

# M10 - THE DISCRETE FOURIER TRANSFORM\*

C. Sidney Burrus

This work is produced by The Connexions Project and licensed under the  
Creative Commons Attribution License †

## Abstract

The Discrete Fourier Transform (DFT) maps a length- $N$  number sequence or signal into a length- $N$  frequency domain complex number sequence which is the transform of the signal. This is a fundamental operation in computational DSP.

## 1 The Discrete Fourier Transform

The description of signals in terms of their sinusoidal frequency content has proven to be as powerful and informative for discrete-time signals as it has for continuous-time signals. It is also probably the most powerful computational tool we will use. We now develop the basic discrete-time methods starting with the discrete Fourier transform (DFT) applied to finite length signals, followed by the discrete-time Fourier transform (DTFT) for infinitely long signals, and ending with the z-transform which uses the powerful tools of complex variable theory.

### 1.1 Definition of the DFT

It is assumed that the signal  $x(n)$  to be analyzed is a sequence of  $N$  real or complex values which are a function of the integer variable  $n$ . The DFT of  $x(n)$ , also called the spectrum of  $x(n)$ , is a length  $N$  sequence of complex numbers denoted  $C(k)$  and defined by

$$C(k) = \sum_{n=0}^{N-1} x(n) e^{-j \frac{2\pi}{N} nk} \quad (1)$$

using the usual engineering notation:  $j = \sqrt{-1}$ . The inverse transform (IDFT) which retrieves  $x(n)$  from  $C(k)$  is given by

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} C(k) e^{j \frac{2\pi}{N} nk} \quad (2)$$

which is easily verified by substitution into (1). Indeed, this verification will require using the orthogonality of the basis function of the DFT which is

$$\sum_{k=0}^{N-1} e^{-j \frac{2\pi}{N} mk} e^{j \frac{2\pi}{N} nk} = \begin{cases} N & \text{if } n = m \\ 0 & \text{if } n \neq m. \end{cases} \quad (3)$$

---

\*Version 1.1: Sep 17, 2006 11:55 am GMT-5

†<http://creativecommons.org/licenses/by/2.0/>

The exponential basis functions,  $e^{-j\frac{2\pi}{N}k}$ , for  $k \in \{0, N-1\}$ , are the  $N$  values of the  $N$ th roots of unity (the  $N$  zeros of the polynomial  $(s-1)^N$ ). This property is what connects the DFT to convolution and allows efficient algorithms for calculation to be developed [1]. They are used so often that the following notation is defined by

$$W_N = e^{-j\frac{2\pi}{N}} \quad (4)$$

with the subscript being omitted if the sequence length is obvious from context. Using this notation, the DFT becomes

$$C(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad (5)$$

One should notice that with the finite summation of the DFT, there is no question of convergence or of the ability to interchange the order of summation. No “delta functions” are needed and the  $N$  transform values can be calculated exactly (within the accuracy of the computer or calculator used) from the  $N$  signal values with a finite number of arithmetic operations.

## 1.2 Matrix Formulation of the DFT

There are several advantages to using a matrix formulation of the DFT. This is given by writing ((1)) or ((5)) in matrix operator form as

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ \vdots \\ C_{N-1} \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & W^0 & \dots & W^0 \\ W^0 & W^1 & W^2 & & \\ W^0 & W^2 & W^4 & & \\ \vdots & & & & \vdots \\ W^0 & & \dots & & W^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad (6)$$

or

$$\mathbf{C} = \mathbf{F}\mathbf{x}. \quad (7)$$

The orthogonality of the basis function in ((1)) shows up in this matrix formulation by the columns of  $\mathbf{F}$  being orthogonal to each other as are the rows. This means that  $\mathbf{F}^T\mathbf{F} = k\mathbf{I}$ , where  $k$  is a scalar constant, and, therefore,  $\mathbf{F}^T = k\mathbf{F}^{-1}$ . This is called a unitary operator.

The definition of the DFT in ((1)) emphasizes the fact that each of the  $N$  DFT values are the sum of  $N$  products. The matrix formulation in ((6)) has two interpretations. Each  $k$ -th DFT term is the inner product of two vectors,  $k$ -th row of  $\mathbf{F}$  and  $\mathbf{x}$ ; or, the DFT vector,  $\mathbf{C}$  is a weighted sum of the  $N$  columns of  $\mathbf{F}$  with weights being the elements of the signal vector  $\mathbf{x}$ . A third view of the DFT is the operator view which is simply the single matrix equation ((7)).

It is instructive at this point to write a computer program to calculate the DFT of a signal. In Matlab [2], there is a pre-programmed function to calculate the DFT, but that hides the scalar operations. One should program the transform in the scalar interpretive language of Matlab or some other lower level language such as FORTRAN, C, BASIC, Pascal, etc. This will illustrate how many multiplications and additions and trigonometric evaluations are required and how much memory is needed. Do not use a complex data type which also hides arithmetic, but use Euler's relations

$$e^{jx} = \cos(x) + j\sin(x) \quad (8)$$

to explicitly calculate the real and imaginary part of  $C(k)$ .

If Matlab is available, first program the DFT using only scalar operations. It will require two nested loops and will run rather slowly because the execution of loops is interpreted. Next, program it using vector

inner products to calculate each  $C(k)$  which will require only one loop and will run faster. Finally, program it using a single matrix multiplication requiring no loops and running much faster. Check the memory requirements of the three approaches.

The DFT and IDFT are a completely well-defined, legitimate transform pair with a sound theoretical basis that do not need to be derived from or interpreted as an approximation to the continuous-time Fourier series or integral. The discrete-time and continuous-time transforms and other tools are related and have parallel properties, but neither depends on the other.

The notation used here is consistent with most of the literature and with the standards given in [3]. The independent index variable  $n$  of the signal  $x(n)$  is an integer, but it is usually interpreted as time or, occasionally, as distance. The independent index variable  $k$  of the DFT  $C(k)$  is also an integer, but it is generally considered as frequency. The DFT is called the spectrum of the signal and the magnitude of the complex valued DFT is called the magnitude of that spectrum and the angle or argument is called the phase.

### 1.3 Extensions of $x(n)$

Although the finite length signal  $x(n)$  is defined only over the interval  $\{0 \leq n \leq (N-1)\}$ , the IDFT of  $C(k)$  can be evaluated outside this interval to give well defined values. Indeed, this process gives the periodic property 4. There are two ways of formulating this phenomenon. One is to periodically extend  $x(n)$  to  $-\infty$  and  $+\infty$  and work with this new signal. A second more general way is evaluate all indices  $n$  and  $k$  modulo  $N$ . Rather than considering the periodic extension of  $x(n)$  on the line of integers, the finite length line is formed into a circle or a line around a cylinder so that after counting to  $N-1$ , the next number is zero, not a periodic replication of it. The periodic extension is easier to visualize initially and is more commonly used for the definition of the DFT, but the evaluation of the indices by residue reduction modulo  $N$  is a more general definition and can be better utilized to develop efficient algorithms for calculating the DFT [1].

Since the indices are evaluated only over the basic interval, any values could be assigned  $x(n)$  outside that interval. The periodic extension is the choice most consistent with the other properties of the transform, however, it could be assigned to zero [5]. An interesting possibility is to artificially create a length  $2N$  sequence by appending  $x(-n)$  to the end of  $x(n)$ . This would remove the discontinuities of periodic extensions of this new length  $2N$  signal and perhaps give a more accurate measure of the frequency content of the signal with no artifacts caused by "end effects". Indeed, this modification of the DFT gives what is called the discrete cosine transform (DCT) [4]. We will assume the implicit periodic extensions to  $x(n)$  with no special notation unless this characteristic is important, then we will use the notation  $\tilde{x}(n)$ .

### 1.4 Convolution

Convolution is an important operation in signal processing that is in some ways more complicated in discrete-time signal processing than in continuous-time signal processing and in other ways easier. The basic input-output relation for a discrete-time system is given by so-called linear or non-cyclic convolution defined and denoted by

$$y(n) = \sum_{m=-\infty}^{\infty} h(m) x(n-m) = h(n) * x(n) \quad (9)$$

where  $x(n)$  is the perhaps infinitely long input discrete-time signal,  $h(n)$  is the perhaps infinitely long impulse response of the system, and  $y(n)$  is the output. The DFT is, however, intimately related to cyclic convolution, not non-cyclic convolution. Cyclic convolution is defined and denoted by

$$\tilde{y}(n) = \sum_{m=0}^{N-1} \tilde{h}(m) \tilde{x}(n-m) = h(n) \circ x(n) \quad (10)$$

where either all of the indices or independent integer variables are evaluated modulo  $N$  or all of the signals are periodically extended outside their length  $N$  domains.

This cyclic (sometimes called circular) convolution can be expressed as a matrix operation by converting the signal  $h(n)$  into a matrix operator as

$$\mathbf{H} = \begin{bmatrix} h_0 & h_{L-1} & h_{L-2} & \cdots & h_1 \\ h_1 & h_0 & h_{L-1} & & \\ h_2 & h_1 & h_0 & & \\ \vdots & & & & \vdots \\ h_{L-1} & & \cdots & & h_0 \end{bmatrix}, \quad (11)$$

The cyclic convolution can then be written in matrix notation as

$$\mathbf{Y} = \mathbf{H}\mathbf{X} \quad (12)$$

where  $\mathbf{X}$  and  $\mathbf{Y}$  are column matrices or vectors of the input and output values respectively.

Because non-cyclic convolution is often what you want to do and cyclic convolution is what is related to the powerful DFT, we want to develop a way of doing non-cyclic convolution by doing cyclic convolution.

The convolution of a length  $N$  sequence with a length  $M$  sequence yields a length  $N + M - 1$  output sequence. The calculation of non-cyclic convolution by using cyclic convolution requires modifying the signals by appending zeros to them. This will be developed later.

### 1.5 Examples of the DFT

It is very important to develop insight and intuition into the DFT or spectral characteristics of various standard signals. A few DFT's of standard signals together with the above properties will give a fairly large set of results. They will also aid in quickly obtaining the DFT of new signals. The discrete-time impulse  $\delta(n)$  is defined by

$$\delta(n) = \begin{cases} 1 & \text{when } n = 0 \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

The discrete-time pulse  $\Pi_M(n)$  is defined by

$$\Pi_M(n) = \begin{cases} 1 & \text{when } n = 0, 1, \dots, M - 1 \\ 0 & \text{otherwise} \end{cases} \quad (14)$$

Several examples are:

- $DFT\{\delta(n)\} = 1$ , The DFT of an impulse is a constant.
- $DFT\{1\} = N\delta(k)$ , The DFT of a constant is an impulse.
- $DFT\{e^{j2\pi Kn/N}\} = N\delta(k - K)$
- $DFT\{\cos(2\pi Mn/N)\} = \frac{N}{2} [\delta(k - M) + \delta(k + M)]$
- $DFT\{\Pi_M(n)\} = \frac{\sin(\frac{\pi}{N}Mk)}{\sin(\frac{\pi}{N}k)}$

These examples together with the properties can generate a still larger set of interesting and enlightening examples. Matlab can be used to experiment with these results and to gain insight and intuition.

## References

- [1] C. S. Burrus and T. W. Parks. *DFT/FFT and Convolution Algorithms*. John Wiley & Sons, New York, 1985.

- [2] John Little Cleve Moler and Steve Bangert. *Matlab User's Guide*. The MathWorks, Inc., South Natick, MA, 1989.
- [3] DSP Committee, editor. *Digital Signal Processing II, selected reprints*. IEEE Press, New York, 1979.
- [4] D. F. Elliott and K. F. Rao. *Fast Transforms: Algorithms, Analyses and Applications*. Academic Press, New York, 1982.
- [5] A. V. Oppenheim and R. W. Schaffer. *Discrete-Time Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1989.