ES_APPM 446-2 Notes
Spectral Methods for Partial Differential Equations

Prof. David Chopp

Spring 2008

Contents

1 Introduction of Spectral Methods

This quarter we will discuss spectral methods for solving partial differential equations. Last quarter we used finite differences to solve equations such as

$$u_t = u_x.$$

Spectral methods are an alternative way to approximate spatial derivatives such as u_x.

Spectral methods break down into two steps. First, the function u is approximated using a finite series. Fourier series are used for periodic functions and Chebyshev expansions otherwise. Second, the finite series approximation is explicitly differentiated.

To put this in perspective to the finite difference methods we employed last quarter, suppose that we approximate $u(x)$ with a quadratic polynomial with nodes at $(j-1)h$, jh, and $(j+1)h$. The polynomial approximation $p(x)$ must satisfy $p(jh) = u(jh) = u_j$, likewise $p((j \pm 1)h) = u_{j\pm 1}$. Solving for the coefficients, we get

$$\begin{aligned} p(x) = \frac{1}{2h^2}(u_{j+1} - 2u_j + u_{j-1})x^2 + \frac{1}{h}\left[\frac{1}{2}(u_{j+1} - u_{j-1}) - j(u_{j+1} - 2u_j + u_{j-1})\right] x \\ + u_j - \frac{j}{2}(u_{j+1} - u_{j-1}) + \frac{j^2}{2}(u_{j+1} - 2u_j + u_{j-1}) \end{aligned}$$

We now differentiate $p(x)$ at $x = jh$ to get an approximation for $u_x(jh)$. If we do this, we get

$$\begin{aligned} p'(jh) &= \frac{1}{h^2}(u_{j+1} - 2u_j + u_{j-1})(jh) + \frac{1}{h}\left[\frac{1}{2}(u_{j+1} - u_{j-1}) - j(u_{j+1} - 2u_j + u_{j-1})\right] \\ &= \frac{1}{2h}(u_{j+1} - u_{j-1}) \end{aligned}$$

We recognize this as $D_0 u_j$ which is the central finite difference approximation for u_x. Likewise, we can compute

$$p''(jh) = \frac{1}{h^2}(u_{j+1} - 2u_j + u_{j-1}) = D_+ D_- u_j \approx u_{xx}(jh)$$

or the standard three point stencil for the second derivative.

Assume that we are solving on an interval $[0, 2\pi]$, and N points are used to divide up the interval, so $h = \frac{2\pi}{N}$. We saw last quarter that the truncation error for $D_0 u_j$ is given by

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

where $\frac{h^2}{6} u_{xxx}$ is the *truncation error*. Note that the approximation is exact if $u(x)$ is a quadratic polynomial. Otherwise, the error drops like $O(1/N^2)$.

Now let us compare to a Fourier approximation. Suppose $u(x)$ is periodic on the interval $[0, 2\pi]$. We can represent the function $u(x)$ by a Fourier series

$$u(x) = \sum_{\ell=-\infty}^{\infty} a_\ell e^{i\ell x}$$

Now suppose we take a finite number of terms, say

$$u(x) \approx u_N(x) = P_N u(x) = \sum_{\ell=-N}^{N} a_\ell e^{i\ell x}.$$

The operator P_N is a projection operator called the *spectral L_2 operator* or the *spectral Galerkin projector*. The derivative is then approximated by

$$u_x \approx u_{N;x}(x) = \sum_{\ell=-N}^{N} i\ell a_\ell e^{i\ell x}.$$

It is reasonable to now ask what the order of accuracy of this method must be. Suppose that $u(x)$ is C^m where m may be $+\infty$ and let r be any integer $r \leq m$. we know that the coefficients of the Fourier expansion are given by

$$a_\ell = \frac{1}{2\pi}\int_0^{2\pi} u(y)e^{-i\ell y}\,dy$$

If we integrate by parts, we get

$$\begin{aligned}
&= \frac{1}{2\pi}\left[\frac{-1}{i\ell}u(y)e^{-i\ell y}|_0^{2\pi} + \int_0^{2\pi}\frac{1}{i\ell}\frac{du}{dx}(y)e^{-i\ell y}\,dy\right]\\
&= \frac{1}{2\pi}\left[\frac{-1}{i\ell}(u(2\pi)-u(0)) + \int_0^{2\pi}\frac{1}{i\ell}\frac{du}{dx}(y)e^{-i\ell y}\,dy\right]\\
&= \frac{1}{2\pi}\frac{1}{i\ell}\int_0^{2\pi}\frac{du}{dx}(y)e^{-i\ell y}\,dy\\
&\quad\vdots\\
&= \frac{1}{2\pi}\frac{1}{(i\ell)^r}\int_0^{2\pi}\frac{d^r u}{dx^r}(y)e^{-i\ell y}\,dy
\end{aligned}$$

Thus,

$$|a_\ell| \leq \frac{C}{\ell^r}$$

for some constant C.

To compute the error for this method, we will look for the largest error on the whole interval $[0, 2\pi]$. Therefore, the error is measured by

$$\begin{aligned}
\max_{x\in[0,2\pi]}|u_x(x) - P_N u(x)| &= \max_{x\in[0,2\pi]}\left|\sum_{|\ell|>N} i\ell a_\ell e^{i\ell x}\right|\\
&\leq \max_{x\in[0,2\pi]}\sum_{|\ell|>N}\ell|a_\ell|\\
&\leq \max_{x\in[0,2\pi]}\sum_{|\ell|>N}\ell\frac{C}{\ell^r}\\
&= O\left(\frac{1}{N^{r-2}}\right)
\end{aligned}$$

To see this last statement, look at $\int_N^{+\infty}\frac{1}{x^r}\,dx$. Therefore, the order of the approximation of u_x is limited only by the smoothness of the function u. If u happens to be C^∞, then this has what is sometimes called *infinite order accuracy*, i.e. convergence is faster than $O(1/N^r)$ for any r. In contrast, the finite difference method was only able to muster $O(1/N^2)$ regardless of the smoothness of the function.

This gives us the motivation to investigate and understand spectral methods further to see what they have to offer and what the pitfalls may be. Lec. 1

2 Approximation Properties of Fourier Series

2.1 Basic Properties of Fourier Space

We are now going to make the example above more precise. Let $L_2(0, 2\pi)$, (or L_2 for short) be the space of all complex valued square integrable periodic functions on the interval $[0, 2\pi]$. Clearly L_2 is a linear infinite dimensional vector space. Also, we can define the inner product on L_2 by

$$\langle u, v\rangle = \int_0^{2\pi} u(x)\bar{v}(x)\, dx.$$

The L_2 norm then becomes

$$||u||^2 = \langle u, u\rangle = \int_0^{2\pi} |u(x)|^2\, dx$$

The addition of the inner product means that L_2 is a Hilbert space. Therefore, we can express any element in L_2 as a linear combination of a linearly independent set of basis functions $\{\phi_j(x)\}_{j=-\infty}^{+\infty}$. Recall that a set $\{\phi_j(x)\}_{j=-\infty}^{+\infty}$ is called orthogonal if

$$\langle \phi_i, \phi_j\rangle = 0 \quad \text{for } i \neq j.$$

The basis is orthonormal if also $||\phi_i|| = 1$ for all i.

Consider the functions $\{e^{ijx}\}_{j=-\infty}^{+\infty}$. We have

$$\begin{aligned}\langle e^{ikx}, e^{ijx}\rangle &= \int_0^{2\pi} e^{ikx}e^{-ijx}\, dx\\ &= \int_0^{2\pi} e^{i(k-j)x}\, dx\\ &= \begin{cases} 2\pi & k = j\\ 0 & \text{otherwise}\end{cases}\end{aligned}$$

Thus, the set $\{e^{ijx}\}_{j=-\infty}^{+\infty}$ is mutually orthogonal, but not orthonormal (though that could be remedied with an appropriate scaling of $1/\sqrt{2\pi}$). In fact, Fourier series theory states that the set $\{e^{ijx}\}_{j=-\infty}^{+\infty}$ is also complete. This means that for any $u \in L_2$, we can write

$$u(x) = \sum_{k=-\infty}^{+\infty} a_k e^{ikx}$$

To determine the coefficients a_k, we take the inner product to get

$$\begin{aligned}\langle u, e^{ijx}\rangle &= \left\langle \sum_{k=-\infty}^{+\infty} a_k e^{ikx}, e^{ijx}\right\rangle\\ &= \sum_{k=-\infty}^{+\infty} a_k \langle e^{ikx}, e^{ijx}\rangle\\ &= 2\pi a_j\end{aligned}$$

Thus,

$$a_j = \frac{1}{2\pi}\langle u, e^{ijx}\rangle = \frac{1}{2\pi}\int_0^{2\pi} u(x)e^{-ijx}\, dx$$

The convergence of the sum is in norm, in other words,

$$\left\| u - \sum_{k=-N}^{N} a_k e^{ikx} \right\| \to 0, \quad \text{as } N \to 0. \tag{1}$$

Now we define the operator P_N where

$$P_N u = \sum_{k=-N}^{N} a_k e^{ikx}.$$

The operator P_N has the property that $P_N^2 = P_N$, and hence P_N is a projection operator. In terms of P_N, we can rewrite equation (1) as

$$\lim_{N\to\infty} ||u - P_N u|| = 0$$

Next, note that

$$\begin{aligned}
||P_N u||^2 &= \langle P_N u, P_N u \rangle \\
&= \left\langle \sum_{k=-N}^{N} a_k e^{ikx}, \sum_{j=-N}^{N} a_j e^{ijx} \right\rangle \\
&= \sum_{k=-N}^{N} a_k \sum_{j=-N}^{N} \bar{a}_j \langle e^{ikx}, e^{ijx} \rangle \\
&= \sum_{k=-N}^{N} a_k \bar{a}_k 2\pi \\
&= 2\pi \sum_{k=-N}^{N} |a_k|^2
\end{aligned}$$

This is the finite dimensional version of Parseval's equation

$$||u||^2 = 2\pi \sum_{k=-\infty}^{\infty} |a_k|^2$$

To prove this result, we begin by using the Schwarz' inequality $|\langle u, v \rangle| \leq ||u||\,||v||$. We then have

$$\begin{aligned}
||u - v||^2 &= \langle u - v, u - v \rangle \\
&= ||u||^2 + ||v||^2 - \langle u, v \rangle - \langle v, u \rangle \\
&\geq ||u||^2 + ||v||^2 - 2|\langle u, v \rangle| \\
&\geq ||u||^2 + ||v||^2 - 2||u||\,||v|| \\
&= (||u|| - ||v||)^2
\end{aligned}$$

Recall that we had $\lim\limits_{N\to\infty} ||u - P_N u|| = 0$, so therefore we have $\lim\limits_{N\to\infty} ||u|| - ||P_N u|| = 0$. Finally, we have

$$\begin{aligned}
||u||^2 &= \lim_{N\to\infty} ||P_N u||^2 \\
&= \lim_{N\to\infty} 2\pi \sum_{k=-N}^{N} |a_k|^2 \\
&= 2\pi \sum_{k=-\infty}^{\infty} |a_k|^2
\end{aligned}$$

which proves Parseval's equality.

2.2 Differentiation

Again, let

$$u = \sum_{k=-\infty}^{\infty} a_k e^{ikx}$$

then formally we can differentiate the series to get

$$\frac{du}{dx} = \sum_{k=-\infty}^{\infty} ika_k e^{ikx}$$

Note that this does not mean that the series converges. In fact, we have made no assumptions about the differentiability of the elements in L_2. However, it is true that the series converges if and only if u is differentiable, and furthermore, the series converges to $\frac{du}{dx}$.

We define the operator D to be the differentiation operator given by

$$Du = \sum_{k=-\infty}^{\infty} ika_k e^{ikx}$$

As such, D is an *unbounded linear operator* on L_2. This means that D is <u>not</u> defined for all $u \in L_2$ and it is not true that there exists a constant C such that

$$||Du|| \leq C||u||.$$

The operator D is also a *diagonal operator* on the basis $\{e^{ikx}\}$ because it maps basis elements onto multiples
Lec. 2 of themselves (like a diagonal matrix in linear algebra).

This brings us to an important point which we came across earlier, the convergence of the derivative series is dependent upon the smoothness of u. If we use Parseval's equation, we get

$$\left|\left|\frac{du}{dx}\right|\right|^2 = 2\pi \sum_{k=-\infty}^{\infty} |ika_k|^2 = 2\pi \sum_{k=-\infty}^{\infty} |k|^2 |a_k|^2.$$

For convergence, it is necessary that $|k|\,|a_k| \to 0$ as $|k| \to +\infty$. Likewise for higher derivatives we get

$$\left|\left|\frac{d^r u}{dx^r}\right|\right|^2 = 2\pi \sum_{k=-\infty}^{\infty} k^{2r} |a_k|^2.$$

For this series to converge, it now must be that $k^{2r}|a_k|^2 \to 0$ as $k \to \pm\infty$. Therefore, the smoother the function u, the faster the high modes must go to zero as $|k| \to +\infty$.

Now, we want to look again at the convergence rate of $||u - P_N u||$. Again using Parseval's equation, we have

$$\begin{aligned}
||u - P_N u||^2 &= 2\pi \sum_{|k|>N} |a_k|^2 \\
&= \frac{2\pi}{N^{2r}} \sum_{|k|>N} N^{2r} |a_k|^2 \\
&\leq \frac{2\pi}{N^{2r}} \sum_{|k|>N} |k|^{2r} |a_k|^2 \\
&\leq \frac{2\pi}{N^{2r}} \sum_{k=-\infty}^{\infty} |k|^{2r} |a_k|^2 \\
&\leq \frac{1}{N^{2r}} \left|\left|\frac{d^r u}{dx^r}\right|\right|^2.
\end{aligned}$$

We can do the same thing for the derivatives. The spectral approximation for u_x is $\frac{d}{dx}P_N u$ and we have

$$\begin{aligned} u_x - \frac{d}{dx}P_N u &= \sum_{|k|>N} ika_k e^{ikx} \\ ||u_x - \frac{d}{dx}P_N u||^2 &= 2\pi \sum_{|k|>N} |k|^2|a_k|^2 \\ &= \frac{2\pi}{N^{2r-2}} \sum_{|k|>N} N^{2r-2}|k|^2|a_k|^2 \\ &\leq \frac{2\pi}{N^{2r-2}} \sum_{|k|>N} |k|^{2r}|a_k|^2 \\ &\leq \frac{2\pi}{N^{2r-2}} \sum_{k=-\infty}^{\infty} |k|^{2r}|a_k|^2 \\ &= \frac{1}{N^{2(r-1)}} \left\| \frac{d^r u}{dx^r} \right\|^2 \end{aligned}$$

where we have assumed u has r derivatives. In general, we have

$$\left\| \frac{d^q}{dx^q}u - \frac{d^q}{dx^q}P_N u \right\| \leq \frac{1}{N^{r-q}} \left\| \frac{d^r u}{dx^r} \right\|$$

for any $q \leq r$.

Note that if u is only C^r, then this puts a limit on the order of accuracy of the spectral approximation. Furthermore, every derivative has one less order of accuracy.

On the other hand, suppose that u is C^∞, then $||u - P_N u|| \leq \frac{1}{N^r} \left\| \frac{d^r u}{dx^r} \right\|$ for any r. Clearly, $\frac{1}{N^r} \to 0$ as $r \to \infty$. If $\left\| \frac{d^r u}{dx^r} \right\| < C$ for some constant independent of r, then this would imply $||u - P_N u|| = 0$. In general, this is not the case, so for most cases this means that $\left\| \frac{d^r u}{dx^r} \right\| \to \infty$ as $r \to \infty$. Thus, the order of accuracy is determined by

$$\left\| \frac{d^q}{dx^q}u - \frac{d^q}{dx^q}P_N u \right\| \leq \inf_{r=0,1,\ldots} \frac{1}{N^{r-q}} \left\| \frac{d^r}{dx^r}u \right\|.$$

This shows why it is difficult to predict the actual order of accuracy of spectral methods.

2.3 Sobolev Norms

We can simplify the error bounds if we introduce Sobolev norms. Define

$$\|u\|_q^2 = \sum_{r=0}^{q} \left\| \frac{d^r u}{dx^r} \right\|^2.$$

Then, if u is C^r, and $q \leq r$, then

$$\|u - P_N u\|_q \leq \frac{C}{N^{r-q}} \|u\|_r.$$

Note that $\|u\|_0$ is equivalent to the standard 2-norm we were using before.

The space of functions with q derivatives is then the q^{th} Sobolev space and $|| \cdot ||_q$ is the norm on that space.

2.4 Spectral Interpolation

The projection P_N enables us to compute derivatives using $2N+1$ degrees of freedom. Given

$$u = \sum_{k=-\infty}^{\infty} a_k e^{ikx},$$

the projection is

$$P_N u = \sum_{k=-N}^{N} a_k e^{ikx}.$$

The operator P_N is called the spectral projection operator because $\langle u - P_N u, P_N u\rangle = 0$.

Recall that

$$a_k = \frac{1}{2\pi}\int_0^{2\pi} u(x) e^{-ikx}\, dx. \tag{2}$$

If we are to implement this numerically, then we are normally only given u in the form of a discrete set of points x_j, say uniformly spaced $x_j = \frac{2\pi}{2N} j$ for $j = 0, \ldots, 2N-1$. The integral (2) is then approximated by the trapezoidal rule. We would then get

$$a_k \approx \frac{1}{2N}\sum_{j=0}^{2N-1} u(x_j) e^{-ikx_j} \tag{3}$$

There is a problem with this approximation. When $k = N$, then

$$e^{iNx_N} = e^{iN\frac{2\pi}{2N}N} = e^{i\pi N} = (-1)^N = e^{-i\pi N} = e^{-iNx_N}.$$

Lec. 3 Thus, on the grid $\{x_j\}$, the modes e^{iNx} and e^{-iNx} cannot be distinguished. In order to get a symmetric set of coefficients, we will assign half to $j = N$ and half to $j = -N$. We thus rewrite the approximation (3) as

$$\tilde{a}_k = \frac{1}{2N}\frac{1}{c_k}\sum_{j=0}^{2N-1} u(x_j) e^{-ikx_j}, \qquad k = -N, \ldots, N$$

where

$$c_k = \begin{cases} 1 & |k| < N \\ 2 & |k| = N \end{cases}$$

Note that we are using only $2N$ points $\{u_j\}_{j=0}^{2N-1}$ to compute $2N+1$ values $\{\tilde{a}_j\}_{j=-N}^{N}$. If this is to be an invertible process, we need to reconcile the difference. In fact, $\{\tilde{a}_j\}_{j=-N}^{N}$ is only $2N$ values because as observed above, $\tilde{a}_N = \tilde{a}_{-N}$.

Now we define $\tilde{P}_N$ to be the projection

$$\tilde{P}_N u = \sum_{k=-N}^{N} \tilde{a}_k e^{ikx}.$$

The operator $\tilde{P}_N$ is called the *spectral interpolation operator* and $\tilde{P}_N u$ is called the *spectral interpolant* or the *pseudo-spectral approximation*. The values $\{x_j\}$ are called the *collocation points*. Similar to the operator P_N, the derivative approximation is obtained by the sum

$$\frac{d}{dx}\tilde{P}_N u = \sum_{k=-N}^{N} \tilde{a}_k ik e^{ikx}.$$

Note too, that this sum can be evaluated at any x, not just at the collocation points.

What we need to check now is, how well does $\tilde{P}_N u$ approximate u. Before we can show this, we will need a simple lemma.

Lemma 1

$$\sum_{k=0}^{2N-1} \theta^k = \begin{cases} \frac{\theta^{2N}-1}{\theta-1} & \theta \neq 1 \\ 2N & \theta = 1 \end{cases}$$

Proof 1
This is obvious if $\theta = 1$. If $\theta \neq 1$, let $S = \sum_{k=0}^{2N-1} \theta^k$, then we get

$$\begin{aligned} \theta S &= \sum_{k=0}^{2N-1} \theta^{k+1} \\ &= \sum_{k=1}^{2N} \theta^k \\ &= \theta^{2N} - 1 + \sum_{k=0}^{2N-1} \theta^k \\ &= \theta^{2N} - 1 + S \end{aligned}$$

Solving for S gives the result.

Now let's evaluate $\tilde{P}_N u$ at x_j:

$$\begin{aligned} \tilde{P}_N u(x_j) &= \sum_{k=-N}^{N} \tilde{a}_k e^{ikx_j} \\ &= \sum_{k=-N}^{N} \tilde{a}_k e^{ik\left(\frac{j\pi}{N}\right)} \\ &= \sum_{k=-N}^{N} \left[\frac{1}{2Nc_k} \sum_{\ell=0}^{2N-1} u_\ell e^{-ikx_\ell} \right] e^{ik\left(\frac{j\pi}{N}\right)} \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \sum_{k=-N}^{N} \frac{1}{c_k} e^{-ik\left(\frac{\ell\pi}{N}\right)} e^{ik\left(\frac{j\pi}{N}\right)} \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \sum_{k=-N}^{N} \frac{1}{c_k} e^{ik(j-\ell)\pi/N} \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \sum_{k=0}^{2N} \frac{1}{c_{k-N}} \left(e^{i(j-\ell)\pi/N}\right)^{k-N} \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \left[\sum_{k=1}^{2N-1} \left(e^{i(j-\ell)\pi/N}\right)^{k-N} + \frac{1}{2}\left(e^{-i(j-\ell)\pi} + e^{i(j-\ell)\pi}\right) \right] \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \left[e^{-i(j-\ell)\pi} \sum_{k=1}^{2N-1} \left(e^{i(j-\ell)\pi/N}\right)^{k} + \frac{1}{2} e^{-i(j-\ell)\pi} \left(1 + e^{i2(j-\ell)\pi}\right) \right] \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \left[e^{-i(j-\ell)\pi} \sum_{k=1}^{2N-1} \left(e^{i(j-\ell)\pi/N}\right)^{k} + e^{-i(j-\ell)\pi} \right] \\ &= \frac{1}{2N} \sum_{\ell=0}^{2N-1} u_\ell \left[e^{-i(j-\ell)\pi} \sum_{k=0}^{2N-1} \left(e^{i(j-\ell)\pi/N}\right)^{k} \right] \end{aligned}$$

Now note that

$$\begin{aligned}\sum_{k=0}^{2N-1}\left(e^{i(j-\ell)\pi/N}\right)^k &= \begin{cases}\frac{\left(e^{i(j-\ell)\pi/N}\right)^{2N}-1}{e^{i(j-\ell)\pi/N}-1} & j\neq\ell\\ 2N & j=\ell\end{cases}\\ &= \begin{cases}0 & j\neq\ell\\ 2N & j=\ell\end{cases}\end{aligned}$$

Thus,

$$\tilde{P}_N u(x_j) = \frac{1}{2N}\sum_{\ell=0}^{2N-1} u_\ell e^{-i(j-\ell)\pi} 2N\delta_{jl} = u_j.$$

This shows that $\tilde{P}_N u(x_j) = u(x_j)$ at the collocation points and can be used to interpolate u. Because of this, $\tilde{P}_N u$ is sometimes called the *Fourier interpolant* of u. Note that this interpolation is not unique because the nodes $e^{\pm iNx}$ cannot be distinguished on the grid $\{x_j\}$. However, it is unique with the additional restriction that $\tilde{a}_N = \tilde{a}_{-N}$.

The next task is to compare $\tilde{P}_N u$ to $P_N u$, which is a good approximation to u, between the collocation points. It is not true that $\tilde{P}_N u = P_N u$. To see this, let $u = e^{i(2N+r)x}$ for some $0 < r < N$. For this u, $P_N u = 0$. On the other hand,

$$u(x_j) = e^{i(2N+r)\left(\frac{2\pi}{2N}j\right)} = e^{irj\pi/N} = e^{irx_j}$$

Thus, the mode $e^{i(2N+r)x}$ cannot be resolved by the grid and instead is seen as a lower order frequency. This problem of high frequency modes being represented on the grid as low frequency modes is called *aliasing.*

In the pseudo-spectral method, the highest frequency that can be resolved by the grid is e^{iNx_j} where there are two gridpoints per wavelength. Of course, the modes e^{iNx_j} and e^{-iNx_j} cannot be distinguished.

This highlights the key difference between spectral and pseudo-spectral approximations. For spectral methods, the high frequency modes vanish while for pseudo-spectral approximations they alias to lower frequency modes.

We can use aliasing to get a relationship between the Fourier coefficients and the coefficients of $\tilde{P}_N u$. Let

$$u = \sum_{k=-\infty}^{\infty} a_k e^{ikx}$$

The pseudo-spectral approximation is

$$\tilde{P}_N u = \sum_{k=-N}^{N} \tilde{a}_k e^{ikx}$$

where

$$\begin{aligned}\tilde{a}_k &= \frac{1}{2N}\frac{1}{c_k}\sum_{j=0}^{2N-1} u_j e^{-ikx_j}\\ &= \frac{1}{2N}\frac{1}{c_k}\sum_{j=0}^{2N-1}\left(\sum_{\ell=-\infty}^{\infty} a_\ell e^{i\ell x_j}\right)e^{-ikx_j}\\ &= \sum_{\ell=-\infty}^{\infty}\frac{1}{2N}\frac{1}{c_k}\sum_{j=0}^{2N-1} a_\ell e^{i(\ell-k)x_j}\\ &= \sum_{\ell=-\infty}^{\infty}\frac{1}{2N}\frac{1}{c_k}a_\ell\sum_{j=0}^{2N-1} e^{i(\ell-k)\left(\frac{\pi}{N}j\right)}\\ &= \sum_{\ell=-\infty}^{\infty}\frac{1}{2N}\frac{1}{c_k}a_\ell\alpha_{k\ell}\end{aligned}$$

where

$$\alpha_{k\ell} = \begin{cases} \frac{e^{i(\ell-k)2\pi}-1}{e^{i(\ell-k)\pi/N}-1} & e^{i(\ell-k)\pi/N} \neq 1 \\ 2N & e^{i(\ell-k)\pi/N} = 1 \end{cases}$$

$$= \begin{cases} 0 & e^{i(\ell-k)\pi/N} \neq 1 \\ 2N & e^{i(\ell-k)\pi/N} = 1 \end{cases}$$

Now, $e^{i(\ell-k)\pi/N} = 1$ if and only if $\ell - k = 2rN$ for some integer r. In other words, $\ell = k + 2rN$. Thus,

$$\tilde{a}_k = \frac{1}{c_k} \sum_{r=-\infty}^{\infty} a_{k+2rN}.$$

Lec. 4

If u is smooth, then all the terms a_{k+2rN} are small except for $r = 0$. This fact allows us to bound the error of the pseudo-spectral approximation by

$$\|u - \tilde{P}_N u\|_q \leq \frac{C\|u\|_p}{N^{p-q}}$$

where p, q are integers, $p > 0$, $0 \leq q \leq p$.

2.5 Computing the Pseudo-Spectral Approximation

In practice, if we are given u on the collocation points, we compute the pseudo-spectral approximation by the formula

$$\tilde{a}_j = \frac{1}{2N} \frac{1}{c_j} \sum_{k=0}^{2N-1} u_k e^{-ijx_k}, \qquad \text{for } j = -N, \ldots, N \tag{4}$$

Then to compute the approximation to the r^{th} derivative, we use

$$\left. \frac{d^r}{dx^r} \tilde{P}_N u \right|_{x=x_k} = \sum_{j=-N}^{N} (ij)^r \tilde{a}_j e^{ijx_k}$$

Notice that equation (4) requires $O(N^2)$ computations in order to compute all the $\tilde{a}_j$. Compare this to finite difference approximations which are $O(N)$. So we take a computational cost hit when using pseudo-spectral methods. The impact can be reduced for particular values of N, say $N = 2^m$ for some integer m. Then the number of operations can be reduced to $O(N \log N)$ using the *Fast Fourier Transform* or FFT. The idea is to organize the order of computing the terms in the sums. We won't discuss this any further in this course.

An alternative approach to computing the derivatives $\frac{d^r u}{dx^r}\big|_{x=x_k}$ is to treat it as a linear function of the values u_k. Let

$$U = U^{(0)} = \begin{bmatrix} u_0 \\ \vdots \\ u_{2N-1} \end{bmatrix} \text{ and } U^{(r)} = \begin{bmatrix} u_0^{(r)} \\ \vdots \\ u_{2N-1}^{(r)} \end{bmatrix}$$

where $u_k^{(r)}$ is the pseudo-spectral approximation of $\frac{d^r}{dx^r} u\big|_{x=x_k}$. We want to then write a differential operator D so that $U^{(r)} = DU^{(r-1)} = D^r U$. The operator D is called the pseudo-spectral differentiation matrix.

There are a couple ways we can derive the entries of the matrix D. One way is to compute D on the unit vectors, namely $E_0, \ldots, E_{2N-1}$ where $E_k = \delta_{jk}$. Let $U = E_\ell$, then

$$\begin{aligned}
\tilde{a}_j &= \frac{1}{2N}\frac{1}{c_j}\sum_{k=0}^{2N-1} u_k e^{-ijx_k} \\
&= \frac{1}{2N}\frac{1}{c_j}\sum_{k=0}^{2N-1} \delta_{k\ell} e^{-ijx_k} \\
&= \frac{1}{2N}\frac{1}{c_j} e^{-ijx_\ell}
\end{aligned}$$

So,

$$\begin{aligned}
\tilde{P}_N u(x) &= \sum_{j=-N}^{N} \tilde{a}_j e^{ijx} \\
&= \sum_{j=-N}^{N} \frac{1}{2N}\frac{1}{c_j} e^{-ijx_\ell} e^{ijx} \\
&= \sum_{j=-N}^{N} \frac{1}{2N}\frac{1}{c_j} e^{ij(x-x_\ell)} \\
D\tilde{P}_N u(x) &= \sum_{j=-N}^{N} ij\frac{1}{2N}\frac{1}{c_j} e^{ij(x-x_\ell)} \\
D\tilde{P}_N u(x_k) &= \sum_{j=-N}^{N} ij\frac{1}{2N}\frac{1}{c_j} e^{ij(x_k-x_\ell)}
\end{aligned}$$

A more concise formula for the entries of D can be obtained by writing down trigonometric polynomials g_k which have the property that $g_k(x_\ell) = \delta_{k\ell}$. For example, let

$$g_k(x) = \frac{1}{2N}\sin(N(x - x_k))\cot((x - x_k)/2)$$

Clearly, $g_k(x_\ell) = \delta_{k\ell}$. The matrix D can then be constructed by computing

$$\left.\frac{d}{dx} g_k(x)\right|_{x=x_j} = D_{jk}$$

Thus, the matrix D has the entries

$$D_{jk} = \begin{cases} \frac{1}{2}(-1)^{j+k}\cot((x_j - x_k)/2) & j \neq k \\ 0 & j = k \end{cases}$$

Lec. 5

The operator D as computed above has some important properties that we will use later.

1. D is a real skew-symmetric matrix, i.e. $D^T = -D$. To see this, we have

$$\begin{aligned}
D_{kj} &= \frac{1}{2}(-1)^{j+k}\cot((x_k - x_j)/2) \\
&= \frac{1}{2}(-1)^{j+k}(-\cot((x_j - x_k)/2)) \\
&= -D_{jk}
\end{aligned}$$

Note that the same is true if we were to define D in terms of finite differences on a periodic function.

2. D has eigenvalues 0, $\pm i$, $\pm 2i$, $\ldots$, $\pm(N-1)i$, 0. Note that if $u = e^{ikx}$, then $Du = ike^{ikx} = iku$ and if $k < N$, the spectral approximation is exact, hence $\{e^{ikx}\}_{k=-(N-1)}^{N-1}$ are eigenvectors. The last eigenvector is $u_k = (-1)^k$ which corresponds to $e^{\pm iNx}$ (and the two modes cannot be distinguished). Applying D to this mode gives 0. This shows that $\|D\| = O(N)$.

3. Note that if the original function U has real data, then $\bar{\tilde{a}}_k = \tilde{a}_{-k}$. We can use this to note that we need only compute $\tilde{a}_k$ for $k = 0, \ldots, N$ and the reverse process means the sum need only be computed using half the sum (ignoring the imaginary part). There are specialized FFT routines which do this job for you. However, using the matrix method doesn't allow you to take advantage of this fact.

4. In practical terms, we do not actually build the matrix D, but do the equivalent using the FFT. Given U, we apply the FFT to get a new vector A which contains the computed Fourier coefficients a_{-N}, $a_{-N+1}, \ldots, a_{N-1}$. Now we must convert this into the corresponding pseudo-spectral approximation to get

$$\tilde{a}_k = a_k, \text{ for } k = -N+1, \ldots, N-1$$
$$\tilde{a}_{-N} = \tilde{a}_N = \frac{1}{2} a_{-N}$$

   Next, we compute the derivative of $\tilde{P}_N u$

$$\frac{d}{dx}\tilde{P}_N u = \frac{d}{dx}\sum_{k=-N}^{N} \tilde{a}_k e^{ikx}$$
$$\sum_{k=-N}^{N} \tilde{d}_k e^{ikx} = \sum_{k=-N}^{N} ik\tilde{a}_k e^{ikx}$$

   where the $\tilde{d}_k$ are the pseudo-spectral coefficients for the derivative. Note that in order to apply the inverse FFT, we must recombine $\tilde{d}_{-N}$ and $\tilde{d}_{-N}$ to get

$$d_{-N} = \tilde{d}_N + \tilde{d}_{-N} = iN\tilde{a}_N - iN\tilde{a}_{-N} = 0$$

   Therefore, the vector of data that we pass back through the inverse Fourier transform is

$$\begin{bmatrix} 0 & i(-N+1)a_{-N+1} & \cdots & ika_k & \cdots & i(N-1)a_{N-1} \end{bmatrix}$$

   It is important to note that the first entry is *not* $-iNa_{-N}$ as might be guessed.

2.6 Gibbs Phenomenon

Suppose $u(x)$ is not smooth. For example, let u be defined by

$$u(x) = \begin{cases} 1 & x_1 \leq x \leq x_2 \\ 0 & \text{otherwise} \end{cases}$$

Notice that our results on convergence assumed at least a C^0 function, so those results are not valid.

Fourier theory states, with some additional restrictions, that $P_N u(x_1) \to \frac{1}{2}(u(x_1^+) + u(x_1^-))$ as $N \to \infty$ where $u(x_1^{\pm})$ are the one-sided limits of u at x_1. However, convergence is not uniform near the discontinuity. For any λ, one can show

$$u_N\left(x + \frac{\lambda}{N + \frac{1}{2}}\right) \sim \frac{1}{2}[u(x^+) + u(x^-)] + [u(x^+) - u(x^-)]\,\mathrm{Si}(\lambda)$$

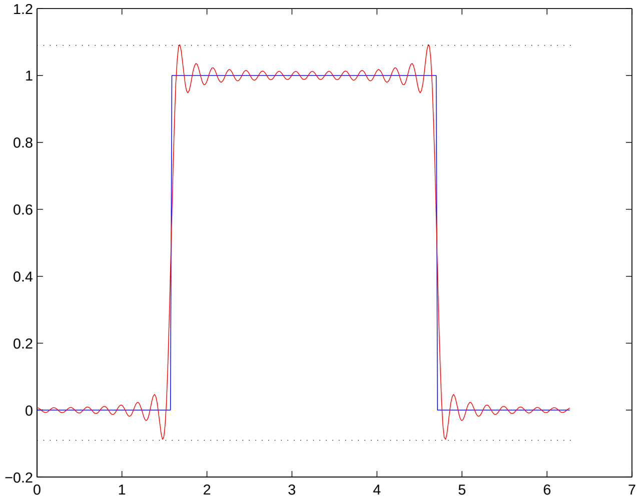


Figure 1: Example of Gibbs phenomemon overshoot.

where

$$\mathrm{Si}(\lambda) = \frac{1}{\pi}\int_0^{\lambda} \frac{\sin(\eta)}{\eta}\, d\eta.$$

What this shows is that if u has a jump discontinuity, then the points which are $O(1/N)$ away from the discontinuity differ from the mean by $O(1)$. Note that $\mathrm{Si}(\lambda)$ has a maximum at $\lambda = \pi$ and one can show that

$$u_N\left(x + \frac{\pi}{N+\frac{1}{2}}\right) - u(x^+) \sim (u(x^+) - u(x^-))(0.09)$$

This is known as the 9% *Gibbs overshoot.* The solution will look like Figure 1. These oscillations are global. Lec. 6 The utility of spectral methods for problems with jump discontinuities is the subject of research.

3 Fourier Methods for Partial Differential Equations

Let us return to partial differential equations and make use of the spectral approximation. Consider a partial differential equation of the form

$$u_t = Su$$
$$u(x,0) = u_0(x)$$

where S is a spatial differential operator and u is assumed to be periodic on $[0, 2\pi]$. Examples of such equations are

- Baby wave equation: $S = \frac{\partial}{\partial x}$, $u_t = u_x$
- Burger's equation: $S = u\frac{\partial}{\partial x}$, $u_t = uu_x$
- Heat equation: $S = \frac{\partial^2}{\partial x^2}$, $u_t = u_{xx}$

There are two methods for solving such a partial differential equation using spectral methods. Pseudo-spectral methods solve the equation using $\tilde{u}_N$ in real space, and Galerkin spectral methods compute u_N by evolving the coefficients in Fourier space.

3.1 Pseudo-spectral method

We introduce collocation points $x_k = \frac{2\pi}{2N}k$, for $k = 0, \ldots, 2N-1$ and let $u_k = u(x_k)$. We wish to solve for $u_k(t)$. Let $\hat{u}_N$ be the trigonometric polynomial which interpolates u_k. The approximate equation is then

$$\frac{du_k}{dt} = S(\hat{u}_N)\big|_{x=x_k}$$

The procedure is then

1. Compute $\hat{u}_N = \tilde{P}_N u$.
2. Solve in time the differential equation

$$\frac{\partial \hat{u}_N}{\partial t} = \tilde{P}_N S(\hat{u}_N), \qquad \hat{u}_N(x,0) = \tilde{P}_N u_0(x)$$

Note that it is not true that $\hat{u}_N = \tilde{P}_N u$ for $t > 0$. If we write the original equation and apply $\tilde{P}_N$ to it we get

$$\begin{aligned} \tilde{P}_N \frac{\partial}{\partial t} u &= \tilde{P}_N S(u) \\ \tilde{P}_N u(x,0) &= \tilde{P}_N u_0(x) \end{aligned}$$

This is different from

$$\begin{aligned} \frac{\partial}{\partial t}(\tilde{P}_N u) &= S(\tilde{P}_N u) \\ \tilde{P}_N u(x,0) &= \tilde{P}_N u_0(x) \end{aligned}$$

In the first case, we transform the entire differential equation into *Fourier space* where the equation is then solved before transforming the result back into real space. This is called a Galerkin method. In the second case, we apply the differential equation to the spectral approximation in *real space*. The two methods are different because they generate error in different ways. The first method has an additional error due to the approximation of the partial differential equation itself.

In the pseudo-spectral method, we must solve for $u_k(t)$, the value of u at the collocation points. The Fourier transform is used only for computing derivatives.

Example 3.1:

Consider the baby wave equation

$$\begin{aligned} u_t &= u_x \\ u(x,0) &= u_0(x) \end{aligned}$$

The easiest way to express the algorithm is in terms of the derivative operator D. We can express D in matrix form, and let $\tilde{u}_k(0) = \tilde{P}_N u(x_k, 0)$. Then the equation is solved by computing the equation

$$\frac{\partial}{\partial t}\tilde{U} = D\tilde{U}$$

where

$$\tilde{U} = \begin{bmatrix} \tilde{u}_0 \\ \vdots \\ \tilde{u}_{2N-1} \end{bmatrix}.$$

The method of stepping in time is not specified, but any high order method could be used.

To illustrate, let us do one step in time using Euler's method. Compute $\tilde{u}_k(0) = \tilde{P}_N u(x_k, 0)$, and express as $\tilde{U}_0$. At the same time, compute $\frac{\partial}{\partial x}\tilde{U}_0 = D\tilde{U}_0$. Now advance $\tilde{U}_0$ by

$$\frac{1}{\Delta t}(\tilde{U}_1 - \tilde{U}_0) = D\tilde{U}_0$$
$$\tilde{U}_1 = (I + \Delta t D)\tilde{U}_0$$

3.2 Conservation Properties

Consider the partial differential equation $u_t = u_x$, and multiply by u to get

$$\int_0^{2\pi} uu_t\,dx = \int_0^{2\pi} uu_x\,dx = \int_0^{2\pi} \frac{1}{2}\frac{\partial}{\partial x}(u^2)\,dx = \frac{1}{2}u^2\bigg|_0^{2\pi} = 0$$

At the same time, we have

$$\int_0^{2\pi} uu_t\,dx = \int_0^{2\pi} \frac{1}{2}\frac{\partial}{\partial t}(u^2)\,dx = \frac{d}{dt}\int_0^{2\pi} \frac{1}{2}u^2\,dx$$

Therefore, we have shown

$$\frac{d}{dt}\int_0^{2\pi} u^2\,dx = 0$$

In fact, this is the conservation of energy property where $\int_0^{2\pi} u^2\,dx$ represents the energy in the system.

How well does the pseudo-spectral method conserve energy? The discrete analog of $\int_0^{2\pi} u^2\,dx$ is given by $\sum_{j=0}^{2N-1} \tilde{u}_j^2$. Define $\tilde{U} = [\tilde{u}_0, \ldots, \tilde{u}_{2N-1}]^T$ to be the vector of data at the collocation points. The pseudospectral method can then be written as $\frac{d}{dt}\tilde{U} = D\tilde{U}$ where D is the pseudospectral differentiation matrix, and we see that

$$\frac{d}{dt}\tilde{U}^T\tilde{U} = 2\tilde{U}^T\frac{d}{dt}\tilde{U} = 2\tilde{U}^T D\tilde{U}.$$

Now, $\tilde{U}^T D\tilde{U}$ is a scalar, so it is equal to its transpose, hence

$$\tilde{U}^T D\tilde{U} = (\tilde{U}^T D\tilde{U})^T = \tilde{U}^T D^T\tilde{U} = -\tilde{U}^T D\tilde{U}$$

where recall that D is skew symmetric. Therefore, $\tilde{U}^T D\tilde{U} = 0$ and hence

$$\frac{d}{dt}\tilde{U}^T\tilde{U} = \frac{d}{dt}\sum_{j=-N}^{N} \tilde{u}_j^2 = 0.$$

One might be led to conclude that this implies the pseudo-spectral method is unconditionally stable. However, we have assumed exact integration in time. Stability of the full numerical method will depend on the choice of time integration. Lec. 7

The next level of complication would be to add variable coefficients. Consider the equation

$$u_t = b(x)u_x$$
$$u(x, 0) = u_0(x)$$

In matrix format, this becomes

$$\frac{d\tilde{U}}{dt} = AD\tilde{U}$$

where D is the differentiation matrix and

$$A = \begin{bmatrix} a(x_0) & & 0 \\ & \ddots & \\ 0 & & a(x_N) \end{bmatrix}.$$

Again, if $a(x) > 0$, then we have a conservation of energy:

$$\begin{aligned}
\frac{d}{dt}\left\langle u, \frac{1}{a}u \right\rangle &= \left\langle \frac{\partial u}{\partial t}, \frac{1}{a}u \right\rangle + \left\langle u, \frac{\partial}{\partial t}\frac{1}{a}u \right\rangle \\
&= \left\langle \frac{\partial u}{\partial t}, \frac{1}{a}u \right\rangle + \left\langle u, \frac{1}{a}\frac{\partial u}{\partial t} \right\rangle \\
&= \left\langle a\frac{\partial u}{\partial x}, \frac{1}{a}u \right\rangle + \left\langle u, \frac{\partial u}{\partial x} \right\rangle \\
&= \left\langle \frac{\partial u}{\partial x}, u \right\rangle + \overline{\left\langle \frac{\partial u}{\partial x}, u \right\rangle} \\
&= 2\Re\left\langle \frac{\partial u}{\partial x}, u \right\rangle \\
&= 2\Re\int_0^{2\pi} \bar{u}\frac{\partial u}{\partial x}\,dx \\
&= \Re\int_0^{2\pi} \bar{u}\frac{\partial u}{\partial x} + u\frac{\partial \bar{u}}{\partial x}\,dx \\
&= \Re\int_0^{2\pi} \frac{\partial}{\partial x}|u|^2\,dx \\
&= |u|^2\Big|_0^{2\pi} = 0
\end{aligned}$$

Likewise, in the discrete case,

$$\begin{aligned}
\frac{1}{2}\frac{d}{dt}\langle \tilde{U}, A^{-1}\tilde{U}\rangle &= \frac{1}{2}\left\langle \frac{\partial \tilde{U}}{\partial t}, A^{-1}\tilde{U} \right\rangle + \frac{1}{2}\left\langle \tilde{U}, A^{-1}\frac{\partial \tilde{U}}{\partial t} \right\rangle \\
&= \frac{1}{2}\langle AD\tilde{U}, A^{-1}\tilde{U}\rangle + \frac{1}{2}\langle \tilde{U}, A^{-1}AD\tilde{U}\rangle \\
&= \tilde{U}^T D\tilde{U} = 0
\end{aligned}$$

We can also apply this technique to the heat equation to see that it properly dissipates energy. Consider the equatiion

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

For the analytic case, we have

$$\begin{aligned}
\frac{d}{dt}\langle u, u\rangle &= 2\,\langle u_t, u\rangle \\
&= 2\,\langle u_{xx}, u\rangle \\
&= 2\int_0^{2\pi} u_{xx}u\,dx \\
&= 2\left[uu_x\big|_0^{2\pi} - \int_0^{2\pi} u_x^2\,dx \right] \\
&= -2\|u_x\|^2
\end{aligned}$$

For the discrete case, we have $\frac{\partial \tilde{U}}{\partial t} = D^2\tilde{U}$, and hence

$$\begin{aligned}
\frac{d}{dt}\left\langle \tilde{U}, \tilde{U} \right\rangle &= 2\left\langle \tilde{U}, \frac{d}{dt}\tilde{U} \right\rangle \\
&= 2\left\langle \tilde{U}, D^2\tilde{U} \right\rangle \\
&= 2\tilde{U}^T D^2 \tilde{U} \\
&= 2\tilde{U}(-D^T D)\tilde{U} \\
&= -2\left(D\tilde{U}\right)^T D\tilde{U} \\
&= -2\left\langle D\tilde{U}, D\tilde{U} \right\rangle .
\end{aligned}$$

This shows that the pseudospectral method will dissipate at the same rate as the analytic equation.

Next, let us consider a non-linear problem such as Burger's equation:

$$u_t = uu_x = \frac{1}{2}(u^2)_x$$

We rewrite this as

$$u_t = \frac{1}{3}(u^2)_x + \frac{1}{3}uu_x$$

Given u defined on the collocation points, compute the spectral interpolants $\tilde{u}_N = \tilde{P}_N u$ and $\tilde{v}_N = \tilde{P}_N u^2$. We then approximate the equation as

$$\frac{\partial}{\partial t}\tilde{u}_k = \frac{1}{3}\frac{\partial}{\partial x}\tilde{v}_N\bigg|_{x=x_k} + \frac{1}{3}u_k\frac{\partial}{\partial x}\tilde{u}_N\bigg|_{x=x_k} .$$

In matrix form, this method becomes

$$\frac{d}{dt}\tilde{U} = \frac{1}{3}D\Lambda\tilde{U} + \frac{1}{3}\Lambda D\tilde{U} \tag{5}$$

where

$$\Lambda = \begin{bmatrix} u_0 & & 0 \\ & \ddots & \\ 0 & & u_{2N-1} \end{bmatrix} .$$

Again, we get a conserved quantity in $\langle u, u \rangle$. From the exact equation, we get

$$\begin{aligned}
\frac{d}{dt}\langle u, u \rangle &= 2\langle u, u_t \rangle \\
&= 2\langle u, uu_x \rangle \\
&= 2\int_0^{2\pi} u^2 u_x \, dx \\
&= \frac{2}{3}\int_0^{2\pi} \frac{\partial}{\partial x}(u^3)\, dx \\
&= \frac{2}{3}u^3\bigg|_0^{2\pi} = 0
\end{aligned}$$

Thus, $\frac{d}{dt}\langle u, u \rangle = 0$.

If we do the same thing using equation (5), we get

$$\begin{aligned}\frac{d}{dt}\tilde{U}^T\tilde{U} &= 2\left(\frac{d}{dt}\tilde{U}\right)^T\tilde{U}\\ &= 2\left(\frac{1}{3}D\Lambda\tilde{U}+\frac{1}{3}\Lambda D\tilde{U}\right)^T\tilde{U}\\ &= \frac{2}{3}\tilde{U}^T(D^T\Lambda^T+\Lambda^T D^T)\tilde{U}\\ &= \frac{2}{3}\tilde{U}^T(-D\Lambda+(D\Lambda)^T)\tilde{U}\\ &= \frac{2}{3}(\tilde{U}^T(D\Lambda)^T\tilde{U}-\tilde{U}^T(D\Lambda)\tilde{U})\end{aligned}$$

But, $\tilde{U}^T(D\Lambda)^T\tilde{U} = (\tilde{U}^T(D\Lambda)^T\tilde{U})^T = \tilde{U}^T(D\Lambda)\tilde{U}$. Thus,

$$\frac{d}{dt}\tilde{U}^T\tilde{U} = \frac{2}{3}(\tilde{U}^T(D\Lambda)\tilde{U}-\tilde{U}^T(D\Lambda)\tilde{U}) = 0.$$

The fact that writing the method in this forms conserves the energy is why writing the equation in the form of equation (5) is done.

3.3 The Galerkin Method

The method we have described so far works with approximate function values. In other words, the approximations are all in real-space. The Galerkin method is a method where the approximation is done in Fourier space.

Consider the equation

$$\begin{aligned}\frac{\partial u}{\partial t} &= Su\\ u(x,0) &= u_0(x)\end{aligned}$$

where S is some differential operator. For fixed N, compute the Fourier coefficients a_k where

$$u(x) \approx P_N u = \sum_{k=-N}^{N} a_k e^{ikx}$$

Note that here we are computing the a_k via the exact formula

$$a_k = \frac{1}{2\pi}\int_0^{2\pi} u_0(x)e^{-ikx}\,dx.$$

If the integral is done numerically, and $u_0(x)$ is given explicitly, use a maximally accurate method to compute a_k so as to avoid initial aliasing problems. We then have $u(x) \approx P_N u(x) = \sum_{k=-N}^{N} a_k(t)e^{ikx}$.

Next, we obtain ordinary differential equations for each of the a_k by substituting $P_N u$ into $\frac{\partial u}{\partial t} = Su$. We get an equation of the form

$$\sum_{k=-N}^{N}\frac{d}{dt}a_k(t)e^{ikx} = S\sum_{k=-N}^{N}a_k(t)e^{ikx} = \sum_{k=-N}^{N}c_k(a_{-N}(t),\ldots,a_N(t),t)a_k(t)e^{ikx}$$

for some functions $c_k(a_{-N}(t),\ldots,a_N(t),t)$ which are independent of x and may depend non-linearly on the $a_k(t)$'s. We then get a system of differential equations of the form

$$\frac{d}{dt}a_k(t) = c_k(a_{-N}(t),\ldots,a_N(t),t) \qquad \text{for } k=-N,\,\ldots,\,N$$

For example, consider the baby wave equation $u_t = u_x$ and assume $u \approx P_N u(x,t) = \sum_{k=-N}^{N} a_k(t)e^{ikx}$. Plugging this approximation into the differential equation gives

$$\sum_{k=-N}^{N} \frac{d}{dt}a_k(t)e^{ikx} = \sum_{k=-N}^{N} ika_k(t)e^{ikx}.$$

Thus, we got the evolution equation for the coefficients

$$\frac{d}{dt}a_k(t) = ika_k(t), \qquad \text{for } k = -N, \ldots, N.$$

The Galerkin method is to compute these individual ordinary differential equations and reconstruct the solution $u(x,t)$ from the coefficients only when necessary. Lec. 8

The Galerkin method works well for linear problems, the difficulty is when the differential operator is not linear, or there are variable coefficients. For example, consider the equation

$$u_t = b(x)u_x$$

Suppose $b(x) = \sum_{j=-\infty}^{\infty} b_j e^{ijx}$ and let $P_N u = \sum_{k=-\infty}^{\infty} a_k e^{ikx}$ where $a_k = 0$ for $k > N$. We then get

$$\begin{aligned}\sum_{k=-\infty}^{\infty} \frac{d}{dt}a_k(t)e^{ikx} &= \left(\sum_{k=-\infty}^{\infty} b_j e^{ijx}\right)\left(\sum_{k=-\infty}^{\infty} ika_k e^{ikx}\right) \\ &= \sum_{k=-\infty}^{\infty} c_k e^{ikx}\end{aligned}$$

where

$$c_k = \sum_{j=-\infty}^{\infty} ija_j b_{k-j} = \sum_{j=-N}^{N} ija_j(t)b_{k-j}.$$

The Galerkin method now becomes a *coupled* linear system of ordinary differential equations given by

$$\frac{d}{dt}a_k(t) = \sum_{j=-N}^{N} ija_j(t)b_{k-j}$$

For another example, consider the Burger's equation,

$$u_t = uu_x.$$

Following the same algebra as above, we see that we get the system of ordinary differential equations

$$\frac{d}{dt}a_k(t) = \sum_{j=-N}^{N} ija_j(t)a_{k-j}(t)$$

Now we have a *non-linear coupled* system of ordinary differential equations which further complicates the computation of the solution. These examples show how products become convolutions for the Galerkin method.

We now will formalize the Galerkin method. For generality, let $\phi_\ell = e^{i\ell x}$ be an orthogonal basis set. Approximate

$$P_N u = \sum_{k=-N}^{N} a_k \phi_k(x)$$

and assume we wish to solve the equation

$$u_t = Su.$$

The Galerkin procedure requires the residual, given by

$$\frac{\partial}{\partial t} P_N u - S P_N u$$

to be in the orthogonal complement of the space spanned by the basis vectors $\{\phi_j\}_{j=-N}^{N}$. For this to be true, it must be that

$$\left\langle \phi_j, \frac{\partial}{\partial t} P_N u - S P_N u \right\rangle = 0$$
$$\frac{d}{dt}\langle \phi_j, P_N u\rangle = \langle \phi_j, S P_N u\rangle$$

Clearly, this is the method we described above where we found the equations for the time evolution of the Fourier coefficients. In other words, we are solving the equation

$$\frac{\partial P_N u}{\partial t} = S P_N u$$

where P_N is the orthogonal spectral projection.

For summary purposes, we present a comparison between Galerkin and Pseudo-spectral methods.

- For non-linear or non-constant coefficient problems, the Galerkin method is more expensive because products are evaluated as convolutions.
- There are applications, such as in turbulence, where the Fourier modes are as important as the real function values, so Galerkin methods make sense in that context.
- Pseudo-spectral methods work in real space and products are handled naturally.
- Galerkin methods project high frequency modes to zero while Pseudo-spectral methods alias high frequencies to lower frequencies.

Lec. 9

3.4 Time Discretization for Pseudo-spectral Methods

Suppose we have the partial differential equation

$$u_t = Su, \qquad u(x, 0) = u_0(x)$$

where again, S is a possibly non-linear differential operator. So far, we have discussed only how to handle the right hand side. Now we will discuss how to advance in time.

When considering the time stepping method, it is important to understand the relative error introduced by the time step method compared to the error in the spatial direction. If a time accurate solution is required, then one must be careful to take small time steps or the time error can dominate the total error and negate the advantage of using spectral methods.

On the other hand, if a steady state solution is sought, or the time evolution is slow compared to the spatial behavior, the larger time steps are appropriate.

3.4.1 Stability Analysis for Time Discretization

Consider the equation

$$u_t = u_x$$

If we again write

$$U = \begin{bmatrix} u_0 \\ \vdots \\ u_{2N-1} \end{bmatrix}$$

where $u_k = u(x_k)$, then the pseudo-spectral approximation to this equation is

$$\frac{d}{dt}U = DU$$

Let $U^n = U(n\Delta t)$. The most general time discretization method can be written as

$$\sum_{j=-r}^{r} \alpha_j U^{n+j} = \sum_{j=-r}^{r} \beta_j p_j(D) U^{n+j}$$

where $p_j(D)$ is a polynomial in D.

If we take $\alpha_{\pm 1} = \frac{\pm 1}{2\Delta t}$, $\beta_0 = 1$, then we get the Leap-Frog method:

$$\frac{U^{n+1} - U^{n-1}}{2\Delta t} = DU^n$$

You may recall from last quarter that we computed

$$\frac{U^{n+1} - U^{n-1}}{2\Delta t} = DU^n + O(\Delta t^2)$$

Now let us discuss the stability. Consider the simpler difference equation

$$\sum_{j=-r}^{r} \alpha_j U^{n+j} = DU^n$$

We look for solutions of the form $U^n = z^n U^0$ where z is a scalar.

$$\sum_{j=-r}^{r} \alpha_j z^{n+j} U^0 = D z^n U^0$$

$$\left(\sum_{j=-r}^{r} \alpha_j z^j \right) z^n U^0 = D z^n U^0$$

Thus, $U^n = z^n U^0$ must be an eigenvector of D. Suppose $D = P \Lambda P^{-1}$ where Λ is the diagonal matrix with the eigenvalues of D, then let $W^n = P^{-1} U^n = z^n P^{-1} U^0$. We thus have a decoupled equation of the form

$$\left(\sum_{j=-r}^{r} \alpha_j z^j \right) W^n = \Lambda W^n$$

$$q(z) W^n = \Lambda W^n$$

Therefore, we have a system of decoupled equations of the form

$$q(z) w_k^n = \lambda_k w_k^n$$

For each equation, we solve for z_k which will depend on λ_k.

Recall that for stability, we require

$$\max_{0\leq k\leq 2N-1} |z_k| \leq 1 + a\Delta t$$

which is sufficient to allow bounded growth, hence stability. Note here that a is a constant which is independent of the choice of N and Δt.

This condition puts a cap on the growth of $|W^n|$. But P depends on N through D, can we make the same conclusion? The answer is yes, because D is a normal matrix ($D^*D = DD^*$). Normal matrices have the property that they can be diagonalized by unitary matrices. This means that P is unitary, and consequently,

$$||W^n||_2 = ||P^{-1}U^n||_2 = ||U^n||$$

Therefore, it is sufficient to show the bound on the growth of the z_k's to prove stability.

Let us apply this technique to the Leap-Frog method for the equation $u_t = u_x$. The method is

$$\frac{U^{n+1} - U^{n-1}}{2\Delta t} = DU^n$$

Let $U^n = z^n U^0$ and plug into the equation to get

$$\frac{z^{n+1}U^0 - z^{n-1}U^0}{2\Delta t} = Dz^nU^0$$
$$\frac{1}{2\Delta t}(z^2 - 1)U^0 = zDU^0$$

Let U^0 be an eigenvector of D with eigenvalue λ, then we have

$$\frac{1}{2\Delta t}(z^2 - 1) = \lambda z$$

Recall that $\lambda = \pm i\ell$ for some integer ℓ, $0 \leq \ell \leq N - 1$ and so we get

$$z^2 - 1 = \pm 2\Delta t i\ell z$$

Solving for z gives

$$z = \mp i\ell\Delta t \pm \sqrt{1 - \ell^2\Delta t^2}$$

If we assume $1 - \ell^2\Delta t^2 \geq 0$, then

$$|z|^2 = (1 - \ell^2\Delta t^2) + \ell^2 + \Delta t^2 = 1$$

Therefore, if $\Delta t \leq \frac{1}{|\ell|}$ for each $\ell = \pm 1, \ldots, \pm N - 1$, then we have stability. The stability limit is then

$$\Delta t \leq \frac{1}{N-1}.$$

Recall that for the collocation points, $\Delta x = \frac{2\pi}{2N} = \frac{\pi}{N}$ and note that for large N, $\frac{1}{N} \sim \frac{1}{N-1}$ so that

$$\frac{1}{N-1} \sim \frac{1}{N} = \frac{\Delta x}{\pi}$$

Thus, we have stability if $\Delta t \leq \frac{1}{\pi}\Delta x$. This is smaller than the stability limit for the corresponding finite difference approximation.

We note here without proof, that solving the corresponding hyperbolic system of equations

$$U_t = AU_x$$

where A is an $m \times m$ matrix, then the stability limit is

$$\Delta t \leq \frac{1}{(N-1)\lambda_{\text{max}}}$$

where λ_{max} is the largest eigenvalue of A in absolute value.

By contrast, we can also analyze Euler's method given by

$$U^{n+1} = U^n + \Delta t D U^n$$

If we assume $U^n = z^n U^0$, then we get

$$\begin{aligned}
z^{n+1}U^0 &= z^n U^0 + \Delta t D z^n U^0 \\
(z-1) &= \Delta t i \ell \\
z &= 1 + \Delta t i \ell \\
|z| &= \sqrt{1 + \Delta t^2 \ell^2} \\
&\leq \sqrt{1 + 2\ell\Delta t + \ell^2 \Delta t^2} \\
&= \sqrt{(1 + \ell\Delta t)^2} \\
&= 1 + \ell \Delta t
\end{aligned}$$

Lec. 10 Therefore, $|z_{\text{max}}| \leq 1 + (N-1)\Delta t$. For stability, we then need $\Delta t = O\left(\frac{1}{N^2}\right)$ which is very restrictive.

We can also study an implicit method such as Crank-Nicolson:

$$\frac{U^{n+1} - U^n}{\Delta t} = \frac{1}{2} D (U^{n+1} + U^n)$$

Again, set $U^n = z^n U^0$ to get

$$\frac{z-1}{\Delta t} U^0 = \frac{1}{2}(z+1) D U^0$$

If U^0 is an eigenvector of D with eigenvalue $i\ell$, then we get

$$\begin{aligned}
\frac{1}{\Delta t}(z-1) &= \frac{1}{2}(z+1) i \ell \\
z\left(1 - \frac{\Delta t}{2} i \ell\right) &= 1 + \frac{\Delta t}{2} i \ell \\
z &= \frac{1 + \frac{\Delta t}{2} i \ell}{1 - \frac{\Delta t}{2} i \ell}
\end{aligned}$$

and hence,

$$\begin{aligned}
|z|^2 &= \frac{1 + \frac{1}{4}\Delta t^2 \ell^2}{1 + \frac{1}{4}\Delta t^2 \ell^2} \\
&= 1
\end{aligned}$$

Therefore, Crank-Nicolson is unconditionally stable. This is not an endorsement for large steps however, because there is still the temporal error that is to be considered.

The implementation of Crank-Nicolson is not trivial. The matrix equation looks like

$$\left(I - \frac{\Delta t}{2} D\right) U^{n+1} = \left(I + \frac{\Delta t}{2} D\right) U^n$$

This time, the matrix on the left is a *full matrix*. It is not a crisis, however, because if F is the FFT operator, then FDF^{-1} is a diagonal matrix, hence $D = F^{-1}\Lambda F$. Substituting into the equation, we get

$$\begin{aligned}
F^{-1}\left(I - \frac{\Delta t}{2}\Lambda\right) FU^{n+1} &= F^{-1}\left(I + \frac{\Delta t}{2}\Lambda\right) FU^{n} \\
\left(I - \frac{\Delta t}{2}\Lambda\right) FU^{n+1} &= \left(I + \frac{\Delta t}{2}\Lambda\right) FU^{n} \\
U^{n+1} &= F^{-1}\left(I - \frac{\Delta t}{2}\Lambda\right)^{-1}\left(I + \frac{\Delta t}{2}\Lambda\right) FU^{n}
\end{aligned}$$

Note that this inversion of D by F does not help in the case of variable coefficient problems. In that case, D must be inverted as a full matrix which is expensive.

Another common high-order time stepping scheme is the Runge-Kutta class of methods. We have already seen that the first order Runge-Kutta class method, i.e. Euler's method, is not a good choice for time stepping. Let us try the second order Runge-Kutta method. The method is

$$\begin{aligned}
U^{(1)} &= U^n + \frac{1}{2}\Delta t D U^n \\
U^{(2)} &= U^n + \Delta t D U^{(1)} \\
U^{n+1} &= U^{(2)}
\end{aligned}$$

In this case,

$$\begin{aligned}
U^{n+1} &= U^{(2)} \\
&= (U^n + \Delta t D U^{(1)}) \\
&= U^n + \Delta t D\left(U^n + \frac{1}{2}\Delta t D U^n\right) \\
&= \left(I + \Delta t D + \frac{1}{2}\Delta t^2 D^2\right) U^n
\end{aligned}$$

We assume $U^n = z^n U^0$, and plug in to get

$$\begin{aligned}
z^{n+1}U^0 &= \left(I + \Delta t D + \frac{1}{2}\Delta t^2 D^2\right) z^n U^0 \\
z &= \left(1 + \Delta t i\ell + \frac{1}{2}\Delta t^2 (i\ell)^2\right) \\
&= \left(1 - \frac{1}{2}\Delta t^2 \ell^2\right) + i\Delta t\ell \\
|z|^2 &= \left(1 - \frac{1}{2}\Delta t^2 \ell^2\right)^2 + \Delta t^2\ell^2 \\
&= 1 - \Delta t^2\ell^2 + \frac{1}{4}\Delta t^4\ell^4 + \Delta t^2\ell^2 \\
&= 1 + \frac{1}{4}\Delta t^4\ell^4
\end{aligned}$$

Thus, as for Euler's method, all modes grow, but can be made weakly stable.

The third and fourth order Runge-Kutta methods are better choices. If we are solving the equation

$$\frac{d}{dt}U = DU$$

with the third order Runge-Kutta method, we get

$$\begin{aligned}
U^{(0)} &= U^n \\
U^{(1)} &= U^n + \alpha_1 \Delta t D U^n \\
U^{(2)} &= U^n + \alpha_2 \Delta t D U^{(1)} \\
U^{(3)} &= U^n + \alpha_3 \Delta t D U^{(2)} \\
U^{n+1} &= U^{(3)}
\end{aligned}$$

where $\alpha_1 = \frac{1}{3}$, $\alpha_2 = \frac{1}{2}$, and $\alpha_3 = 1$.

To analyze this method, we again assume $U^n = z^n U^0$ and combine all the steps to get

$$\begin{aligned}
U^{(0)} &= z^n U^0 \\
U^{(1)} &= z^n U^0 + \alpha_1 \Delta t D z^n U^0 \\
U^{(2)} &= z^n U^0 + \alpha_2 \Delta t D (z^n U^0 + \alpha_1 \Delta t D z^n U^0) \\
z^{n+1} U^0 = U^{n+1} = U^{(3)} &= z^n U^0 + \alpha_3 \Delta t D [z^n U^0 + \alpha_2 \Delta t D (z^n U^0 + \alpha_1 \Delta t D z^n U^0)]
\end{aligned}$$

Thus, we get

$$z U^0 = [I + \alpha_3 \Delta t D + \alpha_2 \alpha_3 \Delta t^2 D^2 + \alpha_1 \alpha_2 \alpha_3 \Delta t^3 D^3] U^0 = A U^0$$

Note that A has the same eigenvectors as D, and the eigenvalues of A are

$$(1 + \alpha_3 \Delta t i \ell - \alpha_2 \alpha_3 \Delta t^2 \ell^2 - \alpha_1 \alpha_2 \alpha_3 \Delta t^3 i \ell^3) \text{ for } \ell = -N-1, \ldots, N-1.$$

Thus, we must have

$$\begin{aligned}
z &= 1 - \alpha_2 \alpha_3 \Delta t^2 \ell^2 + i(\alpha_3 \Delta t \ell - \alpha_1 \alpha_2 \alpha_3 \Delta t^3 \ell^3) \\
|z|^2 &= (1 - \alpha_2 \alpha_3 \Delta t^2 \ell^2)^2 + (\alpha_3 \Delta t \ell - \alpha_1 \alpha_2 \alpha_3 \Delta t^3 \ell^3)^2 \\
&= 1 - 2\alpha_2 \alpha_3 \Delta t^2 \ell^2 + \alpha_2^2 \alpha_3^2 \Delta t^4 \ell^4 + \alpha_3^2 \Delta t^2 \ell^2 - 2\alpha_1 \alpha_2 \alpha_3^2 \Delta t^4 \ell^4 + \alpha_1^2 + \alpha_2^2 \alpha_3^2 \Delta t^6 \ell^6 \\
&= 1 - \frac{1}{12} \Delta t^4 \ell^4 + \frac{1}{36} \Delta t^6 \ell^6
\end{aligned}$$

If we want $|z|^2 \leq 1$, then we get

$$\begin{aligned}
1 - \frac{1}{12} \Delta t^4 \ell^4 + \frac{1}{36} \Delta t^6 \ell^6 &\leq 1 \\
\frac{1}{36} \Delta t^2 \ell^2 &\leq \frac{1}{12} \\
\Delta t^2 &\leq \frac{3}{\ell^2} \\
\Delta t &\leq \frac{\sqrt{3}}{|\ell|}
\end{aligned}$$

Therefore, the stability requirement is $\Delta t \leq \frac{\sqrt{3}}{(N-1)}$.

Fourth order Runge-Kutta can also used. This is the same as third order Runge-Kutta except there are four stages.

$$\begin{aligned}
U^{(1)} &= U^n + \alpha_1 \Delta t D U^n \\
U^{(2)} &= U^n + \alpha_2 \Delta t D U^{(1)} \\
U^{(3)} &= U^n + \alpha_3 \Delta t D U^{(2)} \\
U^{(4)} &= U^n + \alpha_4 \Delta t D U^{(3)} \\
U^{n+1} &= U^{(4)}
\end{aligned}$$

where $\alpha_1 = \frac{1}{4}$, $\alpha_2 = \frac{1}{3}$, $\alpha_3 = \frac{1}{2}$, and $\alpha_4 = 1$. In this case, the stability bound is

$$\Delta t \leq \frac{2\sqrt{2}}{N-1} \approx \frac{2.8}{N-1}$$

For non-linear problems, the Runge-Kutta methods tend to give only second order accuracy but are also more robust than the Leap-Frog method. Lec. 11

3.4.2 Parabolic Problems

Next, let us look at the stability of parabolic problems. Consider the heat equation

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

The discrete pseudo-spectral approximation is

$$\frac{dU}{dt} = D^2 U$$

Clearly, the eigenvalues of D^2 are $O(N^2)$, so we have to expect a more restrictive time step for explicit methods. Let us first try Euler's method.

$$U^{n+1} = U^n + \Delta t D^2 U^n$$

Plug in $U^n = z^n U^0$ to get

$$\begin{aligned} z^{n+1}U^0 &= z^n U^0 + \Delta t D^2 z^n U^0 \\ z &= 1 + \Delta t (i\ell)^2 \\ &= 1 - \Delta t \ell^2 \end{aligned}$$

Thus,

$$\begin{aligned} |1 - \Delta t \ell^2| &\leq 1 \\ -1 \leq 1 - \Delta t &\ell^2 \leq 1 \\ -2 \leq -\Delta t &\ell^2 \leq 0 \\ \Delta t &\leq \frac{2}{\ell^2} \end{aligned}$$

So we must have $\Delta t \leq \frac{2}{(N-1)^2}$.

For a larger time step, we could try the Crank-Nicolson method

$$\frac{1}{\Delta t}(U^{n+1} - U^n) = \frac{1}{2} D^2 (U^{n+1} + U^n)$$

Plugging in $U^n = z^n U^0$, we get

$$\begin{aligned} \frac{1}{\Delta t}(z^{n+1}U^0 - z^n U^0) &= \frac{1}{2} D^2 (z^{n+1}U^0 + z^n U^0) \\ z\left(I - \frac{\Delta t}{2} D^2\right) U^0 &= \left(I + \frac{\Delta t}{2} D^2\right) U^0 \\ z\left(1 - \frac{\Delta t}{2}(i\ell)^2\right) &= \left(1 + \frac{\Delta t}{2}(i\ell)^2\right) \\ z &= \frac{1 - \frac{\Delta t}{2}\ell^2}{1 + \frac{\Delta t}{2}\ell^2} \\ &\leq 1 \end{aligned}$$

Thus, Crank-Nicolson is again unconditionally stable.

Backward Euler is also a good choice for a method if steady state solutions are desired, but not if a time accurate solution is needed.

$$\frac{U^{n+1}-U^n}{\Delta t} = D^2 U^{n+1}$$

This method is also unconditionally stable.

While Crank-Nicolson and Backward Euler are unconditionally stable, they also will require inversion of a full matrix. The diagonalization of D^2 using the FFT operator F can again be employed for the linear constant coefficient problem.

By contrast, The Leap Frog method is not a good choice for parabolic problems (as we also saw for finite differences). Plugging $U^n = z^n U^0$ into the equation becomes

$$\begin{aligned}
\frac{U^{n+1}-U^{n-1}}{2\Delta t} &= D^2 U^n \\
\frac{z^{n+1}U^0 - z^{n-1}U^0}{2\Delta t} &= D^2 z^n U^0 \\
(z^2-1) &= 2\Delta t z(-\ell^2) \\
z^2 + 2\Delta t \ell^2 z - 1 &= 0 \\
z &= \frac{-2\Delta t\ell^2 \pm \sqrt{4\Delta t^2\ell^4+4}}{2} \\
&= -\Delta t\ell^2 \pm \sqrt{1+\Delta t^2\ell^4}
\end{aligned}$$

Hence,

$$z = -\Delta t\ell^2 - \sqrt{1+\Delta t^2\ell^4} < -1,$$

and therefore, Leap Frog is not stable.

As for finite differences, we can also apply semi-implicit methods to avoid the time step restrictions of some equations. For example, consider an equation of the form

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

where $N(u) = uu_{xx}$ or some other type of reaction term. We can combine Crank-Nicolson for the parabolic terms and use an explicit multistep method for the non-linear term.

$$\frac{U^{n+1}-U^n}{\Delta t} = \frac{1}{2}D^2(U^{n+1}+U^n) + \left(\frac{3}{2}N(U^n) - \frac{1}{2}N(U^{n-1})\right)$$

The resulting matrix on the left hand side can be inverted using the FFT operator as before.

3.5 High-mode Filtering and Cutting

When using a pseudo-spectral method for a partial differential equation, you are approximating u by a partial Fourier sum.

$$u_N = \sum_{|k|\le N} \tilde{a}_k e^{ikx}$$

If the function u is not well resolved, then aliasing can cause high mode oscillations to appear. In addition, many non-linear problems have instabilities causing the growth of high mode oscillations. One way to control high mode oscillations is to use filtering.

Filtering is an adjustment of the wave numbers. Suppose that

$$\frac{\partial}{\partial x} u_N = \sum_{k=-N}^{N} ika_k e^{ikx}$$

then we instead use

$$\frac{\partial}{\partial x} U_N = \sum_{k=-N}^{N} ikf(k)a_k e^{ikx}$$

The function $f(k)$ is called a *filter function*. Examples of filter functions are

$$f(k) = \begin{cases} 1 & |k| \leq k_0 \\ 0 & \text{otherwise} \end{cases} \quad \text{or} \quad f(k) = \begin{cases} e^{-\alpha(|k|-|k_0|)^2} & |k| > k_0 \\ 1 & |k| \leq k_0 \end{cases}$$

A high decay rate can be achieved with the filter function

$$f(k) = \begin{cases} 1 & |k| \leq k_0 \\ e^{-\alpha(|k|-k_0)^4} & |k| > k_0 \end{cases}$$

Rewriting

$$e^{-\alpha(|k|-k_0)^4} = e^{-\tilde{\alpha}\left(\frac{|k|-k_0}{N-k_0}\right)^4}$$

we can see that for wave mode $k = N$, the amount that the highest mode is cut is

$$e^{-\tilde{\alpha}\left(\frac{N-k_0}{N-k_0}\right)^4} = e^{-\tilde{\alpha}}$$

The parameters $\tilde{\alpha}$ and γ where $k_0 \sim \gamma N$, $\gamma < 1$ are tunable according to the problem.

Obviously, filters will reduce the accuracy of the spectral approximation, but it is essential for solving
Lec. 12 problems which involve jump discontinuities or are highly non-linear with rapid spatial variation.

One common filter when using Leap-Frog is to use

$$f(k) = \frac{\sin(k\Delta t)}{k\Delta t}$$

To see how to use such a filter function, consider the wave equation

$$u_t = u_x$$

and the Leap-Frog approximation

$$U^{n+1} - U^{n-1} = 2\Delta t D U^n$$

The normal algorithm proceeds as follows. Given

$$U^n = \begin{bmatrix} u_0 \\ \vdots \\ u_{2N-1} \end{bmatrix},$$

1. Compute the pseudo-spectral approximation

$$\tilde{\alpha}_k = \frac{1}{2N}\frac{1}{c_k}\sum_{j=0}^{2N-1} u_j e^{-ikx_j}$$

2. Compute the derivative approximation u_x

$$u_x \approx \frac{d}{dx}\tilde{P}_N u(x) = \sum_{k=-N}^{N} ik\tilde{a}_j e^{ijx}$$

3. Evaluate the approximate derivative at the collocation points

$$u_x(x_j) \approx \frac{d}{dx}\tilde{P}_N u(x)\bigg|_{x=x_j} = \sum_{k=-N}^{N} ik\tilde{a}_k e^{ikx_j}$$

Now, let us look at the expression $2\Delta t DU$. The j^{th} element of this expression is

$$2\sum_{k=-N}^{N} ik\Delta t\tilde{a}_k e^{ikx_j}$$

Now we use the filter function to replace this with

$$\begin{aligned} &2\sum_{k=-N}^{N} ik\Delta t f(k)\tilde{a}_k e^{ikx_j} \\ &= 2\sum_{k=-N}^{N} ik\Delta t\frac{\sin(k\Delta t)}{k\Delta t}\tilde{a}_k e^{ikx_j} \\ &= 2\sum_{k=-N}^{N} i\sin(k\Delta t)\tilde{a}_k e^{ikx_j} \end{aligned}$$

Let us now see where this filter function comes from. Recall that the time step restrictions on Leap-Frog are determined by the high wave modes corresponding to the largest eigenvalues of D. Note that an exact solution of $u_t = u_x$ is $u(x,t) = e^{ik(x+t)}$ and if we apply Leap-Frog to this solution, we get

$$\begin{aligned} \frac{1}{2\Delta t}(u(t+\Delta t) - u(t-\Delta t)) &= \frac{1}{2\Delta t}(e^{ik(x+t+\Delta t)} - e^{ik(x+t-\Delta t)}) \\ &= \frac{1}{2\Delta t}(e^{ik\Delta t} - e^{-ik\Delta t})e^{ik(x+t)} \\ &= \frac{1}{2\Delta t}(2i\sin(k\Delta t))e^{ik(x+t)} \\ &= i\frac{\sin(k\Delta t)}{\Delta t}e^{ik(x+t)} \\ &= ik\frac{\sin(k\Delta t)}{k\Delta t}u(t) \end{aligned}$$

On the other hand, the pseudo-spectral method would give

$$\frac{1}{2\Delta t}(\tilde{u}(t+\Delta t) - \tilde{u}(t-\Delta t)) = ik\tilde{u}(t)$$

This shows that $f(k) = \frac{\sin(t\Delta t)}{k\Delta t}$ is a good choice for the filter function.

If we look at the replacement for $2\Delta t DU$, we get

$$2\sum_{k=-N}^{N} i\sin(k\Delta t)\tilde{a}_k e^{ikx_j}.$$

We see that we have replaced the large eigenvalues of D by eigenvalues of size 1. This means that we get unconditional stability while maintaining spectral accuracy.

Applying this to the heat equation, we replace $-k^2\tilde{a}_k$ with $-\sin(k^2\Delta t)\tilde{a}_k$.

4 Chebyshev Method

One drawback to the pseudo-spectral method is that it assumes a periodic solution. We will now expand our class of possible functions to those which are not assumed to be periodic. To do this, we will do expansions in Chebyshev polynomials.

We begin the discussion of Chebyshev polynomials with a discussion of cosine series. Let $g(\theta)$ be defined for $0 \leq \theta \leq \pi$. Now define $f(\theta)$ on the interval $0 \leq \theta \leq 2\pi$ by

$$f(\theta) = \begin{cases} g(\theta) & 0 \leq \theta \leq \pi \\ g(2\pi - \theta) & \pi \leq \theta \leq 2\pi \end{cases}$$

The function $f(\theta)$ is periodic on $[0, 2\pi]$, so we can compute the Fourier series for f as

$$f(\theta) = \sum_{k=-\infty}^{\infty} a_k e^{ik\theta}$$

where

$$\begin{aligned} a_k &= \frac{1}{2\pi} \int_0^{2\pi} f(\theta) e^{-ik\theta}\, d\theta \\ &= \frac{1}{2\pi} \left[\int_0^{\pi} g(\theta) e^{-ik\theta} + \int_{\pi}^{2\pi} g(2\pi - \theta) e^{-ik\theta}\, d\theta \right] \\ &= \frac{1}{2\pi} \left[\int_0^{\pi} g(\theta) e^{-ik\theta} - \int_{\pi}^{0} g(\theta) e^{-ik(2\pi-\theta)}\, d\theta \right] \\ &= \frac{1}{2\pi} \left[\int_0^{\pi} g(\theta) (e^{ik\theta} + e^{-ik\theta})\, d\theta \right] \\ &= \frac{1}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \end{aligned}$$

Lec. 13 From this we can see that $a_k = a_{-k}$, and we get

$$\begin{aligned} f(\theta) &= \sum_{k=-\infty}^{\infty} \left(\frac{1}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \right) e^{ik\theta} \\ &= \sum_{k=-\infty}^{-1} \left(\frac{1}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \right) e^{ik\theta} + \frac{1}{\pi} \int_0^{\pi} g(\theta)\, d\theta + \sum_{k=1}^{\infty} \left(\frac{1}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \right) e^{ik\theta} \\ &= \frac{1}{\pi} \int_0^{\pi} g(\theta)\, d\theta + \sum_{k=1}^{\infty} \left(\frac{1}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \right) (e^{ik\theta} + e^{-ik\theta}) \\ &= \frac{1}{\pi} \int_0^{\pi} g(\theta)\, d\theta + \sum_{k=1}^{\infty} \left(\frac{2}{\pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta \right) \cos(k\theta) \end{aligned}$$

Thus,

$$g(\theta) = \sum_{k=0}^{\infty} a_k \cos(k\theta)$$

where

$$a_k = \frac{2}{c_k \pi} \int_0^{\pi} g(\theta) \cos(k\theta)\, d\theta$$

and

$$c_k = \begin{cases} 1 & k \neq 0 \\ 2 & k = 0 \end{cases}$$

Recall that convergence of the Fourier series for f depended upon the smoothness of f. In turn, the smoothness of f depends upon both the smoothness of g and also the number of odd derivatives that match at $\theta = \pi$. To see this, consider

$$\begin{aligned} &\int_0^\pi g(\theta)\cos(k\theta)\,d\theta \\ &= \frac{1}{k}\sin(k\theta)g(\theta)\Big|_0^\pi - \int_0^\pi \frac{1}{k}g'(\theta)\sin(k\theta)\,d\theta \\ &= -\int_0^\pi \frac{1}{k}g'(\theta)\sin(k\theta)\,d\theta \\ &= \frac{1}{k^2}g'(\theta)\cos(k\theta)\Big|_0^\pi - \int_0^\pi \frac{1}{k^2}g''(\theta)\cos(k\theta)\,d\theta \end{aligned}$$

If $g'(0) = g'(\pi) = 0$, then

$$\begin{aligned} a_k &\sim -\int_0^\pi \frac{1}{k^2}g''(\theta)\cos(k\theta)\,d\theta \\ &= -\frac{1}{k^3}g''(\theta)\sin(k\theta)\Big|_0^\pi + \int_0^\pi \frac{1}{k^3}g'''(\theta)\sin(k\theta)\,d\theta \\ &= \int_0^\pi \frac{1}{k^3}g'''(\theta)\sin(k\theta)\,d\theta \\ &= -\frac{1}{k^4}g'''(\theta)\cos(k\theta)\Big|_0^\pi + \int_0^\pi \frac{1}{k^4}g^{(iv)}(\theta)\cos(k\theta)\,d\theta \\ &= O\left(\frac{1}{k^4}\right) \end{aligned}$$

If $g'(0) \neq g'(\pi)$, then

$$a_k \sim \frac{1}{k^2}((-1)^k g'(\pi) - g'(0)) = O\left(\frac{1}{k^2}\right)$$

This is true regardless of how smooth g is. If integration by parts continues, we get terms of the form

$$g^{(\ell)}(\theta)\frac{\cos(k\theta)}{k^{\ell+1}}\Big|_0^\pi, \qquad \ell = 1,\, 3,\, 5,\, \ldots$$

which will be zero only if $g^{(\ell)}(0) = g^{(\ell)}(\pi) = 0$, for $\ell = 1,\, 3,\, 5,\, \ldots$ and this does not vanish otherwise.

Thus, if g does not match derivatives at the boundary, we will see Gibbs phenomenon and high mode oscillations due to the non-smooth behavior of the function g at the boundary.

4.1 Chebyshev Expansion

Now let $f(x)$ be a function defined on $[-1, 1]$ and consider the mapping $x = \cos(\theta)$, $0 \leq \theta \leq \pi$ and let

$$g(\theta) = f(\cos(\theta))$$

Now, $g(\theta)$ is defined on $[0, \pi]$ and

$$g'(\theta) = f'(\cos(\theta))(-\sin(\theta))$$

so $g'(0) = g'(\pi) = 0$. From this, we claim that g has a cosine expansion. Furthermore, $g^{(2\ell+1)}(0) = g^{(2\ell+1)}(\pi) = 0$ for all $\ell = 0, 1, 2, \ldots$. Therefore, we can expect a very good cosine series approximation for $g(\theta)$.

Let

$$g(\theta) = \sum_{k=0}^{\infty} a_k \cos(k\theta)$$

and define $T_k(x) = \cos(k \cos^{-1}(x))$, then we have

$$\begin{aligned} f(x) &= f(\cos(\theta)) \\ &= g(\theta) \\ &= \sum_{k=0}^{\infty} a_k \cos(k\theta) \\ &= \sum_{k=0}^{\infty} a_k \cos(k \cos^{-1}(x)) \\ &= \sum_{k=0}^{\infty} a_k T_k(x) \end{aligned}$$

Lec. 14 The function $T_k(x)$ is the k^{th} *Chebyshev polynomial*.

We now wish to derive some properties of the $T_n(x)$. First, $T_n(x)$ is an n^{th} degree polynomial. To see this, we will use induction. Clearly,

$$\begin{aligned} T_0(x) &= \cos(0) = 1 \\ T_1(x) &= \cos(\cos^{-1}(x)) = x \end{aligned}$$

Now for the inductive step,

$$\begin{aligned} T_{n+1}(x) &= \cos((n+1)\cos^{-1}(x)) \\ &= \cos(n \cos^{-1}(x))\cos(\cos^{-1}(x)) - \sin(n\cos^{-1}(x))\sin(\cos^{-1}(x)) \\ T_{n-1}(x) &= \cos((n-1)\cos^{-1}(x)) \\ &= \cos(n \cos^{-1}(x))\cos(\cos^{-1}(x)) + \sin(n\cos^{-1}(x))\sin(\cos^{-1}(x)) \\ T_{n+1}(x) + T_{n-1}(x) &= 2xT_n(x) \end{aligned}$$

Thus,

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

The inductive hypothesis gives that $T_n(x)$ and $T_{n-1}(x)$ are polynomials of degree n and $n-1$ respectively, and therefore, T_{n+1} must be a polynomial of degree $n+1$.

Now, we wish to expand a function $f(x)$ as

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x)$$

so we must determine the values of the a_k. From the derivation, we have that

$$a_k = \frac{2}{c_k \pi} \int_0^{\pi} f(\cos(\theta)) \cos(k\theta)\, d\theta$$

Let $x = \cos(\theta)$, $dx = -\sin(\theta)\,d\theta = -\sqrt{1-x^2}\,d\theta$, and we have

$$\begin{aligned} a_k &= \frac{2}{c_k\pi}\int_{-1}^{1}\frac{f(x)\cos(k\cos^{-1}(x))}{\sqrt{1-x^2}}\,dx \\ &= \frac{2}{c_k\pi}\int_{-1}^{1}\frac{f(x)}{\sqrt{1-x^2}}T_k(x)\,dx \end{aligned}$$

The function $w(x) = (1-x^2)^{-1/2}$ is called the *Chebyshev weight function.*

Next, note that

$$\begin{aligned} &\int_0^\pi \cos(k\theta)\cos(\ell\theta)\,d\theta \\ &= \int_0^\pi \frac{1}{2}(\cos((k+\ell)\theta) + \cos((k-\ell)\theta))\,d\theta \\ &= \frac{1}{2}\left(\frac{1}{k+\ell}\sin((k+\ell)\theta) + \frac{1}{k-\ell}\sin((k-\ell)\theta)\right)\Bigg|_0^\pi \\ &= 0 \qquad \text{if } k \neq \ell \end{aligned}$$

If $k = \ell \neq 0$, then

$$\begin{aligned} &\int_0^\pi \cos^2(k\theta)\,d\theta \\ &= \int_0^\pi \frac{1+\cos(2\theta)}{2}\,d\theta \\ &= \left(\frac{\theta}{2} + \frac{1}{4}\sin(2\theta)\right)\Bigg|_0^\pi \\ &= \frac{\pi}{2} \end{aligned}$$

If $k = \ell = 0$, then $\int_0^\pi d\theta = \pi$. Now translate this integral back into x to get

$$\int_{-1}^{1} w(x)T_k(x)T_\ell(x)\,dx = \begin{cases} 0 & k \neq \ell \\ \frac{\pi}{2} & k = \ell \neq 0 \\ \pi & k = \ell = 0 \end{cases}$$

This shows that if we define the inner product $\langle u, v\rangle_w$ as

$$\langle u, v\rangle_w = \int_{-1}^{1} w(x)u(x)\bar{v}(x)\,dx$$

then the $T_n(x)$ are mutually orthogonal. Let C be the space of all functions u such that

$$\langle u, u\rangle_w = \int_{-1}^{1} w(x)|u(x)|^2\,dx < \infty$$

Then C is a Hilbert space with norm $||u|| = \langle u, u\rangle_w^{1/2}$ and inner product $\langle\cdot,\cdot\rangle_w$. Note that $w(x) = \frac{1}{\sqrt{1-x^2}}$ is singular at $x = \pm 1$, but still integrable. Lec. 15

The Chebyshev expansion is equivalent to the cosine expansion of $f(\cos(\theta))$ and using the same integration by parts argument we see that the coefficients $a_k = O\left(\frac{1}{k^r}\right)$ if f has r derivatives. It is important to realize that this is true regardless of the behavior of f at ±1. Therefore, the Chebyshev expansion is an analogous approximation to the Fourier expansion for functions which are not periodic.

Next, let us find the derivative of T_n. Consider again $x = \cos(\theta)$. Thus, for any function u,

$$\frac{du}{d\theta} = \frac{du}{dx}\frac{dx}{d\theta} = \frac{du}{dx}(-\sin(\theta)) = \frac{du}{dx}(-\sqrt{1-x^2})$$

and we can write

$$\frac{d}{d\theta} = -\sqrt{1-x^2}\frac{d}{dx}$$

Now, since

$$\frac{d^2}{d\theta^2}\cos(n\theta) + n^2\cos(n\theta) = 0$$

we use the $x = \cos(\theta)$ transformation to get

$$\begin{aligned}
\frac{d^2}{d\theta^2}T_n(x) + n^2 T_n(x) &= 0 \\
\frac{d}{d\theta}\left(-\sqrt{1-x^2}\frac{d}{dx}T_n(x)\right) + n^2 T_n(x) &= 0 \\
-\sqrt{1-x^2}\frac{d}{dx}\left(-\sqrt{1-x^2}\frac{d}{dx}T_n(x)\right) + n^2 T_n(x) &= 0 \\
\frac{d}{dx}\left(\sqrt{1-x^2}\frac{d}{dx}T_n(x)\right) + \frac{n^2}{\sqrt{1-x^2}}T_n(x) &= 0 \\
\frac{d}{dx}\left(\frac{1}{w}\frac{d}{dx}T_n(x)\right) + n^2 w T_n(x) &= 0 \qquad (6)
\end{aligned}$$

with $T_n(\pm 1)$ bounded. Equation (6) is a singular Sturm-Liouville problem solved by T_n with eigenvalue n^2. Unfortunately, differentiation in x is <u>not</u> diagonal in the basis $T_n(x)$. In fact, it is easy to see that $\frac{d}{dx}T_n(x)$ is a polynomial of degree $n-1$, one degree less than $T_n(x)$.

Next, let us look at the structure of the functions $T_n(x)$. Let's find the zeros of $T_n(x)$. To do this, we must solve the equation

$$\begin{aligned}
0 &= \cos(n\cos^{-1}(x)) \\
n\cos^{-1}(x) &= \frac{(2\ell+1)\pi}{2} \\
\cos^{-1}(x) &= \frac{(2\ell+1)\pi}{2n} \\
x &= \cos\left(\frac{(2\ell+1)\pi}{2n}\right)
\end{aligned}$$

So the roots of $T_n(x)$ are at $x_\ell = \cos\left(\frac{(2\ell+1)\pi}{2n}\right)$ for $\ell = 0, \ldots, n-1$.

Clearly, $|T_n(x)| \leq 1$ since $T_n(x) = \cos(n\cos^{-1}(x))$ and $T_n(x) = (-1)^n$ whenever $n\cos^{-1}(x) = \ell\pi$ and hence $x = \cos\left(\frac{\ell\pi}{n}\right)$ for $\ell = 0, \ldots, n$. These points cluster at the boundary much the same way that the roots cluster near the boundary. In fact, $T_n(x)$ oscillates between ± 1. This is called the *equi-oscillation property.*

Next, note that $T_{2n}(x)$ is an even function and $T_{2n-1}(x)$ is an odd function. If $T_n(x) = \sum_j a_j x^j$, then the a_j's alternate in sign.

Next, we need to analyze the accuracy of the Chebyshev approximations. Suppose f is defined on $[-1, 1]$. Note here that the interval $[a, b]$ can be mapped to $[-1, 1]$ by the mapping

$$S = \frac{2x}{b-a} - \frac{b+a}{b-a}.$$

Let f have the Chebyshev expansion

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x)$$

where

$$\begin{aligned}
a_k &= \frac{2}{c_k\pi}\int_{-1}^{1} w(x)f(x)T_k(x)\,dx \\
&= \frac{2}{c_k\pi}\int_{-1}^{1} f(x)(w(x)T_k(x))\,dx \\
&= -\frac{2}{k^2c_k\pi}\int_{-1}^{1} f(x)\frac{d}{dx}\left(\sqrt{1-x^2}\frac{dT_k}{dx}\right)dx \qquad \text{(from (6))} \\
&= -\frac{2}{k^2c_k\pi}f(x)\sqrt{1-x^2}\frac{dT_k}{dx}\bigg|_{x=-1}^{1} + \frac{2}{k^2c_k\pi}\int_{-1}^{1} f'(x)\sqrt{1-x^2}\frac{dT_k}{dx}\,dx \\
&= \frac{2}{k^2c_k\pi}\int_{-1}^{1} f'(x)\sqrt{1-x^2}\frac{dT_k}{dx}\,dx
\end{aligned}$$

Note here that this does not mean that $a_k = O\left(\frac{1}{k^2}\right)$. In fact, $\frac{dT_k}{dx} = O(k)$ and hence $a_k = O\left(\frac{1}{k}\right)$. Integrate by parts again to get

$$\begin{aligned}
&= \frac{2}{k^2c_k\pi}f'(x)\sqrt{1-x^2}T_k(x)\bigg|_{-1}^{1} - \frac{2}{k^2c_k\pi}\int_{-1}^{1}\frac{d}{dx}\left(f'(x)\sqrt{1-x^2}\right)T_k(x)\,dx \\
&= \frac{-c_k}{k^2\pi}\int_{-1}^{1}\frac{d}{dx}\left(\sqrt{1-x^2}f'(x)\right)T_k(x)\,dx \\
&= \frac{-c_k}{k^2\pi}\int_{-1}^{1}(1-x^2)^{1/4}\frac{d}{dx}\left(\sqrt{1-x^2}f'(x)\right)\frac{T_k(x)}{(1-x^2)^{1/4}}\,dx
\end{aligned}$$

From this we get

$$\begin{aligned}
|a_k| &= \frac{2}{k^2c_k\pi}\left|\int_{-1}^{1}(1-x^2)^{1/4}\frac{d}{dx}(\sqrt{1-x^2}f'(x))\frac{T_k(x)}{(1-x^2)^{1/4}}\,dx\right| \\
&\le \frac{2}{k^2c_k\pi}\left(\int_{-1}^{1}\sqrt{1-x^2}\left[\frac{d}{dx}(\sqrt{1-x^2}f'(x))\right]^2dx\right)^{1/2}\left(\int_{-1}^{1}\frac{T_k^2(x)}{\sqrt{1-x^2}}\,dx\right)^{1/2} \\
&= \frac{2}{k^2c_k\pi}\left(\int_{-1}^{1}\sqrt{1-x^2}\left[\frac{d}{dx}(\sqrt{1-x^2}f'(x))\right]^2dx\right)^{1/2}\langle T_k,T_k\rangle_w^{1/2} \\
&= \frac{2}{k^2c_k\pi}\left(\int_{-1}^{1}\sqrt{1-x^2}\left[\frac{d}{dx}(\sqrt{1-x^2}\frac{d}{dx})f(x)\right]^2dx\right)^{1/2}\left(\frac{\pi c_k}{2}\right)^{1/2} \\
&= \frac{1}{k^2}\left(\int_{-1}^{1}\sqrt{1-x^2}\left[\frac{d}{dx}(\sqrt{1-x^2}\frac{d}{dx})f(x)\right]^2dx\right)^{1/2}\left(\frac{2}{c_k\pi}\right)^{1/2}
\end{aligned}$$

Therefore, if f has two derivatives, then a_k decays as $\frac{1}{k^2}$. In general, we can continue this process and conclude that if f has $2r$ derivatives, then a_k decays as $a_k = O\left(\frac{1}{k^{2r}}\right)$.

Next, we need to project a function f onto a subspace of the T_k, so we define the Chebyshev Galerkin approximation (spectral approximation)

$$f_N(x) = P_Nf(x) = \sum_{k=0}^{N} a_kT_k(x)$$

Thus, P_N is an N^{th} degree polynomial and has the first $N+1$ terms of the Chebyshev expansion.

To see how good this approximation is, we need to estimate $|f - P_N f|$. We hope that $|f - P_N f| = O\left(\frac{1}{N^{2r}}\right)$ provided f has $2r$ derivatives. To give a complete picture, we must first recall the Sobolev norms which are now weighted by w. The q^{th} weighted Sobolev norm is given by

$$||f||_{q,w}^2 = \sum_{j=0}^{q} \left\| \frac{d^j}{dx^j} f(x) \right\|_w^2$$

Based upon the Sobolev norm, if f has q derivatives, then it is possible to show

$$\|f - f_N\|_{0,w} \leq \frac{C}{N^q} \|f\|_{q,w}$$

and if $1 \leq r \leq q$,

$$\|f - f_N\|_{r,w} \leq \frac{C}{N^{q-2r+\frac{1}{2}}} \|f\|_{q,w}$$

This shows that the weighted L_2 norm of the error is $O\left(\frac{1}{N^r}\right)$ and that approximating the r^{th} derivative results in the loss of $2r$ powers of N (as opposed to r in pseudo-spectral).

Next, let us construct the corresponding pseudo-spectral approximation. To obtain the results above, we must compute the coefficients a_k exactly. However, we may only know the function f at discrete data points. Therefore, we need a quadrature rule as we did for Fourier series. We have

$$a_k = \frac{2}{c_k \pi} \int_{-1}^{1} \frac{f(x) T_k(x)}{\sqrt{1 - x^2}}\, dx = \frac{2}{c_k \pi} \int_0^{\pi} f(\cos(\theta)) \cos(k\theta)\, d\theta$$

Now introduce a uniform grid in θ, $\theta_j = \frac{j\pi}{N}$, for $j = 0, \ldots, N$ and use the trapezoidal rule approximation to get

$$a_k \approx \frac{2}{c_k \pi} \frac{\pi}{N} \sum_{j=0}^{N} \frac{f(\cos(\theta_j)) \cos(k\theta_j)}{\gamma_j} = \tilde{a}_k$$

where $\gamma_0 = \gamma_N = 2$, $\gamma_j = 1$ for $1 \leq j \leq N - 1$. Pulling back to x, we have that $x_j = \cos\left(\frac{j\pi}{N}\right)$ and we get

$$\tilde{a}_k = \frac{2}{N c_k} \sum_{j=0}^{N} \frac{f(x_j) T_k(x_j)}{\gamma_j}$$

We must modify this to accommodate for the highest mode, $T_N(x_j) = (-1)^j$ and we get

$$\tilde{a}_k = \frac{2}{N \gamma_k} \sum_{j=0}^{N} \frac{f(x_j) T_k(x_j)}{\gamma_j}, \qquad \tilde{f}_N(x) = \sum_{k=0}^{N} \tilde{a}_k T_k(x)$$

Lec. 16 is the *pseudo-spectral Chebyshev approximation.*

To look at the accuracy of the quadrature rule we are using, consider the following: Let $g(\theta)$ be defined on $[0, \pi]$, and we have the trapezoidal rule

$$\int_0^{\pi} g(\theta)\, d\theta \approx \frac{\pi}{N} \sum_{j=0}^{N} \frac{g(\theta_j)}{\gamma_j}$$

Suppose that $g(\theta) = \cos(\ell\theta)$, then

$$\int_0^{\pi} \cos(\ell\theta)\, d\theta = \begin{cases} \pi & \text{if } \ell = 0 \\ 0 & \text{if } \ell \neq 0 \end{cases}$$

If $\ell = 0$, the approximation gives

$$\frac{\pi}{N}\sum_{j=0}^{N}\frac{1}{\gamma_j} = \frac{\pi}{N}\left(\frac{1}{2}+N-1+\frac{1}{2}\right) = \pi$$

If $\ell = 2m$, then

$$\begin{aligned}
\frac{\pi}{N}\sum_{j=0}^{N}\frac{\cos(2m\theta_j)}{\gamma_j} &= \frac{\pi}{N}\sum_{j=0}^{N}\frac{1}{\gamma_j}\cos\left(2m\frac{\pi}{N}j\right) \\
&= \frac{\pi}{2N}+\frac{\pi}{N}\sum_{j=1}^{N-1}\cos\left(2m\frac{\pi}{N}j\right)+\frac{\pi}{2N} \\
&= \frac{\pi}{2N}-\frac{\pi}{N}+\frac{\pi}{N}\sum_{j=0}^{N-1}\cos\left(2m\frac{\pi}{N}j\right)+\frac{\pi}{2N} \\
&= \frac{\pi}{2N}\sum_{j=0}^{N-1}\left(e^{i2m\frac{\pi}{N}j}+e^{-i2m\frac{\pi}{N}j}\right) \\
&= \frac{\pi}{2N}\left(\sum_{j=0}^{N-1}\left(e^{i2m\frac{\pi}{N}}\right)^j+\sum_{j=0}^{N-1}\left(e^{-i2m\frac{\pi}{N}}\right)^j\right) \\
&= \frac{\pi}{2N}\left(\frac{e^{i2m\frac{\pi}{N}N}-1}{e^{i2m\frac{\pi}{N}}-1}\right)+\frac{\pi}{2N}\left(\frac{e^{-i2m\frac{\pi}{N}N}-1}{e^{-i2m\frac{\pi}{N}}-1}\right) \\
&= 0
\end{aligned}$$

If $\ell = 2m+1$, then note that $\cos\left((2m+1)\frac{\pi}{N}(N-j)\right) = -\cos\left((2m+1)\frac{\pi}{N}j\right)$ so that if N is odd, all the terms cancel, and if N is even, all the terms cancel except the middle term which is $\cos\left((2m+1)\frac{\pi}{2M}M\right) = 0$. Thus, we again have $\frac{\pi}{N}\sum_{j=0}^{N}\frac{1}{\gamma_j}\cos(\ell\theta_j) = 0$. Finally, if $\ell = 2N$, then

$$\frac{\pi}{N}\sum_{j=0}^{N}\frac{1}{\gamma_j}\cos\left(2N\frac{\pi}{N}j\right) = \frac{\pi}{N}\sum_{j=0}^{N}\frac{1}{\gamma_j} = \pi$$

Now, if we carry back the approximation

$$\int_0^{\pi} g(\theta)\,d\theta \approx \frac{\pi}{N}\sum_{j=0}^{N}\frac{g(\theta_j)}{\gamma_j}$$

via $x = \cos(\theta)$, we get

$$\int_{-1}^{1}\frac{f(x)}{\sqrt{1-x^2}}\,dx \approx \frac{\pi}{N}\sum_{j=0}^{N}\frac{f(x_j)}{\gamma_j}$$

and this is exact for polynomials up to degree $2N-1$. Using this result, we can show the interpolation result given $f(x)$. Define

$$\tilde{P}_N f = \sum_{j=0}^{N}\tilde{a}_j T_j(x)$$

where

$$\tilde{a}_j = \frac{2}{\gamma_j N}\sum_{k=0}^{N}\frac{f(x_k)}{\gamma_k}T_j(x_k)$$

Then $\tilde{P}_N f$ is the unique N^{th} degree polynomial which interpolates f at the collocation points $x_j = \cos\left(\frac{j\pi}{N}\right)$.

To see this, let p_N be the N^{th} degree polynomial which interpolates f at $\{x_j\}$. Then

$$p_N(x) = \sum_{k=0}^{N} b_k T_k(x)$$

for some constants b_k. We must show that $b_k = \tilde{a}_k$. By definition of p_N, we must have that

$$f(x_j) = \sum_{k=0}^{N} b_k T_k(x_j), \qquad \text{for } j = 0, \ldots, N$$

Now, for $\ell < N$,

$$\begin{aligned}\sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_\ell(x_j) &= \sum_{j=0}^{N} \frac{1}{\gamma_j} \sum_{k=0}^{N} b_k T_k(x_j) T_\ell(x_j) \\ &= \sum_{k=0}^{N} b_k \sum_{j=0}^{N} \frac{1}{\gamma_j} T_k(x_j) T_\ell(x_j) \\ &= \sum_{k=0}^{N} b_k \frac{N}{\pi} \int_{-1}^{1} w(x) T_k(x) T_\ell(x)\, dx\end{aligned}$$

because the quadrature is exact for polynomial of degree $\leq 2N - 1$

$$\begin{aligned}&= \sum_{k=0}^{N} b_k \frac{N}{\pi} \frac{\pi c_k}{2} \delta_{\ell k} \\ &= b_\ell \frac{N c_\ell}{2} \\ &= b_\ell \frac{N \gamma_\ell}{2}\end{aligned}$$

If $\ell = N$, then

$$\begin{aligned}\sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_N(x_j) &= \sum_{k=0}^{N} b_k \sum_{j=0}^{N} \frac{1}{\gamma_j} T_k(x_j) T_N(x_j) \\ &= \sum_{k=0}^{N-1} b_k \sum_{j=0}^{N} \frac{1}{\gamma_j} T_k(x_j) T_N(x_j) + b_N \sum_{j=0}^{N} \frac{1}{\gamma_j} T_N^2(x_j) \qquad (7)\end{aligned}$$

Now note that

$$\begin{aligned}\sum_{j=0}^{N} \frac{1}{\gamma_j} T_N^2(x_j) &= \sum_{j=0}^{N} \frac{1}{\gamma_j} \cos^2\left(N \frac{\pi}{N} j\right) \\ &= \sum_{j=0}^{N} \frac{1}{\gamma_j} \left[\cos^2\left(N \frac{\pi}{N} j\right) - \sin^2\left(N \frac{\pi}{N} j\right)\right] \\ &= \sum_{j=0}^{N} \frac{1}{\gamma_j} \cos\left(2N \frac{\pi}{N} j\right) \\ &= \sum_{j=0}^{N} \frac{1}{\gamma_j} = N\end{aligned}$$

Plugging this into equation (7) gives

$$\begin{aligned}
\sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_N(x_j) &= \sum_{k=0}^{N-1} b_k \sum_{j=0}^{N} \frac{1}{\gamma_j} T_k(x_j) T_N(x_j) + N b_N \\
&= \sum_{k=0}^{N-1} b_k \frac{N}{\pi} \int_{-1}^{1} w(x) T_k(x) T_N(x)\, dx + N b_N \\
&= \sum_{k=0}^{N-1} b_k \frac{N}{\pi} \delta_{kN} \frac{\pi}{2} + N b_N \\
&= N b_N \\
&= b_N \frac{N \gamma_N}{2}
\end{aligned}$$

Therefore,

$$\sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_\ell(x_j) = b_\ell \frac{N \gamma_\ell}{2}$$

$$b_\ell = \frac{2}{\pi \gamma_\ell} \frac{\pi}{N} \sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_\ell(x_j) = \tilde{a}_\ell$$

Lec. 17

Let us now put together the pieces. Given $f(x)$ on $[-1,1]$, we evaluate $f(x_j)$ where $x_j = \cos\left(\frac{\pi}{N} j\right)$ for $j = 0$,...,N, and compute for $\ell = 0, \ldots, N$

$$\tilde{a}_\ell = \frac{2}{N \gamma_\ell} \sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) T_\ell(x_j)$$

and the pseudo-spectral Chebyshev approximation becomes

$$\tilde{f}_N(x) = \sum_{\ell=0}^{N} \tilde{a}_\ell T_\ell(x)$$

and for $j = 0, \ldots, N$, $\tilde{f}_N(x_j) = f(x_j)$.

It is worth noting that the evaluation of the $\tilde{a}_k$ can be simplified by using the fact that since $x_j = \cos(\theta_j)$,

$$\begin{aligned}
T_k(x_j) &= \cos(k \cos^{-1}(\cos(\theta_j))) \\
&= \cos(k \theta_j) \\
&= \cos\left(kj \frac{\pi}{N}\right)
\end{aligned}$$

Therefore,

$$\tilde{a}_k = \frac{2}{N \gamma_k} \sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) \cos\left(kj \frac{\pi}{N}\right)$$

This relationship can also be used to demonstrate that aliasing is a problem for this method as well.

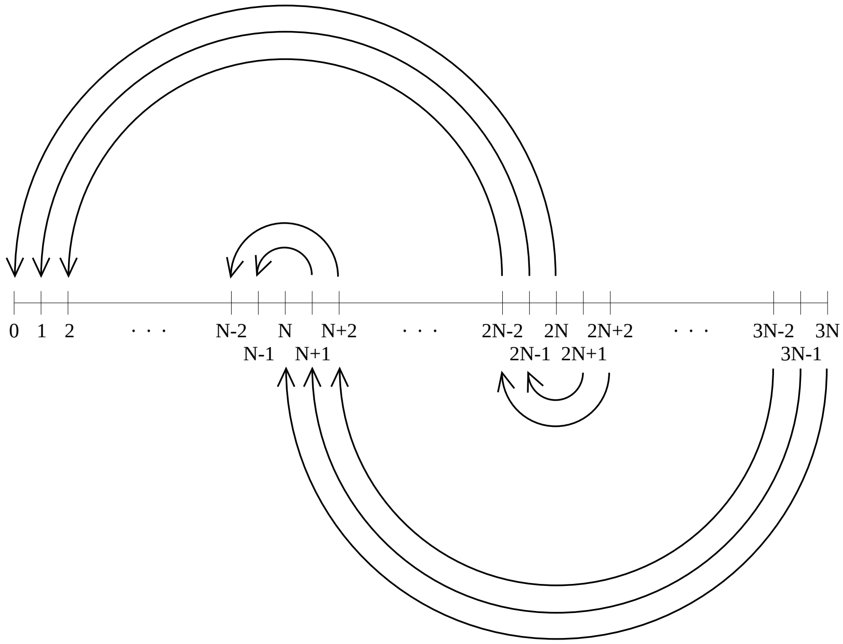
Consider the mode $T_{N+r}(x)$ for $0 < r \le N$. We have

$$\begin{aligned}
T_{N+r}(x_j) &= \cos\left((N+r)j\frac{\pi}{N}\right) \\
&= \cos\left(j\pi + rj\frac{\pi}{N}\right) \\
&= \cos(j\pi)\cos\left(rj\frac{\pi}{N}\right) - \sin(j\pi)\sin\left(rj\frac{\pi}{N}\right) \\
&= \cos(j\pi)\cos\left(rj\frac{\pi}{N}\right) + \sin(j\pi)\sin\left(rj\frac{\pi}{M}\right) \\
&= \cos\left(j\pi - rj\frac{\pi}{N}\right) \\
&= \cos\left((N-r)j\frac{\pi}{N}\right) \\
&= T_{N-r}(x_j)
\end{aligned}$$

Therefore, T_{N+r} aliases to the mode T_{N-r}. In particular, this means that T_{2N} aliases to $T_0 \equiv 1$. Similarly, consider the mode $T_{2mN+r}(x)$, $0 \le r < 2N$, $m \ge 0$. In this case, we get

$$\begin{aligned}
T_{2mN+r}(x_j) &= \cos\left((2mN+r)j\frac{\pi}{N}\right) \\
&= \cos(2mj\pi)\cos\left(rj\frac{\pi}{N}\right) - \sin(2mj\pi)\sin\left(rj\frac{\pi}{N}\right) \\
&= \cos\left(rj\frac{\pi}{N}\right) \\
&= T_r(x_j)
\end{aligned}$$

To illustrate, the pattern of aliasing can be seen in the figure below:



Aliasing relationships of Chebyshev polynomials

Just like for the Fourier case, we can relate the Chebyshev expansion coefficients a_k to the pseudo-spectral Chebyshev coefficients $\tilde{a}_k$. The relationship is similar as well where

$$\tilde{a}_k = a_k + \sum a_\ell$$

where the sum is taken over all ℓ for which T_ℓ aliases to T_k. The polynomials which alias to T_k are T_j where $j = 2mN + k$ for $m = 1, 2, \ldots$ and $j = (2m-1)N + N - k$ for $m = 1, 2, \ldots$. It can be shown that

$$\|f - \tilde{P}_N f\|_\infty \leq C \log(N) \|f - P_N f\|_\infty$$

($\|f - \tilde{P}_N f\|_\infty$ is called the *spectral interpolation error*). Sobolev norm estimates for the pseudo-spectral approximation similar to the bounds for the spectral approximation can also be derived.

Next, we will see how to use the FFT to implement this method. Recall that we can now compute the $\tilde{a}_k$ by means of a cosine transform.

$$\tilde{a}_k = \frac{\pi}{N} \frac{2}{\pi \gamma_k} \sum_{j=0}^{N} \frac{1}{\gamma_j} f(x_j) \cos\left(\frac{jk\pi}{N}\right)$$

where $\gamma_0 = \gamma_N = 2$, $\gamma_j = 1$ otherwise. Suppose f is a real-valued function, then

$$\tilde{a}_k = \frac{\pi}{N} \frac{2}{\pi \gamma_k} \frac{1}{2} \left[f(x_0) \cos\left(\frac{0k\pi}{N}\right) + f(x_N) \cos\left(\frac{NK\pi}{N}\right) \right] + \frac{\pi}{N} \frac{2}{\pi \gamma_k} \sum_{j=1}^{N-1} f(x_j) \cos\left(\frac{jk\pi}{N}\right) \tag{8}$$

Now, define $f_j = f(x_j)$ and set $f_{2N-j} = f_j$ for $j = 1, \ldots, N-1$ and we have

$$\sum_{\ell=N+1}^{2N-1} f_\ell \cos\left(\frac{\ell \pi k}{N}\right) = \sum_{j=1}^{N-1} f_j \cos\left(\frac{k\pi j}{N}\right)$$

Plugging this into equation (8) gives

$$\begin{aligned}
\tilde{a}_k &= \frac{\pi}{N} \frac{2}{\pi \gamma_k} \frac{1}{2} \left[f_0 \cos\left(\frac{0k\pi}{N}\right) + f_N \cos\left(\frac{Nk\pi}{N}\right) \right] \\
&\quad + \frac{\pi}{N} \frac{2}{\pi \gamma_k} \frac{1}{2} \sum_{j=1}^{N-1} f_j \cos\left(\frac{jk\pi}{N}\right) + \frac{\pi}{N} \frac{2}{\pi \gamma_k} \frac{1}{2} \sum_{j=N+1}^{2N-1} f_j \cos\left(\frac{jk\pi}{N}\right) \\
&= \frac{1}{N\gamma_k} \sum_{j=0}^{2N-1} f_j \cos\left(\frac{jk\pi}{N}\right) \\
&= \frac{1}{N\gamma_k} \operatorname{Re}\left[\sum_{j=0}^{2N-1} f_j e^{\frac{ijk\pi}{N}} \right]
\end{aligned}$$

This last term is easily evaluated using an FFT.

4.2 Differentiation of $T_k(x)$

Before we can use this expansion to solve partial differential equations, we need to be able to compute the derivative of the expansion. Consider

$$\tilde{P}_N f = \sum_{j=0}^{N} \tilde{a}_j T_j(x) \approx f(x)$$

Then,

$$f_x \approx \frac{d\tilde{P}_N f}{dx} = \sum_{j=0}^{N} \tilde{a}_j T_j'(x)$$

Since $\frac{d\tilde{P}_N f}{dx}$ is a polynomial of degree $N-1$ and the $\{T_N\}$ are complete, it follows that we can express $\frac{d\tilde{P}_N f}{dx}$ as a sum

$$\frac{d\tilde{P}_N f}{dx} = \sum_{j=0}^{N} \tilde{a}_j \frac{dT_j}{dx} = \sum_{j=0}^{N} b_j T_j(x)$$

Clearly, $b_N = 0$ because $\frac{d\tilde{P}_N f}{dx}$ is a polynomial of degree $N-1$. Computing the remainder of the b_j's is more complicated than in the Fourier case because the $\frac{d}{dx}$ operator is not a diagonal operator.

We can generate a recursive relation for the T_k's by looking at

$$\begin{aligned} T_m(x) &= \cos(m\theta), \qquad x = \cos(\theta) \\ \frac{d}{dx}T_m(x) &= \frac{dT}{d\theta}\frac{d\theta}{dx} = -m\sin(m\theta)\frac{d\theta}{dx} \\ \frac{dx}{d\theta} &= -\sin(\theta), \text{ so } \frac{d\theta}{dx} = \frac{-1}{\sin(\theta)} \end{aligned}$$

Hence, we get the relations

$$\begin{aligned} \frac{d}{dx}T_m(x) &= \frac{m\sin(m\theta)}{\sin(\theta)} \\ \frac{d}{dx}T_{m-2}(x) &= \frac{(m-2)\sin((m-2)\theta)}{\sin(\theta)} \end{aligned}$$

Now,

$$\begin{aligned} \sin(m\theta) &= \sin((m-1+1)\theta) = \sin((m-1)\theta)\cos(\theta) + \sin(\theta)\cos((m-1)\theta) \\ \sin((m-2)\theta) &= \sin((m-1-1)\theta) = \sin((m-1)\theta)\cos(\theta) - \sin(\theta)\cos((m-1)\theta) \end{aligned}$$

Then we get

$$\begin{aligned} \frac{1}{m}\frac{d}{dx}T_m(x) &= \frac{1}{\sin(\theta)}(\sin((m-1)\theta)\cos(\theta) + \sin(\theta)\cos((m-1)\theta)) \\ \frac{1}{m-2}\frac{d}{dx}T_{m-2}(x) &= \frac{1}{\sin(\theta)}(\sin((m-1)\theta)\cos(\theta) - \sin(\theta)\cos((m-1)\theta)) \\ \frac{1}{m}\frac{d}{dx}T_m(x) - \frac{1}{m-2}\frac{d}{dx}T_{m-2}(x) &= 2\cos((m-1)\theta) \\ &= 2T_{m-1}(x) \end{aligned}$$

for $m \geq 3$. For smaller m, we have

$$\begin{aligned} &T_0(x) = 1, \qquad T_0'(x) = 0 \\ &T_1(x) = x, \qquad T_1'(x) = 1 \\ &T_2(x) = 2x^2 - 1, \qquad T_2'(x) = 4x = 4T_1 \end{aligned}$$

Lec. 18

Returning to the computation of the b_k, recall

$$\begin{aligned}
\langle \tilde{P}_N f'(x), T_\ell(x)\rangle_w &= \left\langle \sum_{k=0}^{N} b_k T_k(x), T_\ell(x) \right\rangle_w \\
&= \sum_{k=0}^{N} b_k \langle T_k(x), T_\ell(x)\rangle_w \\
&= \sum_{k=0}^{N} b_k \frac{\pi c_\ell}{2}\delta_{k\ell} \\
&= b_\ell \frac{\pi c_\ell}{2}
\end{aligned}$$

where

$$c_\ell = \begin{cases} 2 & \ell = 0 \\ 1 & \ell > 0 \end{cases}$$

Also,

$$\langle \tilde{P}_N f'(x), T_\ell(x)\rangle_w = \left\langle \sum_{k=0}^{N} \tilde{a}_k T'_k(x), T_\ell(x) \right\rangle_w = \sum_{k=0}^{N} \tilde{a}_k \langle T'_k(x), T_\ell(x)\rangle_w$$

Therefore,

$$b_\ell = \frac{2}{\pi c_\ell} \sum_{k=0}^{N} \tilde{a}_k \langle T'_k, T_\ell\rangle_w$$

So now we have to compute $\langle T'_k, T_\ell\rangle_w$. We break this into three cases.

Case 1: $\ell \geq k$. In this case, if we express $T'_k(x)$ as $\sum_{m=0}^{k-1} \alpha_m T_m(x)$, we get

$$\left\langle \sum_{m=0}^{k-1} \alpha_m T_m(x), T_\ell(x) \right\rangle_w = \sum_{m=0}^{k-1} \alpha_m \langle T_m, T_\ell\rangle_w = 0$$

because $m < \ell$ for $m = 0, \ldots, k-1$. Thus, $\langle T'_k, T_\ell\rangle_w = 0$.

Case 2: $k + \ell$ is even. If k is odd, then T'_k is an even function with only even powers of x. At the same time, ℓ must also be odd, and hence T_ℓ is an odd function and thus, $w(x)T'_k(x)T_\ell(x)$ is an odd function, so

$$\langle T'_k(x), T_\ell(x)\rangle_w = \int_{-1}^{1} w(x)T'_k(x)T_\ell(x)\,dx = 0$$

Case 3: $k + \ell$ is odd. If $\ell \neq 0$, then $k = \ell + 2r - 1$ for some $r \geq 1$. If $r = 1$, then $k = \ell + 1$ and the recursion relation becomes

$$2T_\ell(x) = \frac{T'_{\ell+1}(x)}{\ell + 1} - \frac{T'_{\ell-1}(x)}{\ell - 1}$$

which is valid for $\ell > 1$ and still alright for $\ell = 1$ if we assume the last term is taken to be zero. Now take the inner product with T_ℓ to get

$$\begin{aligned}
2\langle T_\ell, T_\ell\rangle_w &= \frac{1}{\ell+1}\langle T'_{\ell+1}, T_\ell\rangle_w - \frac{1}{\ell-1}\langle T'_{\ell-1}, T_\ell\rangle_w \\
2(\ell+1)\frac{\pi c_\ell}{2} &= \langle T'_k, T_\ell\rangle_w \\
\pi k = \pi(\ell+1) = (\ell+1)\pi c_\ell &= \langle T'_k, T_\ell\rangle_w
\end{aligned}$$

Thus,

$$\langle T'_k, T_\ell\rangle_w = \pi k$$

Now from the recursion relation, we have

$$2T_{\ell+2r}(x) = \frac{1}{\ell+2r+1}T'_{\ell+2r+1}(x) - \frac{1}{\ell+2r-1}T'_{\ell+2r-1}(x)$$

Taking the inner product with T_ℓ gives

$$2\langle T_{\ell+2r}, T_\ell\rangle_w = \frac{1}{\ell+2r+1}\langle T'_{\ell+2r+1}, T_\ell\rangle_w - \frac{1}{\ell+2r-1}\langle T'_{\ell+2r-1}, T_\ell\rangle_w \tag{9}$$

By the inductive hypothesis, $\langle T'_{\ell+2r-1}, T_\ell\rangle_w = (\ell+2r-1)\pi$. Thus,

$$\begin{aligned}\langle T'_{\ell+2r+1}, T_\ell\rangle_w &= \frac{\ell+2r+1}{\ell+2r-1}\langle T'_{\ell+2r-1}, T_\ell\rangle_w \\ &= \frac{\ell+2r+1}{\ell+2r-1}(\ell+2r-1)\pi \\ &= (\ell+2r+1)\pi\end{aligned}$$

Therefore, $\langle T'_k, T_\ell\rangle_w = k\pi$ provided $\ell \neq 0$.

Next, if $\ell = 0$, then $T'_1(x) = 1 = T_0(x)$ and

$$\langle T'_1, T_0\rangle_w = \langle T_0, T_0\rangle_w = 1\cdot\pi$$

Using the same inductive argument as in equation (9), we can conclude $\langle T'_k, T_0\rangle_w = k\pi$.

Recall now that we are looking for the b_k such that

$$\sum_{k=0}^{N} b_k T_k(x) = \sum_{k=0}^{N} \tilde{a}_k T'_k(x)$$

and we found that

$$\begin{aligned}b_\ell &= \frac{2}{\pi c_\ell}\sum_{k=0}^{N}\tilde{a}_k\langle T'_k, T_\ell\rangle_w \\ &= \frac{2}{c_\ell}\sum_{k=\ell+1}^{N}\tilde{a}_k\frac{k}{2}(1-(-1)^{\ell+k})\end{aligned}$$

We can derive a recursion relation for the b_ℓ's to reduce the computational cost of this formula. If $0 \le \ell < N-1$, then

$$\begin{aligned}b_\ell &= \frac{2}{c_\ell}\left(\tilde{a}_{\ell+1}(\ell+1)\frac{1}{2}(1-(-1)^{2\ell+1}) + \sum_{k=\ell+2}^{N}\tilde{a}_k k\frac{1}{2}(1-(-1)^{k+\ell})\right) \\ &= \frac{2}{c_\ell}(\ell+1)\tilde{a}_{\ell+1} + \frac{2}{c_\ell}\sum_{k=\ell+3}^{N}\tilde{a}_k k\frac{1}{2}(1-(-1)^{k+\ell})\end{aligned}$$

(Note that $k = \ell+2$ produces a zero term)

$$= \frac{2}{c_\ell}(\ell+1)\tilde{a}_{\ell+1} + b_{\ell+2}$$

If $\ell = N-1$, then

$$b_{N-1} = \frac{2}{c_{N-1}}\tilde{a}_N N\frac{1}{2}(1-(-1)^{N+N-1}) = 2\tilde{a}_N N\frac{1}{2}2 = 2N\tilde{a}_N$$

Lec. 19 Combining these different results, we get the formulae for b_ℓ to be

$$\begin{aligned} b_N &= 0 \\ b_{N-1} &= 2N\tilde{a}_N && (10) \\ b_\ell &= \frac{1}{c_\ell}(2(\ell+1)\tilde{a}_{\ell+1} + b_{\ell+2}), \qquad \ell = 0, \dots, N-2 && (11) \end{aligned}$$

where

$$c_\ell = \begin{cases} 2 & \ell = 0 \\ 1 & \ell > 0 \end{cases}.$$

This is the Chebyshev recursion relation to express f'_N as a sum of the T_k.

Note that this formulation is equally valid for the spectral approximation $P_N f'(x) = \sum_{k=0}^{N} a_k T'_k(x)$ where the $\tilde{a}_k$ are replaced with a_k.

An important note about parallelization and the recursion relations in (10) is that this type of recursion relation must be computed sequentially and does not easily lend itself to parallel computation.

We can now outline the procedure for computing

$$\frac{d}{dx}\tilde{P}_N f\bigg|_{x=x_k} \approx \frac{d}{dx} f\bigg|_{x=x_k}$$

1. Compute the $\tilde{a}_k$ where
$$\tilde{a}_k = \frac{\pi}{N}\frac{2}{\pi\gamma_k}\sum_{j=0}^{N}\frac{f(x_j)}{\gamma_j}\cos\left(\frac{k\pi j}{N}\right).$$
2. Compute the b_k's using the recursive formula in (10).
3. Evaluate at the collocation points
$$\frac{d}{dx}\tilde{P}_N f\bigg|_{x=x_k} = \sum_{j=0}^{N} b_j \cos\left(\frac{k\pi j}{N}\right)$$

Thus, the pseudo-spectral Chebyshev method reduces to two cosine transforms and a recursion relation.

Alternatively, we can write the differential operator as a matrix. As in the Fourier case, let $F = \begin{bmatrix} f_0 \\ \vdots \\ f_N \end{bmatrix}$ and write $F_x = DF$ where F_x is the vector with entries $\frac{dP_N f}{dx}\Big|_{x=x_k}$ and D is an $(N+1)\times(N+1)$ matrix. D is now called the Chebyshev differentiation matrix.

In order to compute the columns of D, we will apply it to N^{th} degree polynomials p_k which have the property that $p_k(x_j) = \delta_{jk}$. The k^{th} column of D is then

$$\begin{bmatrix} p'_k(x_0) \\ \vdots \\ p'_k(x_N) \end{bmatrix}$$

In fact, we can construct a formula for $p_k(x)$. Recall that $T_N(x)$ has an extremum at each collocation point $x_1, \dots, x_{N-1}$. Therefore, $T'_N(x_j) = 0$ for $j = 1, \dots, N-1$. This means that $(1-x^2)T'_N(x)$ must vanish at $x_0, \dots, x_N$. Finally, we want a polynomial that is non-zero at x_k and so we divide by $(x - x_k)$ to get

$$\frac{(1-x^2)T'_N(x)}{x - x_k}$$

We also need $p_k(x_k) = 1$, so we need to scale this polynomial appropriately. In order to do that we will need some things about $T_N'(x)$. From $T_N(x) = \cos(N\cos^{-1}(x))$, we get

$$\begin{aligned}
T_N'(x) &= \sin(N\cos^{-1}(x))\frac{N}{\sqrt{1-x^2}} \\
T_N''(x) &= \frac{-N^2}{1-x^2}\cos(N\cos^{-1}(x)) - \frac{Nx}{(1-x^2)^{3/2}}\sin(N\cos^{-1}(x))
\end{aligned}$$

From these we get that $T_N'(x_j) = 0$ for $j = 1, \ldots, N-1$ and

$$\begin{aligned}
T_N'(x_N) = T_N'(-1) &= \lim_{x\to -1^+} \frac{N\sin(N\cos^{-1}(x))}{\sqrt{1-x^2}} \\
&= \lim_{x\to -1^+} \frac{N\cos(N\cos^{-1}(x))\frac{-N}{\sqrt{1-x^2}}}{\frac{-x}{\sqrt{1-x^2}}} \\
&= \lim_{x\to -1^+} \frac{N^2}{x}\cos(N\cos^{-1}(x)) \\
&= N^2(-1)^{N+1}
\end{aligned}$$

Likewise,

$$T_N'(x_0) = \lim_{x\to +1^-} \frac{N^2}{x}\cos(N\cos^{-1}(x)) = N^2$$

Lec. 20 Also,

$$\begin{aligned}
T_N'(x_j) &= 0 \\
T_N''(x_j) &= \frac{-N^2}{1-x_j^2}\cos\left(N\frac{\pi j}{N}\right) = \frac{N^2}{1-x_j^2}(-1)^{j+1}, \quad j = 1, \ldots, N-1 \\
T_N''(x_0) &= \frac{1}{3}N^2(N^2-1) \\
T_N''(x_N) &= \frac{1}{3}N^2(1-N^2)(-1)^{N+1} \\
T_N'''(x_j) &= 3\frac{N^2(-1)^{j+1}x_j}{(1-x_j^2)^2}
\end{aligned}$$

Now, assume

$$p_k(x) = \frac{A_k(1-x^2)T_N'(x)}{x-x_k}, \quad \text{for } k = 1,\ldots,N-1$$

Then,

$$\begin{aligned}
p_k(x_k) &= \lim_{x\to x_k} \frac{A_k(1-x^2)T_N'(x)}{x-x_k} \\
&= \lim_{x\to x_k} A_k[(1-x^2)T_N''(x) - 2xT_N'(x)] \\
&= A_k[(1-x_k^2)T_N''(x_k) - 2x_kT_N'(x_k)]
\end{aligned}$$

and we have for $0 < k < N$,

$$\begin{aligned}
1 &= p_k(x_k) \\
&= A_k(1-x_k^2)T_N''(x_k) \\
&= A_k(1-x_k^2)\left(\frac{-N^2}{1-x_k^2}\right)\cos(\pi k) \\
&= A_kN^2(-1)^{k+1}
\end{aligned}$$

and hence $A_k = \frac{(-1)^{k+1}}{N^2}$. For $k = 0$, we have ($x_0 = 1$),

$$\begin{aligned} 1 &= p_0(x_0) \\ &= \frac{A_0(1-x)(1+x)T_N'(x_0)}{x - x_0} \\ &= -A_0(1+x_0)N^2 \\ &= -2A_0N^2 \end{aligned}$$

and hence $A_0 = \frac{(-1)^{0+1}}{2N^2}$. Similarly, $A_N = \frac{(-1)^{N+1}}{2N^2}$ and therefore, we get

$$A_k = \frac{(-1)^{k+1}}{\gamma_k N^2}$$

and hence

$$p_k(x) = A_k \frac{(1-x^2)T_N'(x)}{x - x_k} = \frac{(1-x^2)T_N'(x)(-1)^{k+1}}{\gamma_k N^2(x - x_k)}$$

Next, in order to get the entries of D, we must evaluate $p_k'(x_j) = D_{jk}$. We get for $j \neq k$, $j = 1,\ldots,N-1$,

$$\begin{aligned} p_k'(x_j) &= \left.\frac{[-2xT_n'(x)(-1)^{k+1} + (1-x^2)T_N''(x)(-1)^{k+1}]\gamma_k N^2(x - x_k) - (1-x^2)T_N'(x)(-1)^{k+1}\gamma_k N^2}{\gamma_k^2 N^4(x-x_k)^2}\right|_{x=x_j} \\ &= \frac{(1-x_j^2)T_N''(x_j)(-1)^{k+1}\gamma_k N^2(x_j - x_k)}{\gamma_k^2 N^4 (x_j - x_k)^2} \\ &= \frac{(1-x_j^2)T_N''(x_j)(-1)^{k+1}}{\gamma_k N^2(x_j - x_k)} \\ &= \frac{(1-x_j^2)\frac{N^2}{1-x_j^2}(-1)^{j+1}(-1)^{k+1}}{\gamma_k N^2(x_j - x_k)} \\ &= \frac{(-1)^{j+k}}{\gamma_k(x_j - x_k)} \end{aligned}$$

If $j = 0$, then

$$\begin{aligned} p_k'(x)|_{x=x_0} &= \frac{-2x_0T_n'(x_0)(-1)^{k+1}\gamma_k N^2(x_0 - x_k)}{\gamma_k^2 N^4(x_0 - x_k)^2} \\ &= \frac{-2N^2(-1)^{k+1}}{\gamma_k N^2(x_0 - x_k)} \\ &= \frac{2(-1)^{k+0}}{\gamma_k(x_0 - x_k)} \\ &= \frac{\gamma_0}{\gamma_k}\frac{(-1)^{k+0}}{x_0 - x_k} \end{aligned}$$

Similarly, we get this result for $j = N$, so that if $j \neq k$, we have

$$p_k'(x_j) = \frac{\gamma_j}{\gamma_k}\frac{(-1)^{k+j}}{x_j - x_k}$$

If $j = k$, $k \neq 0$, N, then

$$p_k'(x)|_{x=x_k} = \frac{-x_k}{2(1-x_k^2)}$$

and

$$\begin{aligned}
p_0(x) &= \frac{-(1-x^2)T_N'(x)}{2N^2(x-1)} \\
&= \frac{(1+x)T_N'(x)}{2N^2} \\
p_0'(x)|_{x=x_0} &= \frac{1}{2N^2}(T_N'(x_0) + 2T_N''(x_0)) \\
&= \frac{1}{2N^2}\left(N^2 + 2\frac{1}{3}N^2(N^2-1)\right) \\
&= \frac{1}{2}\left(1 + \frac{2}{3}(N^2-1)\right) \\
&= \frac{1}{2}\left(\frac{1}{3} + \frac{2}{3}N^2\right) \\
&= \frac{1+2N^2}{6}
\end{aligned}$$

$$\begin{aligned}
p_N(x) &= \frac{(1-x^2)T_N'(x)(-1)^{N+1}}{2N^2(x+1)} \\
&= \frac{(1-x)T_N'(x)(-1)^{N+1}}{2N^2} \\
p_N'(x)|_{x=x_N} &= \frac{1}{2N^2}(-T_N'(x_N)(-1)^{N+1} + (1-x_N)T_N''(x_N)(-1)^{N+1}) \\
&= \frac{1}{2N^2}\left(-N^2(-1)^{N+1}(-1)^{N+1} + 2\frac{1}{3}N^2(1-N^2)(-1)^{N+1}(-1)^{N+1}\right) \\
&= \frac{1}{2}\left(-1 + \frac{2}{3}(1-N^2)\right) \\
&= \frac{-(1+2N^2)}{6}
\end{aligned}$$

Putting it all together, we get the matrix entries

$$d_{jk} = \begin{cases} \frac{\gamma_j}{\gamma_k}\frac{(-1)^{j+k}}{x_j - x_k} & j \neq k \\ -\frac{1}{2}\frac{x_k}{1-x_k^2} & j = k, \quad k = 1, \ldots, N-1 \\ \frac{2N^2+1}{6} & j = k = 0 \\ \frac{-(2N^2+1)}{6} & j = k = N \end{cases}$$

Notice that $D^T \neq -D$. In fact, D has terms that are $O(N^2)$. While it is obviously true for d_{00}, d_{NN}, it is also true near the endpoints. If N is large and k is small, then $x_k = \cos\left(\frac{k\pi}{N}\right) \approx 1 - \frac{1}{2}\left(\frac{k\pi}{N}\right)^2$. Thus, $x_k - x_0 \approx 1 - \frac{1}{2}\left(\frac{k\pi}{N}\right)^2 - 1 = O\left(\frac{1}{N^2}\right)$. Thus, $d_{k0} = O(N^2)$. The same is true near the other endpoints as well.

Since the entries of D are $O(N^2)$, then we expect $\|D\| = O(N^2)$. Therefore, it we are solving $u_t = u_x$, using $\frac{d}{dt}U = DU$, then the time steps will be bounded by $O\left(\frac{1}{N^2}\right)$ as opposed to $O\left(\frac{1}{N}\right)$ as in the Fourier and finite difference cases. Lec. 21

To compute higher derivatives, the operator D can be applied repeatedly or one can use the recursion relation. In that case, the recursive operation to compute the b_k's is repeated to get higher derivatives. For example, to compute $\frac{d^2}{dx^2}\tilde{P}_N f$, use the FFT to compute the $\tilde{a}_k$. From this, we compute the b_k's:

$$\begin{aligned} b_N &= 0 \\ b_{N-1} &= 2N\tilde{a}_N \\ b_k &= \frac{1}{\gamma_k}(2(k+1)\tilde{a}_{k+1} + b_{k+2}), \quad k = N-2, \dots, 0 \end{aligned}$$

To get the second derivative, set $\tilde{a}_k = b_k$ and then recompute the b_k's using the same recursive procedure. Finally, the FFT is used to compute $\frac{d^2}{dx^2}\tilde{P}_N f$ at the collocation points.

4.3 Boundary Conditions and Stability Analysis

We now have most of the pieces to solve partial differential equations using the Chebyshev polynomials. We are missing the procedure for incorporating boundary conditions. Let us consider first, the problem

$$\begin{aligned} u_t &= u_x, \quad -1 \le x \le 1 \qquad (12) \\ u(x,0) &= u_0(x) \\ u(1,t) &= g(t) \end{aligned}$$

We assume $g(0) = u_0(1)$. Recall we have the collocation points $x_j = \cos\left(\frac{\pi j}{N}\right)$ and we are trying to compute $u_j(t)$ where $u_j(t)$ is the approximate solution to equation (12) at time t at x_j.

The procedure for solving (12) using Chebyshev pseudo-spectral methods is similar to the Fourier case, except we must impose the boundary conditions at $x_0 = 1$. First we initialize the values u_j via $u_j = u_0(x_j)$. Then we solve

$$\frac{d}{dt}U = DU$$

Finally, we set $u_0(t) = g(t)$.

It is important to note that D is a global operator, so we cannot simply replace one line of the system $\frac{d}{dt}U = DU$ because this would cause errors in the computation of the derivative. Instead, we compute the derivative as we did for the Fourier method and then replace the results with the boundary conditions.

Note that no special procedure is necessary at the boundary as was the case for finite differences. There is no difficulty in computing the derivatives at the boundary.

Next, we will look at the time discretization methods. To do this, we will define a new matrix $\hat{D}$ which has the property that

$$D = \begin{bmatrix} d_{00} & \cdots & d_{0N} \\ \vdots & & \\ d_{N0} & & \hat{D} \end{bmatrix}$$

We therefore have

$$\left.\frac{d}{dx}\tilde{P}_N u\right|_{x=x_j} = \hat{D}\hat{U} + d_{j0}u_0, \quad j = 1, \dots, N$$

Substituting in the boundary conditions dives

$$\left.\frac{d}{dx}\tilde{P}_N u\right|_{x=x_j} = \hat{D}\hat{U} + d_{j0}g(t), \quad j = 1, \dots, N$$

where $\hat{U} = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}$. For homogeneous boundary data, $(g \equiv 0)$, we get

$$\frac{d}{dt}\hat{U} = \hat{D}\hat{U}.$$

Next let's try a simple time-stepping scheme such as forward Euler. In that case, a time step is

$$\frac{1}{\Delta t}(\hat{U}^{n+1} - \hat{U}^{n}) = \hat{D}\hat{U}^{n}$$

Again, we assume $\hat{U}^n = z^n \hat{U}^0$ to find the stable range for Δt. We get

$$\frac{1}{\Delta t}(z-1)\hat{U}^0 = \hat{D}\hat{U}^0$$

and hence we must have that $\hat{U}^0$ is an eigenvector of $\hat{D}$ with eigenvalue λ. Using this, we get

$$\frac{1}{\Delta t}(z-1) = \lambda$$

and $z = 1 + \Delta t \lambda$. The problem is calculating λ.

What can we say about the eigenvalues of $\hat{D}$? We can say that $\hat{D}$ is not skew symmetric and is not normal. Also, we have already seen that it must have eigenvalues that are $O(N^2)$. It is known that the eigenvalues have negative real part.

Let us now do some stability analysis for the Chebyshev method. Consider the equation $u_t = u_x$. We have shown that this results in the pseudo-spectral approximation

$$\frac{d\hat{U}}{dt} = \hat{D}\hat{U}$$

We reduce this to the scalar equation $\frac{du}{dt} = \lambda u$ where λ is an eigenvalue for $\hat{D}$. What we must show for stability is that solving this ordinary differential equation results in only bounded growth. We saw last quarter that this reduces to showing

$$|u^n| \leq Ce^{KT}|u^0|$$

where C, K are supposed to be independent of N, Δt. In practice, K depends on both N, Δt and in order to prevent uncontrolled growth, K is required to be small. For example, if $K = N^2\Delta t$, then the stability requirement is $N^2\Delta t < \epsilon$ for some ϵ. Of course, the longer the method is run, the worse the solutions get (T is increasing).

In order to get absolute stability, i.e. $|u^n| \leq |u^0|$, we must find values of $\lambda\Delta t$ in the complex plane such that this is guaranteed. The set of all values $\lambda\Delta t$ in the complex plane for a given time-stepping method is called the *region of absolute stability.* These regions are well known for the most common time-stepping methods. Of course, for this to apply to the pseudo-spectral methods, $\lambda\Delta t$ must lie in the region of absolute stability for every eigenvalue λ of $\hat{D}$.

Unfortunately, the eigenvalues of $\hat{D}$ are not known explicitly, but can be computed numerically. We do know that they are of order $O(N^2)$.

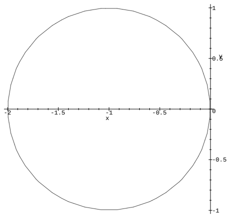
We look now at different time-stepping methods and see what their region of absolute stability looks like. We consider first, Euler's method. For Euler's method we have

$$u^{n+1} = u^n + \Delta t \lambda u^n$$

Plugging in $u^n = z^n u^0$, we see that

$$z = 1 + \Delta t \lambda$$

For what values of $\Delta t\lambda$ do we get $|z| \leq 1$?



Region of absolute stability for Forward Euler

To see this, let $\Delta t\lambda = \alpha + i\beta$, then $1 = |z|^2 = |1+\alpha+i\beta|^2 = (1+\alpha)^2 + \beta^2$ which is a circle of radius 1 centered at -1 in the complex plane.

Recall that for the Fourier pseudo-spectral method, the D operator had all pure-imaginary eigenvalues, and hence $\Delta t\lambda$ will never be in the region of absolute stability.

It is important to note here the eigenvalues of $\hat{D}$ have negative real part and if Δt is taken sufficiently small, the $\Delta t\lambda$ can all be made to lie inside the disk.

Next, let us consider the Runge-Kutta fourth order method. Given the ordinary differential equation $y' = f(y,t)$, the Runge-Kutta fourth order method for one time step is

$$\begin{aligned}
K_1 &= \Delta t f(y_n, t_n) \\
K_2 &= \Delta t f\left(y_n + \frac{1}{2}K_1, t_n + \frac{1}{2}\Delta t\right) \\
K_3 &= \Delta t f\left(y_n + \frac{1}{2}K_2, t_n + \frac{1}{2}\Delta t\right) \\
K_4 &= \Delta t f(y_n + K_3, t_n + \Delta t) \\
y_{n+1} &= y_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)
\end{aligned}$$

To use this scheme, we apply it to the system

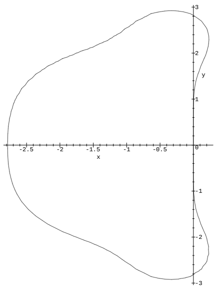
$$\frac{d\hat{U}}{dt} = \hat{D}\hat{U} + d_{j0}g(t)$$

as we saw earlier. Note that the right hand side has a time dependence. This is alright because the Runge-Kutta fourth order method intermediate steps also include the time t to evaluate the right hand side.

We can analyze the region of absolute stability for this method and we get (where $f(y,t) = \lambda y$)

$$\begin{aligned}
K_1 &= \Delta t\lambda z^n y_0 \\
K_2 &= \Delta t\lambda\left(z^n y_0 + \frac{1}{2}\Delta t\lambda z^n y_0\right) \\
K_3 &= \Delta t\lambda\left(z^n y_0 + \frac{1}{2}\Delta t\lambda\left(z^n y_0 + \frac{1}{2}\Delta t\lambda z^n y_0\right)\right) \\
K_4 &= \Delta t\lambda\left(z^n y_0 + \Delta t\lambda\left(z^n y_0 + \frac{1}{2}\Delta t\lambda\left(z^n y_0 + \frac{1}{2}\Delta t\lambda z^n y_0\right)\right)\right) \\
z^{n+1}y_0 &= \left(1 + \Delta t\lambda + \frac{1}{2}\Delta t^2\lambda^2 + \frac{1}{6}\Delta t^3\lambda^3 + \frac{1}{24}\Delta t^4\lambda^4\right)
\end{aligned}$$

and the plot of the region of stability can be seen below:



Region of absolute stability for Runge-Kutta 4

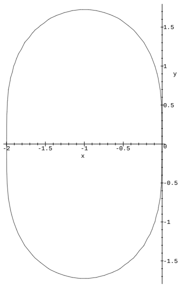
We find that we get absolute stability for $\Delta t \leq \frac{K}{N^2}$ where $K \approx 30$ for Chebyshev methods.

There are alternative forms of the Runge-Kutta methods which require less storage, however, the time value at each stage is ambiguous causing problems with time-dependent boundary conditions. Also, for non-constant coefficient problems, these low-storage variants reduce to second order accurate.

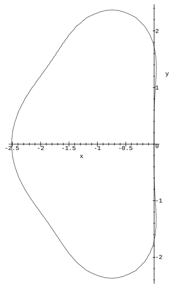
The second order Runge-Kutta method is

$$\begin{aligned}
K_1 &= \Delta t f(y_n, t_n) \\
K_2 &= \Delta t f(y_n + K_1, t_n + \Delta t) \\
y_{n+1} &= y_n + \frac{1}{2}(K_1 + K_2)
\end{aligned}$$

This method is similar to Euler's method in that the region of absolute stability does not include the imaginary axis. This means that this method is absolutely stable only for the Chebyshev method and not for the Fourier methods. For the Chebyshev method stability is achieved for $\Delta t \leq \frac{K}{N^2}$ where $K \approx 16$. The region of absolute stability for Runge-Kutta 2 and 3 are below:



Region of absolute stability for Runge-Kutta 2



Region of absolute stability for Runge-Kutta 3

In general, how does one impose the boundary conditions using a Runge-Kutta scheme? Consider again the system

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

At each stage of the Runge-Kutta process you can either (A) use the equation

$$\frac{d\hat{U}}{dt} = \hat{D}\hat{U} + d_{j0}g(t)$$

which imposes the boundary conditions at each stage, or (B) Use the equation

$$\frac{dU}{dt} = DU$$

at each stage, then impose $u(1,t) = g(t)$ at the end.

In practice, method A is better if you are using the full fourth-order method. It allows a larger time step and is best when the boundary data is not time-dependent. Method B requires a smaller time step for stability, but experiments have shown that this method is more accurate when the boundary data is time dependent.

4.4 Adams-Bashforth Methods

Another class of explicit methods are based upon a multi-step approach similar to Leap Frog. In this case, more than one previous time level is used to advance to the new time level. These methods require a separate start-up procedure.

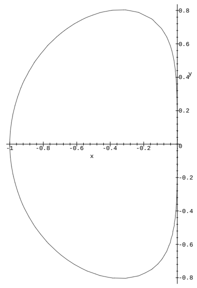
For the ordinary differential equation $y' = f(y,t)$, the Adams-Bashforth second order method is

$$y_{n+1} = y_n + \Delta t \left[\frac{3}{2} f(y_n, t_n) - \frac{1}{2} f(y_{n-1}, t_{n-1}) \right]$$

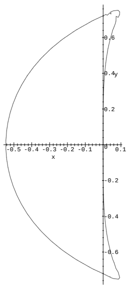
The third order method is

$$y_{n+1} = y_n + \Delta t \left[\frac{23}{12} f(y_n, t_n) - \frac{16}{12} f(y_{n-1}, t_{n-1}) + \frac{5}{12} f(y_{n-2}, t_{n-2}) \right]$$

Again, this method can be made absolutely stable for $\Delta t \leq \frac{K}{N^2}$ where $K \approx 9$ for the second order method and decreasing for higher order methods. The regions of absolute stability for the second and third order methods are shown below:



Region of absolute stability for Adams-Bashforth 2



Region of absolute stability for Adams-Bashforth 3

4.5 Adams-Moulton Methods

A companion set of methods called the Adams-Moulton are multi-step implicit methods and are unconditionally stable. We have seen the first and second order methods already, they are backward Euler and Crank-Nicolson respectively. The third order method is

$$y_{n+1} = y_n + \frac{1}{12}\Delta t(5f(y_{n+1}, t_{n+1}) + 8f(y_n, t_n) - f(y_{n-1}, t_{n-1}))$$

In our context, we then must solve

$$\left(I - \frac{5}{12}\Delta t\hat{D}\right)\hat{U}^{n+1} = \left(I + \frac{8}{12}\Delta t\hat{D}\right)\hat{U}^n - \frac{1}{12}\Delta t\hat{D}\hat{U}^{n-1}$$

It is common practice to turn this implicit method into an explicit method by turning it into a predictor-corrector. A predictor is the Adams-Bashforth method of one lower order as the Adams-Moulton method. Then the Adams-Moulton step is used as a corrector. In other words, a third order predictor-corrector pair can be written as

$$\begin{aligned}\hat{y}_{n+1} &= y_n + \frac{1}{2}\Delta t(3f(y_n, t_n) - f(y_{n-1}, t_{n-1}))\\ y_{n+1} &= y_n + \frac{1}{12}\Delta t(5f(\hat{y}_{n+1}, t_{n+1}) + 8f(y_n, t_n) - f(y_{n-1}, t_{n-1}))\end{aligned}$$

Of course we lose the unconditional stability of the fully implicit method, but the resulting method is still more stable and more accurate than the Adams-Bashforth method alone.

4.6 Implicit Methods

Two common implicit methods are Backward Euler and Crank-Nicolson. For the equation

$$\frac{d\hat{U}}{dt} = \hat{D}\hat{U}$$

we have

$$\begin{aligned} \hat{U}^{n+1} &= \hat{U}^n + \Delta t \hat{D}\hat{U}^{n+1} \\ \hat{U}^{n+1} - \hat{U}^n &= \frac{1}{2}\Delta t(\hat{D}\hat{U}^{n+1} + \hat{D}\hat{U}^n) \end{aligned}$$

respectively. Both of these methods can be rewritten in a δ-formulation as

$$\begin{aligned} \hat{U}^{n+1} - \hat{U}^n &= \Delta t\hat{D}(\hat{U}^{n+1} - \hat{U}^n + \hat{U}^n) \\ \left(I - \frac{1}{2}\Delta t\hat{D}\right)\delta^n &= \Delta t\hat{D}\hat{U}^n \end{aligned}$$

The update step is then $\hat{U}^{n+1} = \hat{U}^n + \delta^n$.

Backward Euler is first order accurate and Crank-Nicolson is second-order accurate. Both methods are unconditionally stable.

In summary, a fully explicit method will have a stability limit of $\Delta t = O(N^{-2})$. The Runge-Kutta methods are generally the best. For the solutions with large spatial variations, the largest problem is the oscillations of the high modes, but this is unavoidable due to the Gibbs phenomenon. For semi-implicit methods, Adams-Bashforth is a good choice for the explicit part. The implicit methods are good for larger time steps. Use Crank-Nicolson for time-accurate solutions and Backward Euler for steady state solutions.

Lec. 22

4.7 Parabolic Partial differential equations

We next consider parabolic problems like the heat equation

$$\begin{aligned} u_t &= u_{xx}, \qquad -1 \le x \le 1 \\ u(x,0) &= u_0(x) \end{aligned}$$

Again, we need boundary conditions in order to make the problem have a unique solution. Two common boundary conditions are Dirichlet boundary conditions, $u(\pm 1, t) = g_\pm(t)$, and Neuman boundary conditions, $u_x(\pm 1, t) = g_\pm(t)$. We will first consider the homogeneous Dirichlet condition $u(\pm 1, t) = 0$.

As in the hyperbolic case, we will trim the vector $U = \begin{bmatrix} u_0 \\ \vdots \\ u_N \end{bmatrix}$ down to $\hat{U} = \begin{bmatrix} u_1 \\ \vdots \\ u_{N-1} \end{bmatrix}$ where here we must impose boundary conditions at each endpoint. As in the hyperbolic case, we then have some left over terms

$$\frac{\partial^2}{\partial x^2}U \approx \hat{D}_2\hat{U} + \hat{\delta}_0 u_0 + \hat{\delta}_N u_N$$

where $\hat{D}_2$, $\hat{\delta}_0$, and $\hat{\delta}_N$ are given by

$$D^2 = \begin{bmatrix} d_{00} & \cdots & d_{0N} \\ \vdots & \hat{D}_2 & \vdots \\ d_{N0} & \cdots & d_{NN} \end{bmatrix} \qquad \hat{\delta}_0 = \begin{bmatrix} d_{1,0} \\ \vdots \\ d_{N-1,0} \end{bmatrix} \qquad \hat{\delta}_N = \begin{bmatrix} d_{1,N} \\ \vdots \\ d_{N-1,N} \end{bmatrix}$$

The stability of any time stepping routine is based upon the eigenvalues of $\hat{D}_2$ which must be calculated numerically.

In fact, the eigenvalues of $\hat{D}_2$ are negative (good) and grow as $O(N^4)$ (bad). As in the hyperbolic case, the rapid growth of the eigenvalues is related to the clustering of the nodes. This means that time steps for explicit methods will be restricted by $\Delta t = O(N^{-4})$. This is very bad and makes explicit time stepping impractical. For this reason, we look at implicit methods.

The two most common methods are backward Euler and Crank-Nicolson. Both methods are unconditionally stable

$$\frac{1}{\Delta t}(\hat{U}^{n+1} - \hat{U}^n) = \hat{D}_2 \hat{U}^{n+1} + \hat{\delta}_0 u_0^{n+1} + \hat{\delta}_N u_N^{n+1}$$
$$\frac{1}{\Delta t}(\hat{U}^{n+1} - \hat{U}^n) = \frac{1}{2}\hat{D}_2(\hat{U}^{n+1} + \hat{U}^n) + \frac{1}{2}\hat{\delta}_0(u_0^{n+1} + u_0^n) + \frac{1}{2}\hat{\delta}_N(u_N^{n+1} + u_N^n)$$

Crank-Nicolson is second order and backward Euler damps higher frequency modes more effectively. These two methods can be combined by

$$\frac{1}{\Delta t}(\hat{U}^{n+1} - \hat{U}^n) = \hat{D}_2(\theta \hat{U}^{n+1} + (1-\theta)\hat{U}^n) + \hat{\delta}_0(\theta u_0^{n+1} + (1-\theta)u_0^n) + \hat{\delta}_N(\theta u_N^{n+1} + (1-\theta)u_N^n)$$

This method is unconditionally stable for any $\frac{1}{2} \leq \theta \leq 1$. A typical value for θ is $\theta = \frac{1}{2} + \alpha \Delta t$. This keeps
Lec. 23 the method second order accurate while improving the damping of high frequency modes.

Next, we will discuss Neumann boundary conditions, $u_x(\pm 1, t) = h_{\pm}(t)$. In this case, we first apply D to U to get

$$\frac{\partial}{\partial x} U \approx DU$$

We now impose the boundary conditions followed by a second application of D to get an approximation for u_{xx}. In matrix form, let

$$D = \begin{bmatrix} d_{0,0} & \cdots & d_{0,N} \\ \vdots & & \vdots \\ d_{1,0} & \cdots & d_{N,N} \end{bmatrix}$$

and then define

$$\hat{D} = \begin{bmatrix} 0 & \cdots & 0 \\ d_{1,0} & \cdots & d_{1,N} \\ \vdots & & \vdots \\ d_{N-1,N} & \cdots & d_{N-1,N} \\ 0 & \cdots & 0 \end{bmatrix}, \qquad H^n = \begin{bmatrix} h_+(t_n) \\ 0 \\ \vdots \\ 0 \\ h_-(t_n) \end{bmatrix}$$

We then have

$$u_{xx} \approx D[\hat{D}U + H]$$

and hence, the Crank-Nicolson method becomes

$$\frac{1}{\Delta t}(U^{n+1} - U^n) = \frac{1}{2} D[\hat{D}(U^{n+1} + U^n) + (H^{n+1} + H^n)]$$

Again, this method can be put into a $\delta-$formulation.

Finally, we can impose mixed boundary conditions such as $u_x + \alpha u = g(t)$ in a similar way.

Note that we are using boundary conditions imposed with spectral accuracy. This is in contrast to finite difference methods where lower order accurate derivatives are needed at the boundary to impose the boundary conditions.

4.8 Hyperbolic Systems

We next look at systems of hyperbolic equations. Consider the system

$$u_t = Au_x$$

where u is an $m-$vector and A is an $m \times m$ matrix. Recall that for this to be a hyperbolic system, A must have all real distinct eigenvalues. If all the eigenvalues of A are the same sign, then we solve this system in the same way as the scalar case.

We will now consider the case where A has both positive and negative eigenvalues in which case we have inflow/outflow boundary conditions. For example, consider the first order system

$$\begin{bmatrix} u \\ p \end{bmatrix}_t = \begin{bmatrix} 0 & \frac{-1}{\rho} \\ -\rho c^2 & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}_x$$
$$p(1) = p(-1) = 0$$

In order to implement the boundary conditions, we must first find the eigenvalues and eigenvectors of A.

$$0 = \det \begin{bmatrix} -\lambda & \frac{-1}{\rho} \\ -\rho c^2 & -\lambda \end{bmatrix} = \lambda^2 - c^2 = (\lambda - c)(\lambda + c)$$

$$\lambda = c, \quad \begin{bmatrix} 1 \\ -\rho c \end{bmatrix} \qquad \lambda = -c, \quad \begin{bmatrix} 1 \\ \rho c \end{bmatrix}$$

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

and we then get

$$\begin{bmatrix} \frac{1}{2} & \frac{-1}{2\rho c} \\ \frac{1}{2} & \frac{1}{2\rho c} \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}_t = \begin{bmatrix} c & 0 \\ 0 & -c \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{-1}{2\rho c} \\ \frac{1}{2} & \frac{1}{2\rho c} \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}_x$$

Multiply this by $2\rho c$ and we get

$$\begin{bmatrix} \rho c & -1 \\ \rho c & 1 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}_t = \begin{bmatrix} c & 0 \\ 0 & -c \end{bmatrix} \begin{bmatrix} \rho c & -1 \\ \rho c & 1 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}_x$$

The characteristic variables are thus $w_1 = \rho c u - p$ and $w_2 = \rho c u + p$ with speeds c and $-c$ respectively.

Now we can write a complete algorithm. Let $U^n = \begin{bmatrix} u_0^n \\ \vdots \\ u_N^n \end{bmatrix}$, $P^n = \begin{bmatrix} p_0^n \\ \vdots \\ p_N^n \end{bmatrix}$, then the Runge-Kutta second order method becomes

1.
$$\hat{U} = U^n - \frac{1}{2}\Delta t \frac{1}{\rho} D P^n$$
$$\hat{P} = P^n - \frac{1}{2}\Delta t \rho c D U^n$$

2.
$$\hat{p}_N = 0$$
$$\hat{p}_0 = 0$$

3.

$$
\begin{aligned}
U^{n+1} &= U^n - \Delta t \frac{1}{\rho} D\hat{P} \\
P^{n+1} &= P^n - \Delta t \rho c D\hat{U}
\end{aligned}
$$

4.

$$
\begin{aligned}
p_N^{n+1} &= 0 \\
p_0^{n+1} &= 0
\end{aligned}
$$

Unfortunately, this does not work because we are not respecting the flow of information. While finite difference methods often have built-in dissipation keeping the method stable, spectral methods typically do not.

To make a stable method, we need to look at the characteristic variables. The variable $w_1 = \rho cu - p$ has speed c, and hence travels right to left, while $w_2 = \rho cu + p$ travels left to right. At the right endpoint, we want w_2 in terms of w_1, i.e.

$$
\begin{aligned}
w_2 &= \alpha w_1 \\
\rho cu + p &= \alpha(\rho cu - p)
\end{aligned}
$$

and we are given that $p = 0$, hence we must have $\alpha = 1$ and we get two equations to solve at the boundary:

$$
\begin{aligned}
\hat{p}_0 &= 0 \\
\rho c\hat{u}_0 + \hat{p}_0 &= \rho c\hat{u}_0 - \hat{p}_0
\end{aligned}
$$

where the right hand side of the above equation comes from the output of the interior method. In other words, the algorithm becomes

1.

$$
\begin{aligned}
\hat{U} &= U^n - \frac{1}{2}\Delta t \frac{1}{\rho} DP^n \\
\hat{P} &= P^n - \frac{1}{2}\Delta t \rho c^2 DU^n
\end{aligned}
$$

2.

$$
\begin{aligned}
\hat{\hat{u}}_j &= \hat{u}_j,\ j = 1,\dots,N-1 \\
\hat{\hat{p}}_j &= \hat{p}_j,\ j = 1,\dots,N-1 \\
\hat{\hat{p}}_0 &= 0 \\
\hat{\hat{p}}_N &= 0 \\
\rho c\hat{\hat{u}}_0 + \hat{\hat{p}}_0 &= \rho c\hat{u}_0 - \hat{p}_0 \\
\rho c\hat{\hat{u}}_N - \hat{\hat{p}}_N &= \rho c\hat{u}_N + \hat{p}_N
\end{aligned}
$$

where the last four equations must be solved simultaneously if necessary.

3.

$$
\begin{aligned}
\tilde{U} &= U^n - \Delta t \frac{1}{\rho} D\hat{\hat{P}}^n \\
\tilde{P} &= P^n - \Delta t \rho c^2 D\hat{\hat{U}}^n
\end{aligned}
$$

4.

$$\begin{aligned}
u_j^{n+1} &= \tilde{u}_j,\ j = 1,\ldots,N-1\\
p_j^{n+1} &= \tilde{p}_j,\ j = 1,\ldots,N-1\\
p_0^{n+1} &= 0\\
p_N^{n+1} &= 0\\
\rho c u_0^{n+1} + p_0^{n+1} &= \rho c \tilde{u}_0 - \tilde{p}_0\\
\rho c u_N^{n+1} - p_N^{n+1} &= \rho c \tilde{u}_N + \tilde{p}_N
\end{aligned}$$

where again, the last four equations must be solved simultaneously if necessary.

4.9 Chebyshev Tau Method

Recall that for the Fourier case, we derived different methods, the pseudo-spectral method and the Galerkin method. In the Galerkin method, we insisted that if we are solving $u_t = Su$, then the residual

$$\frac{\partial u_N}{\partial t} - Su_N$$

must be orthogonal to each of the Fourier modes. This resulted in evolution equations for the Fourier coefficients themselves. The method had the advantage of no aliasing, but often involved convolutions.

The equivalent method for initial boundary value problems is complicated by the fact that the T_k do not necessarily satisfy the boundary conditions (unlike the Galerkin method). Consider the initial boundary value problem

$$\begin{aligned}
u_t &= u_x, \qquad -1 \le x \le 1\\
u(x,0) &= g(x)\\
u(1,t) &= h(t)
\end{aligned}$$

Ordinarily, we would expand $u \approx \sum_{j=0}^{N} a_j T_j(x)$. However, there is nothing to enforce the boundary conditions.

In order to enforce the boundary condition, we change the expansion to become

$$u = u_{N+1} \approx \sum_{j=0}^{N} a_j T_j(x) + a_{N+1} T_{N+1}(x)$$

where the extra a_{N+1} is used to enforce the boundary condition. As in the Galerkin method, we require that the residual be orthogonal to each of the $T_0,\ldots, T_N$, i.e.

$$\left\langle T_k, \frac{\partial u_{N+1}}{\partial t} - \frac{\partial u_{N+1}}{\partial x} \right\rangle_w = 0, \qquad k = 0,\ldots,N$$

Recall that we have $\langle T_k, T_\ell\rangle_w = \frac{c_k\pi}{2}\delta_{k\ell}$. Thus, we have

$$\begin{aligned}
\left\langle T_k, \frac{\partial u_{N+1}}{\partial t}\right\rangle_w &= \left\langle T_k, \sum_{j=0}^{N+1} \frac{da_j}{dt} T_j(x)\right\rangle_w\\
&= \sum_{j=0}^{N+1} \frac{da_j}{dt}\langle T_k, T_j\rangle_w\\
&= \sum_{j=0}^{N+1} \frac{da_j}{dt}\frac{c_k\pi}{2}\delta_{kj}\\
&= \frac{c_k\pi}{2}\frac{da_k}{dt}
\end{aligned}$$

and

$$\left\langle T_k, \frac{\partial u_{N+1}}{\partial x} \right\rangle_w = \sum_{j=0}^{N+1} a_j \langle T_k, T_j' \rangle_w$$
$$= \sum_{j=k+1}^{N+1} a_j \pi j \frac{1}{2}(1 - (-1)^{j+k})$$

Therefore, for $k = 0, \ldots, N$, we have

$$\frac{da_k}{dt} = \frac{2}{\pi c_k} \pi \frac{1}{2} \sum_{j=k+1}^{N+1} a_j j (1 - (-1)^{j+k})$$
$$= \frac{1}{c_k} \sum_{j=k+1}^{N+1} j a_j (1 - (-1)^{j+k})$$

and the boundary conditions equation

$$a_{N+1} = \frac{1}{T_{N+1}(1)} \left(h(t) - \sum_{j=0}^{N} a_j T_j(1) \right)$$

Using the fact that $T_j(1) = 1$, this becomes

$$a_{N+1} = h(t) - \sum_{j=0}^{N} a_j$$

Plugging this back into the first equation, we get the evolution equation for the a_k's:

$$\frac{da_k}{dt} = \frac{1}{c_k} \sum_{j=k+1}^{N} j a_j (1 - (-1)^{j+k}) + \frac{1}{c_k}(N+1) \left(h(t) - \sum_{j=0}^{N} a_j \right) (1 - (-1)^{N+k+1})$$

and the reconstruction is

$$u \approx \sum_{k=0}^{N} a_k T_k(x) + \left(h(t) - \sum_{k=0}^{N} a_k \right) T_{N+1}(x)$$
$$= h(t) + \sum_{k=0}^{N} a_k (T_k(x) - T_{N+1}(x))$$

Note that more than one boundary condition would result in additional expansion terms:

$$u \approx \sum_{k=0}^{N} a_k T_k(x) + \sum_{k=1}^{r} a_{N+k} T_{N+k}(x)$$

where the first term is the Galerkin type approximation and the second term is used to enforce the r boundary conditions.

5 Wavelets

Wavelets are collections of basis functions that make up a *multi-resolution analysis*, basis functions which can operate essentially at all wavelengths, and can be scaled in length hierarchically to produce varying levels of detail for a given approximation. We present here a very basic introduction to the subject.

5.1 Scaling functions

The key to wavelets is the idea of a *multi-resolution analysis* generated by a scaling function, ϕ. A multi-resolution analysis is a nested sequence of function spaces V_j such that

$$0 \subset \cdots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots \subset L^2(\mathbb{R}).$$

In each of these function spaces, we will assume the standard inner product of $L^2(\mathbb{R})$:

$$\langle u, v \rangle = \int_{-\infty}^{\infty} u(x)v(x)\, dx.$$

Let $\phi \in L^2(\mathbb{R})$ be a function such that $\langle \phi, \phi \rangle = 1$, then define V_j to be the space spanned by the set of functions $\{\phi_{j,k}\}_{k\in\mathbb{Z}}$ where

$$\phi_{j,k}(x) = 2^{j/2}\phi(2^j x - k).$$

We then have

$$\begin{aligned}\langle \phi_{j,k}, \phi_{j,\ell} \rangle &= \int_{-\infty}^{\infty} \left(2^{j/2}\phi(2^j x - k)\right)\left(2^{j/2}\phi(2^j x - \ell)\right)\, dx \\ &= 2^j \int_{-\infty}^{\infty} \phi(2^j x - k)\phi(2^j x - \ell)\, dx\end{aligned}$$

If we substitute $y = 2^j x - k$, then $dy = 2^j dx$ and we have

$$= \int_{-\infty}^{\infty} \phi(y)\phi(y - (\ell - k))\, dy.$$

Now, if ϕ is such that the support of ϕ is confined to an interval of length one, then

$$\int_{-\infty}^{\infty} \phi(y)\phi(y - (\ell - k))\, dy = \begin{cases} 0 & k \neq \ell \\ 1 & k = \ell \end{cases}.$$

Therefore, the set $\{\phi_{j,k}\}_{k\in\mathbb{Z}}$ form an orthonormal basis for the space V_j.

Example 5.1:

Let

$$\phi(x) = \begin{cases} 1 & 0 \leq x < 1 \\ 0 & \text{otherwise} \end{cases}.$$

Then the basis functions can be seen as various square hat functions as illustrated below:

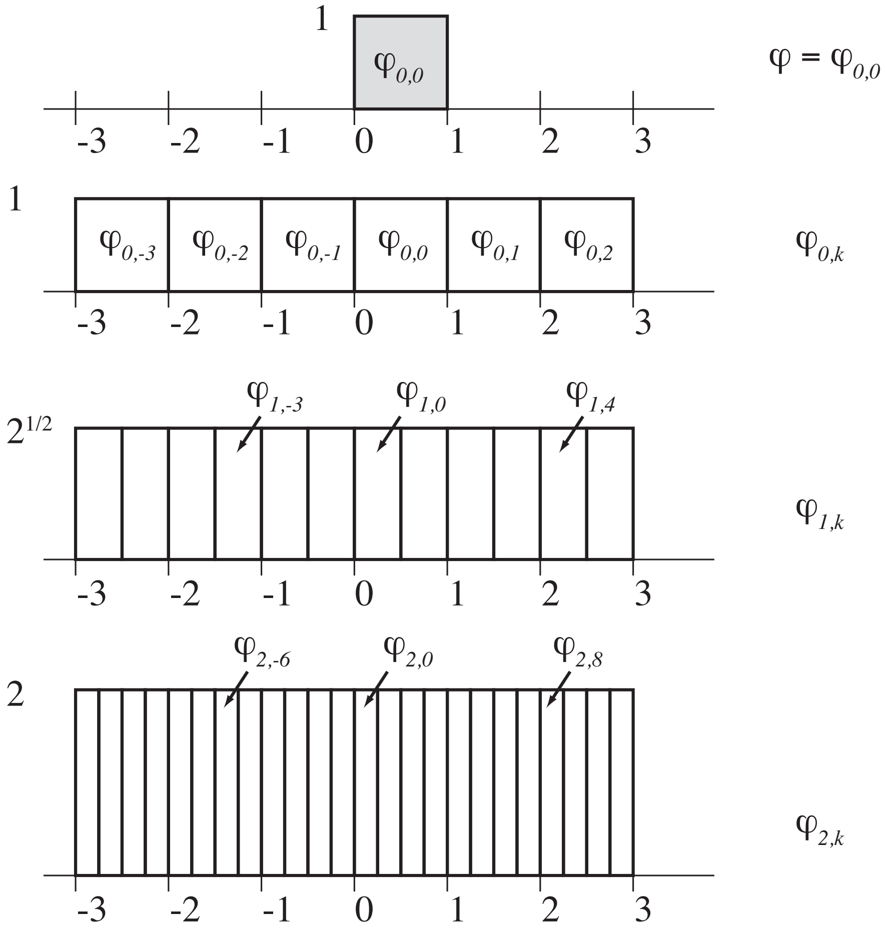


Illustration of nested wavelet basis functions for V_0, V_1, and V_2

Next, define a set of coefficients, h_n, which solve the *refinement equation*:

$$\phi(x) = 2\sum_n h_n \phi(2x-n) = \sqrt{2}\sum_n h_n \phi_{1,n}(x). \tag{13}$$

This equation is also called a *dilation equation* and represents a connection between the nested spaces. More generally, we get that

$$\phi_{j,k}(x) = \sqrt{2}\sum_n h_n \phi_{j+1,n+2k}(x).$$

To see this, we start by taking the inner product of (13) with $\phi_{1,m}(x)$ to get

$$\langle \phi_{0,0}, \phi_{1,m}\rangle = \sqrt{2}\sum_n h_n \langle \phi_{1,n}, \phi_{1,m}\rangle = \sqrt{2}h_m.$$

Thus, we get a formula for the h_n:

$$h_n = \frac{1}{\sqrt{2}}\langle \phi_{0,0}, \phi_{1,n}\rangle.$$

Next, note that

$$\langle \phi_{j,k}, \phi_{j+1,n+2k}\rangle = \int_{-\infty}^{\infty} 2^{j/2}\phi(2^j x - k)2^{(j+1)/2}\phi(2^{j+1}x - (n+2k))\,dx$$

Using the same substitution as before, $y = 2^j x - k$, then $2^{j+1}x - (n + 2k) = 2y - n$, and $dy = 2^j dx$,

$$\begin{aligned} &= \int_{-\infty}^{\infty} \phi(y)\sqrt{2}\phi(2y - n)\, dy \\ &= \langle \phi_{0,0}, \phi_{1,n} \rangle. \end{aligned}$$

Now suppose

$$\phi_{i,j}(x) = \sum_n \alpha_n \phi_{j+1,n+2k}(x),$$

for some coefficients, α_n. Taking the inner product of this equation with $\phi_{j+1,m+2k}$, we get

$$\langle \phi_{j,k}, \phi_{j+1,m+2k} \rangle = \sum_n \alpha_n \langle \phi_{j+1,n+2k}, \phi_{j+1,n+2k} \rangle = \alpha_m.$$

Therefore,

$$\alpha_m = \langle \phi_{j,k}, \phi_{j+1,m+2k} \rangle = \langle \phi_{0,0}, \phi_{1,n} \rangle = \sqrt{2}h_m$$

which is what we were trying to show.

Example 5.2:

For the basis functions generated by $\phi(x) = \begin{cases} 1 & 0 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$, we can compute the coefficients h_n:

$$\sqrt{2}h_0 = \langle \phi_{0,0}, \phi_{1,0} \rangle = \int_0^1 \sqrt{2}\phi(2x)\, dx = \int_0^{1/2} \sqrt{2}\, dx = \frac{1}{\sqrt{2}}.$$

Similarly,

$$\begin{aligned} \sqrt{2}h_1 &= \langle \phi_{0,0}, \phi_{1,1} \rangle = \int_0^1 \sqrt{2}\phi(2x - 1)\, dx = \int_{1/2}^1 \sqrt{2}\, dx = \frac{1}{\sqrt{2}}, \\ \sqrt{2}h_{-1} &= \langle \phi_{0,0}, \phi_{1,-1} \rangle = \int_0^1 \sqrt{2}\phi(2x + 1)\, dx = 0. \end{aligned}$$

So for this particular choice of wavelet generator, the corresponding coefficients are

$$h_n = \begin{cases} \frac{1}{2} & n = 0, 1 \\ 0 & \text{otherwise} \end{cases}.$$

Referring back to the figure in the previous example, it is clear that the nesting of the basis functions would lead to this conclusion.

Before continuing, we should note a few properties of the coefficients, h_n. The previous example illustrates one important result, namely

$$\sum_n h_n \equiv 1.$$

Furthermore, we have

$$\begin{aligned} \delta_{j-k} &= \langle \phi_{0,j}, \phi_{0,k} \rangle \\ &= \left\langle \sqrt{2}\sum_m h_m \phi_{1,m+2j}, \sqrt{2}\sum_n h_n \phi_{1,n+2k} \right\rangle, \\ &= 2\sum_m \sum_n h_m h_n \langle \phi_{1,m+2j}, \phi_{1,n+2k} \rangle, \\ &= 2\sum_n h_n h_{n+2(k-j)}. \end{aligned}$$

We say that *compactly supported* in $[k, \ell]$ if $h_n = 0$ for all $n < k$ and $n > \ell$.

5.2 The Orthogonal Complement

Given the coefficients h_n given from the scaling function, $\phi(x)$, we next define a new wavelet function, $\psi(x)$ given by

$$\psi(x) = 2\sum_n (-1)^n h_{1-n}\phi(2x-n) = \sqrt{2}\sum_n (-1)^n h_{1-n}\phi_{1,n}(x).$$

Starting with this, we can define a new set of basis functions, $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$, where

$$\psi_{j,k} = 2^{j/2}\psi(2^j x - k).$$

This basis is also orthonormal. For orthogonality, we have

$$\begin{aligned}
\langle \psi_{j,k}, \psi_{j,\ell}\rangle &= \langle 2^{j/2}\psi(2^j x - k), 2^{j/2}\psi(2^j x - \ell)\rangle,\\
&= \left\langle 2^{j/2}\cdot 2\sum_m (-1)^m h_{1-m}\phi(2(2^j x - k) - m), 2^{j/2}\cdot 2\sum_n (-1)^n h_{1-n}\phi(2(2^j x - \ell) - n)\right\rangle\\
&= \left\langle 2^{1/2}\sum_m (-1)^m h_{1-m} 2^{(j+1)/2}\phi(2^{j+1}x - (m+2k)), 2^{1/2}\sum_n (-1)^n h_{1-n} 2^{(j+1)/2}\phi(2^{j+1}x - (n+2\ell))\right\rangle,\\
&= 2\sum_m\sum_n (-1)^{m+n} h_{1-m}h_{1-n}\langle \phi_{j+1,m+2k}, \phi_{j+1,n+2\ell}\rangle,\\
&= 2\sum_n (-1)^{n+(n+2\ell-2k)} h_{1-n}h_{1-(n+2\ell-2k)},\\
&= 2\sum_n h_{1-n}h_{1-n+2(k-\ell)},\\
&= 2\sum_n h_n h_{n+2(k-\ell)} = \delta_{k-\ell}.
\end{aligned}$$

Therefore, the $\psi_{j,k}$ form an orthonormal set of functions.

Furthermore, we have

$$\begin{aligned}
\langle \psi_{j,k}, \phi_{j,\ell}\rangle &= \langle 2^{j/2}\psi(2^j x - k), \phi_{j,\ell}\rangle,\\
&= \left\langle \sqrt{2}\sum_m (-1)^m h_{1-m}\phi_{j+1,m+2k}, \sqrt{2}\sum_n h_n\phi_{j+1,n+2\ell}\right\rangle,\\
&= 2\sum_m\sum_n (-1)^m h_{1-m}h_n\langle \phi_{j+1,m+2k}, \phi_{j+1,n+2\ell}\rangle,\\
&= 2\sum_n (-1)^n h_{1-n+2(k-\ell)}h_n,\\
&= 2\sum_{n=1+k-\ell}^{\infty} (-1)^n h_{1-n+2k-2\ell}h_n + 2\sum_{n=k-\ell}^{-\infty} (-1)^n h_{1-n+2k-2\ell}\\
&= 2\sum_{n=0}^{\infty} (-1)^{n+1+k-\ell} h_{1-n-1-(k-\ell)+2(k-\ell)}h_{n+1+k-\ell} + 2\sum_{n=-k+\ell}^{\infty} (-1)^n h_{1+n+2k-2\ell}h_{-n}\\
&= 2\sum_{n=0}^{\infty} (-1)^{n+k-\ell+1} h_{-n+k-\ell}h_{n+1+k-\ell} + 2\sum_{n=0}^{\infty} (-1)^{n-k+\ell} h_{1+n+k-\ell}h_{-n+k-\ell}\\
&= 0.
\end{aligned}$$

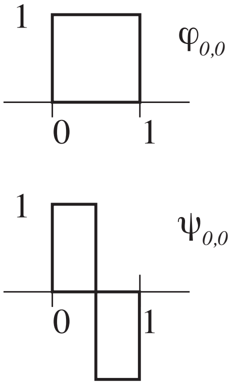
Thus, $\{\psi_{j,k}\}_{k\in\mathbb{Z}}$ and $\{\phi_{j,k}\}_{k\in\mathbb{Z}}$ form an othogonal complement of basis functions for V_{j+1}.

Example 5.3:

Recall that for $\phi(x) = \begin{cases} 1 & 0 \le x < 1 \\ 0 & \text{otherwise} \end{cases}$, we found that $h_0 = h_1 = \frac{1}{2}$, and $h_n = 0$ for $n \neq 0, 1$. We can then construct $\psi(x)$ by

$$\begin{aligned} \psi(x) &= 2\sum_n (-1)^n h_{1-n}\phi(2x-n) \\ &= 2\left(\frac{1}{2}\phi(2x) - \frac{1}{2}\phi(2x-1)\right) \\ &= \phi(2x) - \phi(2x-1) \\ &= \begin{cases} 1 & 0 \le x < 1/2 \\ -1 & 1/2 \le x < 1 \\ 0 & \text{otherwise} \end{cases}. \end{aligned}$$

The resulting scaling and wavelet functions are shown below



Haar scaling ($\phi_{0,0}$) and wavelet ($\psi_{0,0}$) functions.

It is now easy to see how the wavelet functions form the complementary basis that takes V_j into V_{j+1}.

To summarize, the ϕ function is called the *father wavelet*, or the *scaling function*, and the ψ function is called the *mother wavelet*, or the *detail function*. The combination of the scaling and detail functions are what allow us to construct a multi-resolution analysis.

One final note about these functions. In higher dimensions, the wavelets are constructed by using tensor products. For example, in two dimensions, there is one scaling function given by

$$\phi(x, y) = \phi(x)\phi(y),$$

where $\phi(x)$ is the corresponding one-dimensional scaling function. However, there are now three wavelet functions:

$$\begin{aligned} \psi_1(x, y) &= \phi(x)\psi(y) \\ \psi_2(x, y) &= \psi(x)\phi(y) \\ \psi_3(x, y) &= \psi(x)\psi(y). \end{aligned}$$

5.3 Discrete Wavelet Transform

Suppose we want to encode given discrete data, given by u_j for $j = 0, \ldots, 2^N - 1$, into a wavelet form for some integer N. It is easy to see that this data can be easily represented in the space V_N, which is spanned by the functions $\{\phi_{N,k}\}_{k\in\mathbb{Z}}$, namely

$$u(x) \approx \sum_k \phi_{N,k}(x) u_k.$$

Now, recall that the space V_N can be decomposed into $V_N = V_{N-1} \oplus W_{N-1}$, where V_{N-1} is the coarser space of scaling functions and W_{N-1} is the corresponding space of wavelet functions. Since $V_{N-1} \subset V_N$, then it must be that for any basis function $\phi_{N-1,k}$, we have

$$\begin{aligned}
\phi_{N-1,k} &= \sum_\ell \langle \phi_{N-1,k}, \phi_{N,\ell} \rangle \phi_{N,\ell} \\
&= \sum_\ell \langle \phi_{N-1,k} \phi_{N,\ell+2k} \rangle \phi_{N,\ell+2k} \\
&= \sum_\ell h_\ell \phi_{N,\ell+2k} \\
&= \sum_\ell h_{\ell-2k} \phi_{N,\ell}.
\end{aligned}$$

Similarly, we have

$$\psi_{N-1,k} = \sum_\ell \langle \psi_{N-1,k}, \phi_{N,\ell} \rangle \phi_{N,\ell},$$

where

$$\langle \psi_{N-1,k}, \phi_{N,\ell} \rangle = (-1)^{\ell-2k} h_{1-(\ell-2k)}.$$

Therefore, we have

$$\begin{aligned}
\phi_{N-1,k} &= \sum_\ell h_{\ell-2k} \phi_{N,\ell} \\
\psi_{N-1,k} &= \sum_\ell (-1)^{\ell-2k} h_{1-\ell+2k} \phi_{N,\ell}.
\end{aligned}$$

Let $g_n = (-1)^n h_{1-n}$, so that

$$\psi_{N-1,k} = \sum_\ell g_{\ell-2k} \phi_{N,\ell}.$$

Now, if we want to represent the data u_j in V_{N-1}, then

$$u(x) \approx \sum_k \langle \phi_{N-1,k}, u \rangle \phi_{N-1,k},$$

where

$$\begin{aligned}
\langle \phi_{N-1,k}, u \rangle &= \left\langle \sum_\ell h_{\ell-2k} \phi_{N,\ell}, u \right\rangle \\
&= \sum_\ell h_{\ell-2k} \langle \phi_{N,\ell}, u \rangle \\
&= \sum_\ell h_{\ell-2k} u_\ell = A_{N-1,k}.
\end{aligned}$$

This is called the *low-pass filter* and sometimes called the *trend*. Similarly, we can construct the values

$$\begin{aligned} D_{N-1,k} &= \langle \psi_{N-1,k}, u \rangle \\ &= \left\langle \sum_\ell g_{\ell-2k} \phi_{N,\ell}, u \right\rangle \\ &= \sum_\ell g_{\ell-2k} \langle \phi_{N,\ell}, u \rangle \\ &= \sum_\ell g_{\ell-2k} u_\ell. \end{aligned}$$

This is called the *band-pass filter* or the *details*.

Note that after applying both filters, we now have 2^{N-1} trend values and 2^{N-1} detail values. This process can be repeated on the trend values until only the details and one trend value are left.

Example 5.4:

Suppose data is $u_0, \ldots, u_{15}$ and assume we are using the Haar wavelet so that $h_0 = h_1 = g_0 = \frac{1}{2}$, $g_1 = -\frac{1}{2}$. The space V_4 is spanned by the basis $\phi_{4,k}$. The trend values are then:

$$\begin{aligned} A_{-1,0} &= \sum_\ell h_\ell u_\ell = \frac{1}{2} u_0 + \frac{1}{2} u_1 \\ A_{-1,1} &= \sum_\ell h_\ell - 2u_\ell = \sum_\ell h_\ell u_{\ell+2} = \frac{1}{2} u_2 + \frac{1}{2} u_3 \\ &\vdots \\ A_{-1,7} &= \frac{1}{2} u_{14} + \frac{1}{2} u_{15} \end{aligned}$$

and the details are

$$\begin{aligned} D_{-1,0} &= \sum_\ell g_\ell u_\ell = \frac{1}{2} u_0 - \frac{1}{2} u_1 \\ D_{-1,1} &= \frac{1}{2} u_2 - \frac{1}{2} u_3 \\ &\vdots \\ D_{-1,7} &= \frac{1}{2} u_{14} - \frac{1}{2} u_{15} \end{aligned}$$

Next, the filters are applied to the remaining trend values:

$$\begin{aligned} A_{-2,0} &= \tfrac{1}{2} A_{-1,0} + \tfrac{1}{2} A_{-1,1} & D_{-2,0} &= \tfrac{1}{2} A_{-1,0} - \tfrac{1}{2} A_{-1,1} \\ A_{-2,1} &= \tfrac{1}{2} A_{-1,2} + \tfrac{1}{2} A_{-1,3} & D_{-2,1} &= \tfrac{1}{2} A_{-1,2} - \tfrac{1}{2} A_{-1,3} \\ A_{-2,2} &= \tfrac{1}{2} A_{-1,4} + \tfrac{1}{2} A_{-1,5} & D_{-2,2} &= \tfrac{1}{2} A_{-1,4} - \tfrac{1}{2} A_{-1,5} \\ A_{-2,3} &= \tfrac{1}{2} A_{-1,6} + \tfrac{1}{2} A_{-1,7} & D_{-2,3} &= \tfrac{1}{2} A_{-1,6} - \tfrac{1}{2} A_{-1,7} \end{aligned}$$

and then

$$\begin{aligned} A_{-3,0} &= \tfrac{1}{2} A_{-2,0} + \tfrac{1}{2} A_{-2,1} & D_{-3,0} &= \tfrac{1}{2} A_{-2,0} - \tfrac{1}{2} A_{-2,1} \\ A_{-3,1} &= \tfrac{1}{2} A_{-2,2} + \tfrac{1}{2} A_{-2,3} & D_{-3,1} &= \tfrac{1}{2} A_{-2,2} - \tfrac{1}{2} A_{-2,3} \end{aligned}$$

and finally,

$$A_{-4,0} = \tfrac{1}{2}A_{-3,0} + \tfrac{1}{2}A_{-3,1} \qquad\qquad D_{-4,0} = \tfrac{1}{2}A_{-3,0} - \tfrac{1}{2}A_{-3,1}$$

The original date can be reconstituted by reversing the steps:

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

In the end, we store the final trend value, and all the intermediate detail values and we can reconstruct the original data. The cost of the transform is $O(N)$.

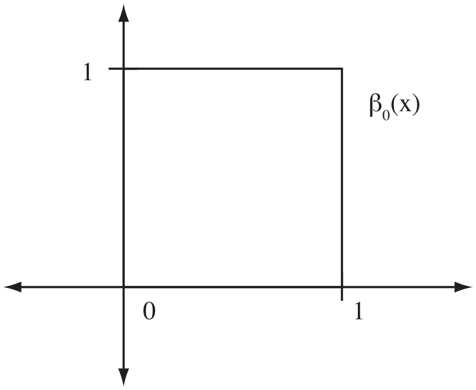
5.4 B-Spline Multiresolution Analysis

The bases we have considered so far are not terribly smooth, and we want smooth basis functions so that we can differentiate them as we have done before. One solution for this is to use B-splines.

We start with the Haar wavelet scaling function, $\phi(x)$, that we have been using as an example so far, but we give it a new name:

$$\beta_0(x) = \begin{cases} 1 & 0 \le x < 1 \\ 0 & \text{otherwise} \end{cases},$$

as shown below:



Haar basis function $\beta_0(x)$

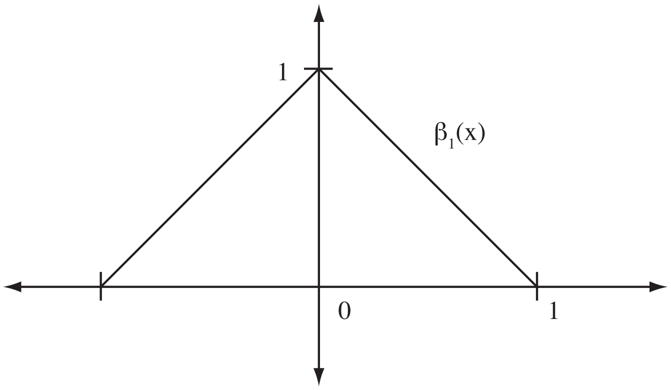
The next higher version will be obtained by taking a convolution:

$$\beta_1(x) = \int_{-\infty}^{\infty} \beta_0(y)\beta_0(x-y)\,dy = \int_0^1 \beta_0(x-y)\,dy = \begin{cases} 2-x & 1 \le x < 2 \\ x & 0 \le x < 1 \\ 0 & x < 0 \text{ or } x \ge 2 \end{cases}.$$

If we shift the result back to 0, we get

$$\beta_1(x) = [x+1]_+ - 2[x]_+ + [x = 1]_+,$$

where $[y]_+ = \max\{y, 0\}$. This basis function is the standard hat function as shown below:



Basis function $\beta_1(x)$

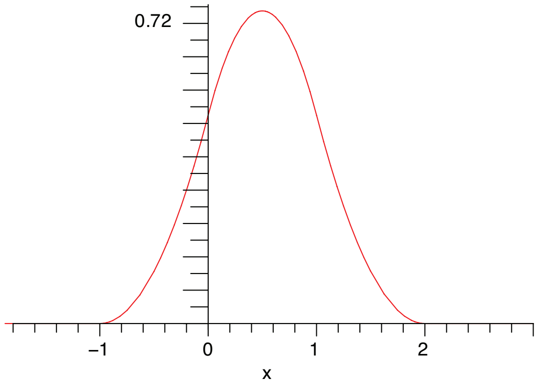
Higher order splines follow the same recipe:

$$\beta_n(x) = \int_{-\infty}^{\infty} \beta_0(y)\beta_{n-1}(x - y + (n \bmod 2))\, dy,$$

where the extra $n \mod 2$ comes from shifting the resulting convolution to be centered around zero for odd values of n. Thus, we can get $\beta_2(x)$ by

$$\begin{aligned}
\beta_2(x) &= \int_{-\infty}^{\infty} \beta_0(x)\beta_1(x-y)\, dy \\
&= \int_0^1 [x+1-y]_+ - 2[x-y]_+ + [x-y-1]_+\, dy \\
&= \int_{x-1}^{x} [u+1]_+ - 2[u]_+ + [u-1]_+\, du \\
&= \frac{1}{2}[u+1]_+^2 - [u]_+^2 + \frac{1}{2}[u-1]_+^2\Big|_{x-1}^{x} \\
&= \frac{1}{2}[x+1]_+^2 - \frac{3}{2}[x]_+^2 + \frac{3}{2}[x-1]_+^2 - \frac{1}{2}[x-2]_+^2,
\end{aligned}$$

and it is illustrated below:



Basis function $\beta_2(x)$

In general, we can write the formula for $\beta_n(x)$ as

$$\beta_n(x) = \begin{cases} \frac{1}{n!}\sum_{j=0}^{n+1}(-1)^j\binom{n+1}{j}\left[x+\frac{n}{2}-j\right]_+^n & n \text{ even,} \\ \frac{1}{n!}\sum_{j=0}^{n+1}(-1)^j\binom{n+1}{j}\left[x+\frac{n+1}{2}-j\right]_+^n & n \text{ odd,.} \end{cases}$$

Note that these spline functions are not suitable scaling functions in our current framework because they are not mutually orthogonal, i.e. $\langle \beta_n(x), \beta_n(x-1)\rangle \neq 0$.

On the other hand, the $\beta_n(x)$ do have the two-scale property:

$$\beta_n(x) = \begin{cases} \frac{1}{2^n}\sum_{j=0}^{n+1}\binom{n+1}{j}\beta_n\left(2x+\frac{n}{2}-j\right) & n \text{ even,} \\ \frac{1}{2^n}\sum_{j=0}^{n+1}\binom{n+1}{j}\beta_n\left(2x+\frac{n+1}{2}-j\right) & n \text{ odd.} \end{cases}$$

Example 5.5:

For $n = 1$, we have

$$\beta_1(x) = [x+1]_+ - 2[x]_+ + [x-1]_+.$$

At the same time, we have

$$\begin{aligned} &\frac{1}{2}(\beta_1(2x+1) + 2\beta_1(x) + \beta_1(2x-1)) \\ &= \frac{1}{2}([2x+2]_+ - 2[2x+1]_+ + [2x]_+ + 2[2x+1]_+ - 4[2x]_+ + 2[2x-1]_+ + [2x]_+ - 2[2x-1]_+ + [2x+2]_+) \\ &= \frac{1}{2}(2[x+1]_+ - 4[x]_+ + 2[x-1]_+) \\ &= \beta_1(x). \end{aligned}$$

So now we just need to orthogonalize these functions. Note that if we take the Fourier transform we get

$$\begin{aligned} \hat{\beta}_0(\omega) &= \int_{-\infty}^{\infty} \beta_0(x)e^{-i\omega x}\,dx \\ &= \int_0^1 e^{-i\omega x}\,dx \\ &= \frac{1}{1\omega}(1-e^{-i\omega}) \\ &= e^{-i\omega/2}\frac{\sin\omega/2}{\omega/2} \end{aligned}$$

Since β_n is constructed as convolutions of β_0, then

$$\hat{\beta}_n(\omega) = e^{-i\kappa\omega/2}\left(\frac{\sin\omega/2}{\omega/2}\right)^{n+1},$$

where $\kappa = n \bmod 2$. Next, define

$$\begin{aligned} b(\omega) &= \sum_\ell \left|\hat{\beta}_n(\omega+2\pi\ell)\right|^2 \\ &= (2\sin\omega/2)^{2(n+1)}\sum_\ell(\omega+2\pi\ell)^{-2(n+1)} \\ &= (2\sin\omega/2)^{2(n+1)}S_{2(n+1)}(\omega), \end{aligned}$$

where

$$S_n(\omega) = \sum_{\ell} (\omega + 2\pi\ell)^{-n}.$$

It can be shown that

$$S_2(\omega) = \frac{1}{4\sin^2 \omega/2},$$

and generally,

$$S_n(\omega) = \frac{(-1)^{n-2}}{(n-1)!} \frac{d^{n-2}}{d\omega^{n-2}} S_2(\omega).$$

The proper scaling function is then

$$\hat{\phi}(\omega) = \frac{\hat{\beta}_n(\omega)}{\sqrt{b(\omega)}}.$$

A concise formula for the scaling function is not available, so instead, let c_k be such that

$$\frac{1}{\sqrt{b(\omega)}} = \sum_k c_k e^{-i\omega k},$$

i.e. the c_k are the discrete Fourier transform of $1/\sqrt{b(\omega)}$, which is possible because $b(\omega)$ is a 2π peeriodic function. Given the c_k, then the scaling function becomes

$$\phi(x) = \sum_k c_k \beta_n(x-k).$$

Note that the c_k must be truncated because there are an infinite number of non-zero entries, but instead decay to zero as $k \to \infty$.

5.5 Biorthogonal Wavelets

The problem with orthogonal wavelets is that it is difficult to build smooth orthogonal functions with compact support. When we built smooth functions, e.g. the B-spline wavelets, they ended up not having compact support. To circumvent the orthogonality condition, we weaken the condition to produce bi-orthogonal basis functions.

Suppose $\mathbf{u}_1$, $\mathbf{u}_2 \in \mathbb{R}^2$ are two linearly independent vectors. Any vector $\mathbf{w} \in \mathbb{R}^2$ can be wrtten as $\mathbf{w} = \alpha_1\mathbf{u}_1 + \alpha_2\mathbf{u}_2$ for some scalars α_1, α_2. If $\mathbf{u}_1$, $\mathbf{u}_2$ are orthogonal, then $\alpha_i = \langle \mathbf{u}_i, \mathbf{w}\rangle$, but if $\langle \mathbf{u}_1, \mathbf{u}_2\rangle \neq 0$, then the situation gets more complicated.

Now suppose $\mathbf{v}_1$, $\mathbf{v}_2$ are such that $\langle \mathbf{v}_i, \mathbf{u}_j\rangle = \delta_{i,j}$, then $\mathbf{v}_1$, $\mathbf{v}_2$ also form a basis of $\mathbb{R}^2$ and

$$\alpha_i = \langle \mathbf{v}_i, \mathbf{w}\rangle.$$

The vectors $\mathbf{v}_1$, $\mathbf{v}_2$ form a *dual basis* of $\mathbf{u}_1$, $\mathbf{u}_2$.

In terms of wavelets, we hope to construct dual scaling functions and dual wavelets so that $\tilde{\phi}$ are the dual scaling functions, $\tilde{\psi}$ are the dual wavelet functions that satisfy the following equations:

$$\begin{aligned} \langle \tilde{\phi}_{j,k}, \phi_{j,\ell}\rangle &= \delta_{k,\ell}, \\ \langle \tilde{\psi}_{j,k}, \phi_{j,\ell}\rangle &= 0, \\ \langle \tilde{\psi}_{j,k}, \psi_{\ell,m}\rangle &= \delta_{j,\ell}\delta_{k,m}, \\ \langle \tilde{\phi}_{j,k}, \psi_{j,k\ell}\rangle &= 0. \end{aligned}$$

and where

$$\begin{aligned} \tilde{\phi}_{j,k} &= 2^{j/2}\tilde{\phi}(2^j x - k), \\ \tilde{\psi}_{j,k} &= 2^{j/2}\tilde{\psi}(2^j x - k). \end{aligned}$$

The set $\{\tilde{\phi}_{j,k}\}_{k\in\mathbb{Z}}$ forms the basis for a dual subspace $\tilde{V}_j$, and similarly $\{\tilde{\psi}_{j,k}\}_{k\in\mathbb{Z}}$ forms a basis for $\tilde{W}_j$.

Now, we still want our scaling and wavelet functions to satisfy the scaling functions:

$$\begin{aligned}
\phi(x) &= \sqrt{2}\sum_k h_k \phi(2x-k),\\
\psi(x) &= \sqrt{2}\sum_k g_k \phi(2x-k),\\
\tilde{\phi}(x) &= \sqrt{2}\sum_k \tilde{h}_k \tilde{\phi}(2x-k),\\
\tilde{\psi}(x) &= \sqrt{2}\sum_k \tilde{g}_k \tilde{\psi}(2x-k).
\end{aligned}$$

To get the coefficients, we take inner products:

$$\begin{aligned}
\phi_{0,0}(x) &= \sum_k h_k \phi_{1,k}(x)\\
\langle \tilde{\phi}_{1,k}, \phi_{0,0}\rangle &= \sum_n h_n \langle \tilde{\phi}_{1,k}, \phi_{1,n}\rangle\\
&= \sum_n h_n \delta_{k,n} = h_k.
\end{aligned}$$

Similarly, we have

$$\begin{aligned}
\langle \phi_{1,k}, \tilde{\phi}_{0,0}\rangle &= \tilde{h}_k,\\
\langle \tilde{\phi}_{1,k}, \psi_{0,0}\rangle &= g_k,\\
\langle \phi_{1,k}, \tilde{\psi}_{0,0}\rangle &= \tilde{g}_k.
\end{aligned}$$

Note that if the $\{\phi_{j,k}\}$ are orthogonal, then $\phi_{j,k} = \tilde{\phi}_{j,k}$, and $\psi_{j,k} = \tilde{\psi}_{j,k}$.

5.5.1 The Bi-orthogonal Wavelet Transform

The wavelet transform carries over from the orthogonal case with only minor modifications. In this case, the coefficients h_n, g_n are used for decomposition, ahd $\tilde{h}_n$, $\tilde{g}_n$ are used for reconstruction. Suppose our given data is given by u_j, then we can write u as an expansion in the dual basis:

$$u = \sum_k \langle \phi_{j,k}, u\rangle \tilde{\phi}_{j,k} + \sum_k \langle \psi_{j,k}, u\rangle \tilde{\psi}_{j,k}.$$

Next, recall that

$$\begin{aligned}
\phi_{j-1,k} &= \sum_\ell h_{\ell-2k}\phi_{j,\ell},\\
\psi_{j-1,k} &= \sum_\ell g_{\ell-2k}\phi_{j,\ell}.
\end{aligned}$$

Let $A_{j,k} = \langle \phi_{j,k}, u\rangle$ and $D_{j,k} = \langle \psi_{j,k}, u\rangle$, then

$$\begin{aligned}
\phi_{j-1,k} &= \sum_\ell h_{\ell-2k}\phi_{j,\ell},\\
\langle u, \phi_{j-1,k}\rangle &= \sum_\ell h_{\ell-2k}\langle u, \phi_{j,\ell},\\
A_{j-1,k} &= \sum_\ell h_{\ell-2k}A_{j,\ell}.
\end{aligned}$$

Similarly, we get $D_{j-1,k} = \sum_\ell g_{\ell-2k} A_{j,\ell}$. These relations show how to compute the trends, $A_{j,k}$, and details, $D_{j,k}$, from the original data.

Recall that $V_j = V_{j-1} \oplus W_{j-1}$, so we can express $\phi_{j,k}$ as a linear combination:

$$\begin{aligned}\phi_{jk} &= sum_\ell \langle \tilde{\phi}_{j-1,\ell}, \phi_{j,k} \rangle \phi_{j-1,\ell} + \sum_\ell \langle \tilde{\psi}_{j-1,\ell}, \phi_{j,k} \rangle \psi_{j-1,\ell} \\ &= \sum_\ell h_{k-2\ell} \phi_{j-1,\ell} + \sum_\ell g_{k-2\ell} \psi_{j-1,\ell}.\end{aligned}$$

Thus,

$$A_{j,k} = \sum_\ell \tilde{h}_{k-2\ell} A_{j-1,\ell} + \sum_\ell \tilde{g}_{k-2\ell} D_{j-1,\ell}.$$

This shows that the transform is the same as before except we use $\tilde{h}_n$, $\tilde{g}_n$ instead of h_n, g_n in the reconstruction.

5.6 Differentiating the B-spline wavelet functions

Recall that the B-spline scaling function is

$$\beta_n(x) = \begin{cases} \frac{1}{n!} \sum_{j=0}^{n+1} (-1)^j \binom{n+1}{j} \left[x + \frac{n}{2} - j\right]_+^n & n \text{ even}, \\ \frac{1}{n!} \sum_{j=0}^{n+1} (-1)^j \binom{n+1}{j} \left[x + \frac{n+1}{2} - j\right]_+^n & n \text{ odd},. \end{cases}$$

Suppose n is even (the case for n odd is analogous), then

$$\begin{aligned}\beta_n'(x) &= \frac{1}{(n-1)!} \sum_{j=0}^{n+1} (-1)^j \binom{n+1}{j} \left[x + \frac{n}{2} - h\right]_+^{n-1} \\ &= \frac{1}{(n-1)!} \left\{ \left[x + \frac{n}{2}\right]_+^{n-1} + \left[x + \frac{n}{2} - (n+1)\right]_+^{n-1} (-1)^{n+1} + \sum_{j=1}^{n} (-1)^j \binom{n+1}{j} \left[x + \frac{n}{2} - j\right]_+^{n-1} \right\}\end{aligned}$$

Now note that

$$\begin{aligned}\binom{n+1}{j} &= \frac{(n+1)!}{j!(n+1-j)!} = \frac{n+1}{j(n+1-j)} \frac{n!}{(j-1)!(n-j)!} \\ &= \left(\frac{1}{j} + \frac{1}{n+1-j}\right) \frac{n!}{(j-1)!(n-j)!} \\ &= \frac{n!}{j!(n-j)!} + \frac{n!}{(j-1)!(n+1-j)!} \\ &= \binom{n}{j} + \binom{n}{j-1}.\end{aligned}$$

So,

$$\begin{aligned}\beta_n'(x) &= \frac{1}{(n-1)!} \left\{ \left[x + \frac{n}{2}\right]_+^{n-1} + (-1)^{n+1} \left[x + \frac{n}{2} - (n+1)\right]_+^{n-1} + \sum_{j=1}^{n} (-1)^j \left(\binom{n}{j} + \binom{n}{j-1}\right) \left[x + \frac{n}{2} - j\right]_+^{n-1} \right\} \\ &= \frac{1}{(n-1)!} \left\{ \sum_{j=0}^{n} (-1)^j \binom{n}{j} \left[x + \frac{n}{2} - j\right]_+^{n-1} + \sum_{j=1}^{n+1} (-1)^j \binom{n}{j-1} \left[x + \frac{n}{2} - j\right]_+^{n-1} \right\} \\ &= \frac{1}{(n-1)!} \left\{ \sum_{j=0}^{n} (-1)^j \binom{n}{j} \left[x + \frac{n}{2} - j\right]_+^{n-1} - \sum_{j=0}^{n} (-1)^j \binom{n}{j} \left[x - 1 + \frac{n}{2} - j\right]_+^{n-1} \right\} \\ &= \beta_{n-1}(x) - \beta_{n-1}(x-1).\end{aligned}$$

Thus, we see that the B-splines have the special property that the derivatives are easy to construct making it possible to do spatial derivatives easily within the framework.