Interpreting Discrete Fourier Transforms

Abstract

We present a systematic way of interpreting the discrete Fourier transform as an approximation to the continuous Fourier transform. The argument is made for general conventions, and several examples are given.

1 Introduction

Often times it is necessary to approximate a continuous Fourier transform (CFT) with a discrete Fourier transform (DFT) when an analytic form of the data is not available. For example, a DFT can be performed on a sampling of time-ordered electric field measurements to approximate the frequency spectrum of the field. In order to perform these transforms, we utilize the numerical packages of programming languages such as IDL, MATLAB, or Mathematica. However, the definition of a Fourier transform is not completely constrained, and freedoms exist that give rise to multiple conventions in the normalization[1], and scaling of the Fourier domain variables. In particular, the numerical package being used may not follow the desired convention. Below we outline a method for interpreting the results of a DFT, in terms of the CFT desired. Then we provide several examples of common conventions.

2 Derivation

We start by assuming we have a continuous function, $f_c(x)$, of which we would like to compute a continuous Fourier transform of the form

$$F_c(u; C_c, p_c) = C_c \int_{-\infty}^{\infty} dx \; f_c(x) e^{i p_c u x} \tag{1}$$

where the constants C_c and p_c are determined by convention of the transform. However, in reality we often have a discrete sample of the continuous function, $f_d(j)$, where j indexes the samples and runs from 0 to $N-1$, N being the number of samples. Then it is necessary to use a discrete Fourier transform. Numerical packages have various conventions for their DFTs, but in general they can be expressed in the form

$$F_d(k; C_d, p_d) = C_d \sum_{j=0}^{N-1} f_d(j) e^{i p_d j k} \tag{2}$$

where k indexes the Fourier domain samples, and also runs from 0 to $N-1$. In some cases, the indices can be shifted. For example, MATLAB counts arrays

[1] There are of course constraints limiting these freedoms, but for this argument we will leave them completely free.

from 1 rather than 0, so the sum would run from 1 to N, and the values in the exponent are adjusted accordingly. For simplicity, we assume counting from 0. The translation is straighforward, and an example is given in Section **??**. We wish to approximate the CFT with the DFT.

In order to proceed, we assume the function $f_c(x)$ is only significantly nonzero within a domain $\mathcal{D}$. Furthermore, we will assume $\mathcal{D} = [0, x_{max}]$. This constraint is not actually necessary, and a translation of the domain results in a phase in the Fourier transform, as seen in Section **??**. Then we can relate the indices in Eq. (**??**) to the axis values with the following expressions:

$$x(j) = j\Delta x \tag{3a}$$
$$u(k) = k\Delta u \tag{3b}$$

where $\Delta x = x_{max}/(N-1)$, and Δu will be derived below. We can now approximate the CFT with a finite sum.

$$\begin{aligned} F_c(u; C_c, p_c) &\approx C_c \int_0^{x_{max}} dx\ f_c(x) e^{ip_c u x} \\ &\approx C_c \sum_{j=0}^{N-1} \Delta x f_c(x(j)) e^{ip_c u j \Delta x} \\ &= C_c \Delta x \sum_{j=0}^{N-1} f_d(j) e^{ip_c u j \Delta x} \end{aligned} \tag{4}$$

In order to force the sum in Eq. (**??**) to the form of the DFT (Eq. (**??**)), we write $u(k)$ in the following way.

$$\begin{aligned} u(k) = k\Delta u &= \frac{p_d k}{p_c \Delta x} \\ \Rightarrow \Delta u &= \frac{p_d}{p_c \Delta x} \end{aligned} \tag{5}$$

This expression for Δu defines our Fourier axis, which is an extremely important part of interpreting a Fourier transform and in particular comparing to other computational conventions.

We can now simplify Eq. **??**.

$$\begin{aligned} F_c(u; C_c, p_c) &\approx C_c \Delta x \sum_{j=0}^{N-1} f_d(j) e^{ip_d j k} \\ &= \frac{C_c}{C_d} \Delta x F_d(k; C_d, p_d) \\ &= \frac{C_c}{C_d} \Delta x F_d(u/\Delta u; C_d, p_d) \end{aligned} \tag{6}$$

In the first line of Eq. **??** we introduced an intermediate index $k = u/\Delta u$ for clarity in replacing the sum with the DFT. Of course any computer package will compute the DFT for integer values of k, and it should be noted that this approximation for the CFT is only used for u equal to integer multiples of Δu.

To summarize, we have related an arbitrary convention of the continuous Fourier transform to an arbitrary convention of the discrete Fourier transform (Eq. **??**). In addition, we have shown the resulting Fourier axis is described by Eq. **??**. For reference, several examples are discussed in Section **??**.

3 Shifting Domains

The above derivation applied to functions significantly non-zero only within a domain $\mathcal{D} = [0, x_{max}]$. We can relax that condition slightly by allowing $\mathcal{D} \to [x_{min}, x_{max}]$. The same process can be followed as in Section **??**, with a simple translation of the variable of integration. The result is a mode-dependent phase, and we can generalize Eq. **??** as

$$F_c(u; C_c, p_c) \approx \frac{C_c}{C_d} \Delta x e^{ip_c u x_{min}} F_d(u/\Delta u; C_d, p_d) \tag{7}$$

where now $\Delta x = (x_{max} - x_{min})/(N-1)$.

A particular case of interest is to shift the real and Fourier domains such that they are centered at zero. This can be achieved by a couple shifts of the arrays. In this case, $x_{min} = -\Delta x(N-1)/2$. The phases in Eq. **??** are now negative, but can be shifted properly to make them positive. This amounts to exchanging the positive and negative x data in $f_d(j)$. This is illustrated in Step A in Fig. **??**.

Once the DFT is performed, our approximation for $F_c(u)$ is still only valid for positive (or in some cases only negative) values of u. But we note that by the periodic nature of a complex phase, for $k > (N-1)/2$, and for $p_d = 2\pi/N$ (as is normally the case), we can make the substituion $k \to -N + k$ and the sum in Eq. **??** remains unchanged. In other words, we can relabel and shift our u axis to center our Fourier axis on $u = 0$, as shown in Steps C and D in Fig. **??**. All these operations can be easily implemented using shifting functions built into many software packages. In MATLAB, for example, `ifftshift` (Step A) and `fftshift` (Step D) are provided for this very reason.

4 Examples

Here we apply the interpretation of Section **??** specifically to the default MATLAB discrete Fourier transform. Further examples are listed in Table **??**. For the forward DFT, MATLAB uses the convention

$$F_d(k) = \sum_{j=1}^{N} f_d(j) e^{-2\pi i(j-1)(k-1)/N},$$

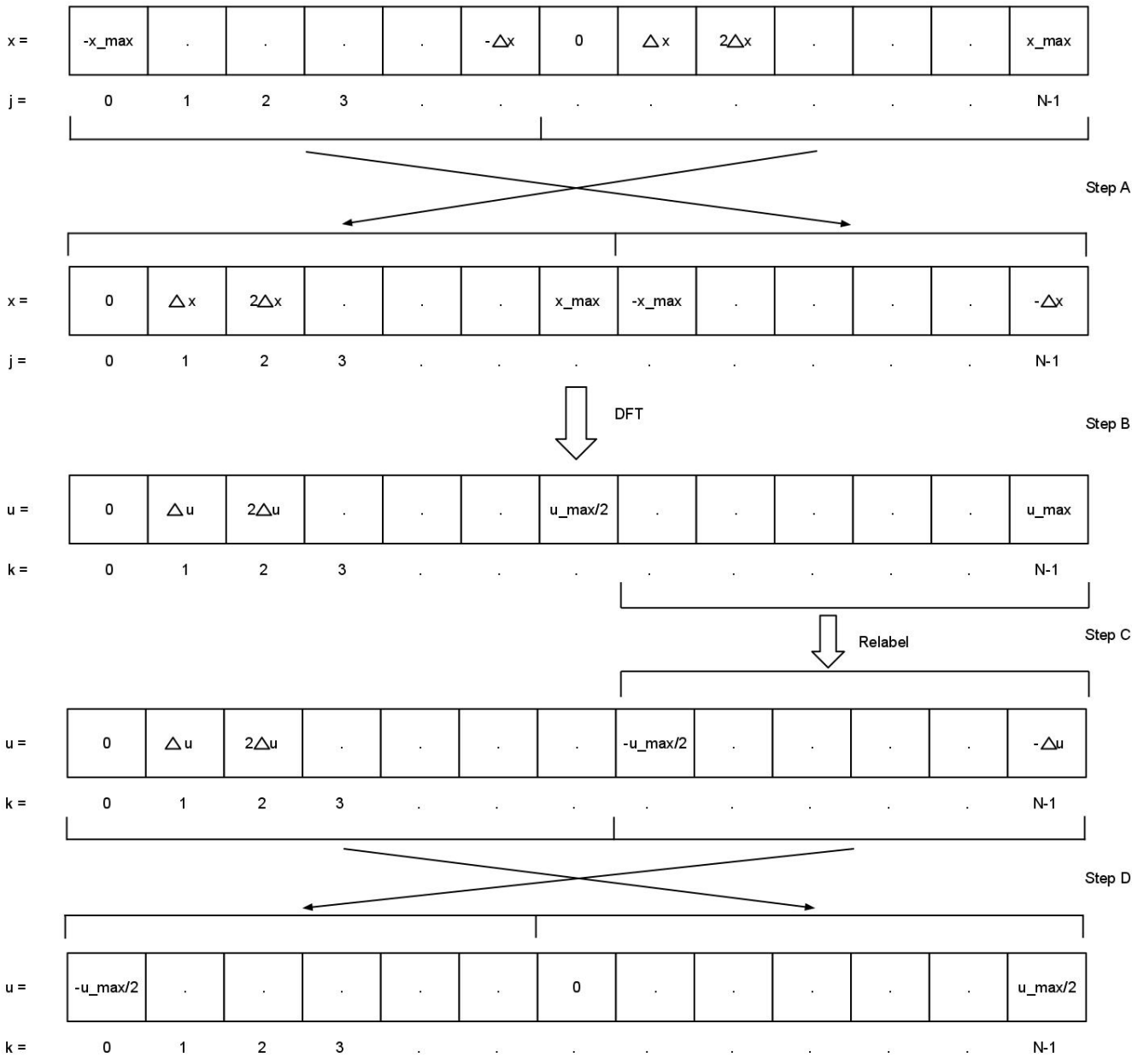

Figure 1: Centering axes on 0. *Step A*: Shift the real axis to be fed into DFT package. *Step B*: Perform discrete Fourier transform. *Step C*: Relabel Fourier modes to equivalent negative modes. *Step D*: Shift Fourier axis to recover correct order.

Package	$F_d(k)$	C_d	p_d	$x(j)$	$u(k)$
IDL	$\frac{1}{N}\sum_{j=0}^{N-1} f_d(j)e^{-2\pi ijk/N}$	$1/N$	$-2\pi/N$	$j\Delta x$	$-k\frac{2\pi}{p_c N\Delta x}$
MATLAB	$\sum_{j=1}^{N} f_d(j)e^{2\pi i(j-1)(k-1)/N}$	1	$-2\pi/N$	$(j-1)\Delta x$	$-(k-1)\frac{2\pi}{p_c N\Delta x}$
Mathematica	$\frac{1}{\sqrt{N}}\sum_{j=1}^{N} f_d(j)e^{2\pi i(j-1)(k-1)/N}$	$\frac{1}{\sqrt{N}}$	$2\pi/N$	$(j-1)\Delta x$	$(k-1)\frac{2\pi}{p_c N\Delta x}$
Python (numpy)	$\sum_{j=0}^{N-1} f_d(j)e^{-2\pi ijk/N}$	1	$-2\pi/N$	$j\Delta x$	$-k\frac{2\pi}{p_c N\Delta x}$

Table 1: Example default parameters for the forward discrete Fourier transform for various numerical packages

so that $C_d = 1$ and $p_d = -2\pi/N$. Note the shift in indices due to the fact that MATLAB counts from 1, rather than zero. This can be accounted for by writing the axes as

$$x(j) = (j-1)\Delta x$$
$$u(k) = (k-1)\Delta u$$

where $\Delta u = p_d/(p_c\Delta x) = -2\pi/(p_c N\Delta x)$.

Let us now assume that we have sampled a function of position, $f_d(j)$, with resolution $\Delta x = 3\text{m}$, and $N = 1001$ ($x_{range} = 3000\text{m}$). We are interested in the Fourier transform of the form

$$F_c(u) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} dx\ f_c(x)e^{iux}$$

so that $C_c = 1/\sqrt{2\pi}$ and $p_c = 1$. Then we use Eq. **??** to approximate the CFT as

$$F_c(u) \approx \frac{3\text{m}}{\sqrt{2\pi}}F_d(-u\frac{3003\text{m}}{2\pi} + 1; 1, 2\pi/N)$$

where $F_d(k, 1, 2\pi/N)$ is the array output from MATLAB's `fft` function, given $f_d(j)$. Note that because the sign of p_c is opposite the sign of p_d, the approximation for the CFT will only be valid for negative Fourier modes. In most applications this difference will not be significant, but in some cases it may be important to account for the axis reversal.

Similar steps can be taken for any DFT convention, and some common default parameters are listed in Table **??**. We also list several common CFT conventions and their corresponding conversions from common numerical packages in Table **??**.

DFT Package	$F_c(u)$ Desired	Amplitude Factor	$\lvert\Delta u\rvert$
IDL	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{iux}$	$\Delta x N/\sqrt{2\pi}$	$2\pi/(N\Delta x)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{iux}$	$\Delta x N/2\pi$	$2\pi/(N\Delta x)$
	$\int dx\ f_c(x)e^{2\pi iux}$	$\Delta x N$	$1/(N\Delta x)$
MATLAB	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{iux}$	$\Delta x/\sqrt{2\pi}$	$2\pi/(N\Delta x)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{iux}$	$\Delta x/2\pi$	$2\pi/(N\Delta x)$
	$\int dx\ f_c(x)e^{2\pi iux}$	Δx	$1/(N\Delta x)$
Mathematica	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{iux}$	$\Delta x\sqrt{N/2\pi}$	$2\pi/(N\Delta x)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{iux}$	$\Delta x\sqrt{N}/2\pi$	$2\pi/(N\Delta x)$
	$\int dx\ f_c(x)e^{2\pi iux}$	$\Delta x\sqrt{N}$	$1/(N\Delta x)$
Python (numpy)	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{iux}$	$\Delta x/\sqrt{2\pi}$	$2\pi/(N\Delta x)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{iux}$	$\Delta x/2\pi$	$2\pi/(N\Delta x)$
	$\int dx\ f_c(x)e^{2\pi iux}$	Δx	$1/(N\Delta x)$

Table 2: Common CFT conventions listed with the normalization factor and axis scaling factors for sample numerical packages.

5 Inverse Fourier Transform

The above argument holds for the inverse Fourier transform as well. Define the inverse CFT as

$$f_c(x;C_c',p_c') = C_c'\int_{-\infty}^{\infty} du\ F_c(k)e^{ip_c'ux} \tag{8}$$

and the inverse DFT as

$$f_d(j;C_d',p_d') = C_d'\sum_{k=0}^{N-1} F_d(k)e^{ip_d'jk}. \tag{9}$$

Then the result for the approximation of the inverse CFT is

$$f_c(x;C_c',p_c') \approx \frac{C_c'}{C_d'}\Delta u f_d(x/\Delta x;C_d',p_d') \tag{10}$$

where $\Delta x = p_d'/p_c'\Delta u$.

We summarize the default parameters for the inverse DFT for various numerical packages in Table **??**, and provide a few applied examples to common CFTs in Table **??**.

Package	$f_d(j)$	C'_d	p'_d	$u(k)$	$x(j)$
IDL	$\sum_{k=0}^{N-1} F_d(k)e^{2\pi ijk/N}$	1	$2\pi/N$	$k\Delta u$	$j\frac{2\pi}{p'_c N\Delta u}$
MATLAB	$\frac{1}{N}\sum_{k=1}^{N} F_d(k)e^{-2\pi i(j-1)(k-1)/N}$	$1/N$	$2\pi/N$	$(k-1)\Delta u$	$(j-1)\frac{2\pi}{p'_c N\Delta u}$
Mathematica	$\frac{1}{\sqrt{N}}\sum_{k=1}^{N} F_d(k)e^{-2\pi i(j-1)(k-1)/N}$	$\frac{1}{\sqrt{N}}$	$-2\pi/N$	$(k-1)\Delta u$	$-(j-1)\frac{2\pi}{p'_c N\Delta u}$
Python (numpy)	$\frac{1}{N}\sum_{k=0}^{N-1} F_d(k)e^{2\pi ijk/N}$	$1/N$	$2\pi/N$	$k\Delta u$	$j\frac{2\pi}{p'_c N\Delta u}$

Table 3: Example default parameters for the inverse discrete Fourier transform for various numerical packages

DFT Package	$f_c(x)$ Desired	Amplitude Factor	$\lvert\Delta x\rvert$
IDL	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{-iux}$	$\Delta u/\sqrt{2\pi}$	$2\pi/(N\Delta u)$
	$\int dx\ f_c(x)e^{-iux}$	Δu	$2\pi/(N\Delta u)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{-2\pi iux}$	$\Delta u/2\pi$	$1/(N\Delta u)$
MATLAB	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{-iux}$	$\Delta uN/\sqrt{2\pi}$	$2\pi/(N\Delta u)$
	$\int dx\ f_c(x)e^{-iux}$	ΔuN	$2\pi/(N\Delta u)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{-2\pi iux}$	$\Delta uN/2\pi$	$1/(N\Delta u)$
Mathematica	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{-iux}$	$\Delta u\sqrt{N/2\pi}$	$2\pi/(N\Delta u)$
	$\int dx\ f_c(x)e^{-iux}$	$\Delta u\sqrt{N}$	$2\pi/(N\Delta u)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{-2\pi iux}$	$\Delta u\sqrt{N}/2\pi$	$1/(N\Delta u)$
Python (numpy)	$\frac{1}{\sqrt{2\pi}}\int dx\ f_c(x)e^{-iux}$	$\Delta uN/\sqrt{2\pi}$	$2\pi/(N\Delta u)$
	$\int dx\ f_c(x)e^{-iux}$	ΔuN	$2\pi/(N\Delta u)$
	$\frac{1}{2\pi}\int dx\ f_c(x)e^{-2\pi iux}$	$\Delta uN/2\pi$	$1/(N\Delta u)$

Table 4: Common inverse CFT conventions listed with the normalization factor and axis scaling factors for sample numerical packages.