Lecture 7 - The Discrete Fourier Transform

7.1 The DFT

The Discrete Fourier Transform (DFT) is the equivalent of the continuous Fourier Transform for signals known only at N instants separated by sample times T (i.e. a finite sequence of data).
Let $f(t)$ be the continuous signal which is the source of the data. Let N samples be denoted $f[0], f[1], f[2], \ldots, f[k], \ldots, f[N-1]$.
The Fourier Transform of the original signal, $f(t)$, would be

$$F(j\omega) \;=\; \int_{-\infty}^{\infty} f(t)e^{-j\omega t}dt$$

We could regard each sample $f[k]$ as an *impulse* having area $f[k]$. Then, since the integrand exists only at the sample points:

$$\begin{aligned} F(j\omega) &= \int_{o}^{(N-1)T} f(t)e^{-j\omega t}dt \\ &= f[0]e^{-j0} + f[1]e^{-j\omega T} + \ldots + f[k]e^{-j\omega kT} + \ldots f(N-1)e^{-j\omega(N-1)T} \end{aligned}$$

$$\text{ie.} \qquad F(j\omega) = \sum_{k=0}^{N-1} f[k]e^{-j\omega kT}$$

We could in principle evaluate this for any ω, but with only N data points to start with, only N final outputs will be significant.

You may remember that the continuous Fourier transform could be evaluated over a finite interval (usually the fundamental period T_o) rather than from $-\infty$ to

$+\infty$ if the waveform was *periodic*. Similarly, since there are only a finite number of input data points, the DFT treats the data as if it were periodic (i.e. $f(N)$ to $f(2N-1)$ is the same as $f(0)$ to $f(N-1)$.)
Hence the sequence shown below in Fig. 7.1(a) is considered to be one period of the periodic sequence in plot (b).

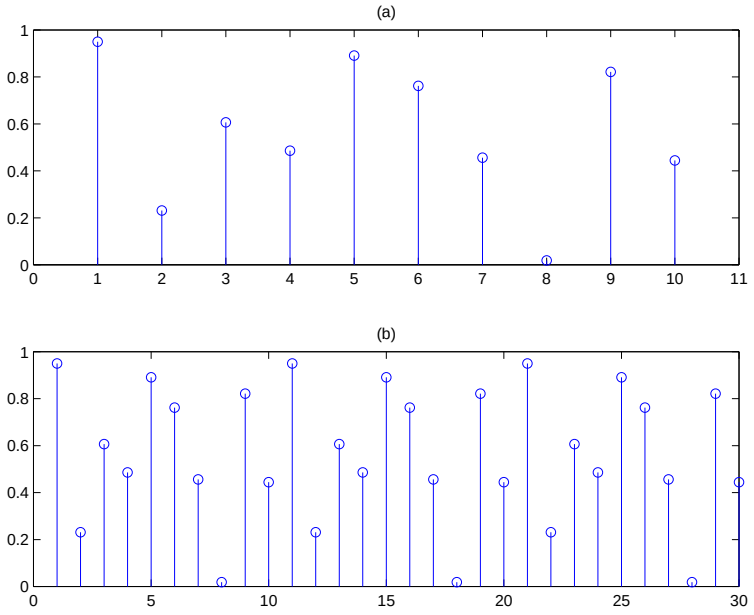

Figure 7.1: *(a) Sequence of $N = 10$ samples. (b) implicit periodicity in DFT.*

Since the operation treats the data as if it were periodic, we evaluate the DFT equation for the fundamental frequency (one cycle per sequence, $\frac{1}{NT}$Hz, $\frac{2\pi}{NT}$ rad/sec.) and its harmonics (not forgetting the d.c. component (or average) at $\omega = 0$).

$$\text{i.e.} \quad \text{set } \omega = 0, \frac{2\pi}{NT}, \frac{2\pi}{NT} \times 2, \ldots \frac{2\pi}{NT} \times n, \ldots \frac{2\pi}{NT} \times (N-1)$$

or, in general

$$F[n] = \sum_{k=0}^{N-1} f[k] e^{-j\frac{2\pi}{N}nk} \qquad (n = 0 : N-1)$$

$F[n]$ is the Discrete Fourier Transform of the sequence $f[k]$.

We may write this equation in matrix form as:

$$\begin{pmatrix} F[0] \\ F[1] \\ F[2] \\ \vdots \\ F[N-1] \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & W & W^2 & W^3 & \dots & W^{N-1} \\ 1 & W^2 & W^4 & W^6 & \dots & W^{N-2} \\ 1 & W^3 & W^6 & W^9 & \dots & W^{N-3} \\ \vdots & & & & & \\ 1 & W^{N-1} & W^{N-2} & W^{N-3} & \dots & W \end{pmatrix} \begin{pmatrix} f[0] \\ f[1] \\ f[2] \\ \vdots \\ f[N-1] \end{pmatrix}$$

where $W = \exp(-j2\pi/N)$ and $W = W^{2N}$ etc. $= 1$.

DFT – example

Let the continuous signal be

$$f(t) = \underbrace{5}_{\text{dc}} + \underbrace{2\cos(2\pi t - 90^o)}_{\text{1Hz}} + \underbrace{3\cos 4\pi t}_{\text{2Hz}}$$

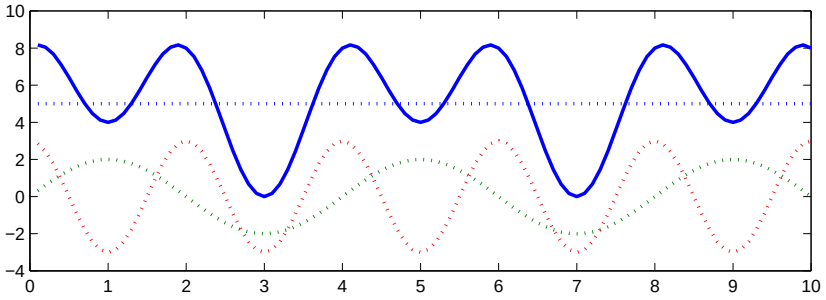

Figure 7.2: *Example signal for DFT.*

Let us sample $f(t)$ at 4 times per second (ie. f_s = 4Hz) from $t = 0$ to $t = \frac{3}{4}$. The values of the discrete samples are given by:

$$f[k] = 5 + 2cos(\tfrac{\pi}{2}k - 90^o) + 3cos\pi k \qquad \text{by putting } t = kT_s = \tfrac{k}{4}$$

i.e. $f[0] = 8, f[1] = 4, f[2] = 8, f[3] = 0, \qquad (N = 4)$

$$\text{Therefore} \quad F[n] = \sum_{0}^{3} f[k] e^{-j\frac{\pi}{2}nk} = \sum_{k=0}^{3} f[k](-j)^{nk}$$

$$\begin{pmatrix} F[0] \\ F[1] \\ F[2] \\ F[3] \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix} \begin{pmatrix} f[0] \\ f[1] \\ f[2] \\ f[3] \end{pmatrix} = \begin{pmatrix} 20 \\ -j4 \\ 12 \\ j4 \end{pmatrix}$$

The magnitude of the DFT coefficients is shown below in Fig. 7.3.

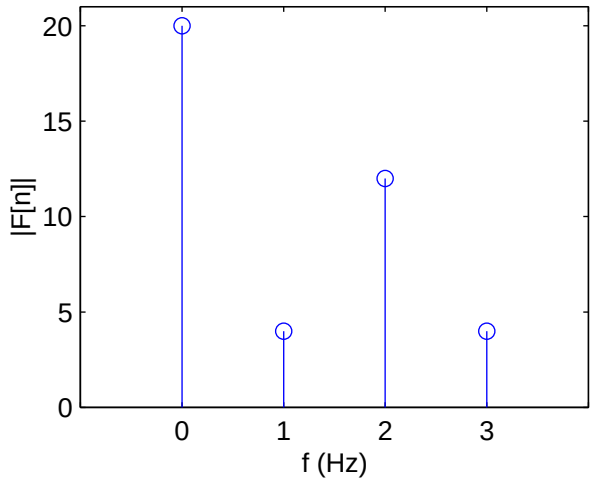

Figure 7.3: *DFT of four point sequence.*

Inverse Discrete Fourier Transform

The inverse transform of

$$F[n] = \sum_{k=0}^{N-1} f[k] e^{-j\frac{2\pi}{N}nk}$$

is

$$f[k] = \frac{1}{N}\sum_{n=0}^{N-1} F[n]e^{+j\frac{2\pi}{N}nk}$$

i.e. the inverse matrix is $\frac{1}{N}$ times the complex conjugate of the original (symmetric) matrix.

Note that the $F[n]$ coefficients are *complex*. We can assume that the $f[k]$ values are *real* (this is the simplest case; there are situations (e.g. radar) in which two inputs, at each k, are treated as a *complex pair*, since they are the outputs from 0° and 90° demodulators).

In the process of taking the inverse transform the terms $F[n]$ and $F[N-n]$ (remember that the spectrum is symmetrical about $\frac{N}{2}$) combine to produce 2 frequency components, only one of which is considered to be valid (the one at the *lower* of the two frequencies, $n \times \frac{2\pi}{T}$ Hz where $n \leq \frac{N}{2}$; the higher frequency component is at an "aliasing frequency" ($n > \frac{N}{2}$)).

From the inverse transform formula, the contribution to $f[k]$ of $F[n]$ and $F[N-n]$ is:

$$f_n[k] = \frac{1}{N}\{F[n]e^{j\frac{2\pi}{N}nk} + F[N-n]e^{j\frac{2\pi}{N}(N-n)k}\} \tag{7.2}$$

$$\text{For all} f[k] \text{ real}, \qquad F[N-n] = \sum_{k=0}^{N-1} f[k]e^{-j\frac{2\pi}{N}(N-n)k}$$

$$\text{But} \quad e^{-j\frac{2\pi}{N}(N-n)k} = \underbrace{e^{-j2\pi k}}_{1 \text{ for all } k}\ e^{+j\frac{2\pi n}{N}k} = e^{+j\frac{2\pi}{N}nk}$$

$$\text{i.e. } F[N-n] = F^*(n) \qquad \text{(i.e. the complex conjugate)}$$

Substituting into the Equation for $f_n[k]$ above gives,

$$f_n[k] = \frac{1}{N}\{F[n]e^{j\frac{2\pi}{N}nk} + F^*(n)e^{-j\frac{2\pi}{N}nk}\} \quad \text{since} \quad e^{j2\pi k} = 1$$

$$\text{ie.} \quad f_n[k] = \frac{2}{N}\{\mathsf{Re}\{F[n]\}\cos\frac{2\pi}{N}nk - \mathsf{Im}\{F[n]\}\sin\frac{2\pi}{N}nk\}$$

$$\text{or} \quad f_n[k] = \frac{2}{N}|F[n]|\cos\{(\frac{2\pi}{NT}n)kT + \arg(F[n])\}$$

i.e. a sampled sinewave at $\frac{2\pi n}{NT}$ Hz, of magnitude $\frac{2}{N}|F[n]|$.

For the special case of $n = 0$, $F[0] = \sum f[k]$ (i.e. sum of all samples) and the contribution of $F[0]$ to $f[k]$ is $f_0[k] = \frac{1}{N}F[0] =$ average of $f[k] =$ d.c. component.

Interpretation of example

1. $F[0] = 20$ implies a d.c. value of $\frac{1}{N}F[0] = \frac{20}{4} = 5$ (as expected)

2. $F[1] = -j4 = F^*[3]$ implies a fundamental component of peak amplitude $\frac{2}{N}|F[1]| = \frac{2}{4} \times 4 = 2$ with phase given by $\arg F[1] = -90^{\circ}$

   $$\text{i.e.} \quad 2\cos(\frac{2\pi}{NT}kT - 90^{\circ}) = 2\cos(\frac{\pi}{2}k - 90^{\circ}) \quad \text{(as expected)}$$

3. $F[2] = 12$ $(n = \frac{N}{2}$ – no other $N - n$ component here) and this implies a component

   $$f_2[k] = \frac{1}{N}F[2]e^{j\frac{2\pi}{N}\cdot 2k} = \frac{1}{4}F[2]e^{j\pi k} = 3cos\pi k \quad \text{(as expected)}$$

   since $\sin \pi k = 0$ for all k

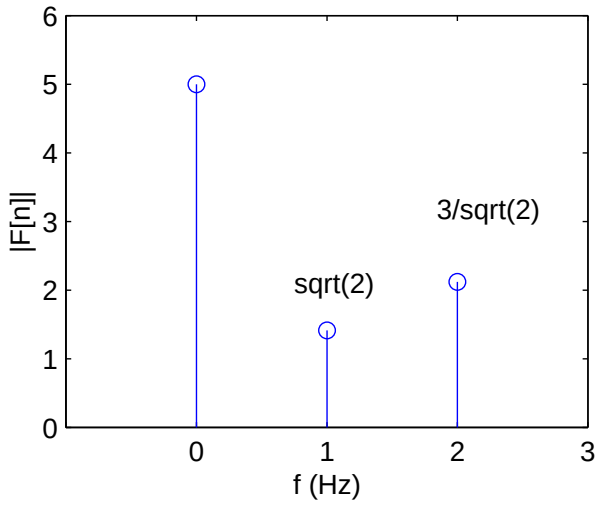

Figure 7.4: *DFT of four point signal.*

Thus, the conventional way of displaying a *spectrum* is not as shown in Fig. 7.3 but as shown in Fig. 7.4 (obviously, the information content is the same):
In typical applications, N is much greater than 4; for example, for $N = 1024$, $F[n]$ has 1024 components, but $513-1023$ are the complex conjugates of $511-1$, leaving $\frac{F[0]}{1024}$ as the d.c. component, $\frac{2}{1024}\frac{|F[1]|}{\sqrt{2}}$ to $\frac{2}{1024}\frac{|F[511]|}{\sqrt{2}}$ as complete a.c. components and $\frac{1}{1024}\frac{F[512]}{\sqrt{2}}$ as the cosine-only component at the highest distinguishable frequency $(n = \frac{N}{2})$.

Most computer programmes evaluate $\frac{|F[n]|}{N}$ (or $\frac{|F[n]|^2}{N}$ for the power spectral density) which gives the correct "shape" for the spectrum, except for the values at $n = 0$ and $\frac{N}{2}$.

7.2 Discrete Fourier Transform Errors

To what degree does the DFT approximate the Fourier transform of the function underlying the data? Clearly the DFT is only an approximation since it provides only for a finite set of frequencies. But how correct are these discrete values themselves? There are two main types of DFT errors: aliasing and "leakage":

7.2.1 Aliasing

This is another manifestation of the phenomenon which we have now encountered several times. If the initial samples are not sufficiently closely spaced to represent high-frequency components present in the underlying function, then the DFT values will be corrupted by aliasing. As before, the solution is either to increase the sampling rate (if possible) or to pre-filter the signal in order to minimise its high-frequency spectral content.

7.2.2 Leakage

Recall that the continuous Fourier transform of a periodic waveform requires the integration to be performed over the interval - ∞ to $+\infty$ or over an integer number of cycles of the waveform. If we attempt to complete the DFT over a non-integer number of cycles of the input signal, then we might expect the transform to be corrupted in some way. This is indeed the case, as will now be shown.

Consider the case of an input signal which is a sinusoid with a fractional number of cycles in the N data samples. The DFT for this case (for $n = 0$ to $n = \frac{N}{2}$) is shown below in 7.5.

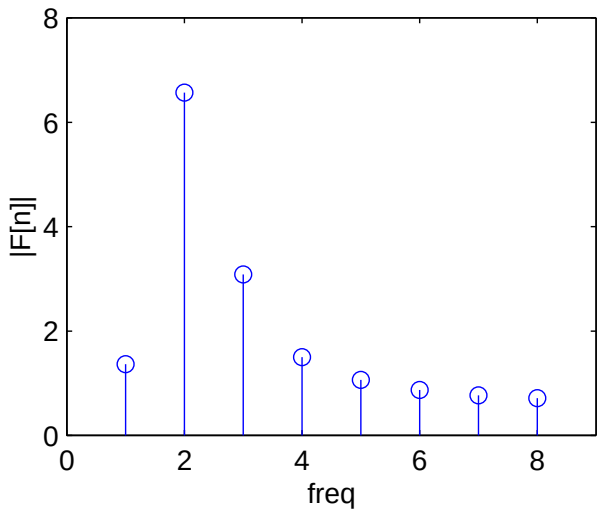

Figure 7.5: *Leakage.*

We might have expected the DFT to give an output at just the quantised frequen-

cies either side of the true frequency. This certainly does happen but we also find non-zero outputs at all other frequencies. This smearing effect, which is known as leakage, arises because we are effectively calculating the Fourier series for the waveform in Fig. 7.6, which has major discontinuities, hence other frequency components.

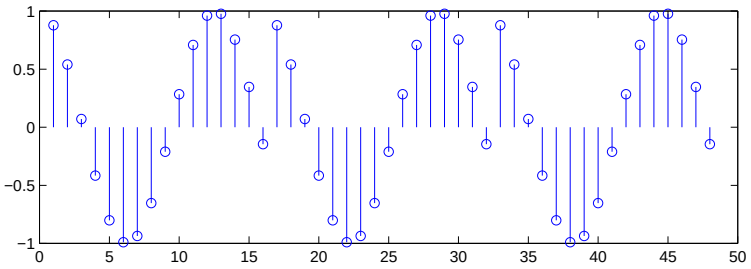

Figure 7.6: *Leakage. The repeating waveform has discontinuities.*

Most sequences of real data are much more complicated than the sinusoidal sequences that we have so far considered and so it will not be possible to avoid introducing discontinuities when using a finite number of points from the sequence in order to calculate the DFT.
The solution is to use one of the *window functions* which we encountered in the design of FIR filters (e.g. the Hamming or Hanning windows). These window functions taper the samples towards zero values at both endpoints, and so there is no discontinuity (or very little, in the case of the Hanning window) with a hypothetical next period. Hence the leakage of spectral content away from its correct location is much reduced, as in Fig 7.7.

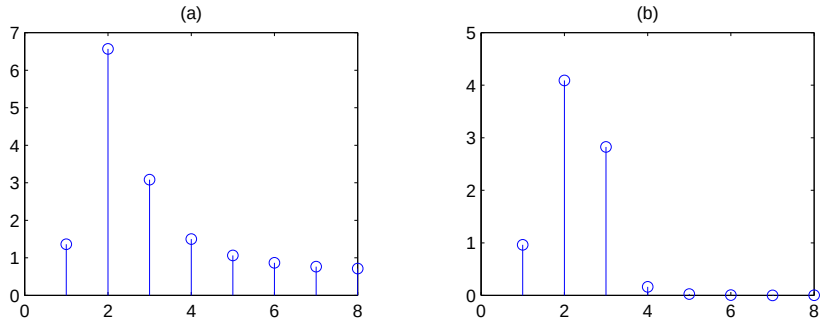

Figure 7.7: *Leakage is reduced using a Hanning window.*

7.3 The Fast Fourier Transform

The time taken to evaluate a DFT on a digital computer depends principally on the number of multiplications involved, since these are the slowest operations. With the DFT, this number is directly related to N^2 (matrix multiplication of a vector), where N is the length of the transform. For most problems, N is chosen to be at least 256 in order to get a reasonable approximation for the spectrum of the sequence under consideration – hence computational speed becomes a major consideration.

Highly efficient computer algorithms for estimating Discrete Fourier Transforms have been developed since the mid-60's. These are known as Fast Fourier Transform (FFT) algorithms and they rely on the fact that the standard DFT involves a lot of redundant calculations:

$$\text{Re-writing} \quad F[n] = \sum_{k=0}^{N-1} f[k] e^{-j\frac{2\pi}{N}nk} \text{ as} F[n] = \sum_{k=0}^{N-1} f[k] W_N^{nk}$$

it is easy to realise that the same values of W_N^{nk} are calculated many times as the computation proceeds. Firstly, the integer product nk repeats for different combinations of k and n; secondly, W_N^{nk} is a periodic function with only N distinct values.

For example, consider $N = 8$ (the FFT is simplest by far if N is an integral power of 2)

$$W_8^1 = e^{-j\frac{2\pi}{8}} = e^{-j45^o} = \frac{1-j}{\sqrt{2}} = a, \text{ say.}$$

Then $a^2 = -j \qquad a^3 = -ja = -a^* \qquad a^4 = -1$

$$a^5 = -a \qquad a^6 = j \qquad a^7 = ja = a^* \qquad a^8 = 1$$

From the above, it can be seen that:

$$W_8^4 = -W_8^0$$
$$W_8^5 = -W_8^1$$
$$W_8^6 = -W_8^2$$
$$W_8^7 = -W_8^3$$

Also, if nk falls outside the range 0-7, we still get one of the above values:

$$\text{eg. if} \quad n = 5 \text{ and } k = 7, \quad W_8^{35} = a^{35} = (a^8)^4 \cdot a^3 = a^3$$

7.3.1 Decimation-in-time algorithm

Let us begin by splitting the single summation over N samples into 2 summations, each with $\frac{N}{2}$ samples, one for k even and the other for k odd.
Substitute $m = \frac{k}{2}$ for k even and $m = \frac{k-1}{2}$ for k odd and write:

$$F[n] = \sum_{m=0}^{\frac{N}{2}-1} f[2m] W_N^{2mn} + \sum_{m=0}^{\frac{N}{2}-1} f[2m+1] W_N^{(2m+1)n}$$

$$\text{Note that} \quad W_N^{2mn} = e^{-j\frac{2\pi}{N}(2mn)} = e^{-j\frac{2\pi}{\frac{N}{2}}mn} = W_{\frac{N}{2}}^{mn}$$

$$\text{Therefore} \quad F[n] = \sum_{m=0}^{\frac{N}{2}-1} f[2m] W_{\frac{N}{2}}^{mn} + W_N^m \sum_{m=0}^{\frac{N}{2}-1} f[2m+1] W_{\frac{N}{2}}^{mn}$$

$$\text{ie.} \quad F[n] = G[n] + W_N^m H[n]$$

Thus the N-point DFT $F[n]$ can be obtained from two $\frac{N}{2}$-point transforms, one on even input data, $G[n]$, and one on odd input data, $H[n]$. Although the frequency index n ranges over N values, only $\frac{N}{2}$ values of $G[n]$ and $H[n]$ need to be computed since $G[n]$ and $H[n]$ are periodic in n with period $\frac{N}{2}$.

For example, for $N = 8$:

- even input data $f[0]f[2]f[4]f[6]$
- odd input data $f[1]f[3]f[5]f[7]$

$$
\begin{aligned}
F[0] &= G[0] + W_8^0 H[0] \\
F[1] &= G[1] + W_8^1 H[1] \\
F[2] &= G[2] + W_8^2 H[2] \\
F[3] &= G[3] + W_8^3 H[3] \\
F[4] &= G[0] + W_8^4 H[0] = G[0] - W_8^0 H[0] \\
F[5] &= G[1] + W_8^5 H[1] = G[1] - W_8^1 H[1] \\
F[6] &= G[2] + W_8^6 H[2] = G[2] - W_8^2 H[2] \\
F[7] &= G[3] + W_8^7 H[3] = G[3] - W_8^3 H[3]
\end{aligned}
$$

This is shown graphically on the flow graph of Fig 7.8:

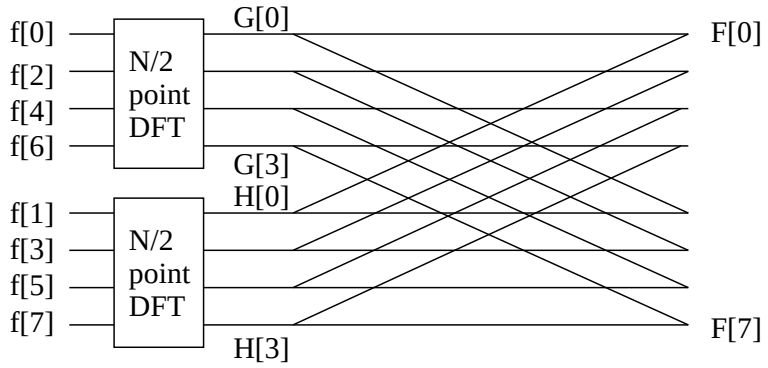

Figure 7.8: *FFT flow graph 1.*

Assuming than N is a power of 2, we can repeat the above process on the two $\frac{N}{2}$-point transforms, breaking them down to $\frac{N}{4}$-point transforms, etc. . . , until we come down to 2-point transforms. For $N = 8$, only one further stage is needed (i.e. there are γ stages, where $N = 2^\gamma$), as shown below in Fig 7.9.

Thus the FFT is computed by dividing up, or *decimating*, the sample sequence $f[k]$ into sub-sequences until only 2-point DFT's remain. Since it is the input, or time, samples which are divided up, this algorithm is known as the *decimation-in-time* (DIT) algorithm. (An equivalent algorithm exists for which the output, or frequency, points are sub-divided – the decimation-in-frequency algorithm.)

The basic computation at the heart of the FFT is known as the *butterfly* because of its criss-cross appearance. For the DIT FFT algorithm, the butterfly computation is of the form of Fig 7.10

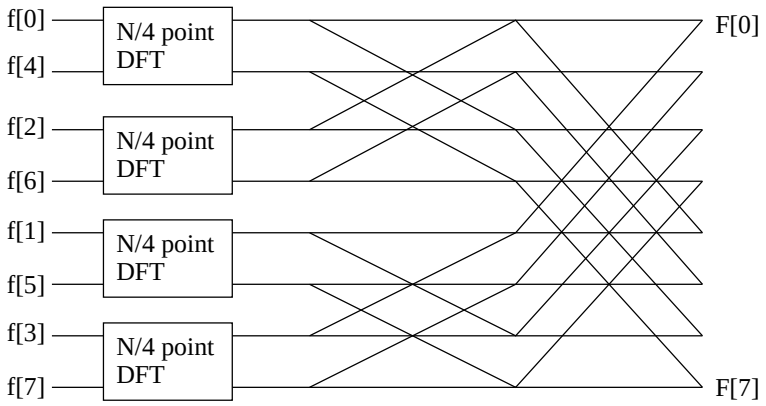

Figure 7.9: *FFT flow graph 2.*

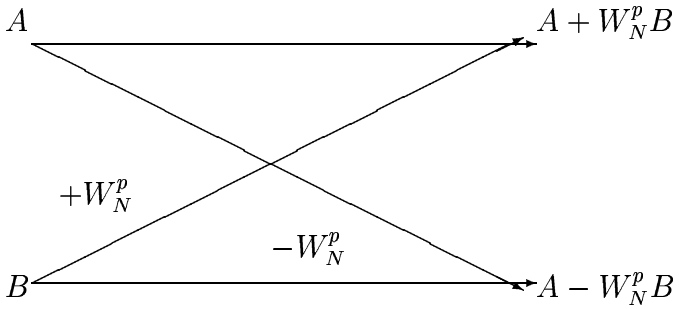

Figure 7.10: *Butterfly operation in FFT.*

where A and B are complex numbers. Thus a butterfly computation requires one complex multiplication and 2 complex additions.

Note also, that the input samples are "bit-reversed" (see table below) because at each stage of decimation the sequence input samples is separated into even- and odd- indexed samples.

(*NB*: the bit-reversal algorithm only applies if N is an integral power of 2).

Index (k)	Binary representation	Bit-reversed Binary	Bit-reversed index
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

7.3.2 Computational speed of FFT

The DFT requires N^2 complex multiplications. At each stage of the FFT (i.e. each halving) $\frac{N}{2}$ complex multiplications are required to combine the results of the previous stage. Since there are $(log_2 N)$ stages, the number of complex multiplications required to evaluate an N-point DFT with the FFT is approximately $N/2log_2 N$ (approximately because multiplications by factors such as W_N^0, $W_N^{\frac{N}{2}}$, $W_N^{\frac{N}{4}}$ and $W_N^{\frac{3N}{4}}$ are really just complex additions and subtractions).

N	N^2 (DFT)	$\frac{N}{2}log_2 N$ (FFT)	saving
32	1,024	80	92%
256	65,536	1,024	98%
1,024	1,048,576	5,120	99.5%

7.3.3 Practical considerations

If N is not a power of 2, there are 2 strategies available to complete an N-point FFT.

1. take advantage of such factors as N possesses. For example, if N is divisible by 3 (e.g. $N = 48$), the final decimation stage would include a 3-point transform.

2. pack the data with zeroes; e.g. include 16 zeroes with the 48 data points (for $N = 48$) and compute a 64-point FFT. (However, you should again be wary of abrupt transitions between the trailing (or leading) edge of the data and the following (or preceding) zeroes; a better approach might be to pack the data with more realistic "dummy values").