The Fourier Transform - A Primer

Hagit Shatkay

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-95-37
November 1995

The Fourier Transform – A Primer

Hagit Shatkay
Department of Computer Science
Brown University
Providence, RI 02912

1 Introduction

The Fourier transform is among the most widely used tools for transforming data sequences and functions (single or multi-dimensional), from what is referred to as the *time domain* to the *frequency domain*. Applications of the transform range from designing filters for noise reduction in audio-signals (such as music or speech), to fast multiplication of polynomials.

The following document provides a brief introduction to the Fourier transform, for those of us who are still aliens in the *frequency domain*. The topic of the Fourier transform and its applications is covered in numerous, stout books (such as [Bra65, OS75, Wea83, BP85, Jac90]), and this paper can not and does not intend to cover the area in full. Its goal is to introduce the basic terminology and the main concepts of the area, providing common ground for further discussion and study.

The rest of the paper is organized as follows: Section 2 introduces the idea of representing sequences and functions through sinusoids. Section 3 shows how complex numbers and exponentials fit into the sinusoids representation framework. Section 4 presents the continuous Fourier transform. In Section 5 we discuss sampling, which is the mean for converting a continuous signal into a discrete sequence. Section 6 presents the discrete Fourier transform, and the prominent related topics – convolution and the fast Fourier transform. Section 7 demonstrates some of the applications of the Fourier transform, and concludes the paper.

2 Functions as Combinations of Sinusoids

Any continuous, periodic function can be represented as a linear combination of sines and cosines. A sine is a function of the form: $Asin(2\pi\omega t + \phi)$, where A is the *amplitude*, ω is the *frequency* measured in cycles (or periods) per second, and ϕ is the *phase*, which is used for getting values other than 0 at $t = 0$. A cosine function has exactly the same components as the sine function, and can be viewed as a shifted sine (or more accurately – a sine with phase $\pi/2$).

Thus, given a function $f(t)$, we can usually rewrite it (or at least approximate it), for some n as:

$$f(t) = \sum_{k=1}^{n}(A_k cos(2\pi\omega_k t) + B_k sin(2\pi\omega_k t)) \tag{1}$$

Both sines *and* cosines are combined, rather than *only* sines, to allow the expression of functions for which $f(0) \neq 0$, in a way that is simpler than adding the phase to the sine in order to make it into a cosine.

As an example of a linear combination of sinusoids consider the function:

$$f_1(t) = 0.5 sin \pi t + 2 sin 4\pi t + 4 cos 2\pi t$$

Its three sinusoidal components and the function f_1 itself are depicted in Figure 1, as *a, b, c* and *d* respectively. The function $f_1(t)$ consists of sines and cosines of 3 frequencies.

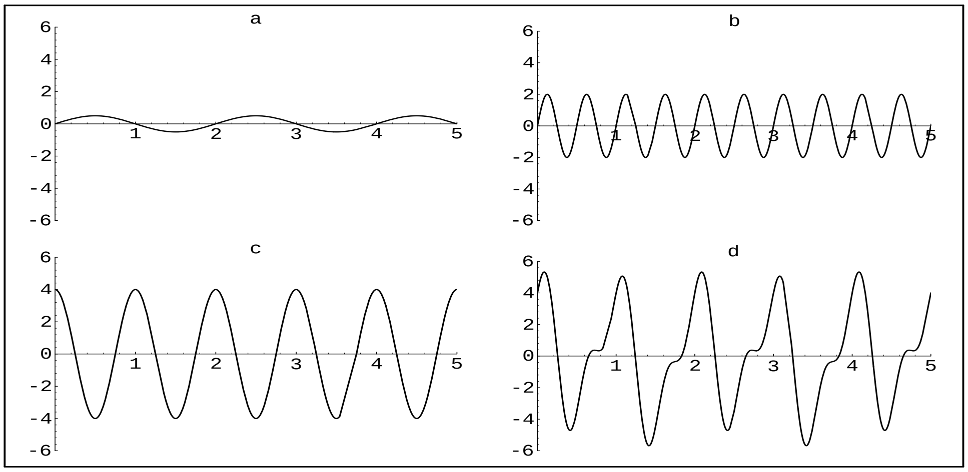

Figure 1: A plot of $f_1(t)$, (d), and its components (a, b, c), for $t = 0..5$

Thus, the frequency analysis of $f_1(t)$, can be summarized in a table such as Table 1, which provides for each frequency of f_1 the amplitude of the sine wave and of the cosine wave with this frequency.

k	**Frequency (ω_k)**	**Cosine Amplitude (A_k)**	**Sine Amplitude (B_k)**
1	1/2	0	1/2
2	2	0	2
3	1	4	0

Table 1: Frequency contents of the function $f_1(t)$

The representation of a periodic function (or of a function that is defined only on a finite interval) as the linear combination of sines and cosines, is known as the *Fourier series* expansion of the function. The Fourier transform is a tool for obtaining such frequency and amplitude information for sequences and functions, which are not necessarily periodic. (Note that sequences are just a special case of functions.)

3 Fitting in Complex Numbers and Exponentials

Another way of writing sinusoids relies on the following equalities:

$$e^{i\theta} = cos(\theta) + isin(\theta) \qquad\qquad e^{-i\theta} = cos(\theta) - isin(\theta) \tag{2}$$

where i is the square root of -1. Both are easily derived from the Taylor series expansion of cos, sin, and e^{θ}. Through addition and subtraction they can be rewritten as:

$$cos(\theta) = \frac{e^{i\theta} + e^{-i\theta}}{2} \qquad\qquad sin(\theta) = \frac{e^{i\theta} - e^{-i\theta}}{2i} \tag{3}$$

Hence, we can substitute the sin and cos expressions of equation 1 by the respective expressions of equation 3 and get:

$$f(t) = \sum_{k=1}^{n} [\frac{A_k}{2}(e^{2\pi i\omega_k t} + e^{-2\pi i\omega_k t}) + \frac{B_k}{2i}(e^{2\pi i\omega_k t} - e^{-2\pi i\omega_k t})] \tag{4}$$

If we denote:

$$\begin{aligned} C_k &= \tfrac{A_k - iB_k}{2} && k > 0 \\ C_k &= \tfrac{A_k + iB_k}{2} && k < 0 \\ C_0 &= 0 \\ \omega_k &= -\omega_{-k} && k < 0 \end{aligned} \tag{5}$$

we can again rewrite $f(t)$:

$$f(t) = \sum_{k=-n}^{n} [C_k e^{2\pi i\omega_k t}] \tag{6}$$

Under this new notation we can rewrite the frequency analysis of Table 1 as shown in Table 2.

k	**Frequency (ω_k)**	C_k
-3	-1	2
-2	-2	$2i$
-1	$-1/2$	$i/4$
0	0	0
1	$1/2$	$-i/4$
2	2	$-2i$
3	1	-2

Table 2: Another form of frequency contents of the function $f_1(t)$

Further manipulation of equation 6 is based on using the polar notation for complex numbers, that is:

$$x + iy = r(cos(\theta) + isin(\theta)) = re^{i\theta}$$

where

$$r = |x + iy| = \sqrt{x^2 + y^2} \quad \text{and} \quad tan(\theta) = \frac{y}{x}$$

Using this representation of complex numbers, we get:

$$C_k e^{2\pi i\omega_k t} = r_k e^{i\phi_k} e^{2\pi\omega_k t} = r_k e^{i(2\pi\omega_k t+\phi_k)} \tag{7}$$

where

$$r_k = \left(\frac{A_k^2 + B_k^2}{4}\right)^{1/2} \qquad \text{and} \qquad tan(\phi_k) = \begin{cases} \frac{-B_k}{A_k} & k > 0 \\ \frac{B_k}{A_k} & k < 0 \end{cases}$$

From all the above we obtain:

$$f(t) = \sum_{k=-n}^{n} r_k e^{i(2\pi\omega_k t+\phi_k)} \tag{8}$$

The full details of the above rewriting can be found in [Wea83].

Using the terminology introduced in Section 2, w_k is the k^{th} *frequency*, r_k is the *amplitude*, and ϕ_k is the *phase*. In the following sections we will be using this terminology as the basis for discussing the Fourier transform.

4 The Continuous Fourier Transform

The continuous Fourier transform of a function $f(t)$ is defined as follows:

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-2\pi i\omega t}dt \tag{9}$$

$$f(t) = \int_{-\infty}^{\infty} F(\omega)e^{2\pi i\omega t}d\omega \tag{10}$$

Equation 10 is the continuous generalization of expressing $f(t)$ as a combination of sinusoids, as discussed in the previous sections. It is known as the *inverse Fourier transform*. Equation 9 provides the means for finding the amplitude for each frequency ω, given that the integral indeed converges. The result of applying the Fourier transform to a function is called the *frequency spectrum* or the *power spectrum* of the function, or in short the *spectrum*.

Here are examples of several useful functions and their respective Fourier transforms:

Example 1 The *Pulse* function is defined as: $p_{1/n}(t) = \begin{cases} \frac{n}{2} & |t| \leq \frac{1}{n} \\ 0 & otherwise \end{cases}$

Its Fourier transform is a *sinc* function, obtained as follows:

$$P_{1/n}(\omega) = \int_{-\infty}^{\infty} p_{1/n}(t)e^{-2\pi i\omega t}dt = \int_{-\frac{1}{n}}^{\frac{1}{n}} \frac{n}{2}e^{-2\pi i\omega t}dt = \frac{n}{2}\frac{e^{2\pi i\omega/n} - e^{-2\pi i\omega/n}}{2\pi i\omega} = \frac{sin(2\pi\omega/n)}{2\pi\omega/n} = sinc(2\pi\omega/n)$$

The respective graphs for $p_{1/n}(t)$ and $P_{1/n}(\omega)$ are shown in Figure 2.

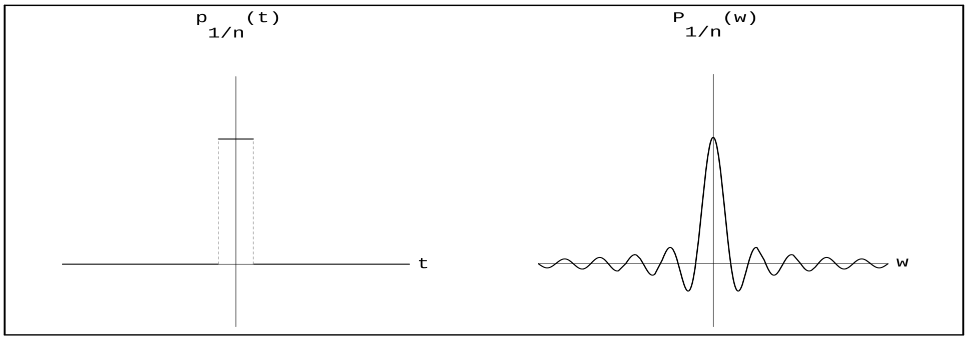

Figure 2: A plot of $p_{1/n}(t)$, and its Fourier transform $P_{1/n}(\omega)$

Example 2 The δ function is defined as: $\delta(t) = \lim_{n\to\infty} p_{1/n}(t)$

This function is also known as the *Dirac delta function* or the *unit impulse function.* It is 0 for all t, except for 0, and we can think of $\delta(0)$ as being ∞.

The Fourier transform of $\delta(t)$ is:

$$\Delta(\omega) = \lim_{n\to\infty} P(\omega) = \lim_{n\to\infty} \frac{sin(2\pi\omega/n)}{2\pi\omega/n} = 1$$

Example 3 Let $f(t)$ be some simple cosine function: $f(t) = cos(2\pi a t)$

Its Fourier transform is:

$$F(\omega) = \int_{-\infty}^{\infty} cos(2\pi a t) e^{-2\pi i \omega t} dt = \frac{\delta(\omega + a) + \delta(\omega - a)}{2}$$

The respective graphs for $f(t)$ and $F(\omega)$ are shown in Figure 3.

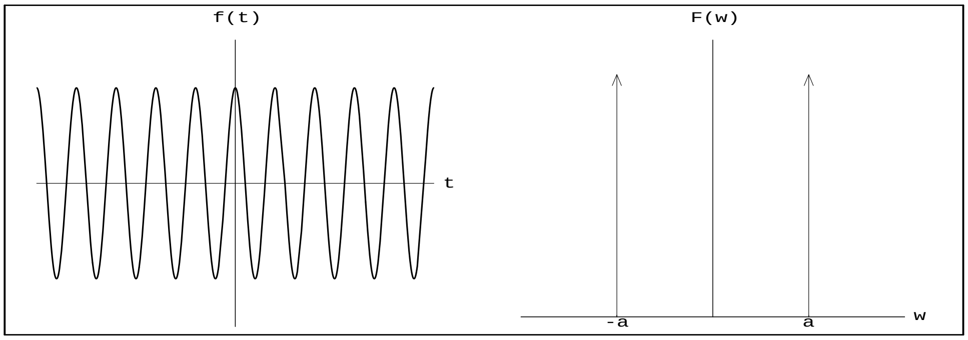

Figure 3: A plot of $f(t) = cos(2\pi a t)$, and its Fourier transform $F(t)$

Table 3 lists some of the Fourier transform properties, which make it so useful in practice. We follow the convention of source functions denoted by small letters, while their Fourier transform results (which are assumed to exist) are capitalized.

	Property	$f(t)$	$F(\omega)$
1.	Linearity	$af_1(t) + bf_2(t)$	$aF_1(\omega) + bF_2(\omega)$
2.	Convolution[1] Theorem	$f_1(t) * f_2(t)$	$F_1(\omega)F_2(\omega)$
3.	Product Theorem	$f_1(t)f_2(t)$	$F_1(\omega) * F_2(\omega)$
4.	Time Shifting	$f(t - t_0)$	$F(\omega)e^{-2\pi i\omega t_0}$
5.	Frequency Shifting	$f(t)e^{-2\pi i\omega_0 t}$	$F(\omega - \omega_0)$
6.	Scaling[2]	$f(at)$	$\|a\|^{-1}F(\omega/a)$
7.	Parseval's Theorem	$\int_{-\infty}^{\infty} \|f(t)\|^2 dt = \int_{-\infty}^{\infty} \|F(\omega)\|^2 d\omega$	

Table 3: Some Basic Properties of the Fourier Transform

Most of the above properties can be proved easily from the definition of the transform and its inverse. Proofs for the more complicated properties (such as Parseval's theorem), as well as some additional properties can be found in [Wea83, OS89, Jac90].

Properties 2 and 3 state, respectively, that convolution in the time domain corresponds to multiplication of coefficients in the frequency domain, and vice versa. This is one of the most useful properties of the transform, and is taken advantage of in filter design, in reasoning about sequence behavior, as well as in fast multiplication of polynomials, as will be discussed later on, when discussing the discrete version of the transform.

Property 4 above states that shifting of the original function in time corresponds to a change of phase of the sinusoids comprising the function. Similarly, property 5 states that a sinusoidal modulation in the function corresponds to a phase shift in frequency.

Parseval's theorem states that the total energy of a signal is the same in the time domain as it is in the frequency domain.

Since most physical signals, from radio waves to seismic phenomena, are continuous, and have a continuous range of frequencies, the functions that are used to model them are continuous as well, and having continuous transform is desirable. However, in many cases, the function describing a phenomenon is unknown. Data that characterizes it needs to be gathered and analyzed. Moreover, measured discrete values at various points in time are relatively easy to obtain, and computers which process such data are inherently discrete as well. Therefore, we are interested in a transform from the *discrete time domain* to the *discrete frequency domain*, and an inverse discrete transform to take us in the opposite direction. The rest of this paper discusses various aspects of the discrete signals and their transforms.

5 Sampling

Discrete sequences most commonly occur as a representation of continuous signals. They are obtained through periodic sampling of the signal.

A correct choice of sampling intervals is crucial for getting a faithful representation of the

[1]The *convolution* of the functions $f(x)$ and $g(x)$ is: $f(x) * g(x) = \int_{-\infty}^{\infty} f(\xi)g(x - \xi)d\xi$.
[2]In particular, $f(-t)$ corresponds to $F(-\omega)$.

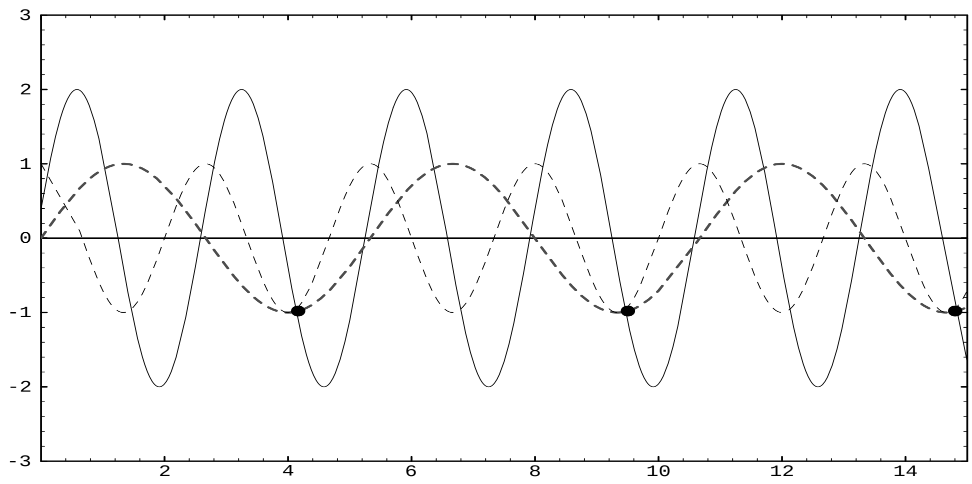

Figure 4: Three sinusoids. The black dots mark the sampling points.

original signal. Consider for instance the three sinusoids depicted in Figure 4. Suppose we sample them only where the three black dots are in the figure, that is, at the intersection points of the three sequences. Clearly, these points don't have enough information to distinguish one sinusoid from the others. Therefore, if we were to reconstruct the original sinusoids from these points, at best one of them would have been recovered correctly. The other two sinusoids could not have been the same as the recovered one. We will see what a "good" sampling is later on in this section.

Given a continuous signal $x_c(t)$, we fix a time interval of length T and obtain a discrete signal $x[n] = x_c(nT)$ for $-\infty < n < \infty$. T is called the *sampling period* and $\omega_s = 1/T$ is called the *sampling frequency.*

At the conceptual level, the discrete sequence $x[n]$ can be viewed as though it is a continuous sequence $x_s(t)$ with value 0 for all $t \neq nT$ and with value $x[n]$ at $t = nT$. Putting it more formally, we define:

$$s(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \quad \text{and} \quad x_s(t) = x_c(t)s(t)$$

where δ is the Dirac delta discussed in Section 4, Example 2. Thus, $x_s(t)$ is the result of multiplying the original continuous signal $x_c(t)$ by an impulse train, $s(t)$. (The reason for referring to $s(t)$ as "an impulse train" is obvious from looking at its graph, given in Figure 5.)

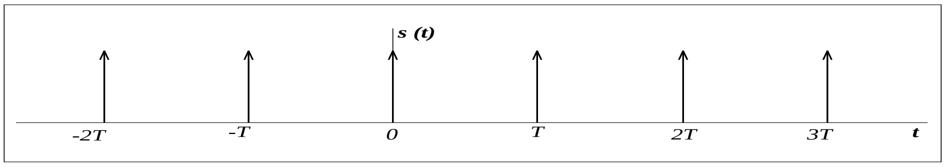

Figure 5: The impulse train $s(t)$

Hence, from the *Product Theorem* of Table 3, we know that the Fourier transform of $x_s(t)$ is the same as the *convolution* of the Fourier transforms of $x_c(t)$ and $s(t)$, which we denote as $X_c(\omega)$ and $S(\omega)$, respectively. $S(\omega)$ is the Fourier transform of an impulse train, which is an impulse train as well (see [OS89] for further details), and is expressed as:

$$S(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} \delta(\omega - k\omega_s)$$

Therefore, we obtain the following:

$$X_s(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_c(\omega - k\omega_s)$$

Figure 6 shows the relationship between X_c and X_s.

Plot (a) in the figure shows the result of the Fourier transform on the original signal, and plot (b) shows the Fourier transform of the impulse train. The result of their convolution,

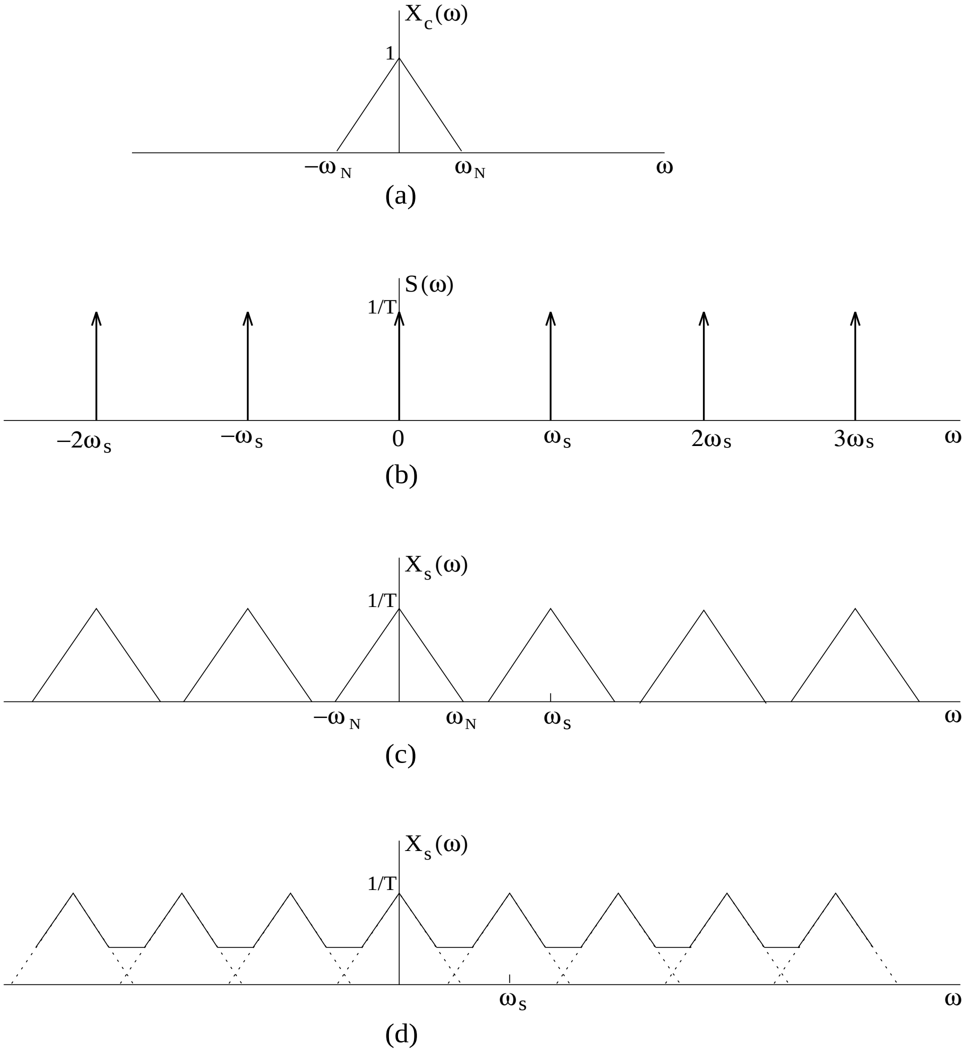

Figure 6: The frequency-domain effects of sampling $x_c(t)$. (a) The spectrum plot for $x_c(t)$. ω_N is its maximum frequency. (b) The spectrum plot for the impulse train. (c) The spectrum of the sampled signal, where $\omega_s > 2\omega_N$. (d) Aliasing when $\omega_s < 2\omega_N$

which is the Fourier transform of the sampled signal, is shown in plot (c). We can see that sampling of a signal $x_c(t)$ in the time domain, corresponds in the frequency domain to duplication of the original signal's spectrum, $X_c(\omega)$, centered around integer multiples of the sampling frequency ω_s. Thus, all we need to do in order to recover the original continuous signal from the sampled one, is to "get rid" of the frequencies introduced by the sampling (which correspond to the duplications), and keep only the frequencies centered around 0, which are the original frequencies. (In terms of the triangles of Figure 6(c), we want to erase all triangles except for the one centered around 0, so that we are left with the spectrum of $x_c(t)$).

This can be achieved by filtering the sampled signal $x_s(t)$, using a *low-pass* filter. Such a filter takes a signal $x(t)$, and eliminates from it all frequencies of absolute value greater than some threshold ω_f. Thus, it produces a signal $y(t)$ whose spectrum is the same as that of $x(t)$ for all frequencies between $[-\omega_f, \omega_f]$, but with no frequencies above ω_f or below $-\omega_f$

By applying such a filter with $\omega_N \leq \omega_f \leq (\omega_s - \omega_N)$ to the sampled signal (padded with 0's on the unsampled intervals), we obtain the original signal, given that precautions were taken to sample frequently enough, such that $\omega_N \leq (\omega_s - \omega_N)$, or equivalently, such that $2\omega_N \leq \omega_s$.

Figure 6(d) demonstrates what happens if the above requirement is not met. We can see (looking at the dashed lines) that the duplications of x_c's spectrum overlap each other, which means that the spectrum of the sampled signal (shown as the solid line) contains frequency amplitudes that were not there to begin with, while original amplitudes of frequencies are lost. Hence, faithful reconstruction of the original signal from its samples is not possible. This phenomenon, of having original amplitudes of frequencies replaced by bogus amplitudes is known as *aliasing*.

The fact we have stated above, that if our sampling frequency ω_s is at least twice the highest frequency of the original signal, ω_n, a faithful reconstruction of the signal from its samples is possible, is exactly the contents of *Nyquist's Theorem*, and a sampling frequency $\omega_s = 2\omega_N$ is called the *Nyquist frequency*.

It is important to note that our reasoning about the frequencies of the sampled sequence with respect to the original sequence is a *mental exercise* rather than an *algorithmic method*. In order to recover the original sequence from the sampled one, we don't need to apply the Fourier transform to the sampled sequence, cut off its high frequencies, and perform the inverse Fourier transform. We simply take the sampled sequence and feed it to a low-pass filter, which processes it and reconstructs the original signal from it. The frequency domain reasoning just showed us why a low-pass filter is a tool for recovering a continuous signal from its samples.

Once a signal is sampled, we have a *discrete* sequence, which is either a faithful representation (if the sampling frequency was at least the Nyquist frequency), or an unfaithful representation of it, (if a lower sampling frequency was used). In any case, a discrete sequence can be processed using discrete methods, which facilitate the usage of digital computers. The discrete form of the Fourier transform, (known as the DFT) is discussed in the following sections.

6 The Discrete Fourier Transform

6.1 Definition of the DFT

The Discrete Fourier Transform (DFT) maps a discrete periodic sequence $f[k]$ (where k is an integer, and the period is N), to another discrete sequence $F[j]$, of frequency coefficients.

It is defined as:[3]

$$F[j] = \sum_{k=0}^{N-1} f[k] e^{-2\pi ikj/N} \qquad 0 \leq j \leq N-1 \tag{11}$$

$$f[k] = \frac{1}{N} \sum_{j=0}^{N-1} F[j] e^{2\pi ikj/N} \qquad 0 \leq k \leq N-1 \tag{12}$$

The interpretation of the above equations is that at point k, the sequence value $f[k]$ is a linear combination of the values of N sinusoids, $e^0, ..., e^{(2\pi/N)k(N-1)}$. The coefficients of the sinusoids are $F[0], ..., F[N-1]$ respectively, and their frequencies are j/N cycles per sample or $2\pi j/N$ radians per sample (where $0 \leq j \leq (N-1)$. We should note that:

$$e^{-2\pi ikj/N} = e^{-2\pi ik(j+N)/N}$$

Thus the function $F[j]$, like the original $f[k]$, is periodic with period N, and therefore the frequency range to be considered is $0..2\pi$ radians/sample, or $0..N$ cycles/sample.

It is also interesting to note that for any frequency other than the j/N's, a discrete sinusoid is not periodic. For a discrete cosine (or sine) $f[k] = Acos(2\pi\omega_0 k)$ to be periodic with period N, it must satisfy:

$$Acos(2\pi\omega_0(k+N)) = Acos(2\pi\omega_0 k)$$

which means: $2\pi\omega_0 N = 2\pi j$, or equivalently, $\omega_0 N = j$, for some integer j. The equality holds only for the frequencies of the form j/N (in units of cycles/samples).

Here are some examples of applying the DFT to discrete sequences:

Example 4 Let $p[k]$ be a discrete pulse function, with periodicity 10, defined as:

$$p[k] = \begin{cases} 1 & 0 + 10m \leq k \leq 4 + 10m \quad \text{For some integer } m \\ 0 & otherwise \end{cases}$$

Its Fourier transform is:

$$P[j] = \sum_{k=0}^{9} p[k] e^{-2\pi ijk/10} = \sum_{k=0}^{4} e^{-2\pi ijk/10} = e^{-4\pi ij/10} \frac{sin(\pi j/2)}{sin(\pi j/10)}$$

The respective graphs for $p[k]$ and $P[j]$ are shown in Figure 7.

[3]The exact formulation of the DFT varies slightly in books. Some have the $\frac{1}{N}$ coefficient in front of the expression for $F[j]$ rather than for $f[k]$, while others have a coefficient $\frac{1}{\sqrt{N}}$ rather than $\frac{1}{N}$ and use it both for $F[j]$ and for $f[k]$. The latter is the form Mathematica uses. All 3 forms are correct as long as they are used consistently.

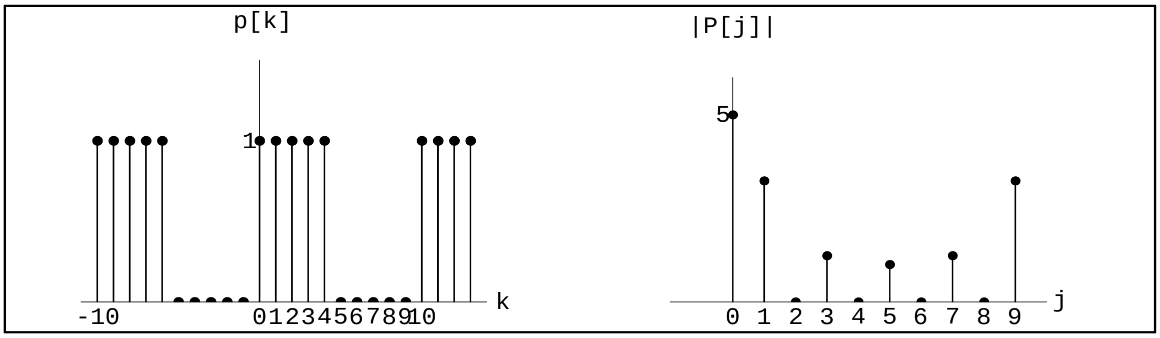

Figure 7: A plot of $p[k]$, and the amplitude of its DFT $|P[j]|$.

Example 5 We have defined the continuous δ function in Example 2, and have used it in the previous section. Its much simpler discrete counterpart, the *discrete* δ function, for $N = 10$ is defined as:

$$\delta[k] = \begin{cases} 1 & k = 0 + 10m \\ 0 & otherwise \end{cases}$$

Its Fourier transform is:

$$\Delta[j] = \sum_{k=0}^{9} \delta[k] e^{-2\pi ijk/10} = e^0 = 1$$

The respective graphs for $\delta[k]$ and $\Delta[j](\omega)$ are shown in Figure 8.

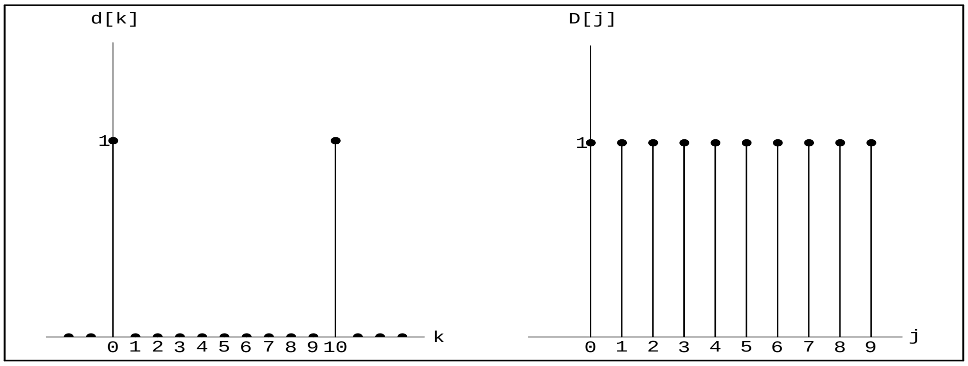

Figure 8: A plot of $\delta[k]$, and its DFT $\Delta[j]$.

To make notation simpler, throughout the rest of this section we denote $e^{2\pi i/N} = W_N$ and obtain:

$$F[j] = \sum_{k=0}^{N-1} f[k](W_N)^{-kj} \qquad 0 \le j \le N-1 \tag{13}$$

$$f[k] = \frac{1}{N} \sum_{j=0}^{N-1} F[j](W_N)^{kj} \qquad 0 \le k \le N-1 \tag{14}$$

W_N is called the *principal N^{th} root of unity* since $(W_N)^N = e^{2\pi i} = cos(2\pi) + isin(2\pi) = 1$. Similarly, $(W_N)^k$, where $0 \le k \le N-1$ are all the N distinct complex roots of unity.

6.2 DFT Properties

Before providing the properties of the DFT, some subtleties that arise from the periodicity of the sequences, must be addressed.

First, we must remember that any sequence we are dealing with in the context of the DFT is periodic, with some integer periodicity N. Given a finite (non-periodic) discrete sequence $x[k]$ of length n, (i.e. $0 \le k \le n-1$), to which we want to apply the Fourier transform, we regard it as though it is periodic with periodicity $N = n$. Thus, we effectively define a new sequence, $y[m]$ for *all* integer m:[4]

$$y[m] = x[m \bmod n] \quad \text{where} \quad m \bmod n = m - n\lfloor x/y \rfloor$$

and apply the DFT to one period of $y[m]$.

Hence, if we regard any sequence $x[k]$ (and its DFT $X[j]$), as periodic with period N, the *shift* of $x[k]$ by l is interpreted as: $x[k+l] = x[(k+l) \bmod N]$ and is called a *circular shift*. The same convention holds for shifting $X[j]$.

Second, when applying the DFT to a combination of two periodic sequences, $x_1[k]$ and $x_2[m]$, we must account for the periodicity of the combination. Since the DFT is defined over a single period, for the DFT of the combination to be well defined, it must have a single periodicity. There are three forms of combinations we have encountered in the continuous case, namely, *linear combination* $(ax_1 + bx_2)$, *multiplication* x_1x_2, and *convolution* $x_1 * x_2$. Both linear combination and multiplication for the continuous case, are defined such that $x_1(t)$ is paired with the corresponding $x_2(t)$. Similarly, in the discrete case each $x_1[i]$ should be combined with the corresponding $x_2[i]$. Thus x_1 and x_2 must be of the same periodicity N, and the resulting sequence is of periodicity N, as well. However, if the two sequences are of two different periodicities, N_1, N_2, (assume, without loss of generality, $N_1 < N_2$), either we disallow their combination, and term it "undefined", or we pad the sequence of the periodicity N_1 with 0's at the end of each period, thus practically converting it into a sequence of periodicity N_2, and therefore the combination is well defined. Whether the 0 padding is reasonable or not depends mostly on the application. Such padding is used, for instance, for supporting the Fast Fourier Transform.

The last kind of combination that needs to be addressed is the convolution. We recall that one of the important properties of the continuous transform is the duality between convolution in the time domain and multiplication in the frequency domain, and vice versa. We want to retain this property in the discrete case. Thus, given two periodic discrete sequences, x_1, x_2, of periodicity[5] N, with respective DFTs $X_1[j], X_2[j]$, we want their convolution result x_3 to have DFT $X_3 = X_1X_2$. By defining the *circular convolution*

[4] See [GKP91] for details on the mod operation.

[5] The period must be the same in order for the (dual) multiplication to be well defined.

of x_1, x_2 to be:

$$x_3[k] = \sum_{m=0}^{N-1} x_1[m]x_2[(k-m) \bmod N] \tag{15}$$

we obtain the desired correspondence [OS75]. As an example of circular convolution consider the two periodic sequences, with $N = 3$:

$$< a_0, a_1, a_2, a_0, a_1, a_2, a_0... > \quad \text{and} \quad < b_o, b_1, b_2, b_o, b_1, b_2, b_o, ... >$$

Their circular convolution is the sequence:

$$< a_0b_0 + a_1b_2 + a_2b_1, \;\; a_0b_1 + a_1b_0 + a_2b_2, \;\; a_0b_2 + a_1b_1 + a_2b_0 >$$

which is also regarded as a discrete periodic sequence with $N = 3$, by repeatedly duplicating these 3 elements, while preserving the above order. Thus, the circular convolution maps a pair of sequences of periodicity N to a third sequence of the same periodicity. (We extend the definition to non-periodic sequences of the same length N, by regarding them as periodic, as defined earlier in this section).

Table 4 lists the discrete counterparts of the properties given in Table 3 for the continuous case. We assume that we are dealing only with well defined combinations.

	Property	$\boldsymbol{f[k]}$	$\boldsymbol{F[j]}$
1.	Linearity	$af_1[k] + bf_2[k]$	$aF_1[j] + bF_2[j]$
2.	Convolution[6] Theorem	$f_1[k] * f_2[k]$	$F_1[j]F_2[j]$
3.	Product Theorem	$f_1[k]f_2[k]$	$F_1[j] * F_2[j]^6$
4.	Time Shifting[7]	$f[k-k_0]$	$F[j]W_N^{-jk_0}$
5.	Frequency Shifting[7]	$f[k]W_N^{kj_0}$	$F[j-j_0]$
7.	Parseval's Theorem	$\sum_{k=0}^{N-1} \|f[k]\|^2 = \frac{1}{N}\sum_{j=0}^{N} \|F[j]\|^2$	

Table 4: Basic Properties of the Discrete Fourier Transform

6.3 The Fast Fourier Transform

One of the most appealing aspects of the DFT is the existence of an efficient procedure for calculating it, using $O(NlogN)$ complex operations, rather than $O(N^2)$ operations required for the naive algorithm.

The algorithm for fast DFT, is known as the *FFT* (the Fast Fourier Transform). It takes advantage of symmetry properties of the complex roots of unity (the W_N's we have defined earlier), and uses repeated clever partitioning of the input sequence into two equally long subsequences, each of which can be separately (and quickly) processed. In order to take full advantage of the repetitive partitioning into equal two parts, the original sequence needs to be of length or periodicity which is a power of 2. If it is not originally so, it is padded with 0's – as mentioned earlier in Section 6.1. The full details of the algorithm are beyond the scope of this paper. An excellent presentation of it, including the fundamental

[6] Circular convolution
[7] Circular shift

mathematical background can be found in [Sav96]. Other good sources for discussion of the FFT and its applications are [CLR89, OS75, Wea83].

The FFT algorithm gives rise to an efficient convolution algorithm, due to the duality between convolution in the time domain and multiplication in the frequency domain. Convolution can be implemented by applying the FFT to the original sequences, multiplying the results, and performing the inverse FFT to obtain the results of the convolution. We should note that in many real applications which require convolution, (such as polynomial multiplication or filtering of signals), the convolution is not circular, it does not regard sequences as periodic, and does not require the sequences to be of the same length. This form of convolution is known as *linear convolution.* Given 2 sequences x_1, x_2 of length M, N respectively, their linear convolution x_3 of length $M+N-1$ is defined as:

$$x_3[k] = \sum_{m=0}^{k} x_1[m]x_2[k-m] \qquad 0 \leq k < M+N-1$$

where $x_1[m]$ is taken to be 0 for $m > M$ and $x_2[m]$ is taken as 0 for $m > N$.

For example, consider the two sequences, of length 4 and 2, respectively:

$$< a_0, a_1, a_2, a_3 > \quad \text{and} \quad < b_o, b_1 >$$

Their linear convolution is:

$$< a_0 b_0,\ \ a_0 b_1 + a_1 b_0,\ \ a_1 b_1 + a_2 b_0,\ \ a_2 b_1 + a_3 b_0,\ \ a_3 b_1 >$$

This sort of convolution, does not preserve the duality with multiplication between the time/frequency domains. In order to benefit from the duality, the linear convolution needs to be expressed as a circular one. The transformation from linear to circular convolution can be achieved through the padding of the two sequences with 0's at their respective ends, thus making them both into sequences of length $M+N-1$. The resulting sequences are both regarded as periodic with periodicity $M+N-1$, and their circular convolution is the same as the linear convolution of the original sequences. More detailed description of this method can be found in [OS75, TAL89].

7 Some Applications and Conclusions

The former sections provided an introduction to the Fourier transform. The motivation behind it, was the wide use of the transform and its properties in various and diverse areas. In what follows we demonstrate several distinct ways in which the transform is applied. The examples differ in the use of DFT properties, and in the resulting benefits.

7.1 Polynomial Multiplication

The canonical example for using the FFT in computer science, is for fast multiplication of polynomials [CLR89, AHU74]. The observation underlying the algorithm is that when multiplying two polynomials, P_1 and P_2 of degrees $N-1$ and $M-1$ respectively, the $M+N-1$ coefficients of the resulting polynomial Q, are the result of convolving the coefficients of P_1 and P_2.

Using the convolution theorem of the DFT, one can execute the polynomial multiplication, by treating the coefficients of P_1 and P_2 as two discrete sequences of length N and M, respectively. The sequences are padded with 0's at their ends, to obtain length that is of the smallest power of 2 that is greater than $N+M-1$. Then the FFT is applied to both sequences, and pointwise multiplication of respective results is carried out. The inverse FFT maps the obtained results to the actual coefficients of the result polynomial. The whole process takes time which is $O((N+M-1)log(N+M-1))$, rather than $O((N+M-1)^2)$.

We note that in this case, the executions of both the FFT and the inverse FFT are actual steps of the algorithm.

7.2 Sequences Retrieval

A very different use of the FFT was recently demonstrated for fast retrieval of an explicitly given sequence, from a large database of stored sequences [AFS93, FRM94]. The problem addressed in this case is the need to compare the whole given query sequence to each of the whole stored sequences, retrieving the sequences which are within a certain Euclidean distance from the query.

Such comparison over a large database with long sequences, takes too much time, and is not feasible. To avoid it, rather than conducting search over the whole sequences, each sequence in the database is represented by its "fingerprints" which are the first few coefficients of its DFT. The query sequence is also transformed, and its first few DFT coefficients are compared against the coefficients stored in the databases.

Since most sequential data stored in databases can be well characterized by its lower frequencies, such representatives are indeed a reliable criteria for comparing sequences. Parseval's' theorem guarantees that sequences that are almost the same in the time domain, are also almost the same in the frequency domain. This ensures that if the search results in sequences that match the query sequence up to some error tolerance,*all* the correct matches have been retrieved, with the possibility of some false matches, which can be discarded later on.

The FFT is used here as a preprocessing step before storing a sequence in the database, and for converting the query sequence into a representation compatible with the database, before the search is executed. The inverse FFT is not used.

7.3 Filtering

The last example for using the DFT, is for reducing noise from data, as is done in the context of constructing a model through delayed coordinate embedding [Sau93]. The algorithm presented in the paper constructs a multi-dimensional model for a given sequence of 1-dimensional observations. An initial step in obtaining the higher dimensionality is the use of a sliding window over the observation sequence, which results in vectors of observations.

To filter out noise from the observation vectors before further processing them, a transformation is applied to each vector, which eliminates the high frequency components of the data. (Usually "noise" corresponds to high frequency data, while the actual data is characterized by low frequency). The transformation is equivalent to applying the DFT, multiplying the resulting vector by a sequence such that the lower coefficients (the low frequency coefficients) are multiplied by numbers close to 1 and the higher frequency coefficients are multiplied by numbers close to 0. The inverse DFT is applied to the result. The overall effect of the three operators, is equivalent to a single transformation (matrix) which is a low-pass filter. Multiplying each vector by this matrix results in a new vector which is the same as the original in its lower frequencies but missing the higher frequencies, thus it is a less "noisy" vector[8].

We note that in this case we don't algorithmically apply the FFT and the inverse FFT to each vector. The Fourier transform calculations were done in order to obtain the low-pass filter matrix, and multiplying a vector by this matrix has an equivalent effect to that of applying the FFT, multiplying by a sequence that eliminates high frequency components, and applying the inverse transform.

7.4 Concluding Remarks

Throughout the paper, we have demonstrated and emphasized that the Fourier transform is a way to represent functions and sequences, as a combination of sinusoids. That is, sequences and functions are spanned using sinusoids as a basis. Sinusoids are a good choice of basis for smooth "rounded " functions, which are periodic, continuous and differentiable at all points. However, for functions and sequences which correspond to square waves, (such as the pulse functions we have seen in the examples), or demonstrate non-periodic local phenomena, other forms of bases may prove simpler to use. The Haar wavelets, which are squared-shaped and compactly-supported functions, form a basis which spans square functions easily, and can be used conveniently to express *local* (non-periodic) high-frequency fluctuations. The wavelet transform is a generalization of the Fourier transform, and is actually a family of transforms. It allows a wide variety of function forms to serve as basis functions. For a quick introduction on the Wavelet transform and its applications see [SDS94]. An extensive discussion on wavelets can be found in [Dau92, Mal89].

The Fourier Transform is a broad subject, of which we have covered only a small fraction. We have taken a rather pragmatic approach, presenting the transform from a mathematical point of view, and providing intuition through examples and applications. A lot of the mathematical detail was omitted. Moreover, we have not addressed the engineering approach to the transform, which uses it to characterize systems. A discussion of what "systems" are, and the status of the Fourier transform with respect to them can be found in [OS89].

[8] Note that in Sauer's paper, in addition to filtering out noise, there is also a reduction in the *order* of the vector. The order reduction is equivalent to applying the inverse DFT only to part of the frequency coefficients (after eliminating the high frequency coefficients).

References

[AFS93] R. Agrawal, C. Faloutsos and A. Swami, *Efficient Similarity Search In Sequence Databases*, In The Fourth International Conference on Foundations of Data Organization and Algorithms, Evanston, Illinois, October 1993.

[AHU74] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms.* Addison & Wesley, 1974.

[BP85] C. S. Burrus and T. W. Parks, *DFT/FFT and Convolution Algorithms, Theory and Implementation.* John Wiley & Sons, 1985.

[Bra65] R. Bracewell, *The Fourier Transform and Its Applications.* McGraw-Hill, 1965.

[CLR89] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, chap. 32. McGraw-Hill, 1989.

[Dau92] I. Daubechies, *Ten Lectures on Wavelets*, Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1992.

[FRM94] C. Faloutsos, M. Ranganathan and Y. Manolopoulos, *Fast Subsequence Matching in Time-Series Databases*, In SIGMOD - Proceedings of Annual Conference, Minneapolis, May 1994.

[GKP91] R. L. Graham, D. E. Knuth and O. Patashnik, *Concrete Mathematics.* Addison & Wesley, 1991.

[Jac90] L. B. Jackson, *Signals, Systems and Transforms.* Addison & Wesley, 1990.

[Mal89] S. G. Mallat, *A Theory of Multiresolution Signal Decomposition: The Wavelet Representation*, IEEE Transactions on Pattern Analysis and Machine Intellidence, 11 , no. 7, 674–693, July 1989.

[OS75] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing.* Prentice-Hall, 1975.

[OS89] A. V. Oppenheim and R. W. Schafer, *Discrete Time Signal Processing.* Prentice-Hall, 1989.

[Sau93] T. Sauer, *Time Series Prediction by Using Delayed Coordinate Embedding*, In A. S. Weigend and N. A. Gershenfeld, editors, Time Series Prediction: Forecasting the Future and Understanding the Past. Addison & Wesley, 1993.

[Sav96] J. E. Savage, *Applied Theory of Computation*, chap. 3. Addison & Wesley, 1996, (to appear).

[SDS94] E. J. Stollnitz, T. D. Derose and D. H. Salesin, *Wavelets for Computer Graphics: A Primer*, Available through anonymous ftp, at cs.washington.edu, at pub/graphics/WaveletPrimer.ps.Z, 1994.

[TAL89] R. Tolimieri, M. An and C. Lu, *Algorithms for Discrete Fourier Transform and Convolution*, chap. 6. Springer-Verlag, 1989.

[Wea83] H. J. Weaver, *Applications of Discrete and Continuous Fourier Analysis*. John Wiley & Sons, 1983.