The Fast Fourier Transform (FFT)

Coding the discrete Fourier transform is not particularly difficult; all one has to do is to form the matrix F_N, conjugate it and divide by N in order to obtain the matrix F_N^{-1} used to form $\mathcal{F}(Z)$ for a given input vector Z. Alternatively we can simply form, for $k = 0, 1, 2, ..., N-1$,

$$(\mathcal{F}(Z))_k = \frac{1}{N} \sum_{j=0}^{N-1} w_{-kj} z_j = \frac{1}{N} \sum_{j=0}^{N-1} e^{-i\,\frac{2\pi kj}{N}} z_j$$

without ever storing the matrix. Either way, forming the product $F_N^{-1}(Z)$ or carrying out the summations just indicated, requires a total of N^2 floating point multiplications. For large values of N, e.g. $N = 1024, 2048$ or 4096 are standard in applications, this can be a very large number of multiplications indeed; it is over one million for $N = 1024$; over 16 million for $N = 4096$.

The *Cooley - Tukey algorithm*, usually called the Fast Fourier Transform, or just FFT, enables one to substantially reduce the number of multiplications required, provided N is a power of 2; i.e., $N = 2^n$ for some positive integer n. It results in a huge increase in computational efficiency for large n and is responsible for making the FFT the widely used analytical tool that it is in current science and technology.

Assuming $N = 2^n$ we have (using the subscript on $\mathcal{F}$ here to indicate the dimension of the transform under discussion rather than the form of the transform as in our **section on convolution equations**)

$$\Big(\mathcal{F}_N(Z)\Big)_k = \frac{1}{N}(\overline{F_N}(Z))_k = \frac{1}{N} \sum_{j=0}^{N-1} e^{-i\,\frac{2\pi kj}{N}} z_j$$

$$= \frac{1}{N} \sum_{\substack{j=0 \\ j\ even}}^{N-1} e^{-i\,\frac{2\pi kj}{N}} z_j + \frac{1}{N} \sum_{\substack{j=0 \\ j\ odd}}^{N-1} e^{-i\,\frac{2\pi kj}{N}} z_j$$

$$= \frac{1}{N} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k 2\ell}{N}}\, z_{2\ell} + \frac{1}{N} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k 2\ell+1}{N}}\, z_{2\ell+1}$$

$$= \frac{1}{N} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell} + \frac{1}{N} e^{-i\,\frac{2\pi k}{N}} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell+1} = \frac{1}{2}\left(\frac{1}{N/2}\overline{F_{N/2}}(Z^0)\right)_k$$

$$+\frac{1}{2}\, e^{-i\,\frac{2\pi k}{N/2}} \left(\frac{1}{N/2}\,\overline{F_{N/2}}(Z^1)\right)_k = \frac{1}{2}\Big(\mathcal{F}_{N/2}(Z^0)\Big)_k + \frac{1}{2}\, e^{-i\,\frac{2\pi k}{N}} \Big(\mathcal{F}_{N/2}(Z^1)\Big)_k.$$

Here Z^0 is the vector of dimension $\frac{N}{2}$ consisting of the even-indexed components of Z and Z^1 is the vector of the same dimension consisting of the odd-indexed components of Z:

$$Z^0 = (z_0\, z_2\, \cdots z_{2N-2})^T; \;\; Z^1 = (z_1\, z_3\, \cdots z_{2N-1})^T.$$

This process can be repeated for each of the Fourier transforms of dimension $N/2$ shown. Thus we have

$$\left(\frac{1}{N/2}\overline{F_{N/2}}(Z^0)\right)_k = \frac{1}{N/2} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell}$$

$$= \frac{1}{N/2} \underset{\ell = 2r}{\textstyle\sum_{\ell=0}^{\frac{N}{2}-1}} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell} + \frac{1}{N/2} \underset{\ell = 2r+1}{\textstyle\sum_{\ell=0}^{\frac{N}{2}-1}} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell}$$

$$= \frac{1}{2}\left(\frac{1}{N/4} \sum_{r=0}^{\frac{N}{4}-1} e^{-i\,\frac{2\pi kr}{N/4}}\, z_{4r}\right) + \frac{1}{2} e^{-i\,\frac{2\pi k}{N/2}} \left(\frac{1}{N/4} \sum_{r=0}^{\frac{N}{4}-1} e^{-i\,\frac{2\pi kr}{N/4}}\, z_{4r+2}\right)$$

$$= \frac{1}{2}\left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{00})\right)_k + \frac{1}{2} e^{-i\,\frac{2\pi k}{N/2}} \left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{10})\right)_k$$

and also

$$\frac{1}{N/2}\Big(\overline{F_{N/2}}(Z^1)\Big)_k = \frac{1}{N/2} \sum_{\ell=0}^{\frac{N}{2}-1} e^{-i\,\frac{2\pi k\ell}{N/2}}\, z_{2\ell+1}$$

$$= \frac{1}{N/2} \sum_{\substack{\ell=0 \\ \ell = 2r}}^{\frac{N}{2}-1} e^{-i\frac{2\pi k\ell}{N/2}} z_{2\ell+1} + \frac{1}{N/2} \sum_{\substack{\ell=0 \\ \ell = 2r+1}}^{\frac{N}{2}-1} e^{-i\frac{2\pi k\ell}{N/2}} z_{2\ell+1}$$

$$= \frac{1}{N/2} \sum_{r=0}^{\frac{N}{4}-1} e^{-i\frac{2\pi kr}{N/4}} z_{4r+1} + \frac{1}{N/2} e^{-i\frac{2\pi k}{N/2}} \sum_{r=0}^{\frac{N}{4}-1} e^{-i\frac{2\pi kr}{N/4}} z_{4r+3}$$

$$= \frac{1}{2}\left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{01})\right)_k + \frac{1}{2} e^{-i\frac{2\pi k}{N/2}} \left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{11})\right)_k .$$

Putting all of this together we now have

$$\left(\frac{1}{N}\overline{F_N}(Z)\right)_k = \frac{1}{4}\left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{00})\right)_k + \frac{1}{4} e^{-i\frac{2\pi k}{N/2}} \left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{10})\right)_k$$

$$+ \frac{1}{4} e^{-i\frac{2\pi k}{N}} \left(\left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{01})\right)_k + e^{-i\frac{2\pi k}{N/2}} \left(\frac{1}{N/4}\overline{F_{N/4}}(Z^{11})\right)_k\right).$$

Here the superscripts indicate the lowest order digits in the binary expansion of the indices j of the z_j included in the $N/4$-dimensional subvector of Z to which $\frac{1}{N/4}\overline{F_{N/4}}$ is applied. Thus Z^{00} includes z_j such that j is divisible by 4; z^{01} includes z_j such that when j is divided by 4 the remainder is 1, and so on.

Now let us analyze the number of multiplications involved. In the first step,

$$(\overline{F_N}(Z))_k = (\overline{F_{N/2}}(Z^0))_k + e^{-i\frac{2\pi k}{N/2}} (\overline{F_{N/2}}(Z^1))_k,$$

a total of N multiplications with the factor $e^{-i\frac{2\pi k}{N/2}}$ are required. It should be noted, however, that on the right hand side the *subscript* k should be computed "modulo" $N/2$; that is, if $k \geq N/2$ then we should subtract $N/2$ to obtain an integer in the range 0 to $N/2 - 1$. (The latter is just an observation; it does not affect the number of multiplications.)

At the second stage we have the equations marked $(*)$ above. Here we have $2N$ multiplications, even though it seems at first glance that we should have $3N$. The reason is that the numbers $e^{-i\,\frac{2\pi k}{N/2}}$ repeat themselves when k exceeds $N/2 - 1$; there are really only $N/2$ of these numbers because if $k = N/2 + \hat{k}$ then

$$e^{-i\,\frac{2\pi k}{N/2}} = e^{-i\,2\pi}\, e^{-i\,\frac{2\pi \hat{k}}{N/2}} = e^{-i\,\frac{2\pi \hat{k}}{N/2}}.$$

If we repeat the above process, expressing the component $\left(\overline{F_N}(Z)\right)_k$ of the original transform in terms of components of successively lower order transforms, i.e., $\left(\overline{F_{N/2^m}}(Z)\right)_k$, we find that the number of multiplications required is mN. Finally when we reach $m = n$ the transform is expressed in terms of the components of the original vector Z, with $nN = \log_2 N\, N$ multiplications involved altogether. For large values of N this can be a huge reduction in the number of required multiplications as compared with the N^2 required for "brute force" computation of the original product $\overline{F_N}(Z)$. For example, if we take $n = 10$; $N = 2^{10} = 1024$, the brute force approach requires $1024^2 = 1048576$ multiplications whereas, taking advantage of the process described above, the number required is $10 \times 1024 = 10240$; more than a hundred-fold reduction.

Finally we should note $F_N(Z)$, as thus constructed, can be multiplied by $1/N$ at the end of the process to obtain $\mathcal{F}_N(Z)$, or, since $N = 2^n$, one can divide by 2 at each stage of the reduction process.

Coding the discrete Fourier transform in the FFT mode requires careful attention to indexing, avoidance of redundant multiplication, etc. We have not provided enough detail in the above exposition to indicate exactly how this would be done in practice. Nevertheless it is important to know about the Cooley-Tukey FFT algorithm because the reduction in the number of required multiplications which it permits, the mathematical essence of the FFT, is at the core of the very large number of successful applications of the discrete Fourier transform in many diverse areas of science and technology.