SPECTRAL METHODS

MARIA CAMERON

Contents

1. Spectral methods

Spectral methods are widely used. If the solution is smooth, then its Fourier series converges exponentially in the number of terms. Moreover, differentiation of the Fourier series in the Fourier space is easy: it is merely a multiplication by ik where k is the wave number and i is the imaginary unit. Spatial derivatives. of order higher than 2 are hard to approximate by finite differences. For example, the Korteweg-de Vries equation is hard to solve without the use of spectral methods.

2. General Fourier Series and the best approximations

Suppose a series of functions $\{\phi_j(x)\}_{j=0}^{\infty}$ belongs to a space with inner product. The 2-norm is defined as $\|\cdot\|_2 = \sqrt{(\cdot,\cdot)}$. The task of the least squares fitting is to find the best approximation of the form $\sum_{j=0}^{N} c_j\phi_j(x)$ for the given function $f(x)$ in the sense that the 2-norm $\|f - \sum_{j=0}^{N} c_j\phi_j(x)\|_2$ is minimal among all possible choices of the coefficients $c_0,\dots,c_N$. For the rest of this section we will deal only with the 2-norm.

If the functions ϕ_j's are orthogonal, i.e.,

$$(\phi_i, \phi_j) = 0 \quad \text{whenever} \quad i \neq j,$$

the best approximation for $f(x)$ of the form $\sum_j c_j \phi_j(x)$ is called a generalized Fourier series for $f(x)$ and the coefficients c_j's giving the best fit are called generalized Fourier coefficients.

Theorem 1. *Let $\{\phi_j\}_{j=0}^{\infty}$ be an orthogonal series of functions. Among all possible choices of $c_0, \dots, c_N$ the choice that minimizes*

$$\left\| f - \sum_{j=0}^{N} c_j \phi_j \right\|_2 \quad \text{is} \quad c_j = \frac{(f, \phi_j)}{(\phi_j, \phi_j)}, \; j = 0, 1, \dots, N.$$

Proof. We consider the squared norm of the error of the approximation.

$$E_N^2 = \left\| f - \sum_{j=0}^{N} c_j \phi_j \right\|^2 = \left(f - \sum_{j=0}^{N} c_j \phi_j, f - \sum_{j=0}^{N} c_j \phi_j \right) =$$

$$\|f\|^2 - 2 \sum_{j=0}^{N} c_j (f, \phi_j) + \left(\sum_{j=0}^{N} c_j \phi_j \right) \left(\sum_{k=0}^{N} c_k \phi_k \right) =$$

$$\|f\|^2 - 2 \sum_{j=0}^{N} c_j (f, \phi_j) + \sum_{j=0}^{N} |c_j|^2 \|\phi_j\|^2.$$

Now we complete the square for the last two terms.

$$E_N^2 = \|f\|^2 + \sum_{j=0}^{N} \left| c_j - \frac{(f, \phi_j)}{\|\phi_j\|^2} \right|^2 \|\phi_j\|^2 - \sum_{j=0}^{N} \|\phi_j\|^2 \frac{(f, \phi_j)^2}{\|\phi_j\|^4} \geq$$

$$\|f\|^2 - \sum_{j=0}^{N} \|\phi_j\|^2 \left[\frac{(f, \phi_j)}{(\phi_j, \phi_j)} \right]^2 \geq 0.$$

The second-to-last inequality is an equality if and only if

$$c_j = \frac{(f, \phi_j)}{(\phi_j, \phi_j)}, \quad j = 0, 1, \dots, N.$$

□

The immediate corollary of this proof is the Bessel's inequality:

$$\|f\|^2 \geq \sum_{j=0}^{N} \|\phi_j\|^2 \left[\frac{(f, \phi_j)}{(\phi_j, \phi_j)} \right]^2.$$

Another immediate corollary is Parseval's equality.

Theorem 2. *A generalized Fourier series converges to f if and only if Parseval's equality holds:*

$$\|f\|^2 = \sum_{j=0}^{\infty} \|\phi_j\|^2 \left[\frac{(f, \phi_j)}{(\phi_j, \phi_j)} \right]^2.$$

2.1. **Trigonometric series.** Let a function $f(x)$ be defined on interval $[0,L]$ and satisfy boundary conditions $f(0)=f(L)=0$. Then we can approximate it using the sine series

$$f(x)\approx\sum_{n=1}^{N}c_n\sin\frac{\pi nx}{L},\quad\text{where}\quad c_n=\frac{2}{L}\int_0^L f(x)\sin\frac{\pi nx}{L}dx.$$

If $f(x)$ is defined on $[0,L]$ and satisfies the Neumann boundary conditions $f('(0)=f'(L)=0$ we can approximate it using the cosine series

$$f(x)\approx\sum_{n=0}^{N}c_n\cos\frac{\pi nx}{L},$$

where

$$c_0=\frac{1}{L}\int_0^L f(x)dx,\quad c_n=\frac{2}{L}\int_0^L f(x)\cos\frac{\pi nx}{L}dx,\ n=1,2,\ldots N.$$

Now suppose that $f(x)$ is defined on $[-L,L]$ and satisfies the periodic boundary conditions $f(-L)=f(L)$, and $f'(-L)=f'(L)$. Then f can be approximated using the full Fourier series

$$f(x)\approx\frac{a_0}{2}+\sum_{n=1}^{N}\left(a_n\cos\frac{\pi nx}{L}+b_n\sin\frac{\pi nx}{L}\right),$$

where

$$a_n=\frac{1}{L}\int_{-L}^{L} f(x)\cos\frac{\pi nx}{L}dx,\quad n=0,\ldots,N,$$
$$b_n=\frac{1}{L}\int_{-L}^{L} f(x)\sin\frac{\pi nx}{L}dx,\quad n=1,\ldots,N.$$

Alternatively, a function $f(x)$ defined on $[-L,L]$ and satisfying periodic boundary conditions can be approximated using the complex series

$$f(x)\approx\sum_{k=-n}^{n}c_ne^{\frac{i\pi kx}{L}},\quad\text{where}\quad c_k=\frac{1}{2L}\int_{-L}^{L}f(x)e^{-\frac{i\pi kx}{L}}dx.$$

When we say "let X_n be a Fourier series for $f(x)$ on $[a,b]$ satisfying the boundary conditions" we mean one of the following:

- $[a,b]=[0,L],\quad f(0)=f(L)=0,\quad X_n(x)=\sin\frac{\pi nx}{L},\quad n=1,2,\ldots.$
- $[a,b]=[0,L],\quad f'(0)=f'(L)=0,\quad X_n(x)=\cos\frac{\pi nx}{L},\quad n=0,1,2,\ldots.$
- $[a,b]=[0,L],\quad f(0)=f'(L)=0,\quad X_n(x)=\sin\frac{\pi(2n+1)x}{2L},\quad n=0,1,2,\ldots.$
- $[a,b]=[0,L],\quad f'(0)=f(L)=0,\quad X_n(x)=\cos\frac{\pi(2n+1)x}{2L},\quad n=0,1,2,\ldots.$
- $[a,b]=[-L,L],\quad f(-L)=f(L),\quad f'(-L)=f'(L),\quad X_n=e^{i\frac{\pi nx}{L}},\quad n\in\mathbb{Z}.$
- $[a,b]=[-L,L],\quad f(-L)=f(L),\quad f'(-L)=f'(L),\quad X_n=\left\{\sin\frac{\pi nx}{L},\cos\frac{\pi nx}{L}\right\}.$

The last one is the classic Fourier series of sines and cosines. Results regarding the convergence of Fourier series are the following:

Theorem 3. *The Fourier series $\sum c_n X_n(x)$ converges to $f(x)$ uniformly on $[a,b]$ provided that*

(1) *f, f', and f'' exist and are continuous on $[a,b]$;*
(2) *$f(x)$ satisfies the boundary conditions.*

Theorem 4. *The Fourier series $\sum c_n X_n(x)$ converges to $f(x)$ on $[a,b]$ in L_2 provided that*

$$\int_a^b |f(x)|^2 dx < \infty.$$

Theorem 5. *Let f and f' be piecewise continuous on $[a,b]$. Then the Fourier series converges pointwise to*

$$\tfrac{1}{2}(f(x+0)+f(x-0)).$$

3. Convergence of Fourier Series

3.1. Algebraic vs Exponential decay of coefficients.

Example Consider the "sawtooth function" (Fig. 1): $f(x)=x$, $x\in[-\pi,\pi]$. Since $f(x)$ is odd, the Fourier Series is the sine series.

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} x\sin(nx)dx = (-1)^{n+1}\frac{2}{n}.$$

The coefficients decay as $O(\frac{1}{n})$. The series converges pointwise to $f(x)$ on $(-\pi,\pi)$. However, $\sup_{x\in(-\pi,\pi)}|f(x)-S_N(x)|$ does not decay as $N\to\infty$. This exemplifies the Gibbs phenomenon.

Example Consider the "half-wave rectifier function" (Fig. 2): $f(x)=0$ for $x\in[-\pi,0]$, $f(x)=\sin x$ for $x\in(0,\pi]$. This function is continuous with a piecewise continuous derivative. Its Fourier coefficients are

$$a_0=\frac{1}{\pi},\quad a_{2n}=-\frac{2}{\pi(4n^2-1)},\quad n>0,\quad a_{2n+1}=0,\quad n\geq 0,$$

$$b_1=\frac{1}{2},\quad b_n=0\quad n>1.$$

The coefficients decay as $O(n^{-2})$ for large n. The series converges pointwise to $f(x)$ on $[-\pi,\pi]$.

Example Consider the function called "symmetric, imbricated Lorentzian" (Fig. 3):

$$f(x)=\frac{1-p^2}{1+p^2-2p\cos x},\qquad 0<p<1.$$

this function is infinitely differentiable. Its Fourier series is given by

$$f(x)=1+2\sum_{n=1}^{\infty}p^n\cos(nx).$$

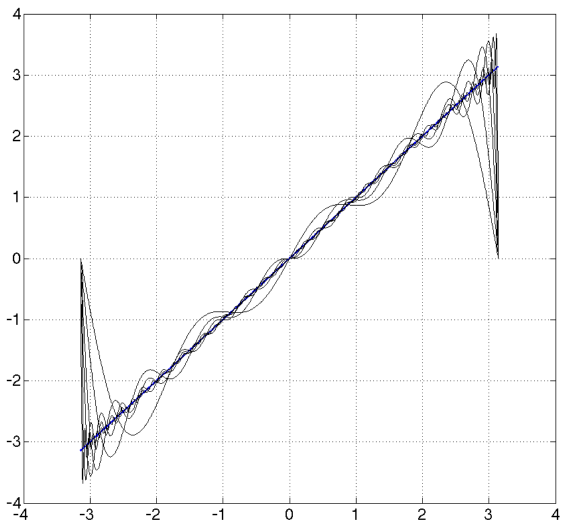

FIGURE 1. $f(x) = x$ on $[-\pi, \pi]$ and the partial sums S_N of its Fourier sine series for $N = 3,\ 6,\ 12,\ 20,\ 40,\ 100$.

One can check it as follows.

$$\begin{aligned} f(x) &= 1 + 2\sum_{n=1}^{\infty} p^n \cos(nx) = 1 + \sum_{n=1}^{\infty} p^n \left(e^{inx} + e^{-inx}\right) \\ &= \frac{1}{1 - pe^{ix}} + \frac{1}{1 - pe^{-ix}} - 1 \\ &= \frac{1 - pe^{-ix} + 1 - pe^{ix} - 1 - p^2 + 2p\cos x}{1 + p^2 - 2p\cos x} \\ &= \frac{1 - p^2}{1 + p^2 - 2p\cos x}. \end{aligned}$$

The Fourier coefficients decay faster than any power of n. In this case we say that the coefficients decay exponentially or spectrally.

Definition 1. *The algebraic index of convergence is the supremum of numbers k for which*

$$\lim_{n\to\infty} |c_n| n^k < \infty,$$

where c_n are the coefficients of the series.

Alternative definition: if

$$c_n \sim O\left(\frac{1}{n^k}\right), \quad n \gg 1,$$

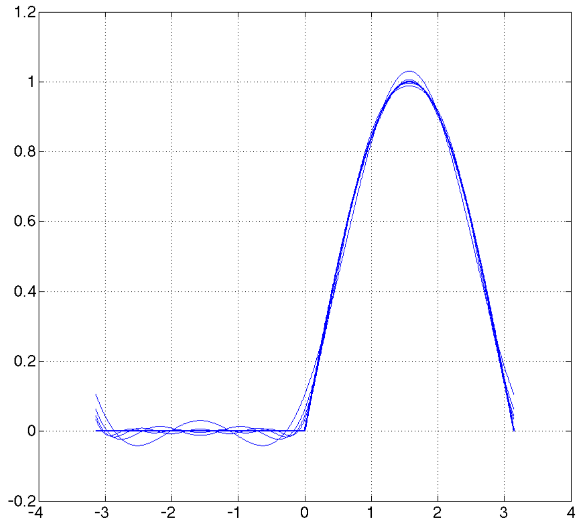

FIGURE 2. Half-wave rectifier and the partial sums S_N of its Fourier series for $n = 2,\ 4,\ 6,\ 8$.

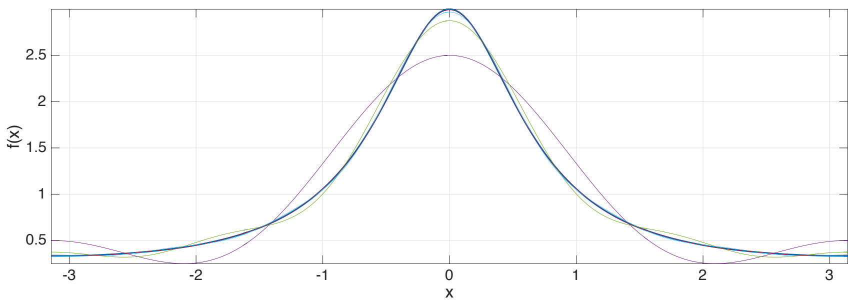

FIGURE 3. Function "symmetric, imbricated Lorentzian" $f(x) = \frac{1-p^2}{1+p^2-2p\cos x}$ for $p = 1/2$, and the partial sums for $n = 2,\ 4,\ 6,\ 8$.

then k is the algebraic index of convergence.

The first form gives an unambiguous definition for the case where $c_n \sim O\left(\frac{\log n}{n}\right)$.

Definition 2. *If the algebraic index of convergence is unbounded than the series converges exponentially or spectrally.*

Definition 3. *The exponential index of convergence r is given by*

$$r = \limsup_{n\to\infty} \frac{\log|\log|c_n||}{\log n}.$$

Equivalently, if

$$c_n \sim O(s\exp[-qn^r]), \quad n \gg 1,$$

then the exponential index of convergence is the exponent r.

Example For the symmetric, imbricated Lorentzian, $c_n = 2p^n = 2e^{-n(-\log p)}$, (note, $\log p < 0$ as $0 < p < 1$), hence $r = 1$, $q = -\log p$, $s = 2$.

3.2. **Decay of coefficients and approximation errors.** Let $f(x)$ be 2π-periodic. The coefficients of the complex-exponential form of a Fourier series are:

$$c_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(x)e^{-inx}dx. \tag{1}$$

Note that if $f(x)$ is real-valued, then

$$f(x) = \sum_{-\infty}^{\infty} c_n e^{inx} = \overline{f(x)} = \overline{\sum_{-\infty}^{\infty} c_n e^{inx}} = \sum_{-\infty}^{\infty} \overline{c_n} e^{-inx}. \tag{2}$$

Therefore, for a real-valued $f(x)$,

$$c_n = \overline{c_{-n}}.$$

If we integrate Eq. (1) by parts, repeatedly integrating the exponent and differentiating f, then after $J+1$ steps we obtain

$$c_n = \frac{1}{2\pi}\sum_{j=0}^{J}(-1)^{j+n}\left(\frac{i}{n}\right)^{j+1}\left\{f^{(j)}(\pi) - f^{(j)}(-\pi)\right\}$$
$$+ \frac{1}{2\pi}\left(-\frac{i}{n}\right)^{J+1}\int_{-\pi}^{\pi} f^{(J+1)}(x)e^{-inx}dx.$$

Exercise Show that for the classic Fourier series $a_n = 2Re(c_n)$ and $b_n = -2Im(c_n)$ and

$$a_n \sim \frac{1}{\pi}\sum_{j=0}^{J}(-1)^{n+j}\left(\frac{f^{(2j+1)}(\pi) - f^{(2j+1)}(-\pi)}{n^{2j+2}}\right) + O(n^{-(2J+4)}), \quad n\to\infty, \quad J \text{ fixed},$$

$$b_n \sim \frac{1}{\pi}\sum_{j=0}^{J}(-1)^{n+j+1}\left(\frac{f^{(2j)}(\pi) - f^{(2j)}(-\pi)}{n^{2j+1}}\right) + O(n^{-(2J+3)}), \quad n\to\infty, \quad J \text{ fixed}.$$

Therefore the following theorem takes place.

Theorem 6. *If*

(1)
$$f(-\pi) = f(\pi),\ f'(-\pi) = f'(\pi),\ \ldots,\ f^{(k-2)}(-\pi) = f^{(k-2)}(\pi),$$

(2) $f^{(k)}$ *is integrable,*

then

$$|a_n| \le \frac{F}{n^k}, \quad |b_n| \le \frac{F}{n^k},$$

where F is a constant that does not depend on n.

The next is the Fourier truncation error theorem.

Theorem 7. *The error of approximating $f(x)$ by the sum of the first N terms of its Fourier series is bounded by the sum of the absolute values of all neglected coefficients.*

Example The "sawtooth function" that is obtained from $f(x) = x$, $x \in (-\pi, \pi)$ by the periodic extension to the real line is not continuous (Fig. 4). Its integral is continuous. Its derivative can be expressed in terms of the Dirac delta-function as follows

$$f'(x) = 1 - 2\pi \sum_{n=-\infty}^{\infty} \delta(x - \pi(2n+1)).$$

You can check it by verifying the identity

$$f(x) = \int_0^x f'(x')dx'.$$

Hence $f'(x)$ is integrable on $[-\pi, \pi]$. Therefore, we need to take $k = 1$ to apply Theorem 6 and get the bound $|b_n| \le \frac{F}{n}$. This bound is correct as we have seen in Example 3.1.

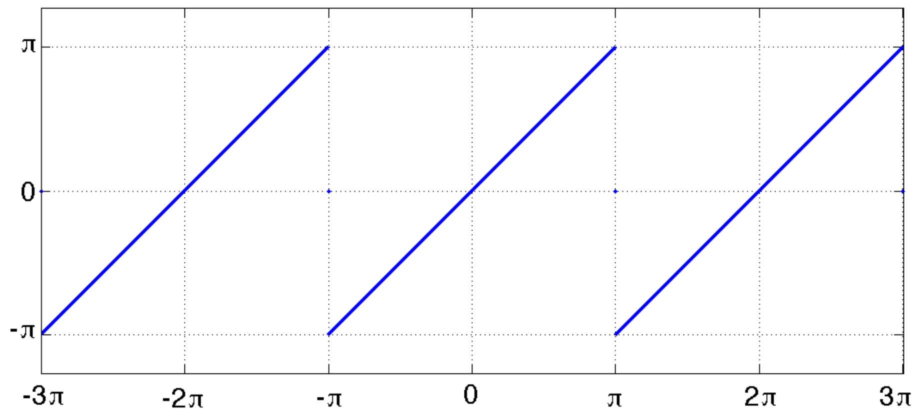

FIGURE 4. The Sawtooth function.

4. DISCRETE FOURIER TRANSFORM IN MATLAB

In MATLAB 2015b, the discrete Fourier transform is defined as follows. Let x be a vector with N entries. The functions `fft` and `ifft` implement the discrete Fourier transform and the inverse discrete Fourier transform. If $y = \texttt{fft}(x)$ and $x = \texttt{ifft}(y)$ then

$$y(k) = \sum_{j=1}^{N} x(j)\omega_N^{(j-1)(k-1)} \tag{3}$$

and

$$x(j) = \frac{1}{N}\sum_{k=1}^{N} y(k)\omega_N^{-(j-1)(k-1)}, \tag{4}$$

where

$$\omega_N = e^{-\frac{2\pi i}{N}}. \tag{5}$$

Exercise Check that the functions `fft` and `ifft` defined by Eqs. (3), (4) and (5) are mutually inverse.

Despite the functions `fft` and `ifft` are mutually inverse, restoring a function using its Fourier modes from 0 to $N-1$ leads to large absolute errors. In practice, `fft` and `ifft` are often used with wave numbers running from $-N/2$ to $N/2-1$. Let N be even. Observe that

$$y(k-N) = \sum_{j=1}^{N} x(j)e^{-\frac{2\pi i(j-1)((k-1)-N)}{N}} = \sum_{j=1}^{N} x(j)e^{-\frac{2\pi i(j-1)(k-1)}{N}} = y(k), \tag{6}$$

as $e^{-2\pi i p} = 1$ for any integer p. This fact allows us to run the wave numbers not from 0 to $N-1$ but from $-N/2$ to $N/2-1$ instead. Matlab provides convenient functions for doing this called `fftshift` and `ifftshift`. These functions swap the left and the right halves of the array. They are mutually inverse. They are identical for even N and differ for odd N. The example below demonstrates their action on a simple example.

```
>> a=[1 2 3 4 5 6];
>> fftshift(a)
ans =
     4     5     6     1     2     3
>> ifftshift(a)
ans =
     4     5     6     1     2     3

>> a=[1 2 3 4 5 6 7];
>> fftshift(a)
ans =
     5     6     7     1     2     3     4
>> ifftshift(a)
ans =
     4     5     6     7     1     2     3
```

Thus, the following sequence of commands is often used:

```
y = fftshift(fft(x));
x = ifft(ifftshift(y));
```

The example below illustrates the restoration of a function from its discrete Fourier transform.

Example Let $f(x) = x(2\pi - x)$, $0 \le x \le 2\pi$. Its Fourier coefficients

$$c_k = \frac{1}{2\pi}\int_0^{2\pi} x(2\pi - x)e^{-ikx}dx$$

are:

$$c_0 = \frac{2\pi^2}{3}, \quad c_k = -\frac{2}{k^2}. \tag{7}$$

Observing that $c_{-k} = c_k$, the function can be written as the following cosine series:

$$x(2\pi - x) = \frac{2\pi^2}{3} - 4\sum_{k=1}^{\infty}\frac{\cos(kx)}{k^2}. \tag{8}$$

The script below produced Figure 5 demonstrating how $f(x)$ is approximated by the truncated cosine series `fN` as well as by the series

$$S_N(x) = \sum_{k=1}^{N}[\texttt{fft}(f)]_k e^{i(k-1)x} \tag{9}$$

and by

$$fs(x) = \sum_{k=1}^{N}[\texttt{fftshift(fft(}f\texttt{))}]_k e^{ix(k-N/2-1)}. \tag{10}$$

```
function fft_demo()
nx = 2000; % for "continuous" x
N = 16; % for discrete Fourier Transform

x=linspace(0,2*pi,2000);
fun = @(x) x.*(2*pi - x);
f = fun(x);
c0 = 2*pi*pi/3;
fN = c0; %ck = -2/k^2;
for k = 1 : N/2
    fN = fN - (4/(k*k))*cos(k*x);
end
fprintf('max|f - fN| = %d\n',max(abs(f - fN)));

figure;
hold on;
grid;
plot(x,f,'Linewidth',2);
```

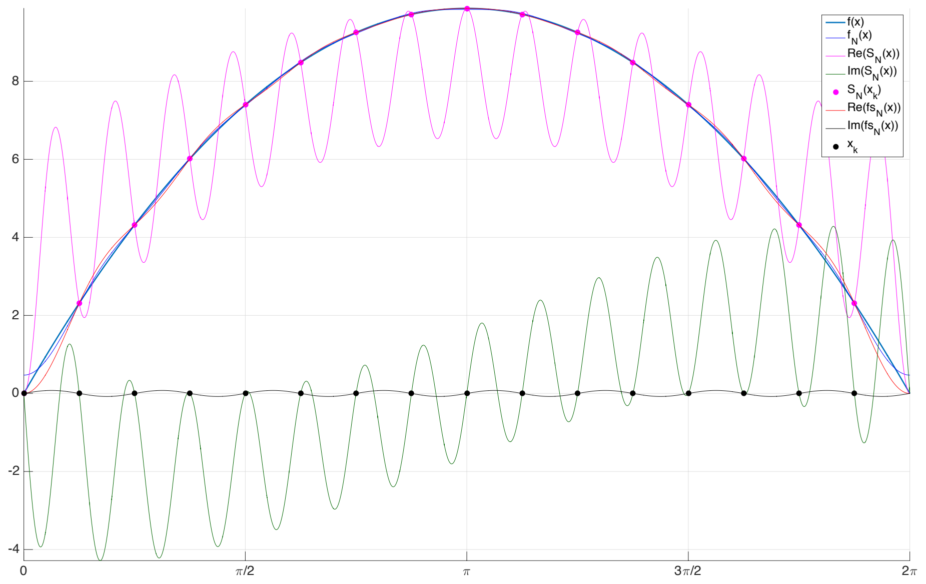

FIGURE 5. An illustration fro Example 4. The function $f(x) = x(2\pi - x)$ is approximated using the truncated Fourier series $fN(x)$ as well as by the interpolation function $SN(x)$ (Eq. (9)) as well as by the shifted interpolation function $fs(x)$ (Eq. (10)).

```
plot(x,fN,'b');

xk = linspace(0,2*pi,N+1);
xk(N+1) = [];
fk = fun(xk);
fkhat = fft(fk);
SN = 0;
for k = 1 : N
    SN = SN + fkhat(k)*exp(1i*x*(k - 1))/N;
end
plot(x,real(SN),'m');
plot(x,imag(SN),'color',[0,102/255,0]);

sN = 0;
for k = 1 : N
    sN = sN + fkhat(k)*exp(1i*xk*(k - 1))/N;
end
```

```
plot(xk,real(sN),'.m','Markersize',20);
fprintf('max|f(xk) - sN(xk)| = %d\n',max(abs(fk - sN)));

fkhatshift = fftshift(fkhat);
fs = 0;
for j = 1 : N
    fs = fs + fkhatshift(j)*exp(1i*x*(j-N/2-1))/N;
end
plot(x,real(fs),'r');
plot(x,imag(fs),'k');
fprintf('max|f - fs| = %d\n',max(abs(f - fs)));

plot(xk,imag(sN),'k.','Markersize',20);

legend('f(x)','f_N(x)','Re(S_N(x))','Im(S_N(x))','S_N(x_k)',...
                                     'Re(fs_N(x))','Im(fs_N(x))','x_k');
set(gca,'Fontsize',20);
axis tight;
ax = gca;
ax.XTick = [0 : pi/2 : 2*pi];
ax.XTickLabel = {'0','\pi/2','\pi','3\pi/2','2\pi'};
end
```

5. Pseudospectral errors

In this Section, we estimate the errors of approximating a function by the interpolation function denoted by $fs(x)$ in Example 4. In the theorem below, the function fs is denoted by S_N. The interpolation function denoted by S_N in Example 4 will not be used because it gives a poor approximation of $f(x)$.

Exercise Let $N > 1$ be even. Show that

$$g_N = \mathcal{R}e\left[\sum_{k=-N/2}^{N/2-1} c_k e^{ikx}\right], \quad \text{where} \quad c_k = \frac{1}{N}\sum_{j=0}^{N-1} f(x_j)e^{-ix_j k}, \tag{11}$$

$x_j = -\pi + \frac{2\pi j}{N}$, $0 \le j \le N-1$, coincides with S_N in Eq. (12) with coefficients a_n and b_n given by Eq. (13).

Theorem 8. *Let the interpolation (or collocation) points be defined by*

$$x_k = -\pi + \frac{2\pi k}{N}, \quad k = 1, 2, \dots, N.$$

Let N be even. Let a function $f(x)$ have the exact, infinite Fourier series representation

$$f(x) = \frac{\alpha_0}{2} + \sum_{n=1}^{\infty} \alpha_n \cos(nx) + \sum_{n=1}^{\infty} \beta_n \sin(nx).$$

The trigonometric polynomial S_N which interpolates to $f(x)$ at the N collocation points, i.e.,

$$S_N(x_k) = f(x_k), \quad k = 1, 2, \dots, N,$$

is

$$S_N = \frac{a_0}{2} + \sum_{n=1}^{N/2-1} a_n \cos(nx) + \sum_{n=1}^{N/2-1} b_n \sin(nx) + \frac{1}{2} a_{N/2} \cos((N/2)x). \tag{12}$$

Then the coefficients of interpolant can be computed without error by the trapezoidal rule

$$a_n = \frac{2}{N} \sum_{k=1}^{N} f(x_k) \cos(nx_k), \quad b_n = \frac{2}{N} \sum_{k=1}^{N} f(x_k) \sin(nx_k) \tag{13}$$

and these coefficients of the interpolant are given by infinite series of the exact Fourier coefficients:

$$a_n = \alpha_n + \sum_{j=1}^{\infty} (\alpha_{n+jN} + \alpha_{-n+jN}), \quad n = 0, 1, 2, \dots, \frac{N}{2},$$

$$b_n = \beta_n + \sum_{j=1}^{\infty} (\beta_{n+jN} - \beta_{-n+jN}), \quad n = 1, 2, \dots, \frac{N}{2} - 1.$$

Furthermore, if f_N is the sum of the first N terms of the exact Fourier series, we have:

$$|f(x) - f_N(x)| \le \left\{ |\beta_{N/2}| + \sum_{n=1+N/2}^{\infty} (|\alpha_n| + |\beta_n|) \right\},$$

while for the trigonometric interpolation the bound is

$$|f(x) - S_N(x)| \le 2 \left\{ |\beta_{N/2}| + \sum_{n=1+N/2}^{\infty} (|\alpha_n| + |\beta_n|) \right\}.$$

That is to say that the error is bounded by twice the sum of the absolute values of all the neglected coefficients.

This theorem is stated in [1] and a reference to its proof is found there.

6. Using Discrete Fourier Transform for solving PDEs

Now we discuss how to use the functions `fft` and `ifft` for solving PDEs.

Example Consider a linear dispersive PDE $u_t + u_{xxx} = 0$ on the interval $-\pi \le x \le \pi$, $t \ge 0$, with periodic boundary conditions and the intial condition $u(x, 0) = u_0(x)$. The periodic boundary conditions suggest the use of trigonometric Fourier series. First we solve it using the Fourier series. Write

$$u(x,t) = \sum_{k=-\infty}^{\infty} c_k(t) e^{ikx}, \quad \text{where} \quad c_k = \frac{1}{2\pi} \int_{-\pi}^{\pi} u(x,t) e^{-ikx} dx.$$

Then

$$u_t = \sum_{k=-\infty}^{\infty} c_k'(t) e^{ikx}, \quad u_{xxx} = \sum_{k=-\infty}^{\infty} c_k(t) (ik)^3 e^{ikx}.$$

Therefore, all harmonics evolve in time independently. For kth harmonic we have

$$c_k'(t) = -(ik)^3 c_k(t) = ik^3 c_k(t). \quad \text{Hence} \quad c_k(t) = c_k(0)e^{ik^3 t}.$$

Finally,

$$u(x,t) = \sum_{k=-\infty}^{\infty} c_k(0) e^{ik^3 t} e^{ikx}, \quad \text{where} \quad c_k(0) = \frac{1}{2\pi}\int_{-\pi}^{\pi} u_0(x) e^{-ikx} dx.$$

Let us take an initial condition whose Fourier transform has a finite number of nonzero terms, for example,

$$u_0(x) = \cos(x)\sin(25x). \tag{14}$$

Then the exact solution is given by

$$u(x,t) = \cos(x + 1876t)\sin(25x + 15700t). \tag{15}$$

Exercise Solve the IVP above using the Fourier transform in the Fourier space. Find phase and group velocities. Verify that the exact solution is given by Eq.(15).

The script `linear_dispersion.m` computes the solution using the discrete Fourier transform and superimposes it with the exact solution (see Fig. 6).

```
function linear_disp_demo()
% Solves u_t + u_{xxx} = 0 using exact time integration and discrete
% Fourier Transform
N = 1024;  % the number of collocation points
nt = 1000;  % the number of time steps
tmax = 0.1;  % the time till that the solution is computed
t = linspace(0,tmax,nt);
x = linspace(-pi,pi,N+1);
x(N+1) = [];
u0 = cos(x).*sin(25*x); % the initial condition
f0 = fftshift(fft(u0));  % Fourier coefficients at time 0
k = [-N/2 : N/2 - 1]; % frequencies

fig = figure;
grid;
hold on;
he = plot(x,u0,'Linewidth',1,'color','r');
h = plot(x,u0,'Linewidth',2,'color','b');
axis([-pi,pi,-1,1]);
drawnow;

for j = 1 : nt
    ft = f0.*exp(1i*k.^3*t(j)); % Fourier coefficients at time t(j)
    ut = ifft(ifftshift(ft));  % solution at time t(j)
     % plot the computed solution at time t(j)
    set(h,'xdata',x,'ydata',real(ut));
     % plot the exact solution at time t(j)
    set(he,'xdata',x,'ydata',cos(x+1876*t(j)).*sin(25*x+15700*t(j)));
    axis([-pi,pi,-1,1]);
    drawnow;
    pause(0.1)
end
```

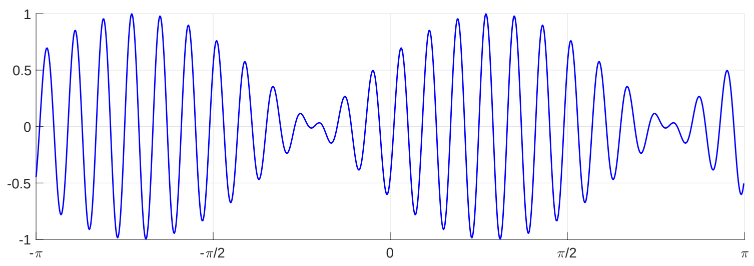

Figure 6. This blue curve: the numerical solution at time $t = 0.1$ of $u_t + u_{xxx} = 0$ with the initial condition (14) obtained using the discrete Fourier transform and exact time integration. The thin red curve: the exact solution at $t = 0.1$ given by Eq. (15).

If we are solving a PDE with periodic boundary conditions on the interval $[0, L]$ then it is very important to scale the frequencies properly. Suppose we are using N-point DFT. Then $x_j = Lj/N$, $j = 0, 1, \dots, N-1$.

$$[\texttt{fft}(u)]_k = \sum_{j=0}^{N-1} f(x_j) e^{-\frac{i2\pi jk}{N}} = \sum_{j=0}^{N-1} f(x_j) e^{-\frac{i2\pi x_j k}{L}}.$$

Therefore, the k-th frequency $f_k = \frac{2\pi k}{L}$, and the vector of frequencies should be set to

$$\texttt{freq} = \frac{2\pi}{L}[0, 1, \dots, N/2-1, -N/2, \dots, -1].$$

Or, if you use `fftshift(fft(u))` then

$$\texttt{freq} = \frac{2\pi}{L}[-N/2, \dots, N/2-1].$$

6.1. **The Korteweg-de Vries equation.**

The Korteweg-de-Vries equation

$$u_t + uu_x + \beta u_{xxx} = 0 \tag{16}$$

originally arose in the theory of surface waves on shallow water. Later it was encountered in numerous other problems. u is proportional to the horizontal component of the velocity which is constant over the channel depth. An analogous equation is also valid for the elevation of the surface over its undisturbed level. The KdV equation has a solution of traveling wave type, the so called soliton:

$$(17)\qquad u = \frac{u_0}{\cosh^2\left(\sqrt{\frac{u_0}{12\beta}}(x - x_0 - st)\right)}, \quad \text{where} \quad u_0 = 3s.$$

This solution is very stable. Let us derive it. We will look for $u(x,t) = w(x - st) \equiv w(y)$, satisfying the boundary conditions $w(\pm\infty) = 0$ and $w'(\pm\infty) = 0$.

$$\begin{aligned}
-sw' + ww' + \beta w''' &= 0\\
-sw' + \left(\frac{w^2}{2}\right)' + \beta w''' &= 0\\
-sw + \frac{w^2}{2} + \beta w'' &= C \quad |\cdot 2w'\\
2\beta w''w' - 2sww' + w^2w' &= 2Cw'\\
\beta(w')^2 - sw^2 + \frac{w^3}{3} &= 2Cw + B
\end{aligned}$$

Since w and w' are zero at $\pm\infty$, $C = 0$ and $B = 0$. We continue.

$$\begin{aligned}
\beta(w')^2 &= sw^2 - \frac{w^3}{3}\\
w' &= \pm\sqrt{\frac{s}{\beta}w^2 - \frac{w^3}{3\beta}}\\
\int \frac{dw}{w\sqrt{s - \frac{w}{3}}} &= \pm\int \frac{dy}{\sqrt{\beta}}
\end{aligned}$$

The integral in the left-hand side can be taken using the substitution

$$\begin{aligned}
&z = \sqrt{s - \tfrac{w}{3}}, \quad dz = -\frac{dw}{6\sqrt{s - \frac{w}{3}}},\\
&z^2 = s - \tfrac{w}{3}, \qquad w = 3s - 3z^2.
\end{aligned}$$

Then we have

$$\begin{aligned}
&\int \frac{dw}{w\sqrt{s - \frac{w}{3}}} = -6\int \frac{dz}{3s - 3z^2}\\
&= -2\int \frac{dz}{s - z^2} = -\frac{2}{\sqrt{s}}\int \frac{dz}{1 - \left(\frac{z}{\sqrt{s}}\right)^2}\\
&= \frac{2}{\sqrt{s}}\int \frac{dp}{p^2 - 1} = \frac{1}{\sqrt{s}}\log\left|\frac{p-1}{p+1}\right|,
\end{aligned}$$

where

$$p = \frac{z}{\sqrt{s}} = \sqrt{1 - \frac{w}{3s}}.$$

This equation shows that $p \le 1$. Now we continue.

$$\frac{1}{\sqrt{s}} \log\left|\frac{p-1}{p+1}\right| = \pm\frac{y}{\sqrt{\beta}} + C$$
$$\left|\frac{p-1}{p+1}\right| = e^{\pm(y-x_0)\sqrt{s/\beta}}.$$

Let us introduce

$$A = \pm(y - x_0)\sqrt{s/\beta}.$$

Opening the absolute value we obtain

$$1 - p = pe^A + e^A$$
$$p(1 + e^A) = 1 - e^A$$
$$p = \frac{1 - e^A}{1 + e^A}.$$

Now we return to the variable w.

$$w = 3s - 3z^2 = 3s(1 - p^2) = 3s\left(1 - \left(\frac{1 - e^A}{1 + e^A}\right)^2\right).$$

Using the identity

$$1 - \left(\frac{1 - e^A}{1 + e^A}\right)^2 = \frac{1}{\cosh^2\frac{A}{2}}$$

we get

$$w = \frac{3s}{\cosh^2\left(\frac{y - x_0}{2}\sqrt{\frac{s}{\beta}}\right)}.$$

Introducing $u_0 = 3s$ and plugging in $y = x - st$ we obtain Eq. (17).

6.2. **Solving nonlinear PDEs using `fft`.** Consider a PDE of the form

$$u_t = \mathcal{L}u + \mathcal{N}(u), \tag{18}$$

where $\mathcal{L}$ is a linear differential operator, and $\mathcal{N}$ is a nonlinear operator. For example, for the Korteweg-de-Vries (KdV) equation (16), $\mathcal{L}u \equiv -u_{xxx}$, while $\mathcal{N}(u) \equiv -(u^2/2)_x$. It is convenient to perform multiplication in the x-space while differentiate in the Fourier space. We introduce a new unknown function $v(x,t)$ via the following variable change:

$$u(x,t) = e^{t\mathcal{L}}v(x,t), \tag{19}$$

where $e^{t\mathcal{L}}$ is the solution operator for the equation $u_t = \mathcal{L}u$. I.e., if $u(x,0) = u_0(x)$ then $u(x,t) = e^{t\mathcal{L}}u_0(x)$. For the KdV equation,

$$e^{t\mathcal{L}}u_0 = \texttt{ifft}(\texttt{ifftshift}(e^{ik^3t}\texttt{fftshift}(\texttt{fft}(u_0)))).$$

Plugging Eq. (19) into Eq. (18) we obtain:

$$u_t = \mathcal{L}e^{t\mathcal{L}}v + e^{t\mathcal{L}}v_t = \mathcal{L}e^{t\mathcal{L}}v + \mathcal{N}\left(e^{t\mathcal{L}}v\right).$$

Canceling $\mathcal{L}e^{t\mathcal{L}}v$ and applying the solution operator $e^{-t\mathcal{L}}$ corresponding to solving $u_t = \mathcal{L}u$ backward in time we obtain the following equation for $v(x,t)$:

$$(20)\qquad v_t = e^{-t\mathcal{L}}\mathcal{N}\left(e^{t\mathcal{L}}v\right).$$

Eq. (20) can be solved using some accurate ODE solver, e.g., the 4th order 4-stage Runge-Kutta method. The Matlab code below developed for solving the KdV equation shows how one can evaluate the right-hand side in Eq. (20). Codes in Python and Julia are available at https://hackmd.io/@NCTUIAM5804/r1sIFj8c8.

```
function KdVrkm()
% solves u_t + u_{xxx} + (0.5u^2)_x = 0, i.e.,
% u_t = -u_{xxx} - (0.5u^2)_x

init_data = 2;


N = 512;
x = linspace(-N/2,N/2,N+1);
x(N + 1) = [];
k = [-N/2 : N/2 - 1];
u = zeros(1,N);
% initial data
if init_data == 1
    u0 = 1./(cosh(x/sqrt(12))).^2;
end
if init_data == 2
    u0 = exp(-(x/20).^2) + exp(-((x+100)/20).^2);
end
dt = 0.1; % time step
figure; clf;
hpic = plot(x,u0,'LineWidth',2,'color','r'); % plot of the numerical solution
hold on;
grid
drawnow
if init_data == 1
    % plot of the exact solution for u_0(x) given by  (*)
    hp = plot(x,u0,'LineWidth',2);
    axis([-N/2 N/2 -0.01 1.01]);
end
%
tmax = 1000;
t = 0;
freq = k.*(2*pi/N); % frequencies
freq3 = freq.^3;
e3=exp(1i*freq3*dt);
while (t<tmax)
    t=t+dt;
    vk=fftshift(fft(u0)); % v in the Fourier space
    k1=rhs(0,vk);
    k2=rhs(0.5*dt,vk+0.5*dt*k1);
    k3=rhs(0.5*dt,vk+0.5*dt*k2);
    k4=rhs(dt,vk+dt*k3);
    vkn=vk+dt*(k1+2*k2+2*k3+k4)/6;
    un=ifft(ifftshift(e3.*vkn)); % return to u in the x-space
    set(hpic,'xdata',x,'ydata',real(un));
    if init_data == 1
        y = -N/2 + mod(x - t/3 + N/2,N);
        set(hp,'xdata',x,'ydata',1./(cosh((y)/sqrt(12))).^2);
        axis([-N/2 N/2 -0.01 1.01]);
    end
    u0=un;
    drawnow
end
end
%%
function rk=rhs(dt,v);
[m N]=size(v);
k=[-N/2 : N/2 - 1];
freq =k.*(2*pi/N);
freq3 = freq.^3;
e3=exp(1i*freq3*dt);
em3=exp(-1i*freq3*dt);
    vk1=v.*e3;          % e^{tL}v in the Fourier space
    v2=ifft(ifftshift(vk1));      % exp(tL)v in the x-space
    v2=0.5*v2.^2;          % [exp(tL)v]^2 in the x-space
   % exp(-tL)[[(exp(tL)v)]_x] in the Fourier space
    rk=em3.*(-1i*freq).*fftshift(fft(v2));
 end
```

7. Aliasing

Have you noticed that sometimes in the movies it seems like a car's wheels rotate in the wrong direction? The reason is aliasing. The movie's shooting speed is 24 frames per second. If a wheel rotates by an angle $\pi < \alpha < 2\pi$ between the frames your brain will process it as if it rotates in the wrong direction with the angular velocity $(\alpha - 2\pi)/24$.

A similar error due to the discrete sampling occurs when we restore a function from its Fourier transform. Let us write a function $f(x)$, $x \in [0, 2\pi]$, as an infinite series

$$f(x) = \sum_{m=-\infty}^{\infty} \alpha_m e^{imx}.$$

The term $\alpha_m e^{imx}$ is the m-th Fourier component of f. Now we consider $f_m(x) \equiv \alpha_m e^{imx}$ on the interval $[0, 2\pi]$. Let us sample it at the points $x_j := \frac{2\pi j}{N}$ and apply the N-point discrete Fourier transform to it . Then for the $\texttt{fft}(f_m)$ we have

$$[\texttt{fft}(f_m)]_k = \sum_{j=0}^{N-1} \alpha_m e^{i2\pi mj/N} e^{-i2\pi jk/N} = \sum_{j=0}^{N-1} \alpha_m e^{i2\pi j(m-k)/N}.$$

Let $m = Nq + r$, where $r \in \{0, 1, \dots, N-1\}$. Then

$$\begin{aligned}
[\texttt{fft}(f_m)]_k &= \sum_{j=0}^{N-1} \alpha_m e^{i2\pi j(m-k)/N} = \sum_{j=0}^{N-1} \alpha_m e^{i\frac{2\pi j}{N}(Nq+r-k)} = \sum_{j=0}^{N-1} \alpha_m e^{i\frac{2\pi j}{N}(r-k)} \\
&\equiv \alpha_m \sum_{j=0}^{N-1} \omega^j, \quad \text{where} \quad \omega := e^{i2\pi(r-k)/N}, \\
&= \alpha_m \begin{cases} \frac{1-\omega^N}{1-\omega}, & \omega \neq 1 \\ N, & \omega = 1 \end{cases} = \alpha_m \begin{cases} 0, & \omega \neq 1 \\ N, & \omega = 1 \end{cases}.
\end{aligned}$$

The last equality follows from the fact $\omega^N = 1$. Since $\omega = 1$ if and only if $r = k$, we have

$$[\texttt{fft}(f_m)]_k = \alpha_m N \delta_{rk},$$

i.e., frequency m higher than $N - 1$ will be aliased to a lower frequency r that correspond to the residual of the division of m by N.

Another important fact to have in mind is that the largest frequency that can be resolved without aliasing by the Fourier transform is $N/2$, not N. The reason is that the frequencies $N/2 < k < N$ are aliased to the negative frequencies between $-N/2$ and 0.

Example Suppose a 25-point discrete Fourier transform is applied to the function $f(x) = \sin(32x) = \frac{1}{2i}\left(e^{i32x} - e^{-i32x}\right)$. Then the inverse discrete Fourier transform is applied to the result. The wave numbers 32 and -32 are aliased to the wave numbers k_1 and k_2 in the range $-12 \le k_1, k_2 \le 12$. It is easy to check that $k_1 = 7$ and $k_2 = -7$. Equivalently, $k_2 = 18$, if we need to convert them to the range $0 \le k_1, k_2 \le 24$.

Example Similarly, $\sin(16x)$ aliases to $-\sin(9x)$ if 25-point DFT is used.

Exercise Suppose you apply an 32-point Fourier transform to the function $\sin(50x)$ and then restore it with the discrete inverse Fourier transform. What should be the result?

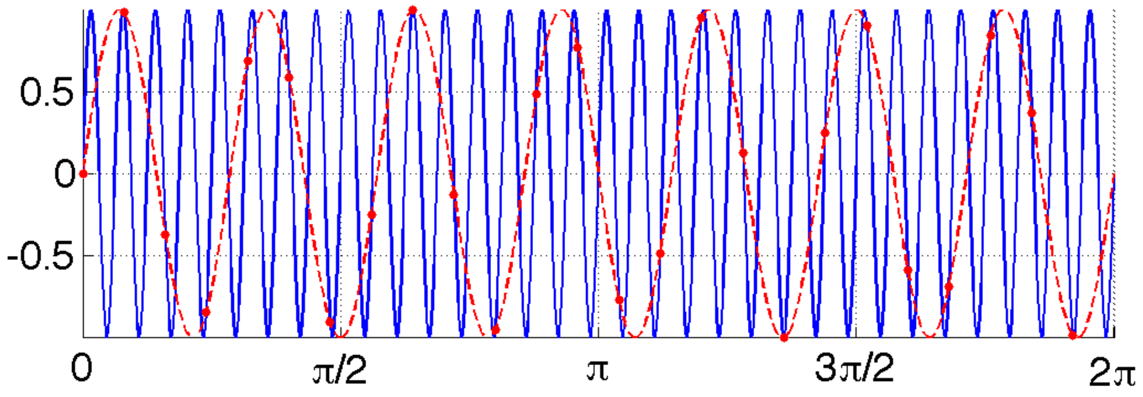

FIGURE 7. A 25-point discrete Fourier transform was applied to the function $f(x) = \sin(32x)$ (the blue curve). Then the result was restored using the inverse discrete Fourier transform. The resulting function is $g(x) = \sin(7x)$ (the red markers and the dashed curve).

8. Fast Fourier Transform

Performing a discrete Fourier transform is equivalent to matrix multiplication:

$$\mathbf{X} = \mathbf{\Omega x}, \quad \Omega_{kj} = e^{-\frac{2\pi i}{N}kj}, \quad k, j = 0, 1, \dots, N-1.$$

The computational cost of the multiplication of a complex $N \times N$ matrix by a complex vector $N \times 1$ is

$$\sim 8N^2 \quad \text{real floating point operations (or flops).}$$

This cost can be dramatically reduced e.g. if N is a power of 2. The Fourier transform performed using the trick explained below is called the Fast Fourier Transform. Its cost is

$$\sim 5N \log_2 N \quad \text{flops.}$$

Consider the discrete Fourier transform with $N = 2M$. Then we can rewrite it as

$$\begin{aligned} X(k) &= \sum_{j=0}^{N-1} x(j)\omega_N^{jk} \\ &= \sum_{j=0}^{N/2-1} \left[x(2j)\omega_N^{k(2j)} + x(2j+1)\omega_N^{k(2j+1)} \right] \\ &\equiv Y_k + \omega^k Z_k, \quad k = 0, 1, \dots, N/2-1. \end{aligned}$$

Note that Y_k and Z_k, $0 \le k \le N/2 - 1$, are the discrete Fourier transforms of the data sets $[x_0, x_2, \dots, x_{N/2-2}]$ and $[x_1, x_3, \dots, x_{N/2-1}]$. The only problem is that Y_k and Z_k are defined only for $k = 0, 1, \dots, N/2 - 1$. To define them for $N/2 \le k < N$ we observe that

$$\omega^{k+N/2} = e^{-i\frac{2\pi(k+N/2)}{N}} = e^{-i\frac{2\pi k}{N} - i\pi} = -e^{i\frac{2\pi k}{N}} = -\omega^k.$$

Hence,

$$X(k+N/2) = Y_k - \omega^k Z_k.$$

Example Let $N = 2^3 = 8$. Let $X = \texttt{fft}(x)$. Then we have

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega^8 & \omega^{10} & \omega^{12} & \omega^{14} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^{12} & \omega^{15} & \omega^{18} & \omega^{21} \\ 1 & \omega^4 & \omega^8 & \omega^{12} & \omega^{16} & \omega^{20} & \omega^{24} & \omega^{28} \\ 1 & \omega^5 & \omega^{10} & \omega^{15} & \omega^{20} & \omega^{25} & \omega^{30} & \omega^{35} \\ 1 & \omega^6 & \omega^{12} & \omega^{18} & \omega^{24} & \omega^{30} & \omega^{36} & \omega^{42} \\ 1 & \omega^7 & \omega^{14} & \omega^{21} & \omega^{28} & \omega^{35} & \omega^{42} & \omega^{49} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} \equiv \begin{bmatrix} Y_0 + Z_0 \\ Y_1 + \omega Z_1 \\ Y_2 + \omega^2 Z_2 \\ Y_3 + \omega^3 Z_3 \\ Y_0 - Z_0 \\ Y_1 - \omega Z_1 \\ Y_2 - \omega^2 Z_2 \\ Y_3 - \omega^3 Z_3 \end{bmatrix}, \tag{21}$$

where Y_k's and Z_k's are the Fourier transform of the even and odd components of x, i.e.,

$$Y = \texttt{fft}([x_0\ x_2\ x_4\ x_6]), \quad Z = \texttt{fft}([x_1\ x_3\ x_5\ x_7]).$$

More exactly,

$$Y_k = \sum_{j=0}^{3} x_{2j}(\omega^2)^{jk}, \quad Z_k = \sum_{j=0}^{3} x_{2j+1}(\omega^2)^{jk}, \quad k = 0,1,2,3.$$

To obtain the last equality in Eq. (21) we have used the following identities:

$$\omega^{(4+l)} \equiv e^{-\frac{2\pi i(4+l)}{8}} = e^{-\frac{8\pi i}{8}} e^{-\frac{2\pi i l}{8}} = -e^{-\frac{2\pi i l}{8}} \equiv -\omega^l, \quad l = 0,1,2,3,$$

$$\omega^{2k(4+l)} \equiv e^{-\frac{2\pi i 2k(4+l)}{8}} = e^{-\frac{8(2k)\pi i}{8}} e^{-\frac{2\pi i l(2k)}{8}} = e^{-\frac{2\pi i l(2k)}{8}} \equiv \omega^{l(2k)}, \quad l = 0,1,2,3.$$

Therefore, instead of doing a single 8-point Fourier transform, we can perform two 4-point Fourier transforms and construct the result of the 8-point Fourier transform out of them.

Further, we proceed recursively. For Y's in Eq. (21) we get:

$$[Y_0, Y_1] = [P_0 + Q_0, P_1 + \omega^2 Q_1], \quad [Y_2, Y_3] = [P_0 - Q_0, P_1 - \omega^2 Q_1],$$

$$[Z_0, Z_1] = [S_0 + T_0, S_1 + \omega^2 T_1], \quad [Z_2, Z_3] = [S_0 - T_0, S_1 - \omega^2 T_1],$$

where

$$[P_0, P_1] = \texttt{fft}([x_0, x_4]), \quad [Q_0, Q_1] = \texttt{fft}([x_2, x_6]),$$

$$[S_0, S_1] = \texttt{fft}([x_1, x_5]), \quad [T_0, T_1] = \texttt{fft}([x_3, x_7]).$$

For the 2-point discrete Fourier transform we have:

$$P_0 = x_0 e^{-i2\pi\cdot 0/2} + x_4 e^{-i2\pi\cdot 0/2} = x_0 + x_4, \quad P_1 = x_0 e^{-i2\pi\cdot 0/2} + x_4 e^{-i2\pi/2} = x_0 - x_4.$$

Hence,

$$P_0 = x_0 + x_4, \quad P_1 = x_0 - x_4, \quad Q_0 = x_2 + x_6, \quad Q_1 = x_2 - x_6,$$

$$S_0 = x_1 + x_5, \quad S_1 = x_1 - x_5, \quad T_0 = x_3 + x_7, \quad T_1 = x_3 - x_7.$$

Now we compute the number of flops in the Fast Fourier Transform. Let $N \equiv 2^m$. Let $F(2^m)$ be the number of flops to be found. Complex addition and subtraction require 2 flops. Complex multiplication requires 6 flops. (Check this!) Therefore,

$$F(2^m) = 2F(2^{m-1}) + 2 \cdot 2^m + 6 \cdot 2^{m-1},$$

i.e., the cost of 2^m-point Fast Fourier Transform is equal to two costs of the Fast Fourier Transform with twice as few points plus the cost of 2^m additions and subtractions for $Y_k \pm \omega^k Z_k$ plus the cost of 2^{m-1} complex multiplications $\omega^k Z_k$. Continuing, we obtain

$$\begin{aligned} F(2^m) &= 2F(2^{m-1}) + 2 \cdot 2^m + 6 \cdot 2^{m-1} \\ &= 2(2F(2^{m-2}) + 2 \cdot 2^{m-1} + 6 \cdot 2^{m-2}) + 2 \cdot 2^m + 6 \cdot 2^{m-1} \\ &= 2^2 F(2^{(m-2)}) + 2 \cdot (2 \cdot 2^m + 6 \cdot 2^{m-1}) \dots \\ &= 2^m F(0) + m(2 \cdot 2^m + 6 \cdot 2^{m-1}) = 2^m F(0) + m(2+3)2^m \\ &= 5N \log_2 N. \end{aligned}$$

Here we have taken into account that $F(0) = 0$ as the one-point DFT, i.e., `fft` of a number a is `fft`$(a) = a$, which requires no floating point operations.

In a similar manner, one can define Fast Fourier Transform for powers of other primes. The graph in Fig. 8 shows the CPU time of the `fft` in Matlab versus N.

References

[1] John P. Boyd, Chebyshev and Fourier Spectral methods, 2nd edition, Dover Publication, Inc., Mineola, New York, 2001

[2] Walter A. Strauss, Partial Differential Equations: An Introduction, 2nd edition, John Wiley and Sons, 2008

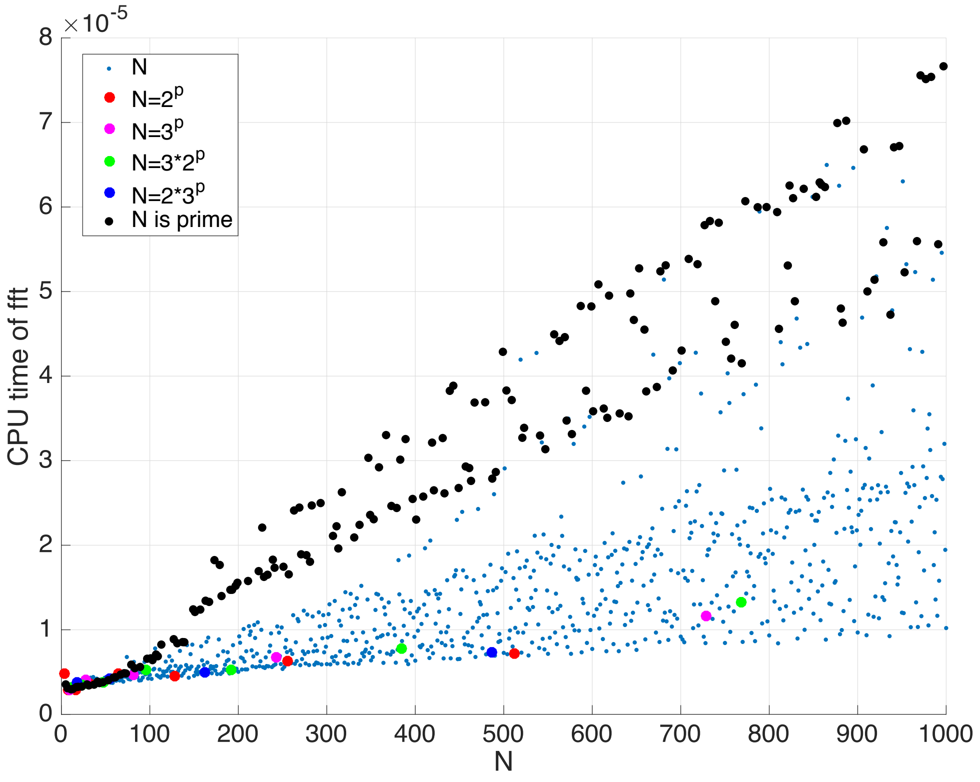

FIGURE 8. The CPU time of the `fft` in Matlab versus N.