

The Discrete Fourier Transform

Frank Dellaert

7th October 2005

The Fourier transform is essential to understanding the effects of linear filters and the phenomenon of aliasing. In class we talked about three variants of the discrete Fourier transform or DFT. In all cases, the idea is to write the $N \times 1$ vector f as a linear combination of basis functions

$$f_j = \sum_k c_k U_{jk}$$

The above is the “synthesis” equation, as it synthesizes the signal out of orthogonal basis functions.

1 DFT using only Real Values

This variant is explained in Chapter 8 of “The Scientist and Engineer’s Guide to Digital Signal Processing” by Steven W. Smith, which can be downloaded for free at <http://www.dspguide.com>.

For an N -point DFT, this variant uses $N + 2$ basis functions, namely $N/2 + 1$ cosines and $N/2 + 1$ sines:

$$f_j = \sum_{k=0}^{N/2} c_k \cos(2\pi jk/N) + \sum_{k=0}^{N/2} s_k \sin(2\pi jk/N)$$

and hence the “Fourier coefficients” are the $N + 2$ values (c_k, s_k) for $k = 0..N/2$. The reason we choose sines and cosines is because they transform very simply under convolution: only their magnitude and phase will be changed. Hence, **we can understand the effect of any linear filtering operation on any function simply by understanding how it acts on the sine and cosine waves.**

However, it is annoying that we do not have N Fourier coefficients, but $N + 2$. The apparent discrepancy is explained by the fact that we always have $s_0 = s_{N/2} = 0$. Also, while synthesis is easy, analysis (the DFT) is not very elegant in this framework.

2 Matrix Form of the Real DFT

It is nicer to drop those two extra sine waves, and write the synthesis in matrix notation:

$$f = Uc$$

where the $N \times N$ matrix U consists of N basis functions as its columns, arranged in the following order:

$$\begin{aligned}
 U_{j0} &= \cos(2\pi j0/N) = 1, 1, 1, 1, \dots \\
 U_{j1} &= \cos(2\pi j1/N) \\
 U_{j2} &= \sin(2\pi j1/N) \\
 U_{j3} &= \cos(2\pi j2/N) \\
 U_{j4} &= \sin(2\pi j2/N) \\
 &\dots \\
 U_{jN} &= \cos(\pi j) = 1, -1, 1, -1, \dots
 \end{aligned}$$

This is the “two scalars and $N/2 - 1$ dials” story from class: the Fourier coefficients are the two scalars (one for the DC component, and one for the highest frequency $N/2$), and $N/2 - 1$ “complex dials” that give the magnitude and phase for frequency 1,2,3... etc up to $N/2 - 1$.

The beneficial consequence of writing the synthesis formula in matrix notation is that computing the Fourier coefficients c is now very easy:

$$c = U^{-1}f$$

This is the “discrete Fourier transform” or DFT.

3 Complex DFT

The problem with the above is that it is not symmetric: for example, there is no way to “multiply” in frequency space. To do that, we need to switch to complex numbers, for which multiplication is defined as:

$$(r_1 e^{i\phi_1})(r_2 e^{i\phi_2}) = (r_1 r_2) e^{i(\phi_1 + \phi_2)}$$

In fact, everything becomes very elegant when instead of thinking of real vectors f and coefficients c , we think of both f and c as complex N -vectors. In that case, we have

$$f = Uc$$

with the $N \times N$ “Fourier matrix” U given as

$$U_{jk} = e^{2\pi ijk/N}$$

The benefit of this form is that the inverse of the Fourier matrix now has a very simple expression

$$U^{-1} = \frac{1}{N} \bar{U}$$

where \bar{U} is the complex conjugate transpose of U . Hence, the discrete Fourier transform is given as (from “Introduction to Applied Mathematics” by Gilbert Strang):

$$c = U^{-1}f = \frac{1}{N} \bar{U}f$$

This is both the easiest and the most standard form of the DFT. It can be implemented in $O(N \log N)$ using the “Fast Fourier Transform” algorithm. However, one annoyance about this form of the DFT is that the DFT of real signals is redundant: the real part will be even, and the imaginary part odd.