The Discrete Fourier Transform

Francis J. Narcowich

October 4, 2005

1 Motivation

We want to numerically approximate coefficients in a Fourier series. The first step is to see how the trapezoidal rule applies when numerically computing the integral $(2\pi)^{-1}\int_0^{2\pi} F(t)dt$, where $F(t)$ is a continuous, 2π-periodic function. Applying the trapezoidal rule with the stepsize taken to be $h = 2\pi/n$ for some integer $n \geq 1$ results in

$$(2\pi)^{-1}\int_0^{2\pi} F(t)dt \approx \frac{1}{n}\sum_{j=0}^{n-1} Y_j,$$

where $Y_j := F(hj) = F(2\pi j/n)$, $j = 1 \ldots n-1$. We remark that we made use of $Y_n = F(2\pi) = F(0) = Y_0$ in employing the trapezoidal rule to arrive at the right hand side of the equation above. Recall that the coefficients in a Fourier series expansion for a continuous, 2π-periodic function $f(t)$ have the form

$$c_k = \frac{1}{2\pi}\int_0^{2\pi} f(t)\exp(-ikt)dt.$$

We can apply the version of the trapezoidal rule derived above to approximately calculate the c_k's, since $f(t)\exp(-ikt)$ is 2π-periodic. Doing so yields

$$c_k \approx \frac{1}{n}\sum_{j=0}^{n-1} f(2\pi ij/n)\exp(-2\pi ijk/n) = \frac{1}{n}\sum_{j=0}^{n-1} y_j\overline{w}^{jk},$$

where $y_j = f(2\pi ij/n)$ and $w = \exp(2\pi i/n)$. If we replace k by $k+n$, the right hand side of the last equation is unchanged, for $\overline{w}^n = \exp(-2\pi i) = 1$. Consequently, only the approximations to c_k for $k = 0 \ldots n-1$ need be calculated. Given these approximations, however, one may recover y_j, $j = 0 \ldots n-1$. To see this, let

$$\hat{y}_k = \sum_{j=0}^{n-1} y_j \overline{w}^{jk},$$

so that $c_k \approx \hat{y}_k/n$. Multiply both sides by $w^{k\ell}$ and sum over k:

$$\sum_{k=0}^{n-1} \hat{y}_k w^{k\ell} = \sum_{j=0}^{n-1} y_j \sum_{k=0}^{n-1} w^{(\ell-j)k}.$$

The sum over k on the right can be evaluated via the algebraic identity

$$\sum_{k=0}^{n-1} z^k = \begin{cases} \dfrac{z^n - 1}{z - 1} & \text{if } z \neq 1 \\ n & \text{if } z = 1. \end{cases}$$

Recalling that $w^n = 1$, setting $z = w^{j-\ell}$ above, and noting that $w^{j-\ell} \neq 1$ unless $j = \ell$, one gets

$$\sum_{k=0}^{n-1} w^{(\ell-j)k} = \begin{cases} 0 & \text{if } j \neq \ell \\ n & \text{if } j = \ell. \end{cases}$$

Consequently, we find that

$$\frac{1}{n} \sum_{k=0}^{n-1} \hat{y}_k w^{k\ell} = y_\ell \, .$$

Thus the y's can be calculated if we know the c's or $\hat{y}$'s .

2 Definition

Let $\mathcal{S}_n$ be the set of periodic sequences of complex numbers with period n. The set $\mathcal{S}_n$ forms a complex vector space under the operations of entry-by-entry addition and entry-by-entry multiplication by a scalar. Let $y = \{y_j\}_{j=-\infty}^{\infty} \in \mathcal{S}_n$, so that $y_{j+n} = y_j$ for all j. We can associate to each y a new sequence $\hat{y}$ via

$$\hat{y}_k = \sum_{j=0}^{n-1} y_j \overline{w}^{jk}.$$

This is the same formula that we used to find $\hat{y}_k$ in §1; the only differences are that the y_j's are do not necessarily come from a continuous function, and that the index k above is not resricted to $\{0, \ldots, n-1\}$. The sequence $\hat{y}$ is periodic with period n. To see this, note that

$$\begin{aligned}
\hat{y}_{k+n} &= \sum_{j=0}^{n-1} y_j \overline{w}^{j(k+n)} = \sum_{j=0}^{n-1} y_j \overline{w}^{jk} \overline{w}^{nj} \\
&= \sum_{j=0}^{n-1} y_j \overline{w}^{jk} \quad [\overline{w}^n = e^{-(2\pi i/n)n} = 1] \\
&= \hat{y}_k
\end{aligned}$$

Put another way, $\hat{y} \in \mathcal{S}_n$. The mapping $y \in \mathcal{S}_n \mapsto \hat{y} \in \mathcal{S}_n$ defines the discrete Fourier transform. We will write $\hat{y} = \mathcal{F}[y]$. In addition, the formula derived in §1 giving y_j's in terms of $\hat{y}_k$'s certainly applies here as well. Thus, after changing the "dummy" indices, one gets this formula for y_j's in terms of $\hat{y}_k$'s:

$$y_j = \frac{1}{n} \sum_{k=0}^{n-1} \hat{y}_k w^{jk} .$$

This is the inversion formula for the DFT. We denote the inverse correspondence $\hat{y} \in \mathcal{S}_n \mapsto y \in \mathcal{S}_n$ by $y = \mathcal{F}^{-1}[\hat{y}]$.

Both $\mathcal{F}$ and $\mathcal{F}^{-1}$ are linear transformations from $\mathcal{S}_n$ to itself. Here are some additional properties that you can verify as exercises.

1. Shifts. If z is the periodic sequence formed from $y \in \mathcal{S}_n$ via $z_j = y_{j+1}$, then $\mathcal{F}[z]_k = w^k \mathcal{F}[y]_k$.

2. Convolutions. If $y \in \mathcal{S}_n$ and $z \in \mathcal{S}_n$, then the sequence defined by $[y * z]_j := \sum_{m=0}^{n-1} y_m z_{j-m}$ is also in $\mathcal{S}_n$. The sequence $y * z$ is called the *convolution* of y and z.

3. The Convolution Theorem: $\mathcal{F}[y * z]_k = \mathcal{F}[y]_k \mathcal{F}[z]_k$.

3 An application

Consider the differential equation

$$u'' + au' + bu = f(t),$$

where f is a continuous, 2π-periodic function of t. There is a well-known analytical method for finding the unique periodic solution to this equation (cf. Boyce & DiPrima, fifth edition, §3.7.2—forced vibrations), provided f is known for all t. On the other hand, if we only know f at the points $t_j = jh$, where again $h = 2\pi/n$ for some integer $n \geq 1$, this method is no longer applicable.

Instead of directly trying to work with the differential equation itself, we will work with a discretized version of it. There a many ways of discretizing; the one that we will use here amounts to making these replacements:

$$\begin{aligned} u'(t) &\longrightarrow \frac{u(t) - u(t-h)}{h}\,, \\ u''(t) &\longrightarrow \frac{u(t+h) + u(t-h) - 2u(t)}{h^2}\,. \end{aligned}$$

Replacing u' and u'' in the differential equation and setting $t = 2\pi j/n$, we get the following difference equation for the sequence $u_k = u(2\pi k/n)$:

$$u_{k+1} + \alpha u_k + \beta u_{k-1} = h^2 f_k\,,$$

where $f_k = f(2\pi k/n)$, $\alpha = bh^2 + ah - 2$, and $\beta = 1 - ah$.

Let $u \in \mathcal{S}_n$ be a solution to the difference equation derived above, and let $\hat{u} = \mathcal{F}[u]$. In addition, let $\hat{f} = \mathcal{F}[f]$. From the inversion formula for the DFT, we have

$$u_k = \frac{1}{n}\sum_{j=0}^{n-1} \hat{u}_j w^{jk} \quad \text{and} \quad f_k = \frac{1}{n}\sum_{j=0}^{n-1} \hat{f}_j w^{jk}.$$

Inserting these in the difference equation then yields, after multiplying by n,

$$\sum_{j=0}^{n-1} \hat{u}_j w^{j(k+1)} + \alpha \sum_{j=0}^{n-1} \hat{u}_j w^{jk} + \beta \sum_{j=0}^{n-1} \hat{u}_j w^{j(k-1)} = \sum_{j=0}^{n-1} h^2 \hat{f}_j w^{jk}.$$

Combining terms and doing an algebraic manipulation then results in this:

$$\sum_{j=0}^{n-1} (w^j + \alpha + \beta\overline{w}^j)\hat{u}_j w^{jk} = \sum_{j=0}^{n-1} h^2 \hat{f}_j w^{jk}.$$

Taking the inverse DFT of both sides and dividing by $w^j + \alpha + \beta\overline{w}^j$, which we assume is never 0, we find that

$$\hat{u}_j = h^2(w^j + \alpha + \beta\overline{w}^j)^{-1}\hat{f}_j\,.$$

Thus we have found the DFT of u. Inverting this then recovers u itself. In the next section we will discuss methods for fast computation of the DFT and its inverse.

4 The Fast Fourier Transform

Let us consider the DFT of a periodic sequence y with period $n = 2N$. The $\hat{y}_k$'s are calculated via

$$\hat{y}_k = \sum_{j=0}^{2N-1} y_j \overline{w}^{jk}.$$

Splitting the sum above into a sum over even and odd integers yields

$$\begin{aligned} \hat{y}_k &= \sum_{j=0}^{N-1} y_{2j}\overline{w}^{2jk} + \sum_{j=0}^{N-1} y_{2j+1}\overline{w}^{(2j+1)k} \\ &= \sum_{j=0}^{N-1} y_{2j}\bar{W}^{jk} + \overline{w}^k \left(\sum_{j=0}^{N-1} y_{2j+1}\bar{W}^{jk} \right), \end{aligned}$$

where $W := \exp(2\pi i/N) = w^2$. We may rewrite this in terms of DFT's with $n \to N$:

$$\hat{y}_k = \mathcal{F}[\{y_0\,, y_2\,, \cdots, y_{2N-2}\}]_k + \overline{w}^k \mathcal{F}[\{y_1\,, y_3\,, \cdots, y_{2N-1}\}]_k.$$

A further savings is possible. In the last equation, let $k \to k+N$ and use these facts: (1) $\mathcal{F}[y^{\text{even}}]$ and $\mathcal{F}[y^{\text{odd}}]$ both have period N. (2) $\overline{w}^{k+N} = \overline{w}^k \exp(-\pi i) = -\overline{w}^k$. The result is that for $0 \le k \le N-1$ we have

$$\left\{ \begin{array}{rl} \hat{y}_k = & \mathcal{F}[\{y_0\,, y_2\,, \cdots, y_{2N-2}\}]_k + \overline{w}^k \mathcal{F}[\{y_1\,, y_3\,, \cdots, y_{2N-1}\}]_k \\ \hat{y}_{k+N} = & \mathcal{F}[\{y_0\,, y_2\,, \cdots, y_{2N-2}\}]_k - \overline{w}^k \mathcal{F}[\{y_1\,, y_3\,, \cdots, y_{2N-1}\}]_k\,. \end{array} \right.$$

Similar formualas can be derived for the inverse DFT; they are:

$$\left\{ \begin{array}{rl} y_k = & \frac{1}{2}\left\{ \mathcal{F}^{-1}[\{\hat{y}_0\,, \hat{y}_2\,, \cdots, \hat{y}_{2N-2}\}]_k + w^k \mathcal{F}^{-1}[\{\hat{y}_1\,, \hat{y}_3\,, \cdots, \hat{y}_{2N-1}\}]_k \right\} \\ y_{k+N} = & \frac{1}{2}\left\{ \mathcal{F}^{-1}[\{\hat{y}_0\,, \hat{y}_2\,, \cdots, \hat{y}_{2N-2}\}]_k - w^k \mathcal{F}^{-1}[\{\hat{y}_1\,, \hat{y}_3\,, \cdots, \hat{y}_{2N-1}\}]_k \right\}. \end{array} \right.$$

(The factor of $\frac{1}{2}$ appears because the inversion formula has a "$1/n$" in it.)

What is the computational "cost" of using the formulas above versus ordinary matrix methods, where there are $4n^2$ real multiplications used in the computation? Set $n = 2^L$ and let K_L be the number of real multiplications required to compute $\mathcal{F}[y]$ by the method above. From the formulas derived above, one sees that to compute $\mathcal{F}[y]$, one needs to compute $\mathcal{F}[y^{\text{even}}]$ and $\mathcal{F}[y^{\text{odd}}]$. This takes $2K_{L-1}$ real multiplications. In addition, one must multiply $\overline{w}^k$ and $\mathcal{F}[y^{\text{odd}}]_k$, for $k = 0, \ldots, 2^{L-1} - 1$, which requires $4 \times 2^{L-1}$ real multiplications. The result is that K_L is related to K_{L-1} via

$$K_L = 2K_{L-1} + 2^{L+1}$$

When $L = 0$, $n = 2^0 = 1$ and no multiplications are required; thus, $K_0 = 0$. Inserting $L = 1$ in the last equation, we find that $K_1 = 1 \times 2^2$. Similarly, setting $L = 2$ then yields $K_2 = 2 \times 2^3$. Similarly, one finds that $K_3 = 3 \times 2^4$, $K_4 = 4 \times 2^5$, and so on. The general formula is $K_L = L \times 2^{L+1} = 2L \times 2^L$. Again setting $n = 2^L$ and noting that $L = \log_2 n$, we see that the number of real multiplications required is $2n \log_2 n$.

To get an idea of how much faster than matrix multiplication this method is, suppose that we want to take the DFT of data with $n = 2^{12} = 4,096$ points. The conventional method requires $2^{26} \approx 7 \times 10^7$ real multiplications. Using the FFT method to get the DFT requires $2 \times 2^{12} \times 12 \approx 10^5$ real multiplications, making the FFT roughly 700 times as fast.

We remark that similar algorithms can be obtained when $n = N_1 N_2 \cdots N_m$, although the fastest one is obtained in the case discussed above. For a discussion of this and related topics, one should consult the references below.

References

[1] J. W. Cooley and J. W. Tukey, "An Algorithm for Machine Computation of Complex Fourier Series," Math. Comp. **19** (1965), 297-301.

[2] Folland, G. B., *Fourier analysis and its applications,* Wadsworth & Brooks/Cole, Pacific Grove, CA, 1992.

[3] Marchuk, G. I., *Methods of numerical mathematics,* Springer-Verlag, Berlin, 1975.

[4] Ralston, A. and Rabinowitz, P. *A first course in numerical analysis,* McGraw-Hill, New York, 1978.