$\lambda_{2,3} = -u_0 \pm \sqrt{\frac{\gamma p_0}{\rho_0}}$$

The value $c_0 = \sqrt{\frac{\gamma p_0}{\rho_0}}$ is the speed of sound in the ambient medium. Thus, the eigenvalues are $-u_0$, $-u_0 \pm c_0$. If the flow is supersonic, i.e. $u_0 > c_0$, then all the eigenvalues are negative. This means that three boundary conditions are required at the inflow end and none at the outflow end.

For a subsonic flow, $u_0 < c_0$, then there are two negative eigenvalues and one positive one. To see what boundary conditions can be specified, let us work a little harder. The matrix can be factored into

$$\begin{bmatrix} 1 & \frac{1}{c_0} & \frac{1}{c_0} \\ 0 & \frac{1}{\rho_0} & -\frac{1}{\rho_0} \\ 0 & c_0 & c_0 \end{bmatrix} \begin{bmatrix} -u_0 & 0 & 0 \\ 0 & -u_0 - c_0 & 0 \\ 0 & 0 & -u_0 + c_0 \end{bmatrix} \begin{bmatrix} 1 & 0 & -\frac{1}{c_0^2} \\ 0 & \frac{\rho_0}{2} & \frac{1}{2c_0} \\ 0 & -\frac{\rho_0}{2} & \frac{1}{2c_0} \end{bmatrix}$$

Note that for subsonic flow, $-u_0 < 0$ and $-u_0 - c_0 < 0$ while $-u_0 + c_0 > 0$. The characteristic variables are now

$$\begin{bmatrix} 1 & 0 & -\frac{1}{c_0^2} \\ 0 & \frac{\rho_0}{2} & \frac{1}{2c_0} \\ 0 & -\frac{\rho_0}{2} & \frac{1}{2c_0} \end{bmatrix} \begin{bmatrix} \rho' \\ u' \\ p' \end{bmatrix} = \begin{bmatrix} \rho' - \frac{p'}{c_0^2} \\ \frac{\rho_0 u'}{2} + \frac{p'}{2c_0} \\ -\frac{\rho_0 u'}{2} + \frac{p'}{2c_0} \end{bmatrix}$$

Thus, at the left endpoint, we must specify boundary conditions for $c_0^2\rho' - p'$ and for $\rho_0 c_0 u' + p'$. At the right endpoint, we must specify the outflow condition $-c_0\rho_0 u' + p'$.

At the left endpoint, we are free to specify say ρ' and u', but not both u' and p'. At the right endpoint, one choice of boundary conditions is called *non-reflecting* or *radiation* boundary conditions. In this case, we assume no disturbances are entering the problem from far downstream (towards large x). Hence, we specify that the condition at outflow be the same at $+\infty$. For example, we could use the boundary condition $-c_0\rho_0 u' + p' = 0$.

---

13 Numerical Approximation of Boundary Conditions

Implementation of boundary conditions numerically can lead to problems in the whole scheme. For example, consider Leap-Frog applied to $u_t = u_x$, $0 \le x \le 1$. From what we have learned about boundary conditions, we must specify a boundary condition at $x = 1$, but not at $x = 0$. However, a grid point placed at $x = 0$ requires a grid point on the left to compute the Leap-Frog step. For this reason, we will need to study numerical boundary conditions.

This problem can also occur at inflow. Imagine the value is specified at $x = 1$ and we are using Leap-Frog (2,4). Then a grid point outside $x = 0$ will be required, what value should it have?

As with numerical methods in the infinite domain, we must also study the effect of numerical boundary conditions on the accuracy and stability of the whole numerical method with boundary conditions. Fortunately, it can be shown that there are no compound difficulties due to a problem having two boundaries, i.e. each boundary condition can be analyzed separately to give a complete analysis of the whole scheme.

With regard to the accuracy of boundary conditions, there is a general theorem which states that an order (p, q) method for the interior requires only a $(p-1, q-1)$ accurate method at the boundary to maintain (p, q) accuracy overall.

For example, if you are using a second order scheme in space, and need extrapolating boundary conditions, you would not want to use

$$u_0^{n+1} = u_1^{n+1}$$

because this is only a 0^{th} order extrapolation. Instead, you would need at least 1^{st} order accuracy such as

$$u_0^{n+1} = 2u_1^{n+1} - u_2^{n+1}$$

For stability, we will use the Kreiss stability theory. In addition to doing the standard von Neuman analysis, we assume $u_j^n = z^n \kappa^j$, and we must show that there is no solution for which $|z| > 1$ and $|\kappa| < 1$ for both the numerical method in the interior and for the boundary condition. Note that this type of analysis is very similar to the stability theory we did in the continuous case where we assumed a solution of the form $u(x,t) = e^{\sigma t} e^{\mu x}$, $\mu < 0$ where z plays the role of $e^{\sigma t}$ and κ plays the role of $e^{\mu x}$. In that case, we saw that specifying boundary conditions eliminated solutions with growth modes when the boundary condition was required. In this discrete case, we want to ensure there is no growth mode (i.e. $|z| > 1$) for an allowed solution of $|\kappa| < 1$.

Example 13.1:

Assume we are applying Leap-Frog to $u_t = u_x$, $x \geq 0$ then the discretization is

$$u_j^{n+1} = u_j^{n-1} + \lambda(u_{j+1}^n - u_{j-1}^n)$$

Plugging in $u_j^n = z^n \kappa^j$, we get

$$z^2 \kappa = \kappa + \lambda(\kappa^2 - 1)z$$

Solving for κ, we get

$$\lambda z \kappa^2 + (1 - z^2)\kappa - \lambda z = 0$$

$$\kappa^2 + \frac{1 - z^2}{\lambda z}\kappa - 1 = 0$$

From this, we can see that there are two solutions, $\kappa_1(z)$ and $\kappa_2(z)$ where $\kappa_1(z)\kappa_2(z) = -1$. We are looking for any solution where $|z| > 1$ and $|\kappa| < 1$, so we will take $|\kappa_1(z)| < 1$ and $|\kappa_2(z)| > 1$. Note that we cannot have both $|z| > 1$ and $|\kappa_1(z)| = |\kappa_2(z)| = 1$ because then $\kappa = e^{i\xi}$ for some ξ, and then plugging $u_j^n = z^n \kappa^j = z^n e^{ij\xi}$ into the numerical method and using the assumption $|z| > 1$ implies that the method violates von Neuman stability. Since we assume we are using a stable method in the interior, this case is ruled out.

What we must now show is that $\kappa_1(z)$ plugged into the boundary condition to ensure that $|z| > 1$ is not allowed. If we try the boundary condition $u_0^{n+1} = u_1^{n+1}$, then we get $1 = \kappa_1(z)$, which is ruled out. On a side note, this type of boundary condition still does not yield a stable method because this is not a sufficient condition to guarantee stability of the whole scheme.

If we do one-sided differencing, we get

$$u_0^{n+1} = u_0^n + \lambda(u_1^n - u_0^n) \tag{11}$$

and we get $z = 1 + \lambda(\kappa_1(z) - 1)$. From above, we find that

$$\kappa_1(z) = \frac{1}{2}\frac{-1 + z^2 - \sqrt{1 - 2\,z^2 + z^4 + 4\,\lambda^2 z^2}}{\lambda\, z}$$

Now we plug this solution into equation (11) to get

$$z = 1 + \lambda\left(\frac{1}{2}\,\frac{-1 + z^2 - \sqrt{1 - 2\,z^2 + z^4 + 4\,\lambda^2 z^2}}{\lambda\, z} - 1\right)$$

Simplifying this equation gives

$$0 = 4z(z-1)^2(\lambda - 1)$$

Thus, the only solutions are $z = 0$, 1. Therefore, $|z| > 1$ is not allowed and the method may be stable.

This analysis is based upon the following theorem.

Theorem 3 (Ryabenkii-Godunov) If a numerical method is von Neuman stable, and the treatment of the boundary is stable, then plugging $u_j^n = z^n \kappa^j$ into both the numerical method and the boundary method will have no solution with both $|z| > 1$ and $|\kappa| < 1$.

Notice that this is only a necessary condition and is not sufficient to prove stability. In practice, this is a very difficult result to establish and gets worse for larger than three point stencil methods.

We return now to the equation $u_t = u_x$, $0 \le x < +\infty$, $u(x,0) = f(x)$. We have already seen that in theory, we are not allowed to impose a boundary condition at $x = 0$, yet if we employ a central difference method, we will need to know how to advance the grid point at the left end.

One way to handle this is to take advantage of our knowledge of characteristics. Since data moves from right to left, we could use the $(1,1)$ upwind scheme

$$\begin{aligned} D^+ u_0^n &= D_+ u_o^n \\ u_0^{n+1} &= u_0^n + \lambda(u_1^n - u_0^n) \end{aligned}$$

On the plus side, it sends data in the right direction, and solves the problem of what to do at the left endpoint. The method is only $(1,1)$, but we saw earlier that this is sufficient to maintain $(2,2)$ accuracy in the interior.

We don't normally use this upwind scheme in the interior except around shocks. For more general problems, we may not always be able to identify the direction of the characteristics so that the correct upwind direction is taken.

In general, we will discuss two kinds of boundary conditions, extrapolation and one-sided differences. However, the two are very similar in their approaches.

13.1 Extrapolating Boundary Conditions

Consider the following situation, we are using a central difference scheme to advance the interior points. The problem is, how do we compute the value u_0^{n+1}? Central difference is not possible because there is no point to the left of u_0^n. Suppose we have computed the values u_j^{n+1} for $j \ge 1$ and need to compute u_0^{n+1}. The 0^{th} order approximation is

$$\begin{aligned} D_+ u_0^{n+1} &= 0 \\ u_0^{n+1} &= u_1^{n+1} \end{aligned}$$

We can get higher order extrapolations by insisting a higher order derivative is zero. An r^{th} order extrapolation method is given by

$$D_+^{r+1} u_0^{n+1} = 0$$

For example, the first order extrapolation is

$$\begin{aligned} 0 &= D_+^2 u_0^{n+1} \\ &= D_+(D_+ u_0^{n+1}) \\ &= \frac{1}{h} D_+(u_1^{n+1} - u_0^{n+1}) \\ &= \frac{1}{h}(D_+ u_1^{n+1} - D_+ u_0^{n+1}) \\ &= \frac{1}{h^2}(u_2^{n+1} - 2u_1^{n+1} + u_0^{n+1}) \end{aligned}$$

Solving for u_0^{n+1} gives

$$u_0^{n+1} = 2u_1^{n+1} - u_2^{n+1}$$

It can be shown that extrapolation of any order is stable if coupled to a method with dissipative error, however there can be degradation in accuracy near the boundary depending on the method used.

13.2 One-Sided Differences

Instead of extrapolating out to the boundary, we can employ one-sided differences. We have already seen one example, namely

$$D^+ u_0^n = D_+ u_0^n$$

Now,

$$\begin{aligned} D_+ u_0^n &= \frac{1}{h}(u_1^n - u_0^n) \\ &= \frac{1}{2h}(2u_1^n - 2u_0^n) \\ &= \frac{1}{2h}(u_1^n - (2u_0^n - u_1^n)) \end{aligned}$$

Suppose $u_{-1}^n = 2u_0^n - u_1^n$, then we have

$$\frac{1}{2h}(u_1^n - u_{-1}^n) = D_0 u_0^n.$$

In fact, one sided differences are equivalent to extrapolation at time n to establish the value u_{-1}^n. Then the interior method can be used right up to and including the boundary. Again, this type of boundary method is stable when coupled to a dissipative method.

We finish this section with a few notes about these extrapolation methods. First, if the equation is a conservation law, e.g. $u_t = (f(u))_x$, then it is generally better to extrapolate the flux itself rather than evaluate f at an extrapolated point. For example, the first order extrapolation would be

$$f_{-1} = 2f_0 - f_1$$

where f_j is the value of $f(u_j)$.

Stencils in space that require more than three points may require more than one grid point to be extrapolated. In this case, extrapolate outward one grid point at a time. For example, a 3^{rd} order extrapolation where u_{-1}, u_{-2} are needed is done by first extrapolating to get u_{-1}.

$$u_{-1} = 4u_0 - 6u_1 + 4u_2 - u_3$$

Then use this to compute u_{-2}

$$u_{-2} = 4u_{-1} - 6u_0 + 4u_1 - u_2$$

Note also that stability is only assured for methods with dissipative error terms. That eliminates Leap-Frog for example. To handle Leap-Frog, we may stably apply one-sided differences. A better one-sided difference for Leap-Frog adds dissipation and is given by

$$u_0^{n+1} = u_0^n + \lambda \left(u_1^n - \frac{1}{2}(u_0^{n+1} - u_0^{n-1}) \right)$$

This too can be used with Leap-Frog to give stable results.

13.3 Linear Systems

Consider the problem

$$\vec{u}_t = A\vec{u}_x, \quad 0 \le x < +\infty$$

where A is an $m \times m$ matrix. If all the eigenvalues are positive, then the characteristics leave the boundary and all characteristic variables must be specified via the boundary conditions. If all the eigenvalues are negative, then all the characteristics enter the boundary and so one-sided differencing or extrapolation must be used.

The hard case is when there is a mixture of inflow and outflow characteristics. The prototypical example of this is the system we looked at earlier, namely

$$\begin{bmatrix} u \\ v \end{bmatrix}_t = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}_x, \quad 0 \le x < +\infty$$

This has eigenvalues ± 1 and characteristic variables $u+v$, $u-v$. We saw before, that the inflow boundary conditions must be determined by the outflow conditions and a specified function, i.e.

$$u(0,t) - v(0,t) = \alpha(u(0,t) + v(0,t)) + g(t)$$

Suppose the boundary condition is give as $v(0,t) = g(t)$. To handle this, we first look at the outgoing variable and note that

$$(u+v)_t = (u+v)_x$$

Since this is outgoing, we use one-sided differences to obtain

$$u_0^{n+1} + v_0^{n+1} = u_0^n + v_0^n + \lambda(u_1^n + v_1^n - (v_0^n + v_0^n))$$

At the same time, we must also solve $v_0^{n+1} = g(t_{n+1})$. Thus, we must solve the simultaneous system

$$\begin{aligned} v_0^{n+1} &= g(t_{n+1}) \\ u_0^{n+1} + v_0^{n+1} &= u_0^n + v_0^n + \lambda(u_1^n + v_1^n - (u_0^n + v_0^n)) \end{aligned}$$

This is called the characteristic boundary treatment.

In general, to solve the inflow/outflow condition, we get m simultaneous equations to solve where r equations are upwind differences applied to the r outgoing characteristic variables and $m-r$ equations are given according to the local boundary conditions. This system of equations is then solved to produce the new value of $\vec{u}_0^{n+1}$.

14 Two-Dimensional Problems

Next we will look at problems in higher spatial dimensions. We will see how the stability criterion change in this new regime. As before, we start with a linear hyperbolic system.

$$\vec{u}_t = A\vec{u}_x + B\vec{u}_y$$

This system is hyperbolic if for any α, β, not both 0, there is a matrix $P_{\alpha,\beta}$ such that

$$P_{\alpha,\beta}^{-1}(\alpha A + \beta B)P_{\alpha,\beta} = D$$

where D is a real diagonal matrix. In particular, this implies that each of the systems $\vec{u}_t = A\vec{u}_x$ and $\vec{u}_t = B\vec{u}_y$ are each hyperbolic. Also, if we assume a solution of the form

$$\vec{u}(x,y,t) = e^{i\omega t}e^{i\alpha x}e^{i\beta y}\vec{u}_0$$

then we get

$$\begin{aligned} i\omega e^{i\omega t}e^{i\alpha x}e^{i\beta y}\vec{u}_0 &= i\alpha A e^{i\omega t}e^{i\alpha x}e^{i\beta y}\vec{u}_0 + i\beta B e^{i\omega t}e^{i\alpha x}e^{i\beta y}\vec{u}_0 \\ \omega\vec{u}_0 &= (\alpha A + \beta B)\vec{u}_0 \end{aligned}$$

Thus, $\vec{u}_0$ must be an eigenvector of $(\alpha A + \beta B)$ with eigenvalue ω.

One important thing to note about this problem is that we cannot reduce this to the scalar problem as we did in the one-dimensional case. Recall that in the one-dimensional case, we could diagonalize the system

and decouple it into a series of scalar equations. In the two- and higher-dimensional cases, this is not possible in general because A and B will most likely not both diagonalize with the same similarity transformation.

We can carry over much of our analysis tools, however. For example, let us apply von Neuman analysis to the two-dimensional Leap-Frog method applied to the scalar equation

$$u_t = u_x + u_y$$
$$u_{j,k}^{n+1} = u_{j,k}^{n-1} + \frac{\Delta t}{\Delta x}(u_{j+1,k}^n - u_{j-1,k}^n) + \frac{\Delta t}{\Delta y}(u_{j,k+1}^n - u_{j,k-1}^n)$$

As before, we assume a solution of the form

$$u_{j,k}^n = z^n e^{ij\xi} e^{ik\eta}$$

where $\xi = \ell_1 \Delta x$ and $\eta = \ell_2 \Delta y$ for wave numbers ℓ_1, ℓ_2. Plugging this into the method gives

$$\begin{aligned}
z^2 &= 1 + z\frac{\Delta t}{\Delta x}(e^{i\xi} - e^{-i\xi}) + z\frac{\Delta t}{\Delta y}(e^{i\eta} - e^{-i\eta}) \\
0 &= z^2 - z\left(\frac{\Delta t}{\Delta x} 2i\sin(\xi) + \frac{\Delta t}{\Delta y} 2i\sin(\eta)\right) - 1 \\
z &= i(\lambda_x \sin(\xi) + \lambda_y \sin(\eta)) \pm \sqrt{1 - (\lambda_x \sin(\xi) + \lambda_y \sin(\eta))^2} \\
|z|^2 &= (\lambda_x \sin(\xi) + \lambda_y \sin(\eta))^2 + (1 - (\lambda_x \sin(\xi) + \lambda_y \sin(\eta))^2) \\
&= 1
\end{aligned}$$

Thus, we get stability provided

$$\begin{aligned}
1 - (\lambda_x \sin(\xi) + \lambda_y \sin(\eta))^2 &\geq 0 \\
|\lambda_x \sin(\xi) + \lambda_y \sin(\eta)| &\leq 1
\end{aligned}$$

Thus, we are guaranteed stability if $\lambda_x + \lambda_y \leq 1$. Note, for example, if $\lambda_x = \lambda_y = \lambda$, then we have

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

which is more restrictive than the one-dimensional case.

If we apply this same approach to the two-dimensional system

$$\vec{u}_t = A\vec{u}_x + B\vec{u}_y$$

then we assume $\vec{u}_{j,k}^n = z^n e^{ij\xi} e^{ik\eta} \vec{u}_0$ and plug in to get as before,

$$((z^2 - 1)I - 2iz(\lambda_x A \sin(\xi) + \lambda_y B \sin(\eta)))\vec{u}_0 = 0$$

This equation can be solved only if $\vec{u}_0$ is an eigenvector of the matrix

$$\lambda_x \sin(\xi) A + \lambda_y \sin(\eta) B$$

Let $\mu_1(\xi, \eta)$, $\ldots$, $\mu_m(\xi, \eta)$ be the m eigenvalues of this matrix, then we are again reduced to the scalar problem and we see that we get stability if

$$\max |\mu_\ell(\xi, \eta)| \leq 1.$$

In particular, if $\lambda_x = \lambda_y = \lambda$ and $\mu_1, \ldots, \mu_m$ are eigenvalues of $\sin(\xi)A + \sin(\eta)B$, then we get

$$\lambda \leq \frac{1}{\max |\mu_\ell(\xi, \eta)|}.$$

If $A = B$, then $\max|\mu_\ell(\xi,\eta)| = 2\max|\mu_\ell^A|$ where μ_ℓ^A are the eigenvalues of A. If A, B have a common set of eigenvectors, then

$$\lambda \le \frac{1}{\max(|\mu_\ell^A| + |\mu_\ell^B|)}.$$

In general, for any matrix norm we have $|\mu_\ell(\xi,\eta)| \le ||A|| + ||B||$ and hence

$$\lambda \le \frac{1}{||A|| + ||B||}.$$

This is usually more restrictive than is necessary in practice.

14.1 Operator Splitting

One way to maneuver around the more restrictive time stepping is done via operator splitting. To begin, consider the scalar differential equation

$$u_t = u_x + u_y$$

An explicit central-difference method would be

$$\begin{aligned} D^+u_{j,k}^n &= D_{x,0}u_{j,k}^n + D_{y,0}u_{j,k}^n \\ u_{j,k}^{n+1} &= u_{j,k}^n + k(D_{x,0}u_{j,k}^n + D_{y,0}u_{j,k}^n) \\ &= (I + kD_{x,0} + kD_{y,0})u_{j,k}^n \end{aligned}$$

For small k, we formally get

$$\begin{aligned} I + k(D_{x,0} + D_{y,0}) &\approx I + k(D_{x,0} + D_{y,0}) + k^2 D_{x,0}D_{y,0} \\ &= (I + kD_{x,0})(I + kD_{y,0}) \end{aligned}$$

This is called operator splitting, and the method becomes

$$\begin{aligned} \hat{u}_{j,k}^n = (I + kD_{y,0})u_{j,k}^n &= u_{j,k}^n + \frac{\lambda}{2}(u_{j,k+1}^n - u_{j,k-1}^n) \\ u_{j,k}^{n+1} = (I + kD_{x,0})\hat{u}_{j,k} &= \hat{u}_{j,k} + \frac{\lambda}{2}(\hat{u}_{j+1,k}^n - \hat{u}_{j-1,k}^n) \end{aligned}$$

Notice that we are therefore solving $u_t = u_y$ and then using the results to compute $u_t = u_x$ to get to the next time step. In this way, our time step restriction is limited by the 1-dimensional operators which is better.

This method of splitting can be applied to most methods, however, we must be careful when applying it to systems of equations. Consider the system of ordinary differential equations

$$\vec{y}' = (A + B)\vec{y}$$

The exact solution is $\vec{y}(t) = e^{(A+B)t}\vec{y}_0$. In particular,

$$\begin{aligned} e^{(A+B)k} &\approx I + (A+B)k + \frac{k^2}{2}(A+B)^2 + \cdots \\ &= I + (A+B)k + \frac{k^2}{2}(A^2 + AB + BA + B^2) \\ &= I + kA + \frac{k^2}{2}A^2 + I + kB + \frac{k^2}{2}B^2 + k^2AB + \frac{k^2}{2}(BA - AB) \\ &= \left(I + kA + \frac{k^2}{2}A^2\right)\left(I + kB + \frac{k^2}{2}B^2\right) + \frac{k^2}{2}(BA - AB) \\ &\approx e^{Ak}e^{Bk} + O(k^2) \end{aligned}$$

Notice that we get $e^{(A+B)k} = e^{Ak}e^{Bk}$ only if A, B commute.

Notice also that if we reverse the roles of A and B, the second order error term simply changes signs. This suggests that if we alternate in applications of the splitting, that the second order error may cancel. Thus, we get

$$\begin{aligned}
e^{Ak}e^{Bk}e^{Bk}e^{Ak} &= \left(I + kA + \frac{k^2}{2}A^2\right)\left(I + kB + \frac{k^2}{2}B^2\right)^2\left(I + kA + \frac{k^2}{2}A^2\right) + O(k^3) \\
&= \left(I + kA + \frac{k^2}{2}A^2 + kB + k^2AB + \frac{k^2}{2}B^2\right) \\
&\qquad \left(I + kA + \frac{k^2}{2}A^2 + kB + k^2BA + \frac{k^2}{2}B^2\right) + O(k^3) \\
&= I + 2kA + 2kB + 2k^2A^2 + 2k^2BA + 2k^2AB + 2k^2B^2 + O(k^3) \\
&= I + 2k(A+B) + \frac{(2k)^2}{2}(A+B)^2 + O(k^3) \\
&= e^{(A+B)2k} + O(k^3)
\end{aligned}$$

Therefore, alternating the order of the operators results in a second order method in time. At the same time, the stability requirement reduces to that for the one-dimensional case. Note that this only works for one-step methods. Multi-step methods such as Runge-Kutta and Leap-Frog require intermediate steps which do not make sense in this framework.

Also, suppose that the operator B requires a time step half as large as A, then it can be applied as

$$e^{Ak}e^{B\frac{k}{2}}e^{B\frac{k}{2}}e^{B\frac{k}{2}}e^{B\frac{k}{2}}e^{Ak}$$

for example. In other words, take one step with operator A with time step k and follow that with two steps of size $k/2$ for operator B, then reverse the order. This has applications in many equations where the two operators have different stability restrictions, such as for example, the reaction diffusion equation

$$u_t = u_{xx} + R(u)$$

where the solution of $u_t = u_{xx}$ is followed by a solution of $u_t = R(u)$. For this type of equation it is common that these two operators will have widely disparate time step restrictions, say on the order of 10 to 1. Using multiple steps for one operator while the other takes a single step is an efficient way to solve this type of equation.

14.2 Alternating Directions Implicit Method

A particular variation of operator splitting is called the alternating directions implicit method. Consider the equation

$$u_t = u_{xx} + u_{yy}$$

(this method applies equally well to the wave equation) and apply Crank-Nicolson to it.

$$u^{n+1} = u^n + \frac{k}{2}(D_{x+}D_{x-}u^{n+1} + D_{x+}D_{x-}u^n) + \frac{k}{2}(D_{y+}D_{y-}u^{n+1} + D_{y+}D_{y-}u^n)$$

Writing this in the form of operators, we get

$$\begin{aligned}
\left(I - \frac{k}{2}D_{x+}D_{x-} - \frac{k}{2}D_{y+}D_{y-}\right)u^{n+1} &= \left(I + \frac{k}{2}D_{x+}D_{x-} + \frac{k}{2}D_{y+}D_{y-}\right)u^n \\
\left(I - \frac{k}{2}D_{x+}D_{x-}\right)\left(I - \frac{k}{2}D_{y+}D_{y-}\right)u^{n+1} &= \left(I + \frac{k}{2}D_{x+}D_{x-}\right)\left(I + \frac{k}{2}D_{y+}D_{y-}\right)u^n \\
u^{n+1} = \left(I - \frac{k}{2}D_{y+}D_{y-}\right)^{-1}\left(I - \frac{k}{2}D_{x+}D_{x-}\right)^{-1}&\left(I + \frac{k}{2}D_{x+}D_{x-}\right)\left(I + \frac{k}{2}D_{y+}D_{y-}\right)u^n
\end{aligned}$$

Next, note that the two matrices $(I - A)$ and $(I + A)$ always commute and hence

$$\left(I - \frac{k}{2}D_{x+}D_{x-}\right)^{-1}\left(I + \frac{k}{2}D_{x+}D_{x-}\right) = \left(I + \frac{k}{2}D_{x+}D_{x-}\right)\left(I - \frac{k}{2}D_{x+}D_{x-}\right)^{-1},$$

so we can write the method as

$$u^{n+1} = \left(I - \frac{k}{2}D_{y+}D_{y-}\right)^{-1}\left(I + \frac{k}{2}D_{x+}D_{x-}\right)\left(I - \frac{k}{2}D_{x+}D_{x-}\right)^{-1}\left(I + \frac{k}{2}D_{y+}D_{y-}\right)u^n$$

Breaking this into two separate steps, we get the method

$$\left(I - \frac{k}{2}D_{x+}D_{x-}\right)u^{n+1/2} = \left(I + \frac{k}{2}D_{y+}D_{y-}\right)u^n \tag{12}$$

$$\left(I - \frac{k}{2}D_{y+}D_{y-}\right)u^{n+1} = \left(I + \frac{k}{2}D_{x+}D_{x-}\right)u^{n+1/2} \tag{13}$$

Now, if we do the von Neuman analysis on equation (12), we get

$$\left(1 - \frac{k}{h^2}(\cos(\xi) - 1)\right)z^{1/2} = \left(1 + \frac{k}{h^2}(\cos(\eta) - 1)\right)$$

$$z^{1/2} = \frac{1 + \beta(\cos(\eta) - 1)}{1 - \beta(\cos(\xi) - 1)}$$

Notice that this has a stability requirement of $\beta \leq 1$. Likewise, for equation (13), we get

$$z^{1/2} = \frac{1 + \beta(\cos(\xi) - 1)}{1 - \beta(\cos(\eta) - 1)}$$

which also has a stability requirement of $\beta \leq 1$. But when we combine these two steps, we get

$$\begin{aligned} z &= \frac{1 + \beta(\cos(\eta) - 1)}{1 - \beta(\cos(\xi) - 1)}\frac{1 + \beta(\cos(\xi) - 1)}{1 - \beta(\cos(\eta) - 1)} \\ &= \frac{1 + \beta(\cos(\eta) - 1)}{1 - \beta(\cos(\eta) - 1)}\frac{1 + \beta(\cos(\xi) - 1)}{1 - \beta(\cos(\xi) - 1)} \\ &\leq 1 \end{aligned}$$

Thus, the method is unconditionally stable even though neither of the intermediate steps is. At the same time, this is a (2,2) method as well.

14.3 Anisotropic Errors

Solving multi-dimensional problems can also result in errors which are not isotropic. Sometimes this is referred to as "grid effects" because the errors vary according to the orientation of the problem with respect to the grid.

Consider the wave equation

$$u_t = u_x + u_y$$

with semi-discrete approximation

$$\dot{u}_{jk} = D_{x0}u_{jk} + D_{y0}u_{jk}$$

For this analysis, we are only concerned with the spatial discretization. Suppose we have a wave traveling in the direction θ, then we expect a solution of the form

$$u(x,t) = e^{i\omega t}e^{i|\ell|(\cos(\theta)x + \sin(\theta)y)}$$

Plugging this into the equation, we get the exact phase of the solution,

$$\omega = |\ell|(\cos(\theta) + \sin(\theta))$$

Next, plug it into the discretization to get

$$\omega_h = \frac{\sin(\xi)}{h} + \frac{\sin(\eta)}{h}$$

where $\xi = |\ell|\cos(\theta)h$ and $\eta = |\ell|\sin(\theta)h$. Finally, we get the general phase error given by

$$\begin{aligned}
\omega - \omega_h &= |\ell|(\cos(\theta) + \sin(\theta)) - \left(\frac{\sin(\xi)}{h} + \frac{\sin(\eta)}{h}\right) \\
&= |\ell|(\cos(\theta) + \sin(\theta)) - \frac{1}{h}\left(\xi - \frac{\xi^3}{6} + \eta - \frac{\eta^3}{6}\right) \\
&= |\ell|(\cos(\theta) + \sin(\theta)) - \frac{1}{h}\left(|\ell|\cos(\theta)h - \frac{1}{6}|\ell|^3\cos^3(\theta)h^3 + |\ell|\sin(\theta)h - \frac{1}{6}|\ell|^3\sin^3(\theta)h^3\right) \\
&= |\ell|(\cos(\theta) + \sin(\theta)) - \left(|\ell|\cos(\theta) - \frac{1}{6}|\ell|^3\cos^3(\theta)h^2 + |\ell|\sin(\theta) - \frac{1}{6}|\ell|^3\sin^3(\theta)h^2\right) \\
&= \frac{1}{6}|\ell|^3h^2(\cos^3(\theta) + \sin^3(\theta))
\end{aligned}$$

Notice that the phase error dependence on θ is different than that for the exact phase error. This means that the amount of error in the phase in not proportional to θ, but varies with the grid. Also note that the maximum error occurs for $\theta = 0$, $\pi/2$, π, and $3\pi/2$ which corresponds to the directions along grid lines. This type of error will manifest itself, for example, in the squaring of a circle as it moves.

14.4 Boundary Conditions in 2D

Consider the system

$$\vec{u}_t = A\vec{u}_x + B\vec{u}_y$$

in a rectangular domain. The hyperbolicity condition ensures that both A and B have m distinct eigenvalues. When enforcing boundary conditions, looking only at derivatives in the normal direction, so if the boundary is $x = 0$, enforce boundary conditions according to the system $\vec{u}_t = A\vec{u}_x$.

In corners, it is difficult to get boundary conditions which work perfectly. Often oscillations or even instability can result there. One approach is to impose both the $x-$ and $y-$ direction boundary conditions at the corners.

15 A Bridge from Finite Differences to Spectral Methods

15.1 A Double-Step Method

Consider the equation

$$\begin{aligned}
u_t = au_x, \quad 0 \le x \le 1, \quad 0 \le t \le 1 \\
u(x+1,t) = u(x,t), \quad \text{(periodic BC)} \\
u(x,0) = u_0(x) \\
a > 0
\end{aligned}$$

Let us solve this using the simplest possible method, namely the $(1,1)$ method

$$\begin{aligned}
D^+u_j^n &= aD_+u_j^n \\
u_j^{n+1} &= u_j^n + \frac{ak}{h}(u_{j+1}^n - u_j^n)
\end{aligned}$$

We can construct a new method from this one by taking two steps of length $k/2$:

$$\begin{aligned} u_j^{n+1/2} &= u_j^n + \frac{ak}{2h}(u_{j+1}^n - u_j^n) \\ u_j^{n+1} &= u_j^{n+1/2} + \frac{ak}{2h}(u_{j+1}^{n+1/2} - u_j^{n+1/2}) \\ &= u_j^n + \frac{ak}{h}(u_{j+1}^n - u_j^n) + \left(\frac{ak}{2h}\right)^2 (u_{j+2}^n - 2u_{j+1}^n + u_j^n) \end{aligned}$$

We begin our analysis of this method by looking at its stability properities. Let $u_j^n = z^n e^{ij\ell h}$, then we get

$$\begin{aligned} z^{1/2} &= 1 + \frac{ak}{2h}(e^{i\ell h} - 1) \\ z &= \left(1 + \frac{ak}{2h}(e^{i\ell h} - 1)\right)^2 \end{aligned}$$

So,

$$\begin{aligned} |z|^2 &= \left|1 + \frac{ak}{2h}(e^{i\ell h} - 1)\right|^4 \\ &= \left[\left(1 + \frac{ak}{2h}(\cos(\ell h) - 1)\right)^2 + \left(\frac{ak}{2h}\sin(\ell h)\right)^2\right]^2 \\ &= \left(1 + \frac{ak}{h}(1 - \cos(\ell h))\left(\frac{ak}{2h} - 1\right)\right)^2 \end{aligned}$$

and we get stability if $\frac{ak}{2h} - 1 \leq 0$, or $\frac{ak}{h} < 2$. This means we have a new method which has a time step restricion twice as large as the original method. Note that we have not violated the CFL condition because the new method also has a stencil which is twice as large.

Next, we can look at the truncation error for our new method:

$$\begin{aligned} k\tau &= u + ku_t + \frac{k^2}{2}u_{tt} + O(k^3) \\ &\quad - \frac{ak}{h}\left(u + hu_x + \frac{h^2}{2}u_{xx} + O(h^3)\right) \\ &\quad - \frac{a^2k^2}{4h^2}(u + 2u_x + 2h^2u_{xx} - 2u - 2hu_x - h^2u_{xx} + u + O(h^3)) \\ &= ku_t + \frac{k^2}{2}u_{tt} - aku_x - \frac{akh}{2}u_{xx} - \frac{a^2k^2}{4}u_{xx} + O(\text{higher order terms}) \\ &= \left(\frac{a^2k^2}{2} - \frac{akh}{2}\right)u_{xx} - \frac{a^2k^2}{4}u_{xx} + O(\text{higher order terms}) \qquad (14) \\ &= \left(\frac{a^2k^2}{4} - \frac{akh}{2}\right)u_{xx} + O(\text{higher order terms}) \\ \tau &= \left(\frac{a^2k}{4} - \frac{ah}{2}\right)u_{xx} + O(\text{higher order terms}) \qquad (15) \end{aligned}$$

Notice that the expression in the parentheses in equation 14 is the error for the original one-step method $D^+u_j^n = D_+u_j^n$ and our new method adds a correction term which exactly halves the time step error, while leaving the spatial error untouched.

So it appears to the uninitiated, that we have essentially gained something from nothing. Our new method has twice the time step length and half the error. In truth, we get one or the other, in other words,

for a fixed time step size k, the new method will have half the error of the original method. Or, we can take double the time step length while achieving the same total error. Of course, we truly have accomplished nothing, because all we have done is made halving the time step in a simple method look complicated.

Nonetheless, let us make it more complicated by framing the discussion as a linear algebra problem. Let

$$U^n = \begin{bmatrix} u_1^n \\ \vdots \\ u_N^n \end{bmatrix}$$

then we can write the original numerical method as a matrix equation:

$$U^{n+1} = A(k)U^n = \begin{bmatrix} 1-\frac{ak}{h} & \frac{ak}{h} & & \\ & \ddots & \ddots & \\ & & \ddots & \frac{ak}{h} \\ \frac{ak}{h} & & & 1-\frac{ak}{h} \end{bmatrix} U^n$$

Hence our 2-step method can be written as

$$U^{n+1} = A(k/2)A(k/2)U^n = A(k/2)^2 U^n$$

The stability analysis can be framed in this by assuming $U^n = z^n V$ for some constant vector V. Plugging this into the matrix equation, we get

$$\begin{aligned} z^{n+1}V &= A(k/2)^2 z^n V \\ zV &= A(k/2)^2 V \end{aligned}$$

So the vector V must be an eigenvector, and the growth rates are governed by the eigenvalues of the matrix $A(k)$.

To compute the eigensystem for $A(k)$ does not seem difficult to do numerically, but analytically it appears at first glance to be laborious. However, we can in fact write the eigenvectors for this matrix down explicitly. The eigenvectors are

$$V(\ell) = \begin{bmatrix} e^{2\pi i \ell/N} \\ e^{2\pi i \ell 2/N} \\ \vdots \\ e^{2\pi i \ell j/N} \\ \vdots \\ e^{2\pi i \ell N/N} \end{bmatrix}, \quad \text{for } \ell = 0,\, 1,\, \ldots,\, N-1$$

These eigenvectors will be obvious when we discuss spectral methods, but for now we can identify these eigenvectors with each of the numerical wave modes that can be represented on a grid of physical length 1 using N grid points, i.e. $h = 1/N$.

Now that we have the eigenvectors, we can easily compute the corresponding eigenvalues by computing

$$A(k)V(\ell) = \begin{bmatrix} 1-\frac{ak}{h} & \frac{ak}{h} & & \\ & \ddots & \ddots & \\ & & \ddots & \frac{ak}{h} \\ \frac{ak}{h} & & & 1-\frac{ak}{h} \end{bmatrix} \begin{bmatrix} e^{2\pi i \ell/N} \\ \vdots \\ \vdots \\ e^{2\pi i \ell N/N} \end{bmatrix}$$

The j^{th} row, for $j = 1,\ldots, N-1$ is then

$$\begin{aligned}(A(k)V(\ell))_j &= \left(1 - \frac{ak}{h}\right) e^{2\pi i\ell j/N} + \frac{ak}{h} e^{2\pi i\ell(j+1)/N} \\ &= \left(1 - \frac{ak}{h} + \frac{ak}{h} e^{2\pi i\ell/N}\right) e^{2\pi i\ell j/N} \\ &= \left(1 + \frac{ak}{h}\left(e^{2\pi i\ell/N} - 1\right)\right) V(\ell)_j\end{aligned}$$

Therefore, the eigenvalues are $\left(1 + \frac{ak}{h}\left(e^{2\pi i\ell/N} - 1\right)\right)$ for $\ell = 0, \ldots, N-1$ for $A(k)$, and hence the eigenvalues for $A(k/2)^2$ are

$$\left(1 + \frac{ak}{2h}\left(e^{2\pi i\ell/N} - 1\right)\right)^2 = \left(1 + \frac{ak}{2h}(\cos(\xi) + i\sin(\xi) - 1)\right)^2$$

where $\xi = 2\pi\ell/N$ as we saw in our original analysis. Thus, we again have $|z| \le 1$ if $\frac{ak}{h} \le 2$.

15.2 An m-step Method

Now, why should we take only two steps? We can just as easily consider an m-step method given by

$$U^{n+1} = A(k/m)^m U^n$$

This method has the benefit that the error term is

$$\frac{a^2 k}{2m} u_{xx} - \frac{ah}{2} u_{xx}$$

and the growth modes are

$$z = \left(1 + \frac{ak}{mh}\left(e^{2\pi i\ell/N} - 1\right)\right)^m$$

which yields $|z| \le 1$ if $\frac{ak}{h} \le m$.

While this method looks appealing, it is actually nothing more than using the original method with a smaller time step. The reason we don't use this reformulation is because the computational cost of computing the matrix $A(k/m)^m$ is greater than using $A(k)$. The brute force method of computing $A(k/m)^m$ is exactly equivalent to taking a smaller time step. However, we know the eigenvalues and eigenvectors of this system, so we can easily diagonalize $A(k/m)$, and hence compute $A(k/m)^m$ for less cost than computing $A(k/3)^3$ by the brute force method. When we diagonalize $A(k/m)$, we get

$$A(k/m) = F\Lambda F^{-1}$$

where

$$F = \begin{bmatrix} V(0) & V(1) & V(2) & \cdots & V(N-1) \end{bmatrix}$$

and Λ is a diagonal matrix with entries

$$\Lambda_{\ell\ell} = \left(1 + \frac{ak}{mh}\left(e^{2\pi i\ell/N} - 1\right)\right)$$

Finally, we get

$$A(k/m)^m = F\Lambda^m F^{-1}$$

15.3 A Connection to the Fourier Transform

Let's take a closer look at the matrix of eigenvectors, F. The entries of F are $F_{j\ell} = e^{2\pi i \ell j/N}$, and so given a vector U, we have

$$(FU)_j = \sum_{\ell=0}^{N-1} F_{j\ell} U_\ell = \sum_{\ell=0}^{N-1} e^{2\pi i \ell j/N} U_\ell$$

Thus, F is none other than matrix form of the discrete Fourier transform. That means that we already know what F^{-1} looks like, namely $(F^{-1})_{j\ell} = \frac{1}{N} e^{-2\pi i \ell j/N}$. To check that we are right, we see that for $r \neq s$,

$$\begin{aligned}
(FF^{-1})_{rs} &= \frac{1}{N} \sum_{\ell=0}^{N-1} e^{2\pi i \ell r/N} e^{-2\pi i \ell s/N} \\
&= \frac{1}{N} \sum_{\ell=0}^{N-1} e^{2\pi i \ell (r-s)/N} \\
&= \frac{e^{2\pi i (r-s)N/N} - 1}{N(e^{2\pi i (r-s)/N} - 1)} \\
&= \frac{e^{2\pi i (r-s)} - 1}{N(e^{2\pi i (r-s)/N} - 1)} \\
&= 0
\end{aligned}$$

because $r - s$ is an integer. If $r = s$, then

$$(FF^{-1})_{ss} = \frac{1}{N} \sum_{\ell=0}^{N-1} e^{2\pi i \ell s/N} e^{-2\pi i \ell s/N} = \frac{1}{N} \sum_{\ell=0}^{N-1} 1 = \frac{N}{N} = 1$$

Thus, F^{-1} is the inverse discrete Fourier trnasform.

When we discuss spectral methods later, we will see this same transformation again. In fact, what we are doing is translating our problem into Fourier space, solving the easier decoupled problem analytically, then transforming back into real space. This is precisely the idea behind spectral methods.

15.4 An ∞-step Method

Of course, we need not stop at a finite number of steps, what if we take $m \to +\infty$? Define $A^\infty = \lim_{m \to +\infty} A(k/m)^m$. We can construct this matrix again, by computing the similarity transformation

$$A^\infty = \lim_{m \to +\infty} F \Lambda^m F^{-1}$$

Recall that

$$\Lambda_{\ell\ell}^m = \left(1 + \frac{ak}{mh}\left(e^{2\pi i \ell/N} - 1\right)\right)^m = \left(1 + \frac{r}{m}\right)^m$$

where $r = \frac{ak}{h}(e^{2\pi i \ell/N} - 1)$, so

$$\begin{aligned}
\lim_{m \to +\infty} \Lambda_{\ell\ell}^m &= \lim_{m \to +\infty} \left(1 + \frac{r}{m}\right)^m \\
&= e^r
\end{aligned}$$

and hence

$$\begin{aligned}
\Lambda_{\ell\ell}^\infty &= e^{\frac{ak}{h}(e^{2\pi i \ell/N} - 1)} \\
&= e^{\frac{ak}{h}(\cos(\xi) + i\sin(\xi) - 1)}, \qquad \text{where } \xi = 2\pi i \ell/N \\
&= e^{\frac{ak}{h}(\cos(\xi) - 1)} e^{i\frac{ak}{h}\sin(\xi)}
\end{aligned}$$

We get unconditional stability from the fact that

$$\begin{aligned} |z| &= e^{\frac{ak}{h}(\cos(\xi)-1)}|e^{i\frac{ak}{h}\sin(\xi)}| \\ &= e^{\frac{ak}{h}(\cos(\xi)-1)} \leq 1 \end{aligned}$$

since $\frac{ak}{h}(\cos(\xi)-1) \leq 0$. In addition to unconditional stability, we also get no time stepping error, and all for the same cost as the finite m-step method.

Now, one may ask why we have worked so hard in order to obtain an unconditionally stable method with a little less error, but there is a fundamental difference. While a standard implicit method can stably take an arbitrarily large time step, it is not practical to use the method for very large time steps because of the presence of an error term in time. The method we describe here does not suffer from this problem, and so can take much larger time steps with better accuracy than the implicit method.

Note that this still does not mean there is no error in time. The spatial error in the method is still present, and this contributes to artificial diffusion. This effect can easily be seen by looking at the growth modes which contain $e^{\frac{ak}{h}(\cos(\xi)-1)}$. The larger the time step k, the larger the amount of diffusion. This means that the solution will degrade over very long time periods. However, this method still is a significant improvement over the corresponding conventional implicit method.

15.5 A Heat Equation Example

Let's look at another example; solve for large time:

$$\begin{aligned} u_t &= au_{xx} \\ u(0,t) &= u(1,t) = 0 \\ u(x,0) &= u_0(x) \end{aligned}$$

If we use the basic explicit method $D^+u_j^n = aD_+D_-u_j^n$, then the matrix equation is

$$U^{n+1} = \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \\ \vdots \\ \vdots \\ u_{N-1}^{n+1} \end{bmatrix} = \begin{bmatrix} 1-2\frac{ak}{h^2} & \frac{ak}{h^2} & & & \\ \frac{ak}{h^2} & 1-2\frac{ak}{h^2} & \frac{ak}{h^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \frac{ak}{h^2} \\ & & & \frac{ak}{h^2} & 1-2\frac{ak}{h^2} \end{bmatrix} \begin{bmatrix} u_1^n \\ u_2^n \\ \vdots \\ \vdots \\ u_{N-1}^n \end{bmatrix}$$

One way to generate the matrix F is to look at the analytic eigenfunctions for the problem. In this case, the eigenfunctions are $\sin(\pi\ell x)$ for $\ell = 1, 2, \ldots, N-1$, so we expect the eigenvectors for the discrete system to be $(V_\ell)_j = \sin(\pi\ell j/N)$, and we can see that this is the case. The eigenvalues are then

$$\begin{bmatrix} 1-2\frac{ak}{h^2} & \frac{ak}{h^2} & & & \\ \frac{ak}{h^2} & 1-2\frac{ak}{h^2} & \frac{ak}{h^2} & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \frac{ak}{h^2} \\ & & & \frac{ak}{h^2} & 1-2\frac{ak}{h^2} \end{bmatrix} \begin{bmatrix} \sin(\pi\ell/N) \\ \vdots \\ \vdots \\ \vdots \\ \sin(\pi\ell(N-1)/N) \end{bmatrix}$$

$$= \begin{bmatrix} 1-2\beta & \beta & & & \\ \beta & 1-2\beta & \beta & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \beta \\ & & & \beta & 1-2\beta \end{bmatrix} \begin{bmatrix} \sin(\pi\ell/N) \\ \vdots \\ \vdots \\ \vdots \\ \sin(\pi\ell(N-1)/N) \end{bmatrix}$$

$$
\begin{aligned}
(AV_\ell)_j &= \beta(\pi\ell(j-1)/N) + (1-2\beta)\sin(\pi\ell j/N) + \beta\sin(\pi\ell(j+1)/N) \\
&= \beta(\sin(\pi\ell j/N)\cos(\pi\ell/N) - \cos(\pi\ell j/N)\sin(\pi\ell/N)) + (1-2\beta)\sin(\pi\ell j/N) \\
&\quad + \beta(\sin(\pi\ell j/N)\cos(\pi\ell/N) + \cos(\pi\ell j/N)\sin(\pi\ell/N)) \\
&= (2\beta\cos(\pi\ell/N) + (1-2\beta))\sin(\pi\ell j/N)
\end{aligned}
$$

and therefore,

$$
\Lambda_{\ell\ell} = 2\beta\cos(\pi\ell/N) + (1-2\beta) = 1 + 2\beta(\cos(\pi\ell/N) - 1)
$$

for $\ell = 1, 2, \ldots, N-1$.

Now, for long time behavior, we must compute

$$
\begin{aligned}
\Lambda_{\ell\ell}^{\infty} &= \lim_{m\to+\infty}\left(1 + 2\frac{ak}{mh^2}(\cos(\pi\ell/N) - 1\right)^m \\
&= e^{2\frac{ak}{h^2}(\cos(\pi\ell/N)-1)}
\end{aligned}
$$

Notice that as $k \to +\infty$, we get $\Lambda_{\ell\ell}^{\infty} \to 0$. This makes sense because the long term steady state solution is $u(x, +\infty) = 0$.

16 Numerical Methods for Stochastic Differential Equations

16.1 Preliminaries

Before going through the analysis for stochastic differential equations, let us first review a few key concepts that we will need for our analysis. This is by no means comprehensive, but just a quick build up so that we can describe the basic equations and numerical methods. For a reference text, see "Numerical Solution of Stochastic Differential Equations", by P. Kloeden and E. Platen.

The *standard Gaussian distribution* for a random variable with mean μ and variance σ^2 is

$$
p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}
$$

This function has a maximum value of $\frac{1}{\sqrt{2\pi}\sigma}$ at $x = \mu$, and points of inflection at $x = \mu \pm \sigma$. A random variable with this distribution is written as $X \sim N(\mu; \sigma^2)$.

A *stochastic process* is a sequence of random variables $X_1, X_2, \ldots, X_n, \ldots$ which may describe the evolution of a probabilistic evolution at times $t_1 < t_2 < \cdots < t_n < \cdots$. We can also define a continuous time stochastic process where for each time t, in some interval $[0, a]$ or $[0, +\infty)$ we define a random variable $X(t) = X_t$.

A *Gaussian process* is a stochastic process where each random vairable X_n has Gaussian distribution. A particular Gaussian process is a *Wiener process* $W = \{W(t), \quad t \geq 0\}$ where for any sequence $t_1 < t_2 < t_3 < \cdots < t_n$, the random variables $W(t_{j+1}) - W(t_j)$ are independent. Also,

$$
W(0) = 0, \quad E(W(t)) = 0, \quad \text{Var}(W(t) - W(s)) = t - s
$$

this is also sometimes called Brownian motion.

An *Ito process* $X = \{X_t, \quad t \geq 0\}$ has the form

$$
X_t = X_0 + \int_0^t a(s, X_s)ds + \int_0^t b(s, X_s)dW_s
$$

for $t \geq 0$, where $X_0 = x_0$ is a possibly random initial condition, $a(t, x)$ is a continuous function called the *drift*, and a $b(t, x)$ is a continuous function called the *diffusion*. In differential form, this equation is written

$$
dX_t = a(t, X_t)dt + b(t, X_t)dW_t
$$

where $W = \{W_t, \quad t \geq 0\}$ is a Wiener process. In order for the Ito process to have a solution in the strong sense, we must make the following assumptions:

(Measurability): $a(t,x)$ and $b(t,x)$ must both be L^2 measurable in the domain $(t,x) \in [t_0, T] \times \mathbb{R}$.

(Smoothness): There exists a constant $K > 0$ such that

$$|a(t,x) - a(t,y)| \leq K|x-y|$$
$$|b(t,x) - b(t,y)| \leq K|x-y|$$

(Stability): There exists a constant $K > 0$ such that

$$|a(t,x)|^2 \leq K^2(1+|x|^2)$$
$$|b(t,x)|^2 \leq K^2(1+|x|^2)$$

(Initial Data): The initial data X_{t_0} has bounded variance, i.e. $E(|X_{t_0}|^2) < \infty$.

Some explicitly computable examples that we will compare against are given below:

Example 16.1:

Additiive noise in an Ito process is modelled by the equation

$$dX_t = -aX_t dt + b dW_t$$

The solution to this equation is

$$X_t = e^{-at}\left(X_0 + b\int_0^t e^{as}\, dW_s\right)$$

Example 16.2:

Multiplicative noise in an Ito process is modelled by the equation

$$dX_t = aX_t dt + bX_t dW_t$$

and the solution is

$$X_t = X_0 e^{\left(\left(a - \frac{1}{2}b^2\right)t + bW_t\right)}$$

and for fixed time steps, we can evaluate this last formula by computing

$$X_{kn} = X_0 e^{\left(\left(a - \frac{1}{2}b^2\right)t + b\sum_{i=1}^{n} W_{ik} - W_{(i-1)k}\right)}$$

16.2 The Euler-Maruyama Method

The *Euler-Maruyama method* is the equivalent of Euler's method for deterministic equations. Let $X = \{X_t, \quad 0 \leq t \leq T\}$ be an Ito process which solves

$$dX_t = a(t, X_t)dt + b(t, X_t)dW_t$$

where X_0 is given and is possibly random. Let $t_n = kn$ be the time steps, then define

$$\begin{aligned} Y_0 &= X_0 \\ Y_{n+1} &= Y_n + a(t_n, Y_n)k + b(t_n, Y_n)(W_{t_{n+1}} - W_{t_n}) \end{aligned}$$

where we hope $Y_n = Y(t_n) \approx X_{t_n}$. Note that if $b \equiv 0$, this reduces to the deterministic Euler method solving the equation $x' = a(t, x)$. Note also that because W is a Wiener process, $\Delta W_n = W_{t_{n+1}} - W_{t_n}$ is a random variable with

$$\begin{aligned} E(\Delta W_n) &= 0 \\ E((\Delta W_n)^2) &= k \end{aligned}$$

16.3 Strong Convergence

In order to evaluate numerical methods, we need to define an error estimate. We begin with the error defined by

$$\text{error} = \epsilon = E(|X_T - Y(T)|)$$

We can use this error estimate to define *strong convergence*. Let Y^k be the approximation using time step size k. We say that Y^k converges strongly to X at time T if

$$\lim_{k \to 0^+} E(|X_t - Y^k(t)|) = 0$$

The order of convergence p is defined as a constant where

$$\epsilon(k) = E(|X_t - Y^k(T)|) \leq Ck^p$$

We shall see that the Euler method has strong convergence rate $p = \frac{1}{2}$.

We say that a method is *strongly consistent* if there is a function $c(k)$ with $\lim_{k \to 0^+} c(k) = 0$ such that

$$E\left(\left|E\left(\frac{Y^k_{n+1} - Y^k_n}{k} \middle| \mathcal{A}_{nk}\right) - a(nk, Y^k_n)\right|^2\right) \leq c(k) \tag{16}$$

$$E\left(\frac{1}{k}\left|Y^k_{n+1} - Y^k_n - E(Y^k_{n+1} - Y^k_n | \mathcal{A}_{nk}) - b(nk, Y^k_n)(W_{nk} - W_{(n-1)k})\right|^2\right) \leq c(k) \tag{17}$$

for all fixed values Y^k_n. Condition (16), in the absence of noise, is equivalent to the deterministic consistency condition. On the other hand, condition (17) gives an indication of pathwise closeness where the random parts of the method and the analytic solution converges to zero.

Theorem 4 A strongly consistent time discrete approximation Y^k of a 1-dimensional autonomous Ito process X with $Y^k(0) = X_0$ converges strongly to X.

Proof 4
For $0 \leq t \leq T$, let

$$Z(t) = \sup_{0 \leq s \leq t} E(|Y^k_{n_s} - X_s|^2)$$

We then have

$$
\begin{aligned}
Z(t) &= \sup_{0\le s\le t} E\left(\left|\sum_{n=0}^{n_s-1}(Y_{n+1}^k - Y_n^k) - \int_0^t a(X_r)\,dr - \int_0^t b(X_r)\,dW_r\right|^2\right)\\
&\le C_1 \sup_{0\le s\le t}\left\{E\left(\left|\sum_{n=0}^{n_s-1}(E(Y_{n+1}^k - Y_n^k|\mathcal{A}_{nk}) - a(Y_n^k)k)\right|^2\right)\right.\\
&\qquad + E\left(\left|\sum_{n=0}^{n_s-1}(Y_{n+1}^k - Y_n^k - E(Y_{n+1}^k - Y_n^k|\mathcal{A}_{kn}) - b(Y_n^k)(W_{(n+1)k} - W_{nk}))\right|^2\right)\\
&\qquad + E\left(\left|\int_0^{n_s k} a(Y_{n_r}^k) - a(X_r)\,dr\right|^2\right) + E\left(\left|\int_0^{n_s k} b(Y_{n_r}^k) - b(X_r)\,dW_r\right|^2\right)\\
&\qquad \left. + E\left(\left|\int_{n_s k}^{s} a(X_r)\,dr\right|^2\right) + E\left(\left|\int_{n_s k}^{s} b(X_r)\,dW_r\right|^2\right)\right\}\\
&\le C_1 \sup_{0\le s\le t}\left\{k^2\sum_{n=0}^{n_s-1} E\left(\left|E\left(\frac{Y_{n+1}^k - Y_n^k}{k}\middle|\mathcal{A}_{nk}\right) - a(Y_n^k)\right|^2\right)\right.\\
&\qquad + k\sum_{n=0}^{n_s-1} E\left(\frac{1}{k}\left|Y_{n+1}^k - Y_n^k - E(Y_{n+1}^k - Y_n^k|\mathcal{A}_{nk}) - b(Y_n^k)(W_{(n+1)k} - W_{nk})\right|^2\right)\\
&\qquad \left. + 2K^2\int_0^{n_s k} Z(r)\,dr + K^2 E\left(\int_{n_s k}^{s}(1+|X_r|^2)\,dr + \int_{n_s k}^{s}(1+|X_r|^2)\,dW_r\right)\right\}\\
&\le C_1 n_s k^2 c(k) + C_1 n_s k c(k) + 2K^2\int_0^t Z(r)\,dr + K^2 k(1+C_2)\\
&\le C_3\int_0^t Z(r)\,dr + C_1 T c(k)(k+1) + kK^2(1+C_2)\\
&\le C_3\int_0^t Z(r)\,dr + C_4(k + c(k))
\end{aligned}
$$

Now we use a lemma called Gronwall's inequality which is given by

Lemma 2 (Gronwall's Inequality) Let $\alpha,\ \beta : [t_0, T] \to \mathbb{R}$ be integrable with $0 \le \alpha(t) \le \beta(t) + L\int_{t_0}^t \alpha(s)\,ds$ for $t \in [t_0, T]$ where $L > 0$. Then

$$\alpha(t) \le \beta(t) + L\int_{t_0}^t e^{L(t-s)}\beta s\,ds$$

for $t \in [t_0, T]$.

We apply Gronwall's inequality to the inequality we have for $Z(t)$, namely

$$Z(t) \le C_3\int_0^t Z(r)\,dr + C_4(k + c(k))$$

Thus, we have

$$
\begin{aligned}
Z(t) &\le C_4(k + c(k)) + C_3\int_0^t e^{C_3(t-r)}C_4(k + c(k))\,dr\\
&\le C_5(k + c(k))
\end{aligned}
$$

Finally, recall that $Z(t) = E(|Y^k(t) - X_t|^2)$. Lyapunov's inequality states that $E(|X|^q) \leq (E(|X|^p))^{q/p}$, so

$$E(|Y^k(T) - X_T|) = \sqrt{Z(T)} \leq \sqrt{C_5(k + c(k))} \to 0$$

and therefore we get convergence.

16.4 Weak Convergence

A computed solution Y^k *converges weakly* to X_T as $k \to 0$ with respect to a class $\mathcal{C}$ of test functions $g : \mathbb{R} \to \mathbb{R}$ if

$$\lim_{k \to 0^+} |E(g(X_T)) - E(g(Y^k(T)))| = 0$$

for all $g \in \mathcal{C}$.

If $\mathcal{C}$ has just the function $g(x) = x$, then this reduces to the usual deterministic convergence criterion. A method is weakly consistent if there is a nonnegative function $c(k)$ with $\lim_{k \to 0^+}$ such that

$$E\left(\left|\left(\frac{Y^k_{n+1} - Y^k_n}{k}\middle| \mathcal{A}_{t_n}\right) - a(t_n, Y^k_n)\right|^2\right) \leq c(k) \tag{18}$$

$$E\left(\left|\left(\frac{1}{k}(Y^k_{n+1} - Y^k_n)^2\middle| \mathcal{A}_{t_n}\right) - b(t_n, Y^k_n)^2\right|^2\right) \leq c(k) \tag{19}$$

The first condition is the same as before, but the second condition is much weaker because only the variance of the increments must match the Ito process. That means we won't expect pathwise convergence. Note that the Euler-Maruyama method has strong order of convergence 1/2, and weak order of convergence 1.

Example 16.3:

An Euler method which is weakly consistent, but not strongly consistent is the method given by

$$Y_{n+1} = Y_n + a(t_n, Y_n)k + b(t_n, Y_n)\xi_n k^{1/2}$$

where the ξ_n are independent 2-point random variables with $P(\xi_n = \pm 1) = 1/2$.

16.5 Strong Taylor Approximations

We have already seen the 1/2 order Euler-Maruyama method. As in the deterministic case, let's take higher order terms in the Ito-Taylor expansion. In order to build up higher order Taylor approximations, we need to know the Taylor expansion, known as the Ito-Taylor expansion (there is also a corresponding Stratonovich calculus paralleling the Ito calculus, but we will concentrate here on just the Ito calculus).

The expansion is built for a particular Ito-process. Let X_t solves

$$dX_t = a(t, X_t)dt + b(t, X_t)dW_t$$

then

$$X_t = X_{t_0} + aI_{(0)} + bI_{(1)} + \left(aa' + \frac{1}{2}b^2a''\right)I_(0,0) + \left(ab' + \frac{1}{2}b^2b''\right)I_{(0,1)} + ba'I_{(1,0)} + bb'I_{(1,1)}$$
$$+ \text{higher order terms}$$

where

$$I_{(0,0)} = \int_{t_0}^{t}\int_{t_0}^{s} dr\, ds = \frac{1}{2}(t-t_0)^2$$
$$I_{(1,1)} = \int_{t_0}^{t}\int_{t_0}^{s} dW_r\, dW_s = \frac{1}{2}((W_t - W_{t_0})^2 - (t-t_0))$$
$$I_{(0,1)} = \int_{t_0}^{t}\int_{t_0}^{s} dr\, dW_s = ?$$
$$I_{(1,0)} = \int_{t_0}^{t}\int_{t_0}^{s} dW_r\, ds = ?$$

While we cannot compute explicit formulae for the last two integrals, we can say some things about their statistics. We will make use of that fact when we do higher order approximations. Note that a', b', etc. indicate partial derivatives with respect to x.

Note too, that when looking for the number of terms needed for a given accuracy, each 1 in the multi-index of I gives an $k^{1/2}$ and each 0 gives a full k. That means that to elevate the Euler-Maruyama method to a fully first order strongly convergent method, we need to include the term $bb'I_{(1,1)}$. This leads us to the Milstein method:

$$Y_{n+1} = Y_n + a(t_n, Y_n)k + b(t_n, Y_n)(W_{t_{n+1}} - W_{t_n}) + \frac{1}{2}b(t_n, Y_n)b_x(t_n, Y_n)((W_{t_{n+1}} - W_{t_n})^2 - k)$$

This method is 1.0 strongly convergent.

If we go to a higher order method, we will need to additionally include $I_{(0,0)}$, $I_{(0,1)}$, $I_{(1,0)}$, $I_{(1,1)}$, $I_{(1,1,1)}$ which will give us an order 1.5 method. The problem with this and higher order methods is that we don't have explicit formulae for many of these multiple Ito integrals. However, the statistics of these integrals can be determined, as well as relationships between the different integrals. To this end, given the Wiener process W, define

$$\Delta Z = I_{(1,0)} = \int_{t_n}^{t_n+k}\int_{t_n}^{s} dW_r\, ds$$

This random variable has statistics

$$E(\Delta Z) = 0, \quad E((\Delta Z)^2) = \frac{k^3}{3}, \quad \text{and } E(\Delta Z \Delta W) = \frac{1}{2}k^2$$

Clearly, ΔZ and ΔW are related. We can compute the pair $(\Delta Z, \Delta W$ from two independent random variables U_1, $U_2 \sim N(0;1)$ to get

$$\Delta W = U_1\sqrt{k}, \quad \Delta Z = \frac{1}{2}k^{3/2}\left(U_1 + \frac{1}{\sqrt{3}}U_2\right)$$

The strong 1.5 order method then becomes

$$Y_{n+1} = Y_n + ak + b\Delta W_n + \frac{1}{2}bb'((\Delta W_n)^2 - k) + a'b\Delta Z + \frac{1}{2}\left(aa' + \frac{1}{2}b^2a''\right)k^2$$
$$+ \left(ab' + \frac{1}{2}b^2b''\right)(k\Delta W - \Delta Z) + \frac{1}{2}b(bb'' + (b')^2)\left(\frac{1}{3}(\Delta W_n)^2 - k\right)\Delta W_n$$

16.6 Weak Ito-Taylor Expansions

By contrast to the previous method, we also construct the weak Ito-Taylor expansion methods using the Ito-Taylor expansion. For second order weak convergence, we need only include the double stochastic integrals

$I_{(0,0)}$, $I_{(0,1)}$, $I_{(1,0)}$, and $I_{(1,1)}$ to get a very similar method as the strongly convergent one

$$Y_{n+1} = Y_n + ak + b\Delta W_n + \frac{1}{2}bb'((\Delta W_n)^2 - k) + a'b\Delta Z + \frac{1}{2}\left(aa' + \frac{1}{2}b^2a''\right)k^2 + \left(ab' + \frac{1}{2}b^2b''\right)(k\Delta W - \Delta Z)$$

However, this is a much stronger method than is necessary for second order weak convergence. In particular, the computation of the two correlated random variables ΔW_n, ΔZ can be replaced by a single 3-point random variable ξ_n where

$$P(\xi = \pm\sqrt{3k}) = \frac{1}{6} \text{ and } P(\xi = 0) = \frac{2}{3}$$

We then get equations for ΔW_n, ΔZ given by

$$\Delta W_n = \xi_n, \text{ and } \Delta Z_n = \frac{1}{2}\xi_n k$$

Plugging this into the above approximation, we get

$$\begin{aligned} Y_{n+1} &= Y_n + ak + b\xi_n + \frac{1}{2}bb'((\xi_n)^2 - k) + a'b\frac{1}{2}\xi_n k + \frac{1}{2}\left(aa' + \frac{1}{2}a''b^2\right)k^2 \\ &\quad + \left(ab' + \frac{1}{2}b''b^2\right)\left(\xi_n k - \frac{1}{2}\xi_n k\right) \\ &= Y_n + ak + b\xi_n + \frac{1}{2}bb'((\xi_n)^2 - k) + \frac{1}{2}k\xi_n\left(ab' + \frac{1}{2}b''b^2 + a'b\right) + \frac{1}{2}\left(aa' + \frac{1}{2}a''b^2\right)k^2 \end{aligned}$$

Similarly, we can build up higher order methods for both weak and strong Ito-Taylor expansions.

16.7 Applications of Stochastic Differential Equations

In this section, we will look at a handful of applications of stochastic differential equations and the use of both strong and weak convergent methods.

16.7.1 Applications of Strong Approximations

Example 16.4:

[Investment Finance] A safe investment is an investment which grows at a deterministic rate, so if P_t is the price of a safe asset, then P_t solves the deterministic equation $dP_t = aP_t\,dt$ for some constant $a > 0$. Similarly, a risky investment would have a similar equation with multiplicative noise (the higher the price, the greater the fluxuations)

$$dQ_t = \alpha Q_t\,dt + \beta Q_t\,dW_t$$

Let us assume $\alpha > a$ since taking on risk should also have a potentially higher return. An investment portfolio should include some of each investment. Let f be the fraction of the portfolio invested in the risky investment, and hence $1 - f$ is the fraction invested in the safe investment. We can then write the total value of the portfolio, X_t by

$$\begin{aligned} dX_t &= f(\alpha X_t\,dt + \beta X_t\,dW_t) + (1-f)aX_t\,dt - c\,dt \\ &= (((1-f)a + f\alpha)X_t - c)\,dt + f\beta X_t\,dW_t \end{aligned}$$

where $c \geq 0$ is the consumption rate. The investor's problem is to optimize the portfolio distribution and consumption rates to optimize some objective function of the portfolio. This then becomes an optimal stochastic control problem.

Example 16.5:

[Satellite Orbital Stability] Suppose a satellite is moving in a circular orbit, and let $x(t)$ be the radial perturbation from the circular orbit. Fluctuations in the atmosphere density and other phenomena cause fluctuations in x. Sagirow proposed a model equation for x given by

$$\ddot{x} + b(1 + a\xi_t)\dot{x} + (1 + a\xi_t)\sin(x) - c\sin(2x) = 0$$

where a, b, c are constants with $b > 0$, $c > 0$. If we let $X_t^1 = x$ and $X_t^2 = \dot{x}$, then this can be written as a system of stochastic differential equations:

$$d\begin{bmatrix} X_t^1 \\ X_t^2 \end{bmatrix} = \begin{bmatrix} X_t^2 \\ -bX_t^2 - \sin(X_t^1) - c\sin(2X_t^1) \end{bmatrix} dt + \begin{bmatrix} 0 \\ -abX_t^2 - b\sin(X_t^1) \end{bmatrix} dW_t$$

Example 16.6:

[Stochastic Annealing] Suppose we wish to compute the global minimum of a given objective function $V : \mathbb{R}^d \to \mathbb{R}$. The gradient descent algorithm uses the equation

$$\dot{x} = -\nabla V(x)$$

The problem with this method is that it is dependent upon the initial condition and can result in finding only a local minimum instead of a global minimum.

One way to circumvent this problem is to include additive noise to this equation to get

$$dX_t = -\nabla V(X_t)\, dt + \sigma(t)\, dW_t$$

where dW_t is a d-dimensional Wierner process. If ∇V is uniformly Lipschitz and has growth bound

$$|\nabla V(x)|^2 \leq K(1 + |x|^2)$$

for some $K > 0$, then for the special choice of

$$\sigma(t) = \frac{c}{\sqrt{\log(t+2)}}$$

with $c > 0$ it can be shown that the distribution of X_t is proportional to $e^{-V(x)/T}$ where $T = \sigma(t)^2 \to 0$, $t \to 0$ and hence $X_t \to \bar{X}$ with the greatest probability where $\bar{X}$ is the global minimum.

16.7.2 Application of Weak Approximations

Example 16.7:

[Numerical Solution of Diffusion Equations] Consider the partial differential equation

$$u_t(t,x) + \frac{1}{2}\Delta u(t,x) + V(t,x)u(t,x) = 0 \tag{20}$$
$$u(T,x) = f(x)$$

for $0 \le t \le T$, $x \in \mathbb{R}^m$ where $V(t,x)$, $f(x)$ are given. Note that this problem has a terminal condition, not an initial condition.

Using the Frynman-Kac formula, we have a solution of the form

$$u(t,x) = E\left(f(W_T) e^{\int_t^T V(s,W_s)\,ds} \middle| W_t = x \right) \tag{21}$$

where $W = W_s, s \ge t, W_t = x$ is an m-dimensional Wiener process. Equations such as this arise, for example, in the analysis of wave scattering in random media.

If we evaluate equation (21) for all t, x, then we have solved equation (20). We can evaluate equation (21) by using a high order weak method to solve the stochastic differential system

$$dX_t^k = dW_s^k, \quad X_0^k = x^k, \quad \text{for } k = 1, \ldots, m$$
$$dX_t^{m+1} = V(s, X_s^1, X_s^2, \ldots, X_s^m)\,dt, \quad X_0^{m+1} = 0$$

Then,

$$u(t,x) = E(f(X_T^1, \ldots, X_T^m) e^{X_T^{m+1}})$$

Since we are only interested in the statistics, i.e. the expected value of a functional of X_t, then we can use the simpler weakly convergent methods to solve this system and achieve good results.

---

17 Vortex Methods

17.1 Analysis and origin of vortices

Vortex methods are used to model incompressible flow. To describe vortex methods, we will derive some equations which describe the evolution of vorticity in a flow.

The momentum equation from the Navier-Stokes equations is

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} = -\frac{1}{\rho}\nabla p + \frac{1}{R}\nabla^2 \vec{u}$$

The incompressibility condition is that $\nabla \cdot \vec{u} = 0$. We make the additional assumption related to incompressibility that the density ρ is constant.

Define the vorticity ξ by

$$\xi = \nabla \times \vec{u} = \left(\frac{\partial w}{\partial y} - \frac{\partial v}{\partial z}, \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x}, \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \right)$$

where $\vec{u} = (u, v, w)$. We will also need the vector identities

$$\frac{1}{2}\nabla(\vec{u} \cdot \vec{u}) = \vec{u} \times (\nabla \times \vec{u}) + (\vec{u} \cdot \nabla)\vec{u}$$
$$\nabla \times (\vec{u} \times \xi) = (\xi \cdot \nabla)\vec{u} - \xi(\nabla \cdot \vec{u}) - (\vec{u} \cdot \nabla)\xi + \vec{u}(nabla \cdot \xi)$$

Using the first identity in the momentum equation gives

$$\frac{\partial \vec{u}}{\partial t} + \frac{1}{2}(\nabla(\vec{u} \cdot \vec{u})) - \vec{u} \times (\nabla \times \vec{u}) = -\frac{1}{\rho}\nabla p + \frac{1}{R}\nabla^2 \vec{u}$$

If we take the curl of this equation we get

$$\frac{\partial(\nabla \times \vec{u})}{\partial t} + \frac{1}{2}\nabla \times (\nabla(\vec{u} \cdot \vec{u})) - \nabla \times (\vec{u} \times (\nabla \times \vec{u})) = -\frac{1}{\rho}\nabla \times \nabla p + \frac{1}{R}\nabla^2(\nabla \times \vec{u})$$

But $\nabla \times \nabla f = 0$, so

$$\nabla \times (\nabla(\vec{u} \cdot \vec{u})) = \nabla \times \nabla p = 0$$

and we are left with

$$\frac{\partial \xi}{\partial t} - \nabla \times (\vec{u} \times \xi) = \frac{1}{R}\nabla^2 \xi$$

Using the second identity, we get

$$\frac{\partial \xi}{\partial t} + (\vec{u} \cdot \nabla)\xi - (\xi \cdot \nabla)\vec{u} + \xi(\nabla \cdot \vec{u}) - \vec{u}(\nabla \cdot \xi) = \frac{1}{R}\nabla^2 \xi$$

From the incompressibility condition, we have $\nabla \cdot \vec{u} = 0$, and the vector identity $0 = \nabla \cdot (\nabla \times \vec{u}) = \nabla \cdot \xi$ gives

$$\frac{\partial \xi}{\partial t} + (\vec{u} \cdot \nabla)\xi - (\xi \cdot \nabla)\vec{u} = \frac{1}{R}\nabla^2 \xi$$

In two dimensions,

$$\xi = (\xi_1, \xi_2, \xi_3) = (0, 0, \xi_3) = \left(0, 0, \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)$$

Also, $(\xi \cdot \nabla)\vec{u} = \xi_3 \frac{\partial}{\partial z}\vec{u} = 0$. Thus, the equation becomes

$$\frac{\partial \xi}{\partial t} + (\vec{u} \cdot \nabla)\xi = \frac{1}{R}\nabla^2 \xi$$

If the flow is also inviscid, then we have the equation

$$\frac{\partial \xi}{\partial t} + (\vec{u} \cdot \nabla)\xi = 0$$

Theorem 5 The equation

$$\frac{\partial \xi}{\partial t} + (\vec{u} \cdot \nabla)\xi = 0$$

holds iff the vorticity is advected under its onw velocity field.

Proof 5
Let $(x(t), y(t))$ be the path of a particle starting at $(x(0), y(0)) = (x_0, y_0)$. The claim is that the vorticity is constant along the path $(x(t), y(t))$. We have

$$\begin{aligned}\frac{\partial}{\partial t}(\xi(x(t), y(t), t)) &= \xi_x \frac{dx}{dt} + \xi_y \frac{dy}{dt} + \xi_t \\ &= \xi_x u + \xi_y v + \xi_t \\ &= \xi_t + (\vec{u} \cdot \nabla)\xi = 0\end{aligned}$$

Thus, $\xi(x(t), y(t), t) = \xi(x_0, y_0, 0)$.

If the domain D of the flow is simply connected, and we use the incompressibility assumption $\nabla \cdot \vec{u} = 0$, then there is a *stream function* ψ such that

$$u = \psi_y, \quad v = -\psi_x$$

Theorem 6 For fixed t, level lines of ψ are parallel to the flow.

Proof 6
Fix t and let $\psi(x(s), y(s), t)$ be constant. Then

$$\frac{d\psi}{ds} = \psi_x x_s + \psi_y y_s = 0$$

Using $\psi_x = -v$, $\psi_y = u$, we get

$$0 = -vx_s + uy_s$$

Thus, $\psi \cdot \vec{u}^{\perp} = 0$ and hence ψ is parallel to u.

Next, we will find an equation for ψ. Notice that

$$\nabla^2 \psi = \psi_{xx} + \psi_{yy} = -v_x + u_y = -\xi_3(x, y, t)$$

The solution to this equation is

$$\psi(x, y, t) = \int_D \frac{1}{2\pi} \log ||(x, y) - (x', y')|| \xi(x', y', t) d(x', y')$$

From this we can now compute the velocity

$$u = \psi_y = \int_D \frac{y}{2\pi ||(x, y) - (x', y')||^2} \xi(x', y', t) d(x', y')$$
$$v = -\psi_x = \int_D \frac{-x}{2\pi ||(x, y) - (x', y')||^2} \xi(x', y', t) d(x', y')$$

If we also have no flow boundary conditions, then we must have $\vec{u} \cdot \vec{n} = 0$ where $\vec{n}$ is the normal to ∂D. To account for the boundary condition we look for a potential function ϕ such that

$$\nabla^2 \phi = 0$$
$$\left. \frac{\partial \phi}{\partial \vec{n}} \right|_{\partial D} = - \left((\psi_y, -\psi_x) \cdot \vec{n} \right)|_{\partial D}$$

so that $\frac{\partial \phi}{\partial \vec{n}}$ exactly cancels the velocity induced by the vorticity. The final velocity is then

$$\vec{u} = (\psi_y, -\psi_x) + \nabla \phi$$

Suppose we have an infinite domain and a single point vortex, i.e.

$$\xi(x, y, 0) = \begin{cases} 0 & (x, y) \neq 0 \\ \text{something} & (x, y) = 0 \end{cases}$$

then we get the stream function

$$\psi(x, y) = -\frac{k}{2\pi} \log ||(x, y)||$$

and the corresponding induced velocity is

$$\left(\frac{ky}{2\pi ||(x, y)||^2}, \frac{-kx}{2\pi ||(x, y)||^2} \right).$$

17.2 Point Vortex Method for Incompressible Inviscid Flow

In general, we saw that the stream function was given by

$$\psi(x,y,t) = \int_D G(x-x',y-y')\xi(x',y',t)d(x',y')$$

where $G(x,y) = \frac{1}{2\pi}\log||(x,y)-(x',y')||$ and the resulting induced velocity is

$$(u(x,y,t),v(x,y,t)) = \int_D K(x-x',y-y')\xi(x',y',t)d(x',y')$$

where $K(x,y) = \frac{(-y,x)}{2\pi||(x,y)||^2}$.

From this we can form the basis of a basic vortex method. Suppose the initial vorticity is given in a domain D. Discretize the domain into small cells, e.g. a square box divided into small squares. Within each cell i, the total vorticity is computed to give the strength of the vorticity in that cell

$$\Gamma_i = \int_{\text{cell}_i} \xi(x,y)d(x,y)$$

Next, we assume that the vorticity distribution is a sum of delta function distributions,

$$\xi(x,y) \approx \sum_{i=1}^{N} \Gamma_i \delta(||(x,y)-(x_i,y_i)||)$$

where (x_i,y_i) is the center of cell i. Now we can compute the induced velocity as

$$\begin{aligned}(u,v) &= \int_D K(x-x',y-y')\sum_{i=1}^{N}\Gamma_i\delta(||(x,y)-(x_i,y_i)||)d(x',y') \\ &= \sum_{i=1}^{N}\frac{\Gamma_i(-(y-y_i),(x-x_i))}{2\pi||(x,y)-(x_i,y_i)||^2}\end{aligned}$$

But what we need is only the velocity at the points (x_i,y_i). At the location (x_i,y_i), the vorticity there moves only according to the induced velocity field from the other vortices. Thus, the velocity field at (x_i,y_i) is (u_i,v_i) where

$$\begin{aligned}u_i &= \sum_{j\neq i}\frac{-\Gamma_j}{2\pi}\frac{y_i-y_j}{(x_i-x_j)^2+(y_i-y_j)^2} \\ v_i &= \sum_{j\neq i}\frac{\Gamma_j}{2\pi}\frac{x_i-x_j}{(x_i-x_j)^2+(y_i-y_j)^2}\end{aligned}$$

To complete the method, we need to handle the boundary conditions. Without viscosity, we can only impose one boundary condition on the sides. One typical solution is to use no-flow boundary conditions, i.e. $\frac{\partial\vec{u}}{\partial n} = 0$. To solve this, we must use a potential flow to balance the induced velocity at the boundary so that they cancel to give no normal velocity. Thus, we must solve,

$$\nabla^2\phi = 0$$

$$\frac{\partial\phi}{\partial n} = -(\psi_y,-\psi_x)\cdot\vec{n} \text{ on } \partial D$$

The potential function ϕ gives a potential velocity

$$(u_i^{\text{pot}},v_i^{\text{pot}}) = \nabla\phi$$

From this we can define a numerical method:

1. Discretize the domain and compute ξ_i, Γ_i, x_i, and y_i from the initial conditions.
2. Compute the induced velocity field (u, v) at (x_i, y_i) and on ∂D.
3. Solve the Poisson problem $\nabla^2\phi = 0$, $\frac{\partial\phi}{\partial n} = -(u, v)$ on ∂D. Then, $(u_i^{\text{pot}}, v_i^{\text{pot}}) = \nabla\phi(x_i, y_i)$.
4. Move the vortices by Euler's method
$$x_i^{n+1} = x_i^n + k(u_i + u_i^{\text{pot}})$$
$$y_i^{n+1} = y_i^n + k(v_i + v_i^{\text{pot}})$$
5. Go back to step 2.

One problem with this method occurs when two vortices approach each other. This will give large velocities which is unrealistic. To handle this, Chorin created the vortex blob method to deal with this problem. It cuts off the maximum velocity but also creates a time step restriction. Hald proved that the vortex blob method converges to the exact solution.

Another problem arises when a large number of vortices are floating around. When viscosity is added to the mix, we will see that new vortices can be created, so the number of vortices can be quite large. One way to handle this is to compute the vorticity on a fixed mesh using the equation

$$\frac{\partial\xi}{\partial t} + (\vec{u}\cdot\nabla)\xi = 0$$

A straigtforward discretization is then

$$D^+\xi_{jk}^n = -u_{jk}^n D_{x0}\xi_{jk}^n - v_{jk}^n D_{y0}\xi_{jk}^n$$

where u_{jk}^n, v_{jk}^n are the computed induced velocity plus potential velocity.

A less dramamtic speed-up is to compute the velocity on a fixed mesh instead of at each vortex. The vortex velocities are then interpolated from the mesh values. This method is called the *particle in cell method* and can be faster if the number of grid points is much less than the number of vortices.

The point vortex and particle in cell methods can be combined for better accuracy by noting that the velocity due to vorticity drops off as $\frac{1}{r^2}$. Thus, we can make a local correction by exchanging the local interpolated velocity for the real two point vortex interaction, but only for vortices that are nearby. This is called the *Local Correction Method*.

17.3 Point Vortex Method for Incompressible Viscous Flow

Now we will add in viscosity. The equation becomes

$$\frac{\partial\xi}{\partial t} + (\vec{u}\cdot\nabla)\xi = \frac{1}{R}\nabla^2\xi$$

Adding viscosity will also require an additional boundary condition. the additional boundary condition we will add is the no-slip condition. The *no-slip condition* is $\vec{u}\cdot\tau = 0$ where τ is the unit tangent to the boundary ∂D. This additional condition can be the source of new vorticity. The amount of vorticity depends on the Reynolds number R. Viscosity also means the vorticity will be diffused. To see how to handle this in the framework of the point vortex method, we will take a little side-trip.

17.3.1 Particle Methods for Diffusion Equations

Suppose we have the heat equation

$$u_t = \frac{1}{R}u_{xx}, \quad u(x, 0) = \delta(x)$$

The exact solution is

$$u(x,t) = \frac{1}{\sqrt{4\pi t/R}} e^{\frac{-x^2}{4t/R}}$$

But this is a Gaussian distribution with mean zero and variance $\frac{2t}{R}$. The same result would arise if N particles are placed at $x = 0$, each with mass $1/N$ and then each particle takes a random jump with mean zero and variance $2t/R$. The amount of heat particles between x and $x + \Delta x$ will be the Gaussian. Now let each particle jump with mean zero and variance $2\Delta t/R$, m times, to get $u(x, m\Delta t)$. This is a way to solve the equation.

17.3.2 Application of Particle Methods to Point Vortex method

Applying this to vortices is straightforward. We assume operator splitting:

$$\xi_t = -(\vec{u} \cdot \nabla)\xi \tag{22}$$

$$\xi_t = \frac{1}{R}\nabla^2 \xi \tag{23}$$

$$\tag{24}$$

The first equation we solve using the techniques described above. The second is solved using the random jump method. Thus, each vortex undergoes a random jump (η_x, η_y) where (η_x, η_y) are drawn from a Gaussian distribution with mean zero and variance $2\Delta t/R$.

Putting it together, we now get

$$\begin{aligned} x_i^{n+1} &= x_i^n + k(u_i + u_i^{\text{pot}}) + \eta_x \\ y_i^{n+1} &= y_i^n + k(v_i + v_i^{\text{pot}}) + \eta_y \end{aligned} \tag{25}$$

We next have to see how to deal with the no-slip boundary condition. To see how to handle this, imagine a stationary fluid on top of an infinitely long flat plate. The plate is then moved along its length at speed U. To see how much vorticity is generated, we must balance the fluid velocity on either side of the plate. Now the vorticity at the plate is given by

$$\xi = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} = -\frac{\partial u}{\partial y} = -2u\delta(y)$$

because the no-flow condition gives $v = 0$ at the plate. Thus, we get a new vortex with strength $-2Uds$ where the boundary is discretized into segments of length ds.

The algorithm above for the point vortex method for inviscid flow can now be amended to include viscosity effects.

1. Initialize the vortices from the initial conditions
2. Compute the induced velocity field $\vec{u}$.
3. Solve the Poisson problem $\nabla^2\phi = 0$, $\frac{\partial\phi}{\partial n} = -\vec{u} \cdot \vec{n}$.
4. Compute the tangential velocity of $\vec{u} + \nabla\phi$ at ∂D.
5. Create new vorticies at the boundary with strength $-2Uds$ to offset the tangential flow at the boundary.
6. Advect and diffuse the vortices using equations 25.
7. Go back to step 2

18 Intro to Numerical Methods for Elliptic Equations

The solution of elliptic equations numerically is in the realm of numerical linear algebra where in the end we must solve a linear system

$$Ax = b.$$

Different numerical methods result in a different A and b, but in the end, a linear system must be solved. In general, A is typically large and sparse.

To discuss how to solve these equations, we will begin by discussing different ways that the matrix A is constructed. We then conclude with methods for solving linear systems.

18.1 Finite Difference Method

Consider the elliptic equation

$$\nabla \cdot (\beta \nabla u) + \kappa u = f$$

where β, κ, f are given functions of x. In one dimension, this equation is

$$\begin{aligned} \frac{\partial}{\partial x}\left(\beta \frac{\partial u}{\partial x}\right) + \kappa u &= f \\ u(0) = a, \quad \frac{\partial u}{\partial x}(1) &= 0. \end{aligned}$$

Here, we have chosen representative boundary conditions.

The discretization of this equation is done in much the same way we would solve the heat equation. Let us concentrate on the first term $\frac{\partial}{\partial x}\left(\beta \frac{\partial u}{\partial x}\right)$. How should this be discretized? A first attempt could be

$$\begin{aligned} & D_+(\beta_j D_- u_j) \\ &= D_+\left(\beta_j \frac{1}{h}(u_j - u_{j-1})\right) \\ &= \frac{1}{h}\left(\beta_{j+1}\frac{1}{h}(u_{j+1} - u_j) - \beta_j \frac{1}{h}(u_j - u_{j-1})\right) \\ &= \frac{1}{h^2}(\beta_{j+1}u_{j+1} - (\beta_j + \beta_{j+1})u_j + \beta_j u_{j-1}) \end{aligned}$$

Let's check the accuracy of this approximation:

$$\begin{aligned} \tau &= \frac{\partial}{\partial x}\left(\beta \frac{\partial u}{\partial x}\right) - \frac{1}{h^2}(\beta(x+h)u(x+h) - (\beta(x) + \beta(x+h))u(x) + \beta(x)u(x-h)) \\ &= \frac{\partial}{\partial x}(\beta u_x) - \frac{1}{h^2}\left(\left(\beta + h\beta_x + \frac{h^2}{2}\beta_{xx} + \frac{h^3}{6}\beta_{xxx}\right)\left(u + hu_x + \frac{h^2}{2}u_{xx}\right)\right. \\ &\qquad \left. - \left(\beta + \beta + h\beta_x + \frac{h^2}{2}\beta_{xx} + \frac{h^3}{6}\beta_{xxx}\right)u + \beta\left(u - hu_x + \frac{h^2}{2}u_{xx} - \frac{h^3}{6}u_{xxx}\right)\right) \\ &= \beta_x u_x + \beta u_{xx} - (u_x\beta_x + \beta u_{xx} + \frac{h}{2}(u_x\beta_{xx} + \beta_x u_{xx}) + O(h^2)) \\ &= -\frac{h}{2}(u_x\beta_{xx} + \beta_x u_{xx}) + O(h^2). \end{aligned}$$

Thus, this is a first order method.

Suppose we had instead approximated the expression with

$$D_-(\beta_j D_+ u_j) = \frac{1}{h^2}(\beta_j u_{j+1} - (\beta_j + \beta_{j-1})u_j + \beta_{j-1}u_{j_1})$$

then we would again get a first order method. If we balance by taking half of each expression, we get better results:

$$\begin{aligned}
&\frac{1}{2}(D_-(\beta_j D_+ u_j) + D_+(\beta_j D_- u_j)) \\
&= \frac{1}{2h^2}(\beta_{j+1}u_{j+1} - (\beta_j + \beta_{j+1})u_j + \beta_j u_{j-1} + \beta_j u_{j+1} - (\beta_j + \beta_{j-1})u_j + \beta_{j-1}u_{j-1}) \\
&= \frac{1}{2h^2}((\beta_j + \beta_{j+1})u_{j+1} - (\beta_{j+1} + 2\beta_j + \beta_{j-1})u_j + (\beta_{j-1} + \beta_j)u_{j-1})
\end{aligned}$$

Computing the truncation in this case results in a second order method.

Putting it all together, we now have a linear system of equations to solve

$$\begin{gathered}
u_0 = a \\
\frac{1}{h^2}((\beta_j + \beta_{j+1})u_{j+1} - (\beta_{j+1} + 2\beta_j + \beta_{j-1})u_j + (\beta_{j-1} + \beta_j)u_{j-1}) + \kappa_j u_j = f_j, \quad \text{for } j = 1,\ldots, N-1 \\
u_N - u_{N-1} = 0
\end{gathered}$$

which gives the matrix A and right hand side b.

18.2 Finite Element Method

The finite element method has a large a rich theory of its own of which we will only scratch the surface. In the finite element method, the solution is constructed from a collection of basis functions, called *trial functions*, and a separate set of functions called *test functions*. These functions arise when constructing the weak form of the equation to be solved.

Consider again our basic elliptic equation

$$\frac{\partial}{\partial x}\left(\beta\frac{\partial u}{\partial x}\right) + \kappa u = f.$$

To get the weak form, multiply this equation by a test function ϕ and integrate over the domain.

$$\int_0^1 \phi\frac{\partial}{\partial x}\left(\beta\frac{\partial u}{\partial x}\right) + \phi\kappa u\,dx = \int_0^1 f\phi\,dx.$$

The idea of weak formulations is to transfer the differentiability of the solution to the test functions. This opens the space of possible solutions to functions which may not be differentiable. This transfer of differentiability is done through integration by parts. In this case, we get

$$\phi\beta\frac{\partial u}{\partial x}\bigg|_0^1 - \int_0^1 \beta\frac{\partial\phi}{\partial x}\frac{\partial u}{\partial x}\,dx + \int_0^1 \phi\kappa u\,dx = \int_0^1 f\phi\,dx.$$

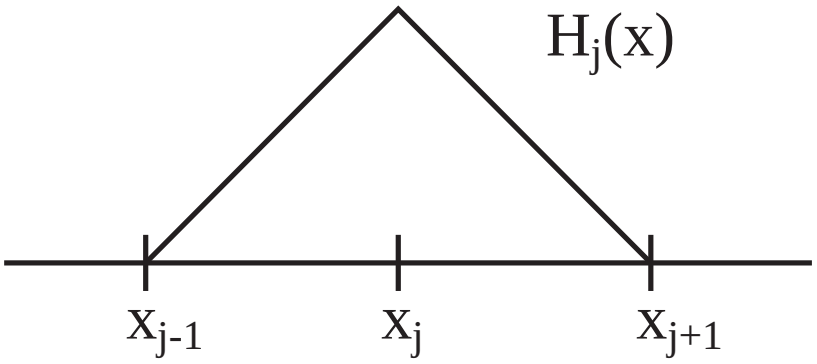
Note how the solution u is no longer required to be twice differentiable.

Next, the solution u is written as a linear combination of basis functions

$$u(x) = \sum_{j=0}^{N} u_j H_j(x)$$

where the u_j are scalar coefficients and the $H_j(x)$ are the basis functions. One easy choice for basis functions is a hat function where

$$H_j(x) = \begin{cases} \frac{1}{h}(x - x_{j-1}) & x_{j-1} \le x \le x_j \\ -\frac{1}{h}(x - x_{j+1}) & x_j < x \le x_{j+1} \\ 0 & \text{otherwise} \end{cases}.$$



First order basis function

Plugging this expansion for u into the weak formulation produces

$$\sum_{j=0}^{N} u_j \phi \beta \frac{\partial H_j}{\partial x}\bigg|_0^1 - \int_0^1 u_j \beta \frac{\partial \phi}{\partial x}\frac{\partial H_j}{\partial x}\,dx + \int_0^1 \phi \kappa u_j H_j\,dx = \int_0^1 f\phi\,dx.$$

Thus, we get a different equation for each choice of test function ϕ, and there are $N+1$ unknowns $u_0,\ldots,$ u_N, so we should choose $N+1$ different test functions. One choice for the ϕ functions are the basis functions again. This makes this a Galerkin finite element mtehod.

Suppose we take $\phi = H_i$, then we have

$$\sum_{j=0}^{N} u_j H_i \beta \frac{\partial H_j}{\partial x}\bigg|_0^1 - \int_0^1 u_j \beta \frac{\partial H_i}{\partial x}\frac{\partial H_j}{\partial x}\,dx + \int_0^1 H_i \kappa u_j H_j\,dx = \int_0^1 f H_i\,dx. \tag{26}$$

If $0 < i,\, j < N$, then we get that

$$\begin{aligned} H_i(0) = H_i(1) &= 0, \\ \int_0^1 u_j \beta \frac{\partial H_i}{\partial x}\frac{\partial H_j}{\partial x}\,dx &= \begin{cases} \int_{x_{j-1}}^{x_{j+1}} u_j \beta \frac{\partial H_i}{\partial x}\frac{\partial H_j}{\partial x}\,dx & |i-j| \le 1 \\ 0 & \text{otherwise} \end{cases}, \\ \int_0^1 H_i \kappa u_j H_j\,dx &= \begin{cases} \int_{x_{j-1}}^{x_{j+1}} \kappa u_j H_i H_j\,dx & |i-j| \le 1 \\ 0 & \text{otherwise} \end{cases}. \end{aligned}$$

Thus, (26) reduces to

$$\sum_{j=i-1}^{i+1} -\int_{x_{j-1}}^{x_{j+1}} u_j \beta \frac{\partial H_i}{\partial x}\frac{\partial H_j}{\partial x}\,dx + \int_{x_{j-1}}^{x_{j+1}} H_i \kappa u_j H_j\,dx = \int_{x_{j-1}}^{x_{j+1}} f H_i\,dx. \tag{27}$$

Now, for simplicity, let us assume β, κ, f are constant, then the integrands simplify and we can explicitly compute their values

$$\int_0^1 \frac{\partial H_i}{\partial x}\frac{\partial H_{i-1}}{\partial x}\,dx = \int_{x_{i-1}}^{x_i} \frac{1}{h}\left(-\frac{1}{h}\right)\,dx = -\frac{1}{h^2}(x_i - x_{i-1}) = -\frac{1}{h},$$

$$\int_0^1 \frac{\partial H_i}{\partial x}\frac{\partial H_i}{\partial x}\,dx = \int_{x_{i-1}}^{x_{i+1}} \frac{1}{h^2}\,dx = \frac{1}{h^2}(2h) = \frac{2}{h},$$

$$\int_0^1 H_i H_{i-1}\,dx = \int_{x_{i-1}}^{x_i} \frac{1}{h}(x - x_{i-1})\left(-\frac{1}{h}\right)(x - x_i)\,dx$$
$$= -\frac{1}{h^2}\int_{x_{i-1}}^{x_i} x^2 - (x_i + x_{i-1})x + x_i x_{i-1}\,dx$$
$$= \frac{1}{6h^2}(x_i - x_{i-1})^3 = \frac{h}{6},$$

$$\int_0^1 H_i^2\,dx = \int_{x_{i-1}}^{x_i} \frac{1}{h^2}(x - x_{i-1})^2\,dx + \int_{x_i}^{x_{i+1}} \frac{1}{h^2}(x - x_{i+1})^2\,dx$$
$$= \frac{2h}{3},$$

and

$$\int_0^1 H_i\,dx = \frac{1}{2}(x_{i+1} - x_{i-1}) = h.$$

Putting these expressions into (27) gives

$$-u_{i-1}\beta\left(-\frac{1}{h}\right) - u_i\beta\frac{2}{h} - u_{i+1}\beta\left(-\frac{1}{h}\right) + \kappa u_{i-1}\frac{h}{6} + \kappa u_i\frac{2h}{3} + \kappa u_{i+1}\frac{h}{6} = fh$$
$$\frac{\beta}{h^2}(u_{i-1} - 2u_i + u_{i+1}) + \frac{\kappa}{6}(u_{i-1} + 4u_i + u_{i+1}) = f \tag{28}$$

Note that (28) looks very close to the method we derived from finite differences. The first term is obviously $D_+D_-u_j$ which is second order accurate. We also have

$$\frac{1}{6}(u_{j-1} + 4u_j + u_{j+1}) = \frac{1}{6}\left(u - hu_x + \frac{h^2}{6}u_{xx} + 4u + u + hu_x + \frac{h^2}{2}u_{xx}\right) = u + \frac{h^2}{6}u_{xx}.$$

Thus, the second term is a second order approximation for u and hence we again obtain a second order numerical method.

Now if β, κ, f, are not constant, we could approximate them by taking, for example,

$$\beta = \sum_{j=0}^{N} \beta_j H_j(x).$$

We then have more complicated integrals such as

$$\int_0^1 H_{i-1}\frac{\partial H_i}{\partial x}\frac{\partial H_{i-1}}{\partial x}\,dx = \int_{x_{i-1}}^{x_i} -\frac{1}{h}(x - x_i)\frac{1}{h}\left(-\frac{1}{h}\right)\,dx = \frac{h}{2}.$$

18.3 Solving the Linear System

We can see now that regardless of which method we are describing, the net result is that we must solve a large, usually sparse, linear system of the form

$$Ax = b.$$

For purposes of developing the method, let us split the matrix A into

$$A = L + D + U$$

where L is the lower triangle of A, D is the diagonal, and U is the upper triangle of A.

Let us consider Jacobi's method. In Jacobi's method, we have

$$\begin{aligned} Ax &= b \\ (L+D+U)x &= b \\ Dx &= -(L+U)x + b. \end{aligned} \tag{29}$$

Turning (29) into an iterative method, we get

$$Dx^(r+1) = -(L+U)x^(r) + b$$

where the superscript indicates the iterate ove x. One advantage of this method is that it is relatively easy to implement because it requires only the inversion of D which is a diagonal matrix.

Note that one important feature of discretizations for elliptic equations is that they are typically *diagonally dominant*. A matrix A is diagonally dominant if for each row i of A,

$$A_{ii} \geq \sum_{j \neq i} A_{ij}.$$

Diagonal dominance gaurantees that D not have any zero entries on the diagonal.

The rate of convergence of the Jacobi method is governed by the iterating matrix

$$M = -D^{-1}(L+U).$$

For convergence, it must be the case that all the eigenvalues of M be less than 1, for if not, there is a growth mode which will prevent convergence. The convergence rate is determined by the slowest decaying mode which is given by the largest magnitude eigenvalue of M. This value is called the *spectral radius*, ρ_s, of M. Roughly speaking, a matrix M with spectral radius ρ_s will require $\ln(10^{-p})/\ln(\rho_s)$ iterations in order to reduce the error in solving for x by a factor of 10^{-p}. For many applications of Jacobi's method, in two dimensions, the spectral radius of M turns out to be

$$\rho_s \approx 1 - \frac{A}{J^2}$$

for some constant $A > 0$. This means that reducing the error in the solution by a factor of 10^{-p} will require

$$r \approx \frac{\ln(10^{-p})}{\ln\left(1 - \frac{A}{J^2}\right)} \approx \frac{-p\ln(10)}{-A/J^2} = \frac{J^2 p \ln(10)}{A} = CpJ^2$$

for some constant C. Here, we have used the approximation $\ln(1-\epsilon) \approx -\epsilon$. This means the number of iterations necessary to reduce the error by a factor of 10 requires the same order of iterations as the number of grid points. This is too computationally expensive to be of use.

The Gauss-Seidel method is given by

$$(L+D)x^{(r+1)} = -Ux^{(r)} + b.$$

This method suffers from the same problems as the Jacobi method in terms of convergence. However, the Gauss-Seidel method can be rescued by increasing the amount of relaxation. Define the residual vector $\xi^{(r)} = Ax^{(r)} - b$. Now, the Gauss-Seidel method becomes

$$\begin{aligned} (L+D)x^{(r+1)} &= (L+D)x^{(r)} - ((L+D+U)x^{(r)} - b) \qquad (30) \\ &= (L+D)x^{(r)} - \xi^{(r)} \\ x^{(r+1)} &= x^{(r)} - (L+D)^{-1}\xi^{(r)} \qquad (31) \\ (L+D)(\delta x^{(r+1)}) &= -\xi^{(r)}. \end{aligned}$$

This shows that the residual gives the correction term for computing the solution x. The Simultaneous Over-Relaxation method (SOR) uses a relaxation parameter ω in (31) to get

$$x^{(r+1)} = x^{(r)} - \omega(L+D)^{-1}\xi^{(r)}.$$

In many cases, this leads to improved performance.

Choosing ω is not easy, however, if the spectral radius for Jacobi's method is known, then the optimal choice for ω is given by

$$\omega = \frac{2}{1+\sqrt{1-\rho^2_{\text{Jacobi}}}}.$$

This leads to the spectral radius of the SOR method

$$\rho_{\text{SOR}} = \left(\frac{\rho_{\text{Jacobi}}}{1+\sqrt{1-\rho^2_{\text{Jacobi}}}}\right)^2.$$

Approximate to get $\omega \approx 2/(1+C/J)$, $\rho_{\text{SOR}} \approx 1 - \frac{A}{J}$, which leads to $r = C\rho J$.

Other methods are the conjugate gradient method, which is also an iterative method, but where it can be accelerated by proper choice of a preconditioner matrix.