Interpolating Data with the Discrete Fourier Transform

Ken Huffman

December 13, 2005

Abstract

The Fourier transform says that any function can be approximated with an infinite series of sines and cosines. But often we need to approximate data that does not fit our conventional idea of a function. In this paper we will derive the discrete Fourier transform, interpolate a data set, and examine few of its applications.

History

The history behind the Fourier transform is really quite fascinating, and one which most students can relate to. According to [1] Jean Baptiste Joseph Fourier presented the first version of his paper on the theory of heat conduction to the Paris Academy in 1807. The academy's judges included such notable names as Lagrange, Laplace, and Legrande. The judges were quite unimpressed, and suggested that Fourier refine his work and resubmit it for the 1812 contest. In 1812 Fourier resubmitted his paper and was awarded the grand prize. However, the judges were still not impressed with Fourier's work. Stating,"the way in which the author arrives at his equations is not exempt from difficulties, and his analysis still leaves something to be desired, be it in generality, or be it even in rigor" [1]. Despite its less than impressive start, Fourier analysis has had a major impact in mathematics and its applications. In fact, some of the most famous names in science and mathematics built their careers on the Fourier analysis. Including Riemann, Lebesgue, and Carl Friedrich Gauss.

Fourier analysis has been used to solve some ancient problems. Including the movements of the planets, and a vibrating string attached at both ends. However, Fourier analysis is not relegated to the past. In 1965 Cooley and Tukey discovered the fast Fourier transform, which has led to breakthroughs in data transmission and compression, as well as numerous other applications in technology.

Deriving the DFT

You may be wondering, what is the difference between the discrete Fourier transform and the Fourier transform? In 1812 Fourier stated that any function can be written as an infinite series of sines and cosines.

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} [a_k \cos(kx) + b_k \sin(kx)] \tag{1}$$

However, many times our function is defined only on a closed interval, or we may only know certain discrete values that lie along the function. In these cases we don't have enough information for an infinite series. The real question becomes, how do we approximate a Fourier transform with this limited information? If we assume we have a function f of a real variable x, we know that it is absolutely integrable on the real line

$$\int_{-\infty}^{\infty} |f(x)| dx < \infty. \tag{2}$$

We can use the Fourier transform to approximate the function f, which states that

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi\omega x}\, dx, \tag{3}$$

where

$$e^{\pm i2\pi\omega x} = \cos(2\pi\omega x) \pm i \sin(2\pi\omega x). \tag{4}$$

If we assume that our function is defined on some interval of length A centered at the origin we can rewrite equation (3), giving us

$$\hat{f}(\omega) = \int_{-\frac{A}{2}}^{\frac{A}{2}} f(x)e^{-i2\pi\omega x}\, dx. \tag{5}$$

We can approximate this integral numerically using the trapezoid rule. If we begin by dividing the interval A into N equally spaced subintervals of length Δx, we see that

$$A = N\Delta x. \tag{6}$$

Rearranging the terms gives us

$$\Delta x = \frac{A}{N}. \tag{7}$$

The trapezoid rule states that

$$\int_{-\frac{A}{2}}^{\frac{A}{2}} g(x)dx \approx \frac{\Delta x}{2}\left\{ g\left(-\frac{A}{2}\right) + 2\sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}-1} g(x_n) + g\left(\frac{A}{2}\right)\right\}. \tag{8}$$

In order to approximate this function using the trapezoid rule, we will make two assumptions:

1. N is an even number.
2. $g(-\frac{A}{2}) = g(\frac{A}{2})$.

Using this last assumption we can simplify equation (8), giving us

$$\int_{-\frac{A}{2}}^{\frac{A}{2}} g(x)dx \approx \Delta x \sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} g(x_n). \tag{9}$$

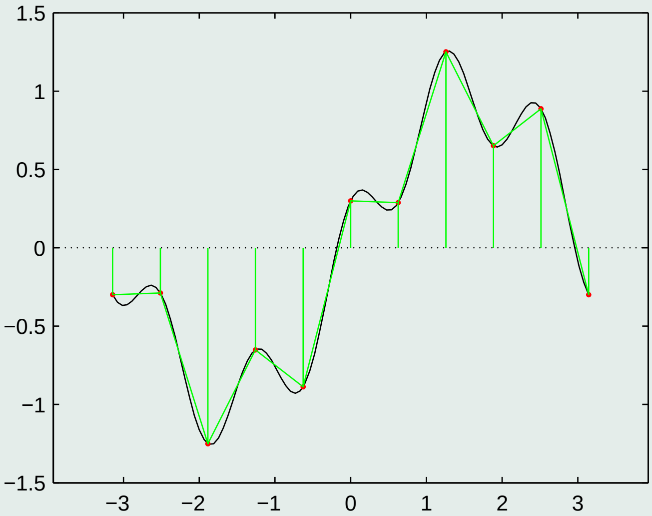

Figure 1: A periodic function $g(x)$ with $g(-\frac{A}{2}) = g(\frac{A}{2})$ and its trapezoidal approximation.

Using the trapezoid rule to approximate the function $\hat{f}$, we see that

$$\hat{f} = \frac{A}{N} \sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} f(x_n) e^{-i2\pi\omega x_n}. \tag{10}$$

The discrete Fourier transform has allowed us to solve one problem, by approximating a discrete solution rather than computing an infinite series. However, in the process we have created a new problem. Equation (10) is valid for all ω, so which frequencies are we supposed to use?

Reciprocity Relations

According to [1] the reciprocity relations are the cornerstone of the DFT. Our function is currently defined on the spatial domain (Figure 2) $[-A/2, A/2]$ and has a grid spacing Δx. This tells us that our grid points are located at the points $x_n = n\Delta x$. According to [1] there is a frequency domain (Figure 2) $[-\Omega/2, \Omega/2]$ that is closely associated with our spatial domain. "This frequency domain will also be equipped with a grid consisting of N equally spaced points separated by a distance $\Delta\omega$"[1]. Therefore, we know that the grid points within our frequency domain are given by the formula $\omega_k = k\Delta\omega$. According to [1] we need to "Imagine all modes (sines and cosines) that have an integer number of periods on $[-A/2, A/2]$ and fit exactly on the interval. Of these waves, consider the wave with the largest possible period. . . . This wave has a frequency of $1/A$ periods per-unit length. This frequency is the *lowest* frequency associated with the interval $[-A/2, A/2]$". We can use this frequency as our fundamental unit for dividing our frequency domain, giving us

$$\Delta\omega = \frac{1}{A}. \tag{11}$$

All of the other frequencies ω will be integer multiples of $\Delta\omega$. In the same way we related the interval A to the product of Δx and N, we can relate the interval Ω to the product of $\Delta\omega$ and N, giving us

$$\Omega = N\Delta\omega. \tag{12}$$

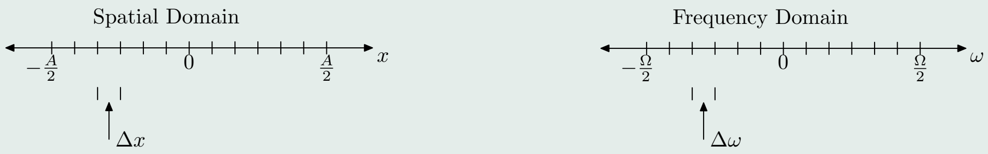

Figure 2: Relationship between the spatial and frequency domains.

Combining equations (11) and (12), we have our first reciprocity relation

$$A\Omega = N. \tag{13}$$

At first glance you may not see the importance of this relationship. However, this relationship states that the lengths of the spatial domain and the frequency domain vary inversely with each other. In order to find our second reciprocity relation we can combine equations (6) and (11), giving us

$$\frac{1}{\Delta\omega} = A = N\Delta x, \tag{14}$$

or

$$\Delta x \Delta\omega = \frac{1}{N}. \tag{15}$$

This relationship shows that the grid spacings in the two domains are inversely related. According to [1] "The first relation tells us that if the number of grid points N is held fixed, an increase in the length of the spatial domain comes at the expense of a decrease in the length of the frequency domain. ... The second reciprocity relation can be interpreted in a similar way. Halving Δx with N fixed also halves the length of the spatial domain". If we return to our approximation with the trapezoid rule, equation (10), we have a term ωx_n in our exponent. If we let f_n be the sampled values $f(x_n)$ for $n = -N/2+1 : N/2$. Then we can approximate $\hat{f}$

at the frequency grid points $\omega_k = k/A$. We can use our reciprocity relations to show that

$$x_n \omega_k = (n\Delta x)(k\Delta\omega) \tag{16}$$
$$= \frac{nA}{N}\frac{k}{A} \tag{17}$$
$$= \frac{nk}{N}. \tag{18}$$

We can substitute this value into equation (10), leaving us with

$$\hat{f} = \frac{A}{N}\sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} f_n e^{-i2\pi nk/N}. \tag{19}$$

Using this formula we can compute the N coefficients F_k. The coefficients are given by the formula

$$F_k = \frac{1}{N}\sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} f_n e^{-i2\pi nk/N}. \tag{20}$$

For purposes of this project we will assume that all the functions we will be approximating will consist of only real valued inputs. However, the coefficients F_k will consist of both real and complex coefficients. According to [1] we can compute the real coefficients using the formula

$$Re\{F_k\} = \frac{1}{N}\sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} f_n \cos\left(\frac{2\pi nk}{N}\right). \tag{21}$$

Similarly, we can compute the complex coefficients using the formula

$$Im\{F_k\} = \frac{1}{N}\sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} -f_n \sin\left(\frac{2\pi nk}{N}\right). \tag{22}$$

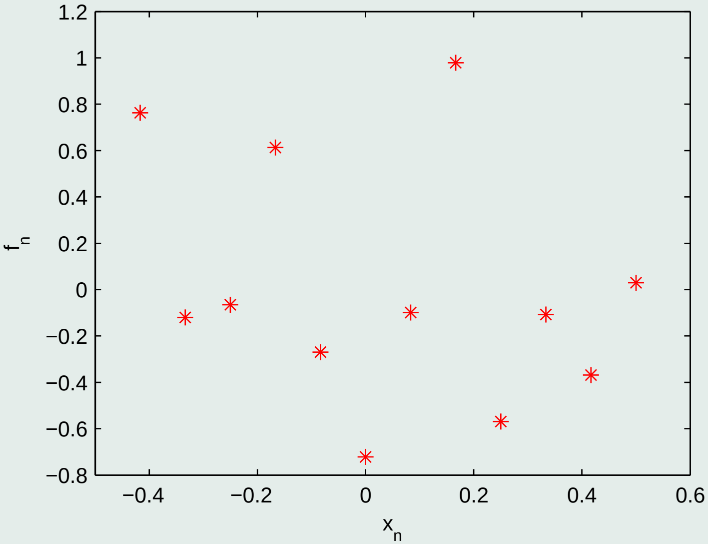

Figure 3: Plot of the Data Points Collected

Up until now we've seen a lot of equations and not much practical application, so let's do an example. Given the twelve equally spaced data points,

n, k	x_n	$Re\{f(x_n)\}$
-5	$-5/12$	0.7630
-4	$-4/12$	-0.1205
-3	$-3/12$	-0.0649
-2	$-2/12$	0.6133
-1	$-1/12$	-0.2697
0	0	-0.7216
1	$1/12$	-0.0993
2	$2/12$	0.9787
3	$3/12$	-0.5689
4	$4/12$	-0.1080
5	$5/12$	-0.3685
6	$6/12$	0.0293

(23)

find a trigonometric function that passes through all twelve points. Looking at Figure 3 which is a plot of these data points. It is difficult, if not impossible, to imagine a function which could pass through all twelve points. However, we can use the DFT to find this function. We can begin computing the real DFT coefficients using equation (21) and substituting $N = 12$ into the summation, giving us

$$Re\{F_k\} = \frac{1}{12} \sum_{n=-5}^{6} f_n \cos\left(\frac{2\pi nk}{12}\right). \tag{24}$$

As you can probably guess this would be tedious calculations if we had to do this by hand. However, this is a very simple calculation for a computer program. Using Matlab we can create a simple program to calculate the coefficients.

```
n=-5:6;
Re_Fn=[.7630 -.1205 -.0649 .6133 -.2697 -.7216 -.0993 ...
    .9787 -.5689 -.1080 -.3685 .0293]';
Re_Fk=zeros(size(n));
Im_Fk=zeros(size(n));
z=zeros(size(n));
for k=1:12
    for i=1:12
        z(i)=(1/12)*(Re_Fn(i))*cos(2*pi*n(i)*n(k)/12);
    end
    z=z';
    Re_Fk(k)=sum(z);
end
Re_Fk=Re_Fk'
```

Similarly, we can calculate the imaginary coefficients using equation (22) and again substituting $N = 12$, giving us

$$Im\{F_k\} = \frac{1}{12}\sum_{n=-5}^{6} -f_n \sin\left(\frac{2\pi nk}{12}\right). \tag{25}$$

By adding a few more lines of code to our previous Matlab program we can calculate the imaginary coefficients.

```
for k=1:12
    for i=1:12
        z(i)=(1/12)*-(Re_Fn(i))*sin(2*pi*n(i)*n(k)/12);
    end
    z=z';
    Im_Fk(k)=sum(z);
end
Im_Fk=Im_Fk'
```

Which leads to the table.

n, k	x_n	$Re\{f(x_n)\}$	$Re\{F_k\}$	$Im\{F_k\}$
-5	$-5/12$	0.7630	0.0684	-0.1093
-4	$-4/12$	-0.1205	-0.1684	0.0685
-3	$-3/12$	-0.0649	-0.2143	-0.0381
-2	$-2/12$	0.6133	-0.0606	0.1194
-1	$-1/12$	-0.2697	-0.0418	-0.0548
0	0	-0.7216	0.0052	0
1	$1/12$	-0.0993	-0.0418	0.0548
2	$2/12$	0.9787	-0.0606	-0.1194
3	$3/12$	-0.5689	-0.2143	0.0381
4	$4/12$	-0.1080	-0.1684	-0.0685
5	$5/12$	-0.3685	0.0684	0.1093
6	$6/12$	0.0293	0.1066	0

(26)

At this point we can begin to see some of the key differences between the Fourier transform and the DFT. The first thing that you might notice is that we have a limited number of frequencies which make up our solution. In fact there are exactly $N/2$ different frequencies. These frequencies and their corresponding k values are

- $k = 0$ is analogous to your initial condition and has no associated frequency.
- $k = \pm 1$ has an associated frequency $\cos(\pi n/6) \mp i \sin(\pi n/6)$.
- $k = \pm 2$ has an associated frequency $\cos(\pi n/3) \mp i \sin(\pi n/3)$.
- $k = \pm 3$ has an associated frequency $\cos(\pi n/2) \mp i \sin(\pi n/2)$.
- $k = \pm 4$ has an associated frequency $\cos(2\pi n/3) \mp i \sin(2\pi n/3)$.
- $k = \pm 5$ has an associated frequency $\cos(5\pi n/6) \mp i \sin(5\pi n/6)$.
- $k = 6$ has an associated frequency $\cos(\pi n)$.

We can also see there is a symmetry in the coefficients. We will use this fact when reconstructing the function from the DFT coefficients.

Creating an Interpolating Function

In order to recreate the function that interpolates these data points we need some way to transform our DFT coefficients into a function. This is accomplished with the inverse discrete Fourier transform which is defined as

$$f(x) = \sum_{k=-N/2+1}^{N/2} F_k e^{i2\pi kx/N}. \tag{27}$$

It is important to note that in contrast to equation (20) we are summing with respect to the variable k as opposed to n. We can expand equation (27) using equation (4) and the symmetry in our Fourier coefficients, giving us

$$f(x) = Re\{F_0\} + 2\sum_{k=1}^{5}\left(Re\{F_k\}\cos\left(\frac{2\pi kx}{N}\right) - Im\{F_k\}\sin\left(\frac{2\pi kx}{N}\right)\right) + Re\{F_6\}\cos\left(\frac{2\pi kx}{N}\right). \tag{28}$$

Using Matlab we can write a program to plot this function onto Figure 3, giving us Figure 4. As you can see this function is a very good fit for our data points. In fact, this interpolating function goes through all twelve points.

What Does This Have To Do with Linear Algebra?

At this point you might be thinking to yourself. This is all very interesting, but what does it have to do with linear algebra? Up to this point most of the mathematics we have used might seem to fit into a Calculus II course. However, some of the fundamental assumptions we have made are based in linear algebra. If we define the function ω as

$$\omega = e^{-i2\pi/N}. \tag{29}$$

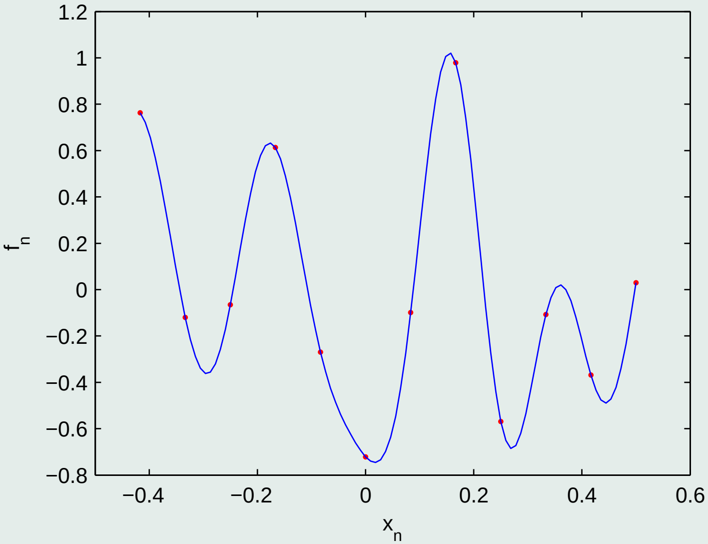

Figure 4:

We can rewrite equation (20) as

$$F_k = \frac{1}{N} \sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} f_n \omega^{nk}. \tag{30}$$

Then we can write out the system of equations represented by equation (30), giving us

$$F_{-\frac{N}{2}+1} = \frac{1}{N}\Big[f_{-\frac{N}{2}+1}\omega^{(-\frac{N}{2}+1)^2} + f_{-\frac{N}{2}+2}\omega^{(-\frac{N}{2}+2)(-\frac{N}{2}+1)} + \cdots$$
$$+ f_0\omega^{0(-\frac{N}{2}+1)} + \cdots + f_{-\frac{N}{2}}\omega^{(\frac{N}{2})(-\frac{N}{2}+1)}\Big]$$
$$F_{-\frac{N}{2}+2} = \frac{1}{N}\Big[f_{-\frac{N}{2}+1}\omega^{(-\frac{N}{2}+1)(-\frac{N}{2}+2)} + f_{-\frac{N}{2}+2}\omega^{(-\frac{N}{2}+2)^2} + \cdots$$
$$+ f_0\omega^{0(-\frac{N}{2}+2)} + \cdots + f_{-\frac{N}{2}}\omega^{(\frac{N}{2})(-\frac{N}{2}+2)}\Big]$$
$$\vdots$$

$$F_0 = \frac{1}{N}\Big[f_{-\frac{N}{2}+1}\omega^{(-\frac{N}{2}+1)(0)} + f_{-\frac{N}{2}+2}\omega^{(-\frac{N}{2}+2)(0)} + \cdots$$
$$+ f_0\omega^{(0)^2} + \cdots + f_{\frac{N}{2}}\omega^{(\frac{N}{2})(0)}\Big]$$
$$\vdots$$

$$
\begin{aligned}
F_{\frac{N}{2}-1} &= \frac{1}{N}\Big[f_{-\frac{N}{2}+1}\omega^{(-\frac{N}{2}+1)(\frac{N}{2}-1)} + f_{-\frac{N}{2}+2}\omega^{(-\frac{N}{2}+2)(\frac{N}{2}-1)} + \cdots \\
&\quad + f_0\omega^{0(\frac{N}{2}-1)} + \cdots + f_{-\frac{N}{2}}\omega^{(\frac{N}{2})(\frac{N}{2}-1)} \Big] \\
F_{\frac{N}{2}} &= \frac{1}{N}\Big[f_{-\frac{N}{2}+1}\omega^{(-\frac{N}{2}+1)(\frac{N}{2})} + f_{-\frac{N}{2}+2}\omega^{(-\frac{N}{2}+2)(\frac{N}{2})} + \cdots \\
&\quad + f_0\omega^{0(\frac{N}{2})} + \cdots + f_{-\frac{N}{2}}\omega^{(\frac{N}{2})^2} \Big].
\end{aligned}
$$

This system of equations may seem intimidating at first. However, if we rewrite the system of equations in a more familiar form they won't be so intimidating. If we let F represent a vector containing our DFT coefficients and we let f be a vector containing our input data. We can rewrite our system of equations, giving us

$$
F = \frac{1}{N} W f, \tag{31}
$$

where

$$
W = \begin{bmatrix}
\omega^{(-\frac{N}{2}+1)^2} & \omega^{(-\frac{N}{2}+2)(-\frac{N}{2}+1)} & \cdots & \omega^{0(-\frac{N}{2}+1)} & \cdots & \omega^{(\frac{N}{2}-1)(-\frac{N}{2}+1)} & \omega^{(-\frac{N}{2}+1)(\frac{N}{2})} \\
\omega^{(-\frac{N}{2}+1)(-\frac{N}{2}+2)} & \omega^{(-\frac{N}{2}+2)^2} & \cdots & \omega^{0(-\frac{N}{2}+2)} & \cdots & \omega^{(\frac{N}{2}-1)(-\frac{N}{2}+2)} & \omega^{(\frac{N}{2})(-\frac{N}{2}+2)} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
\omega^{(-\frac{N}{2}+1)(0)} & \omega^{(-\frac{N}{2}+2)(0)} & \cdots & \omega^{(0)^2} & \cdots & \omega^{(\frac{N}{2}-1)(0)} & \omega^{(\frac{N}{2})(0)} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
\omega^{(-\frac{N}{2}+1)(\frac{N}{2}-1)} & \omega^{(-\frac{N}{2}+2)(\frac{N}{2}-1)} & \cdots & \omega^{0(\frac{N}{2}-1)} & \cdots & \omega^{(\frac{N}{2}-1)^2} & \omega^{(\frac{N}{2})(\frac{N}{2}-1)} \\
\omega^{(-\frac{N}{2}+1)(\frac{N}{2})} & \omega^{(-\frac{N}{2}+2)(\frac{N}{2})} & \cdots & \omega^{0(\frac{N}{2})} & \cdots & \omega^{(\frac{N}{2}-1)(\frac{N}{2})} & \omega^{(\frac{N}{2})^2}
\end{bmatrix}.
$$

Let's return to our example. If we let

$$f = \begin{bmatrix} 0.7630 \\ -0.1205 \\ -0.0649 \\ 0.6133 \\ -0.2697 \\ -0.7216 \\ -0.0993 \\ 0.9787 \\ -0.5689 \\ -0.1080 \\ -0.3685 \\ 0.0293 \end{bmatrix}. \qquad (32)$$

Then we can determine our DFT coefficients using equation (31). Using Matlab with the commands:

```
N=12;
f=[.7630;-.1205;-.0649;.6133;-.2697;-.7216;...
     -.0993;.9787;-.5689;-.1080;-.3685;.0293]
h=-5:6;
h=h'
H=h*h'
w=exp(-i*2*pi/12);
W=w.^H;
F=(1/N)*W*f
```

We see that our DFT coefficients are:

```
F =
   0.0684 - 0.1093i
  -0.1684 + 0.0685i
  -0.2143 - 0.0381i
  -0.0606 + 0.1194i
  -0.0418 - 0.0548i
   0.0052
  -0.0418 + 0.0548i
  -0.0606 - 0.1194i
  -0.2143 + 0.0381i
  -0.1684 - 0.0685i
   0.0684 + 0.1093i
   0.1066 + 0.0000i .
```

Which confirms our previous solution. At first glance this may not appear to be symmetric like our previous solution. However, if you remember that when k is a negative number it negates the complex DFT coefficient, you will see that we have exactly the same solution as before.

As you can see using linear algebra to solve for our DFT coefficients is much easier than our previous method. The DFT has allowed us to do something truly amazing. Assuming that we have a set of discrete data points that exist spatial domain, in the spatial domain we are unable to create a function to interpolate this data. Therefore, we transfer those data points into the frequency domain where we are able to dissect the frequencies and their amplitudes. Then, we use the IDFT to shift back into our spatial domain where we plot our interpolating function $f(x)$.

Applications of the DFT

We already have examined a very powerful application of the DFT. We have shown that the DFT can be used to interpolate any data set which exhibits a periodic behavior. Over the centuries the DFT has been used in some very interesting applications. In 1754 Clairaut and Lagrange were looking for a way to explain their astronomical data. Since their data involved discrete quantities they used a finite series of sines and cosines to fit their data. This is perhaps the first known application of the DFT. In the early 1800's Gauss was trying to determine the orbit of the asteroid Pallas and used his own version of the DFT to fit his data. Fast-forward almost 200 years and the DFT is still being used in some of the most interesting applications of our time. One example is using the DFT in signal processing. We obtain a signal from an input device such as a telephone. However, in the interest of saving our limited resources we only want to transmit a minimum amount of data, while still preserving the integrity of the signal. This is a perfect application for the DFT. We can divide the input signal into its major frequencies, determine the DFT coefficients (amount of each frequency), transmit the DFT coefficients and associated frequencies, and re-create the signal on the other end of the line. The DFT also has applications in seismic migration, image reconstruction, and digital filtering to name just a few. Despite having been discovered almost 200 years ago, the DFT is still on the cutting edge of mathematics and its applications.

References

[1] Briggs, William, and Van Emden Henson. The DFT: An Owners Manual for the Discrete Fourier Transform. Philadelphia: SIAM, 1995.

[2] Leon, Steven J. Linear Algebra with Applications. 6th ed. New York, NY: Prentice Hall, 2002.

[3] Polking, John, Albert Boggess, and David Arnold. Differential Equations with Boundary Value Problems. Upper Saddle River, NJ: Pearson Education, 2002.

[4] Professor David Arnold, College of the Redwoods.

[5] Strang, Gilbert. Introduction to Linear Algebra. 3rd ed. Wellesley, MA: Wellesley- Cambridge Press, 2003.