

The Discrete Fourier Transform (Amath 571 – Bretherton):

1 Definition

Let $y_n, n = 1, \dots, N$ be a sequence of N possibly complex values. The *discrete Fourier transform* (DFT) of this sequence is the sequence $Y_m, m = 1, \dots, N$, where

$$Y_m = \sum_{n=1}^N y_n e^{-2\pi i(m-1)(n-1)/N} \quad (1)$$

The inverse transform is

$$y_n = \frac{1}{N} \sum_{m=1}^N Y_m e^{2\pi i(m-1)(n-1)/N} \quad (2)$$

Direct use of these formulas would require $O(N^2)$ flops to obtain the sequence Y_m from the sequence y_n , or vice versa. However, when N is a power of 2, a very efficient algorithm called the fast Fourier transform (FFT) can calculate the DFT in $O(N \log N)$ flops. The FFT can be generalized (with a little loss of efficiency) to all N which can be factored into products of small primes (e. g. 2, 3, and 5). Because software packages use the FFT algorithm whenever possible, the word FFT is often (loosely) used in place of DFT. In MATLAB, $Y = \text{fft}(y)$ calculates the DFT of the sequence y , and $y = \text{ifft}(Y)$ calculates the inverse DFT of Y . MATLAB uses the fast FFT algorithm only if N is a power of 2.

2 Relation to Fourier analysis

There are myriad uses of the DFT, ranging from highly accurate numerical differentiation to time series analysis to digital filtering. From our perspective, the DFT is a tool for reconstructing and manipulating a periodic function $y(x)$ of some wavelength L , based on its values at the discrete set of N equally spaced points

$$x_n = (n - 1)L/N. \quad (3)$$

The utility of the DFT is due to its close relation to the complex Fourier coefficients for $y(x)$,

$$\hat{y}_M = \frac{1}{L} \int_0^L y(x) \exp(-ik(M)x) dx, \quad M = 0, \pm 1, \pm 2, \dots \quad (4)$$

where the wavenumber associated with Fourier coefficient M is

$$k(M) = 2\pi M/L \quad (5)$$

Recall that $y(x)$ can be reconstructed from its Fourier coefficients:

$$y(x) = \sum_{M=-\infty}^{\infty} \hat{y}_M \exp(ik(M)x) \quad (6)$$

It is a well known result of Fourier analysis that the Fourier coefficients \hat{y}_M tend to zero as $|M| \rightarrow \infty$. In particular, if $y(x)$ is R times differentiable over the entire interval $[0, L]$ then the Fourier coefficients can be shown by successive integrations by parts to be $O(|M|^{-(R+1)})$. Thus a periodic step function (which is not continuous at the steps) has Fourier coefficients which are $O(|M|^{-1})$, a sawtooth (which is continuous, but whose derivative is not continuous at the teeth) has Fourier coefficients which are $O(|M|^{-2})$, and an infinitely differentiable function has Fourier coefficients which decay faster than any power of $|M|$. A good approximation to $y(x)$ can be obtained even when (6) is truncated to a finite range $M = (-N/2, N/2)$, as long as $|\hat{y}_M|^2$ is sufficiently small (in practice, 10^{-6} of the largest Fourier mode amplitude is usually OK) for all $|M| > N/2$. For smooth functions $y(x)$, the required N can be small (10 or less). For a step function, the required N may be very large.

The solid lines in the left panels of Figure 1 shows a step, sawtooth, and C^∞ function periodic over $[0, 1]$. In the right panel, the exact (analytically derived) squared amplitudes of the Fourier coefficients are denoted by '+'. This is called the *Fourier power spectrum* of $y(x)$. The analytical form of the Fourier coefficients in the three cases are:

$$\text{Step: } y(x) = \text{sgn}(x - 0.5) \quad \longleftrightarrow \quad \hat{y}_M = -2i/(\pi M), \quad M \text{ odd,}$$

$$\text{Sawtooth: } y(x) = |2x - 1| - 0.5 \quad \longleftrightarrow \quad \hat{y}_M = 2/(\pi M)^2, \quad M \text{ odd,}$$

$$\text{Swell: } y(x) = (1 - 0.6 \cos(x))^{-1} \quad \longleftrightarrow \quad \hat{y}_M = 1.25 \cdot 3^{-M}.$$

For the 'swell' function, the squared amplitudes of all Fourier coefficients for $|M| > 6$ are less than 10^{-6} of the leading Fourier coefficient, so a very accurate representation of the swell can be obtained with only a few Fourier modes.

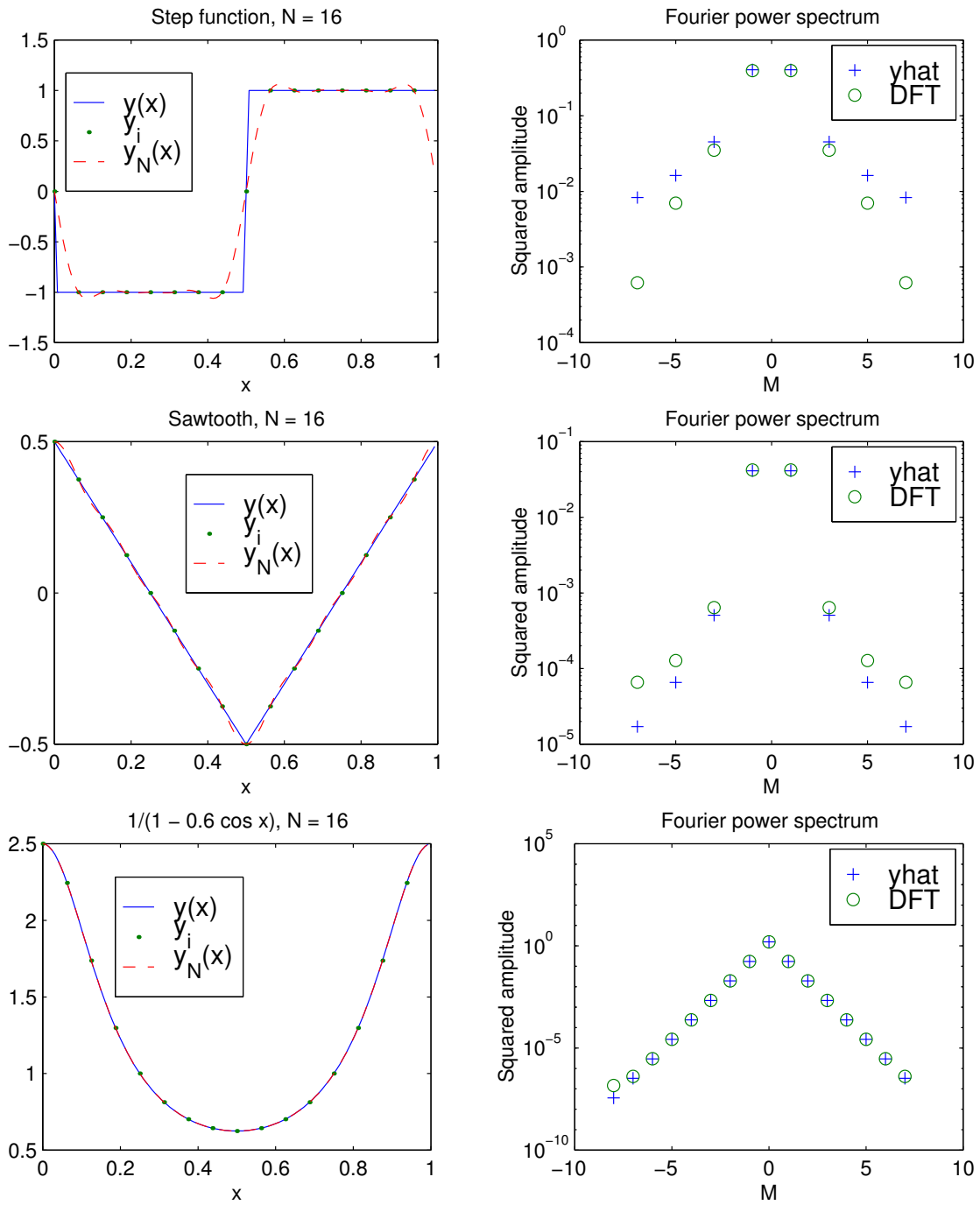


Figure 1: Left panels: Function $y(x)$ (solid), grid points y_i , and FFT-based interpolating function $y_N(x)$. Right panels: Power spectrum of Fourier components (+) and of DFT (normalized by dividing DFT amplitudes by N).

We can approximate the integral for each Fourier coefficient in (4) using a Riemann sum, noting that the spacing between grid points x_n is $\Delta x = L/N$:

$$\begin{aligned}\hat{y}_M &\approx \frac{1}{L} \sum_{n=1}^N y(x_n) \exp(-ik(M)x_n) \Delta x \\ &= \frac{1}{N} \sum_{n=1}^N y(x_n) \exp(-i[2\pi M/L][(n-1)L/N])\end{aligned}\tag{7}$$

$$= Y_m/N,\tag{8}$$

where

$$m = \text{mod}(M, N) + 1, \quad (1 \leq m \leq N).$$

Thus, each Fourier coefficient \hat{y}_M can be approximated as $1/N$ times a corresponding element of the DFT, Y_m . This approximation will be good as long as the integrand only varies slightly in a grid spacing. This requires that $y(x)$ be a smooth function of x and that the exponential $\exp(ik_M x)$ also not vary too much between grid points, i. e. that $|k_M \Delta x| (= 2\pi|M|/N) \ll 1$. If N has been chosen so that all the significant Fourier modes have $|M| \ll N/2$, these Fourier coefficients will all be well approximated with the DFT.

Hence, to interpret the DFT in terms of the Fourier expansion of a continuous function, we associated with the sequence of terms in the DFT

$$m = 1, 2, \dots, N\tag{9}$$

a sequence of M 's

$$M_m = 0, 1, 2, \dots, N/2 - 1, -N/2, \dots, -1,\tag{10}$$

and a corresponding sequence of wavenumbers $k_m = k(M_m) = 2\pi M_m/L$. Then

$$\hat{y}_M \approx Y_m/N\tag{11}$$

$$y(x) \approx y_N(x) = \sum_{M=-N/2}^{N/2-1} (Y_m/N) \exp(ik(M)x) = \sum_{m=1}^N (Y_m/N) \exp(ik_m x)\tag{12}$$

The right panel of Figure 1 compares the amplitudes of the normalized DFT coefficients Y_m/N to the amplitudes of the corresponding exact Fourier coefficients \hat{y}_M . For the small M modes, the agreement is extremely good. For larger M , the agreement is less good, especially when $y(x)$ is not very smooth. The

chain-dashed line on the left panel of Figure 1 is the DFT-based approximation $y_N(x)$ to the original function using $N = 16$ grid points. For the C^∞ swell function, it is extremely good. For the step function, ‘ringing’ near the edges of the step is evident. In general, a DFT-based approximation is much more accurate for smooth $y(x)$; if $y(x)$ is $C^R[0, L]$, $\max |y_N(x) - y(x)| = O(N^{-R})$.

3 Use of DFT for interpolation and differentiation

The DFT-based approximation $y_N(x)$ to the continuous function $y(x)$ can be used for interpolation. The matlab function `y =interpft(yn,Nintp)` evaluates an interpolating function based on the N -vector of values `yn` (assumed equally spaced through a wavelength) at `Nintp` grid points (also equally spaced through a wavelength). This is done by taking the DFT of `yn`, filling out the middle of the resulting vector with zeros to bring it to length `Nintp`, renormalizing by multiplying by `Nintp/N`, then taking an inverse DFT.

We can also approximate dy/dx at the gridpoints using the DFT. To do this, the following MATLAB algorithm can be used (assuming L and N have already been defined);

```
Y = fft(y)
M = [0:(N/2-1) (-N/2):(-1)];
k = 2*pi*M/L;
dydx = real(ifft(1i*k.*Y))
```

This algorithm is extremely accurate for smooth functions $y(x)$ well resolved by the grid, but is less accurate for functions with derivative discontinuities. This can be seen in Figure 2. The derivative of a sawtooth function is the periodic step function shown at left. The DFT estimates of its derivative oscillate around the step function. In contrast, for the C^∞ ‘swell’ function at right, the DFT gives an excellent approximation to dy/dx at the gridpoints despite the small number of gridpoints used.

Finally, note that since $L\hat{y}_0$ is the integral I of $y(x)$ over one wavelength $[0, L]$, the first DFT coefficient (which is just the sum of all the gridpoint values y_i) also gives an excellent approximation to this integral:

$$I \approx I_N = LY_1/N$$

For smooth functions, I_N converges to I faster than any power of N , so only a few grid points are required

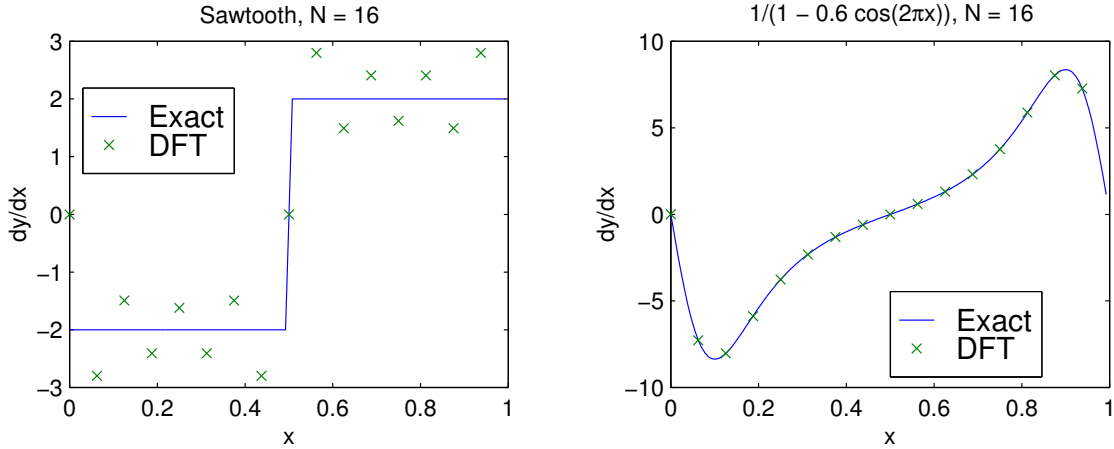


Figure 2: Exact and DFT-computed derivative of $[0, 1]$ periodic sawtooth and swell functions.

to get an excellent approximation. For instance, for the swell function with $N = 16$, $I_N = 1.2500$ while the exact integral, which can be done using complex variable theory, is $I = 1/\sqrt{1 - 0.6^2} = 5/4$, exactly the same.

4 Use of DFT for localized functions on unbounded domains

Frequently, the DFT is used for approximating localized functions on the unbounded domain $(-\infty, \infty)$. The method is to restrict $y(x)$ to a finite domain $[X_1, X_2]$ which is chosen large enough so that $y(x)$ is very small (10^{-3} or less of its maximum value) outside this domain. Then $y(x)$ is treated as periodic on this domain. For example, we can use a DFT to numerically differentiate $\exp(-x^2/2)$ in this manner (Figure 3). If we restrict the domain to $[-4, 4]$ and take $N = 16$ grid points, dy/dx has only small errors at the grid points (left) compared to the exact value $-x \exp(-x^2/2)$. The maximum error, 0.0013, occurs near $x = -4$, suggesting that it is primarily due to the domain truncation. The power spectrum of the DFT (right) falls off rapidly with $|M|$ up through $|M| = 6$, then levels off. The leveling is due to the derivative discontinuity in $y(x)$ when interpreted as a periodic function on the truncated domain. In this example, enlarging the domain somewhat would make $y(x)$ so small at the boundary that the domain truncation effects could be made totally negligible. To avoid degrading accuracy by increasing the grid spacing, correspondingly more grid points (and hence a larger FFT) would be required. In this problem, the computational effort is minute,

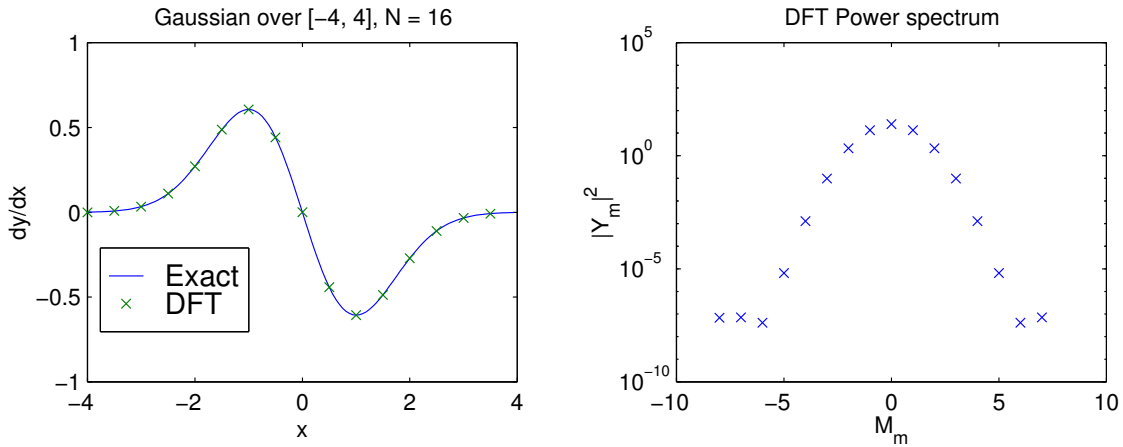


Figure 3: Left: Exact derivative of $y(x) = \exp(-x^2/2)$, and DFT-computed derivative with $N = 16$ grid points, truncating domain to $[-4, 4]$. Right: Power spectrum of the DFT of $y(x)$

but in other problems, an optimal choice of domain truncation may involve a tradeoff (for a given allowable computational effort) between wanting a large domain and wanting a sufficiently fine grid spacing.

5 Other noteworthy properties of the DFT

The DFT obeys the convolution theorem

$$\text{DFT}(w * y) = W_m Y_m \tag{13}$$

where

$$(w * y)_n = \sum_{p=1}^N w_p y_{n-p} \tag{14}$$

is the convolution of w and y . A general linear filter with weights w_p applied to a sequence y_n has the form

$$y_n^{(f)} = \sum_{p=-\infty}^{\infty} w_p y_{n-p} = (w * y)_n \tag{15}$$

If it is applied to a periodic sequence, the filter can efficiently be studied and implemented using a DFT. Hence, the convolution theorem makes the DFT a fundamental tool in digital filtering, a vast area which we will not discuss here.

If \mathbf{y} is the vectors of gridpoint values y_i , then the DFT and inverse DFT can be written in matrix form

$$\mathbf{Y} = N^{1/2} F \mathbf{y}$$

$$\mathbf{y} = N^{-1/2} F^\dagger \mathbf{Y} \quad (16)$$

where \mathbf{Y} is the DFT of \mathbf{y} , and

$$F_{mn} = N^{-1/2} \exp(-2\pi i(m-1)(n-1)) \quad (17)$$

and F^\dagger is the conjugate transpose of F . It is easily verified that F is a unitary matrix (i. e. $FF^\dagger = F^\dagger F = I$).

From this representation it is easy to derive *Parseval's theorem*

$$\begin{aligned} \sum_{m=1}^N |Y_m|^2 / N &= \mathbf{Y}^\dagger \mathbf{Y} / N \\ &= \mathbf{y}^\dagger F^\dagger F \mathbf{y} \\ &= \mathbf{y}^\dagger \mathbf{y} \\ &= \sum_{n=1}^N |y_n|^2. \end{aligned} \quad (18)$$

The DFT also has some easily derived symmetry properties that are sometimes employed to reduce the required amount of storage or computation. If y_n is real, then $Y_{N+1-m} = Y_M^*$. If in addition $y(x)$ is an even [odd] function of x (even: $y(L-x) = y(x)$, odd: $y(L-x) = -y(x)$) then its DFT is real [imaginary].