Lecture 10

- Discrete Fourier Transform (DFT)
- Fast Fourier Transform (FFT)

What you will learn in Lecture 10

10.1 Discrete Fourier Transform

10.2 Fast Fourier Transform

10.1 Discrete Fourier Transform

10.1 Discrete Fourier Transform

Additional Example 1

Find the Discrete Fourier Transform of $\{f[0], f[1], f[2], f[3]\} = \{0, 1, 0, 0\}, N = 4.$

Solution

By using $$\mathcal{F}[k] = \sum_{n=0}^{3} f[n]\, e^{-i\frac{2\pi}{N}nk}$$

$$\mathcal{F}[0] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 0}$$

$$= 0\cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} + 1\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} + 0\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} + 0\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 0}$$

$$= e^{0} \quad = 1$$

$$\mathcal{F}[1] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 1}$$

$$= 0\cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} + 1\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} + 0\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} + 0\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 1}$$

$$= \underline{e^{-i\frac{\pi}{2}} \quad = \cos\frac{\pi}{2} - i\sin\frac{\pi}{2}} \quad = -i$$

By introducing Euler's Formula

10.1 Discrete Fourier Transform

Additional Example 1

Find the Discrete Fourier Transform of $\{f[0], f[1], f[2], f[3]\} = \{0, 1, 0, 0\}, N = 4.$

$$\mathcal{F}[2] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 2}$$

$$= 0 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} + 1 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 2}$$

$$= e^{-i\pi} \quad = \cos\pi - i\sin\pi \quad = -1$$

$$\mathcal{F}[3] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 3}$$

$$= 0 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} + 1 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 3}$$

$$= e^{-i\frac{3\pi}{2}} = \cos\frac{3\pi}{2} - i\sin\frac{3\pi}{2} \quad = i$$

10.1 Discrete Fourier Transform

Additional Example 2

Find the Discrete Fourier Transform of $\{f[0], f[1], f[2], f[3]\} = \{1, 2-i, -i, -1+2i\}, N = 4.$

Solution

By using $\mathcal{F}[k] = \sum_{n=0}^{3} f[n]\, e^{-i\frac{2\pi}{N}nk}$

$$\mathcal{F}[0] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 0}$$

$$= 1\cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} + (2-i)\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} + (-i)\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} + (-1+2i)\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 0}$$

$$= 1 + (2-i) + (-i) + (-1+2i) \quad = 2$$

$$\mathcal{F}[1] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 1}$$

$$= 1\cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} + (2-i)\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} + (-i)\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} + (-1+2i)\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 1}$$

$$= 1 + (2-i)\cdot e^{-i\frac{\pi}{2}} + (-i)\cdot e^{-i\pi} + (-1+2i)\cdot e^{-i\frac{3\pi}{2}} \quad \underline{= -2-2i}$$

By introducing Euler's Formula $e^{-ia} = \cos a - i \sin a$

Additional Example 2

Find the Discrete Fourier Transform of $\{f[0], f[1], f[2], f[3]\} = \{1, 2-i, -i, -1+2i\}, N = 4.$

$$\mathcal{F}[2] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 2}$$

$$= 1 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} + (2-i)\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} + (-i)\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} + (-1+2i)\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 2}$$

$$= 1 + (2-i)\cdot e^{-i\pi} + (-i)\cdot e^{-i2\pi} + (-1+2i)\cdot e^{-i3\pi} \quad = -2i$$

$$\mathcal{F}[3] = f[0]e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} + f[1]e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} + f[2]e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} + f[3]e^{-i\frac{2\pi}{4}\cdot 3\cdot 3}$$

$$= 1 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} + (2-i)\cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} + (-i)\cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} + (-1+2i)\cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 3}$$

$$= 1 + (2-i)\cdot e^{-i\frac{3\pi}{2}} + (-i)\cdot e^{-i3\pi} + (-1+2i)\cdot e^{-i\frac{9\pi}{2}} = 4 + 4i$$

Additional Example 3

Find the Discrete Fourier Transform of $\{f[0], f[1], f[2], f[3]\} = \{8, 4, 8, 0\}, N = 4$.

Solution

By using $\mathcal{F}[k] = \sum_{n=0}^{3} f[n]\, e^{-i\frac{2\pi}{N}nk}$

$$\left.\begin{aligned}
\mathcal{F}[0] &= 8 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} + 4 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} + 8 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 0} \\
\mathcal{F}[1] &= 8 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} + 4 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} + 8 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 1} \\
\mathcal{F}[2] &= 8 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} + 4 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} + 8 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 2} \\
\mathcal{F}[3] &= 8 \cdot e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} + 4 \cdot e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} + 8 \cdot e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} + 0 \cdot e^{-i\frac{2\pi}{4}\cdot 3\cdot 3}
\end{aligned}\right\} \Rightarrow$$

Let $\boldsymbol{W}_N = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{(N-1)} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & W_N^{(N-1)} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix}$

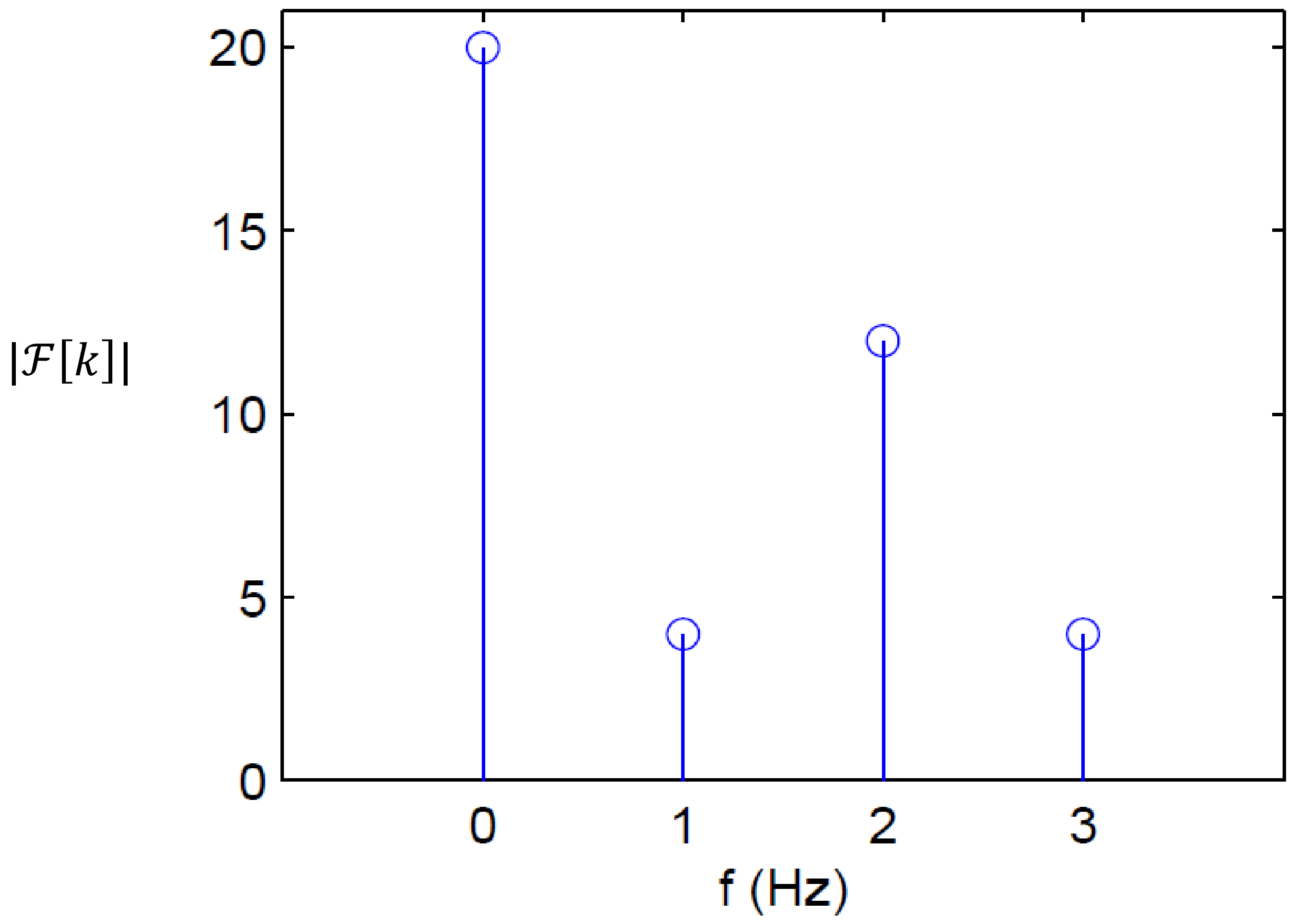
$$\begin{bmatrix} \mathcal{F}[0] \\ \mathcal{F}[1] \\ \mathcal{F}[2] \\ \mathcal{F}[3] \end{bmatrix} = \begin{bmatrix} e^{-i\frac{2\pi}{4}\cdot 0\cdot 0} & e^{-i\frac{2\pi}{4}\cdot 1\cdot 0} & e^{-i\frac{2\pi}{4}\cdot 2\cdot 0} & e^{-i\frac{2\pi}{4}\cdot 3\cdot 0} \\ e^{-i\frac{2\pi}{4}\cdot 0\cdot 1} & e^{-i\frac{2\pi}{4}\cdot 1\cdot 1} & e^{-i\frac{2\pi}{4}\cdot 2\cdot 1} & e^{-i\frac{2\pi}{4}\cdot 3\cdot 1} \\ e^{-i\frac{2\pi}{4}\cdot 0\cdot 2} & e^{-i\frac{2\pi}{4}\cdot 1\cdot 2} & e^{-i\frac{2\pi}{4}\cdot 2\cdot 2} & e^{-i\frac{2\pi}{4}\cdot 3\cdot 2} \\ e^{-i\frac{2\pi}{4}\cdot 0\cdot 3} & e^{-i\frac{2\pi}{4}\cdot 1\cdot 3} & e^{-i\frac{2\pi}{4}\cdot 2\cdot 3} & e^{-i\frac{2\pi}{4}\cdot 3\cdot 3} \end{bmatrix} \begin{bmatrix} f[0] \\ f[1] \\ f[2] \\ f[3] \end{bmatrix}$$

i.e. $\boldsymbol{\mathcal{F}} = \boldsymbol{W}_N \boldsymbol{f}$

Therefore, we obtain $\begin{bmatrix} \mathcal{F}[0] \\ \mathcal{F}[1] \\ \mathcal{F}[2] \\ \mathcal{F}[3] \end{bmatrix} = \begin{bmatrix} 20 \\ -4i \\ 12 \\ 4i \end{bmatrix}$

10.1 Discrete Fourier Transform

The magnitude of the DFT coefficients is shown below in Figure.



DFT of 4-point sequence

THEOREM 2 TRANSFORMS OF CONVOLUTIONS

For any two N-sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ we have

$$\mathcal{F}_N[\boldsymbol{x} * \boldsymbol{y}] = \mathcal{F}_N[\boldsymbol{x}]\mathcal{F}_N[\boldsymbol{y}]$$

$$\mathcal{F}_N^{-1}[\boldsymbol{X}\boldsymbol{Y}] = \boldsymbol{x} * \boldsymbol{y}$$

10.1 Discrete Fourier Transform

EXAMPLE 3 Convolution of sequences

Let $\boldsymbol{x} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \end{bmatrix}$ $\quad \boldsymbol{y} = \begin{bmatrix} -1 \\ 1 \\ 3 \\ -2 \end{bmatrix}$ Find $\boldsymbol{x} * \boldsymbol{y}$

Solution

We obtain DFT by using definition as $\boldsymbol{X} = \begin{bmatrix} 2.5 \\ -0.5i \\ -0.5 \\ 0.5i \end{bmatrix}$ $\quad \boldsymbol{Y} = \begin{bmatrix} 0.5 \\ -2 + 1.5i \\ 1.5 \\ -2 - 1.5i \end{bmatrix}$

Then we have $\boldsymbol{XY} = \begin{bmatrix} 1.25 \\ 0.75 + i \\ -0.75 \\ 0.75 - i \end{bmatrix}$

Therefore $\boldsymbol{x} * \boldsymbol{y} = \mathcal{F}_N^{-1}[\boldsymbol{XY}] = \begin{bmatrix} 1 \\ 2 \\ -0.5 \\ 0 \end{bmatrix}$

*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.

How correct are these discrete values themselves?

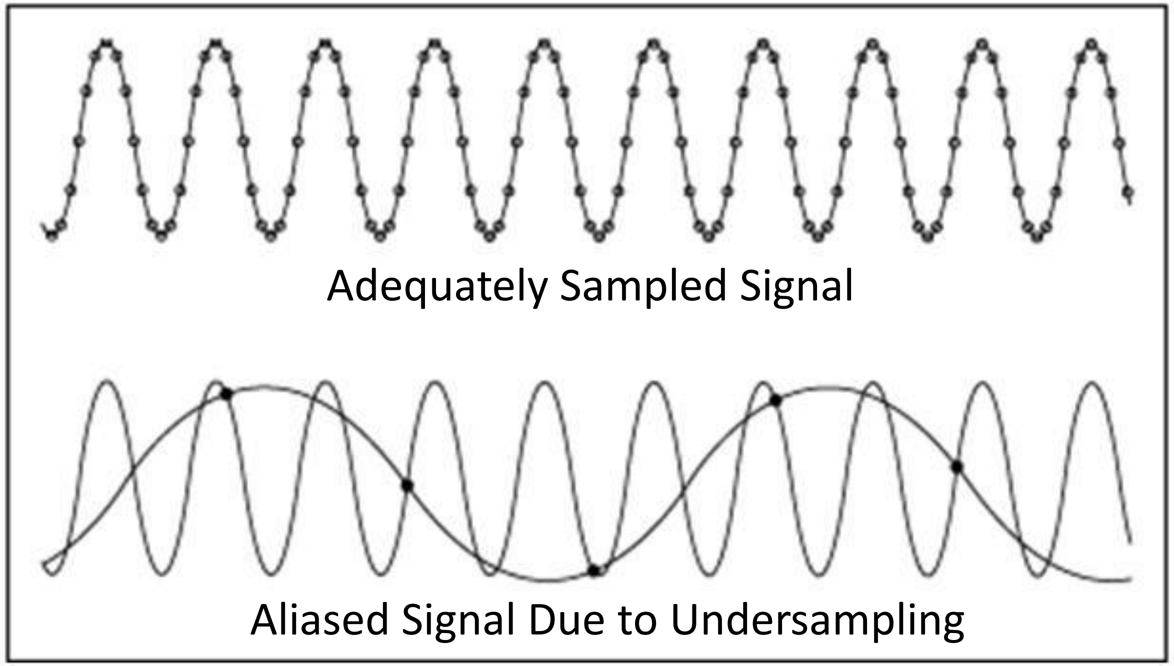
There are two main types of DFT errors: **aliasing** and **leakage**.

10.1 Discrete Fourier Transform

*Aliasing

If the initial samples are not sufficiently closely spaced (undersampling) 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 minimize its high frequency spectral content**.

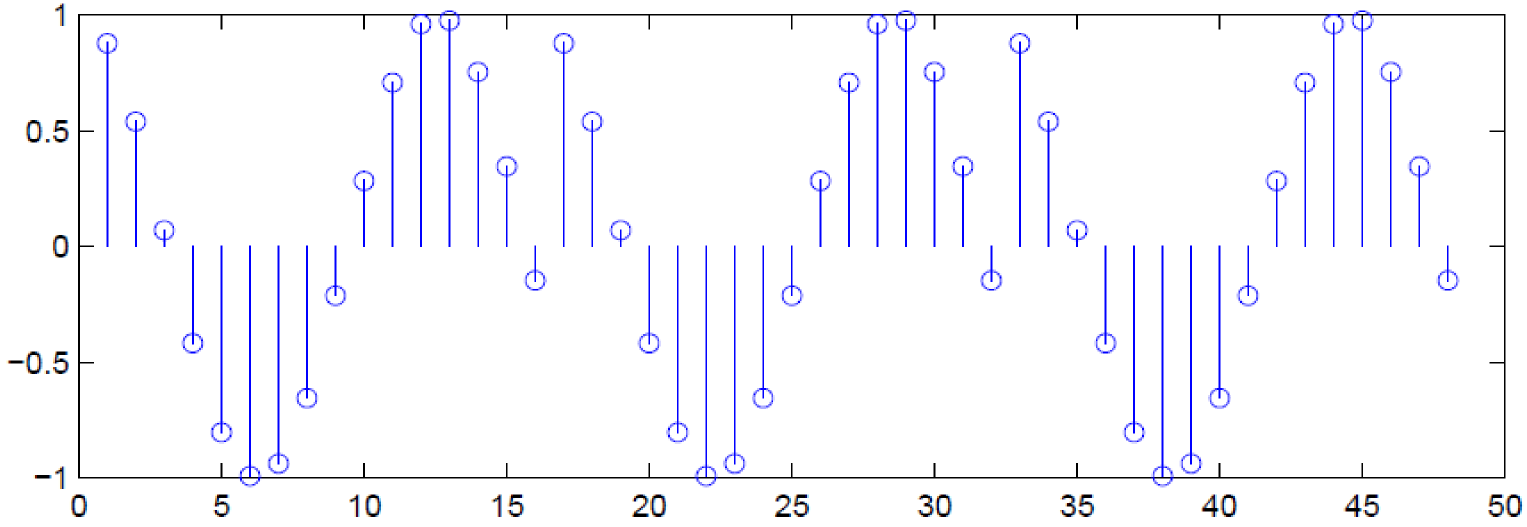


10.1 Discrete Fourier Transform

*Leakage

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**.

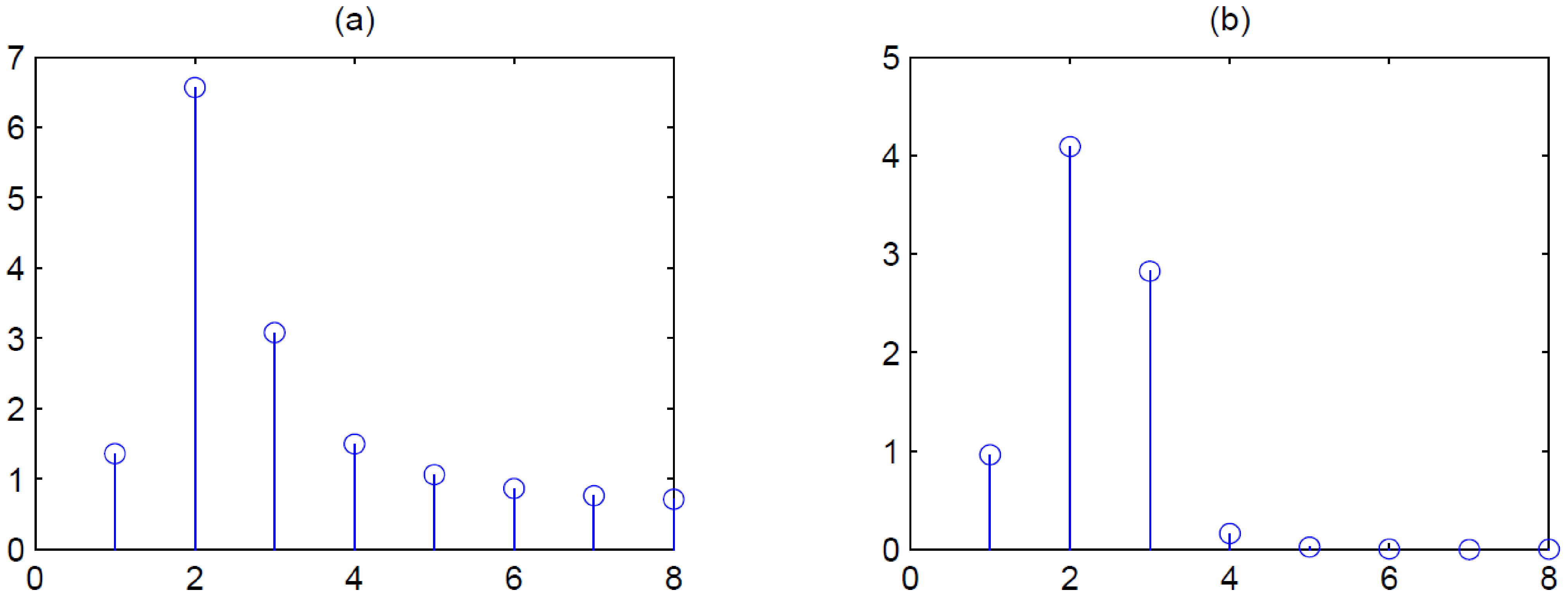
This smearing effect, which is known **as leakage**, arises because we are effectively calculating the Fourier series for the waveform in Figure, which has major discontinuities, hence other frequency components.



Leakage. The repeating waveform has discontinuities.

10.1 Discrete Fourier Transform

The solution is to use one of the *window functions* which we encountered in the design of finite impulse response (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.



Leakage is reduced using a Hanning window.

10.2 Fast Fourier Transform

10.2 Fast Fourier Transform (FFT)

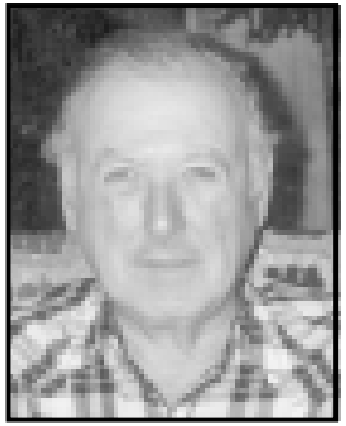
- The Cooley and Tukey Fast Fourier Transform (FFT) algorithm is a turning point to the computation of DFT
- Before that, DFT was never practical except running by some very expensive computers

Top 10 Algorithms in the 20th Century

1946: The Metropolis Algorithm
1947: Simplex Method
1950: Krylov Subspace Method
1951: The Decompositional Approach to Matrix Computations
1957: The Fortran Optimizing Compiler
1959: QR Algorithm
1962: Quicksort
1965: Fast Fourier Transform
1977: Integer Relation Detection
1987: Fast Multipole Method

10.2 Fast Fourier Transform (FFT)

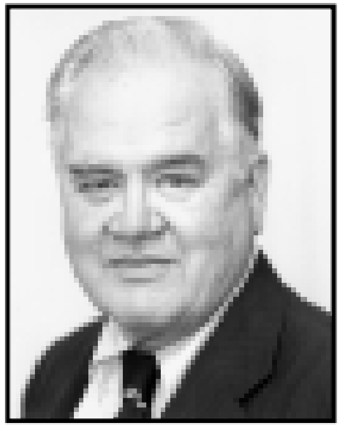
- The Cooley and Tukey Fast Fourier Transform (FFT) algorithm is a turning point to the computation of DFT
- Before that, DFT was never practical except running by some very expensive computers



James Cooley

1965: James Cooley of the IBM T.J. Watson Research Center and John Tukey of Princeton University and AT&T Bell Laboratories unveil the **fast Fourier transform**.

Easily the most far-reaching algo-rithm in applied mathematics, the FFT revolutionized signal processing. The underlying idea goes back to Gauss (who needed to calculate orbits of asteroids), but it was the Cooley–Tukey paper that made it clear how easily Fourier transforms can be computed. Like Quicksort, the FFT relies on a divide-and-conquer strategy to reduce an ostensibly $O(N^2)$ chore to an $O(N \log N)$ frolic. But unlike Quick- sort, the implementation is (at first sight) nonintuitive and less than straightforward. This in itself gave computer science an impetus to investigate the inherent complexity of computational problems and algorithms.



John Tukey

10.2 Fast Fourier Transform (FFT)

Basic principle of FFT

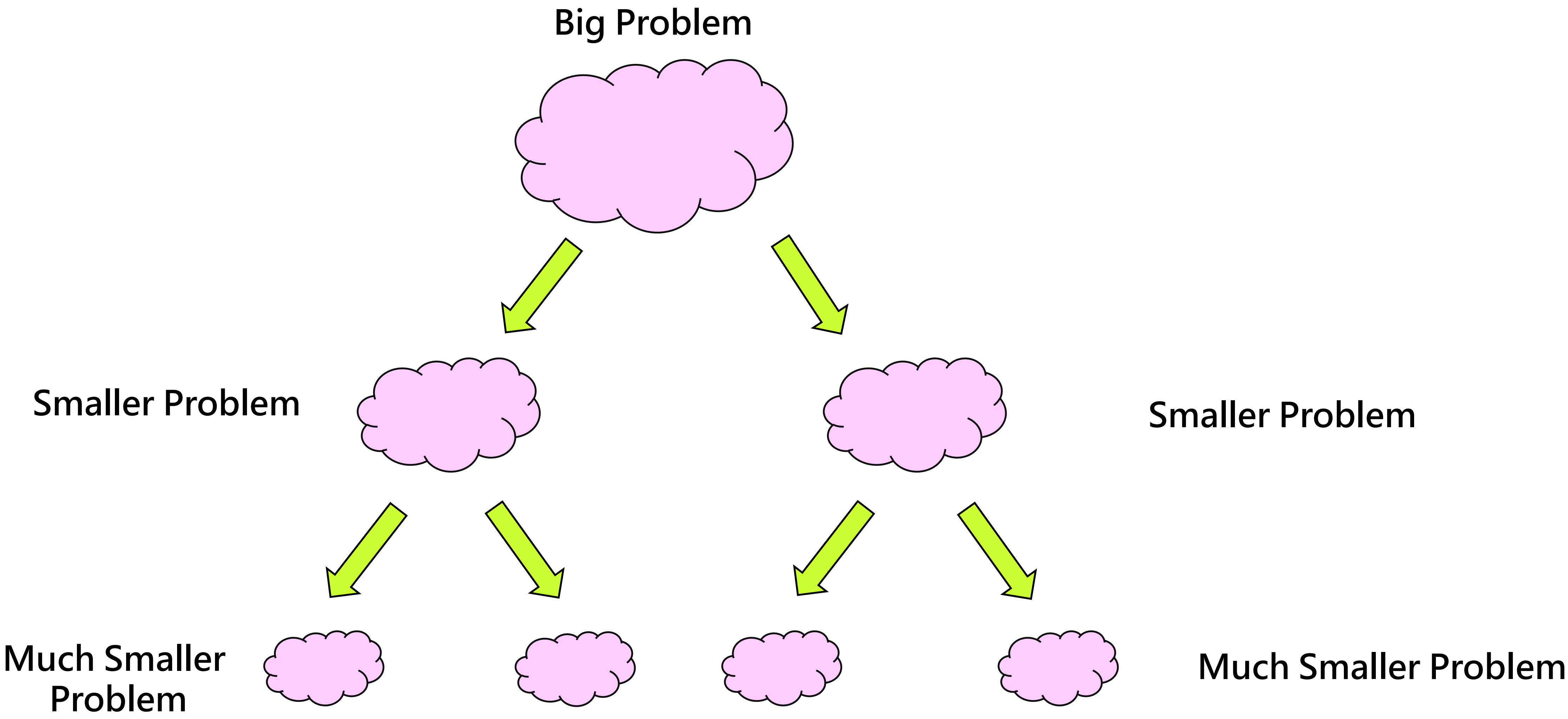
Divide and Conquer

→ To break down a big problem to a number of smaller problems and tackle them individually

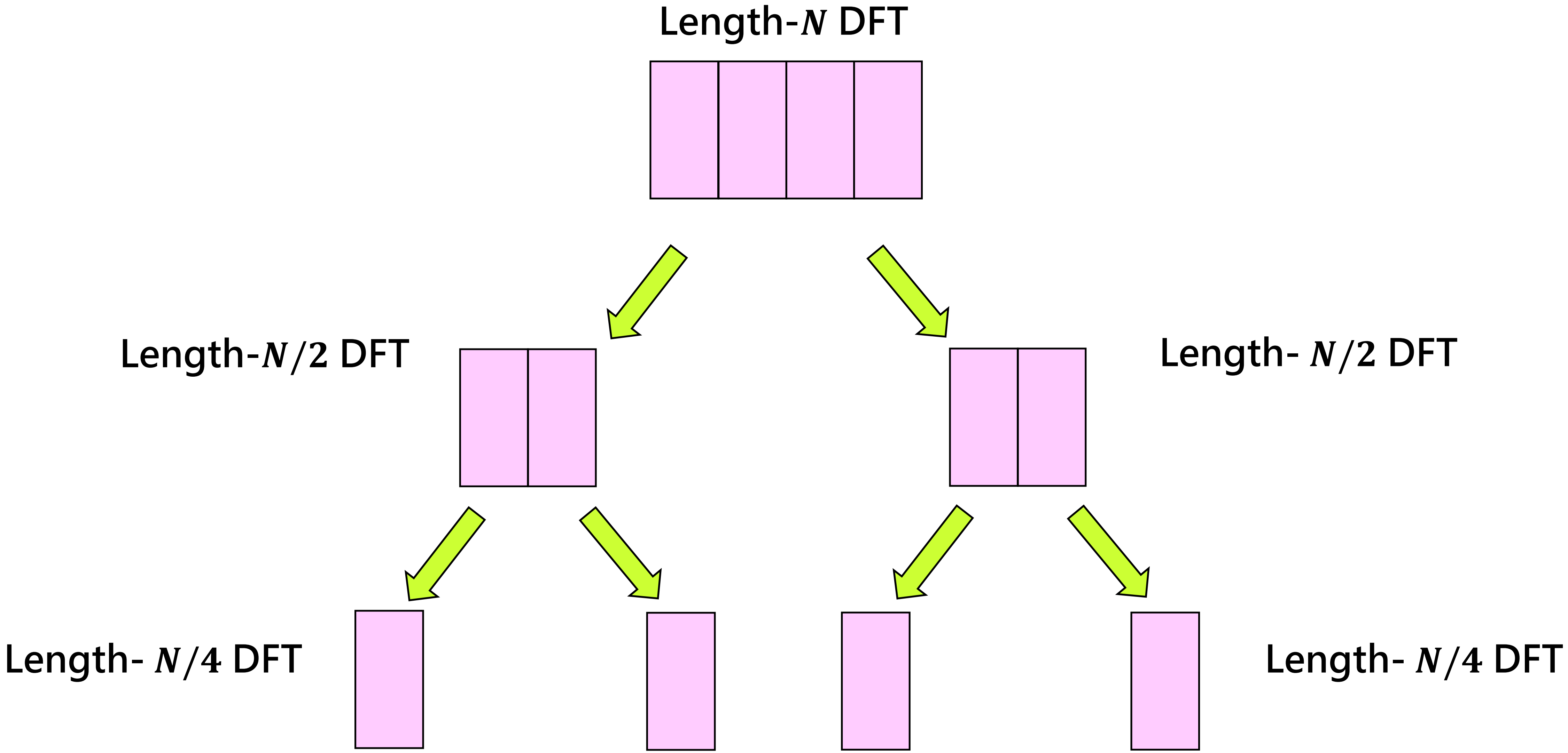
Need to satisfy the following criterion

$$\sum [cost\ (subproblem) + cost\ (overhead)] < cost\ (original\ problem)$$

10.2 Fast Fourier Transform (FFT)



10.2 Fast Fourier Transform (FFT)



10.2 Fast Fourier Transform (FFT)

Let $\boxed{W_N^{nk} = e^{-i\frac{2\pi}{N}nk}}$

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

It is easy to realize 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.

10.2 Fast Fourier Transform (FFT)

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

$$W_8^1 = e^{-i\frac{2\pi}{8}} = e^{-i\frac{\pi}{4}} = \frac{1-i}{\sqrt{2}}$$

Similarly

$$W_8^2 = -i \quad W_8^3 = -ia \quad W_8^4 = -1$$

$$W_8^5 = -a \quad W_8^6 = i \quad W_8^7 = ia \quad W_8^8 = 1$$

From the above, it can be seen that:

$$W_8^4 = -W_8^0$$

$$W_8^5 = -W_8^1$$

$$W_8^6 = -W_8^2$$

$$W_8^7 = -W_8^3$$

10.2 Fast Fourier Transform (FFT)

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

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

10.2 Fast Fourier Transform (FFT)

$$\mathcal{F}[k] = \sum_{n=0}^{N-1} f[n] W_N^{nk} \quad \text{For } k = 0, 1, \dots, N-1$$

Let us begin by **splitting the single summation over N samples into 2 summations**, each with $\frac{N}{2}$ samples, **one for even** and **the other for odd**.

$$\mathcal{F}[k] = \sum_{\text{even } n} f[n] W_N^{nk} + \sum_{\text{odd } n} f[n] W_N^{nk}$$

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

Note that

$$W_N^{2mk} = e^{-i\frac{2\pi}{N}(2mk)} = e^{-i\frac{2\pi}{\frac{N}{2}}mk} = W_{\frac{N}{2}}^{mk}$$

Therefore

$$\mathcal{F}[k] = \underbrace{\sum_{m=0}^{\frac{N}{2}-1} f[2m] W_{\frac{N}{2}}^{mk}}_{\frac{N}{2}\text{ - periodic}} + W_N^k \underbrace{\sum_{m=0}^{\frac{N}{2}-1} f[2m+1] W_{\frac{N}{2}}^{mk}}_{\frac{N}{2}\text{ - periodic}}$$

i.e.

Overhead

$$\mathcal{F}[k] = G[k] + W_N^k H[k]$$

Length-N/2 DFT of even data

Length-N/2 DFT of odd data

10.2 Fast Fourier Transform (FFT)

Thus the N-point DFT $\mathcal{F}[k]$ can be obtained from two $\frac{N}{2}$ - point transforms, one on even input data, $G[k]$, and one on odd input data, $H[k]$.

Although the frequency index k ranges over N values, only $\frac{N}{2}$ values of $G[k]$ and $H[k]$ need to be computed since $G[k]$ and $H[k]$ are periodic in k with period $\frac{N}{2}$.

10.2 Fast Fourier Transform (FFT)

For example, for $N = 8$

- Even input data $f[0], f[2], f[4], f[6]$
- Odd input data $f[1], f[3], f[5], f[7]$

$$\mathcal{F}[0] = G[0] + W_8^0 H[0]$$

$$\mathcal{F}[1] = G[1] + W_8^1 H[1]$$

$$\mathcal{F}[2] = G[2] + W_8^2 H[2]$$

$$\mathcal{F}[3] = G[3] + W_8^3 H[3]$$

$$\mathcal{F}[4] = G[0] + W_8^4 H[0] = G[0] - W_8^0 H[0]$$

$$\mathcal{F}[5] = G[1] + W_8^5 H[1] = G[1] - W_8^1 H[1]$$

$$\mathcal{F}[6] = G[2] + W_8^6 H[2] = G[2] - W_8^2 H[2]$$

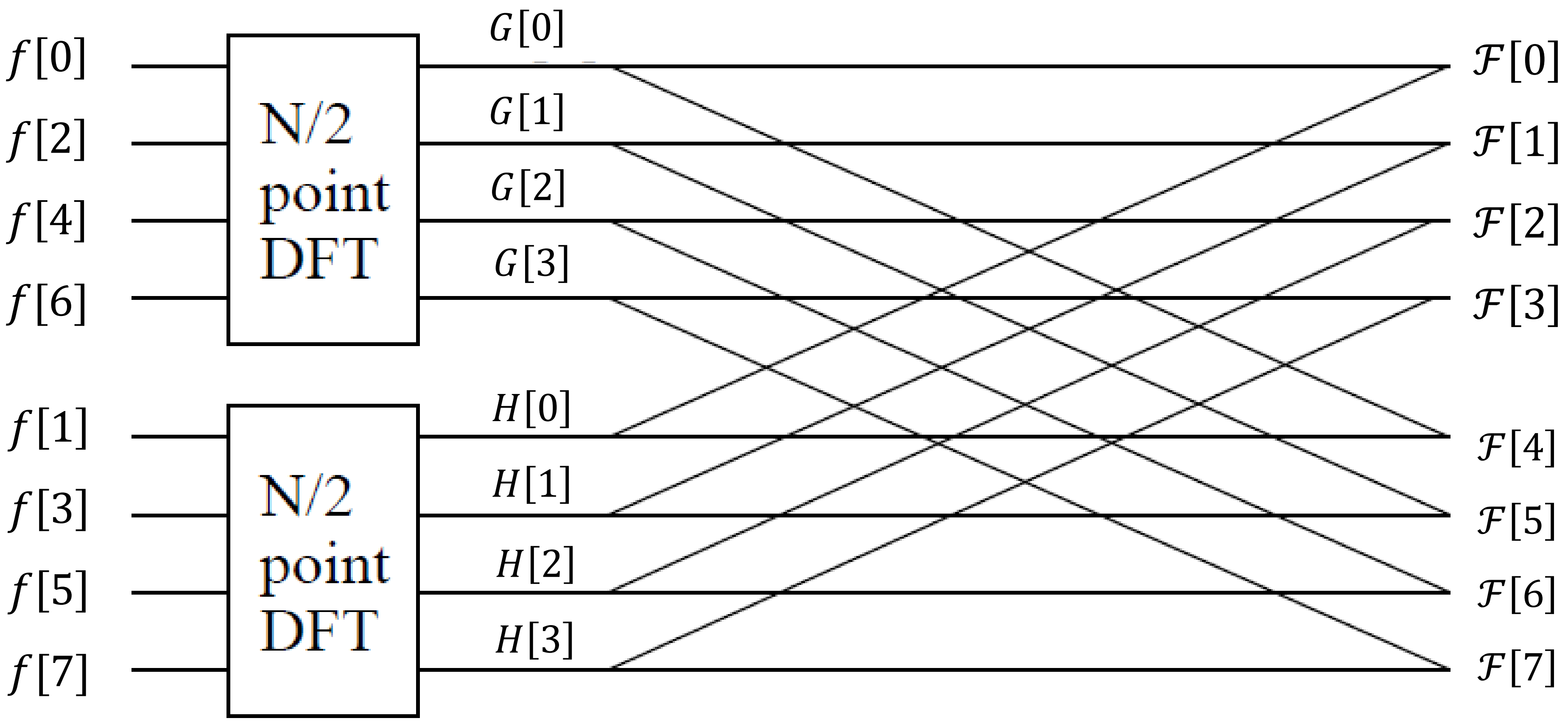
$$\mathcal{F}[7] = G[3] + W_8^7 H[3] = G[3] - W_8^3 H[3]$$

Q: How many multiplications in this stage?

$4 \quad \text{i.e.} \quad \frac{N}{2}$

10.2 Fast Fourier Transform (FFT)

This is shown graphically on the flow graph of Figure.



10.2 Fast Fourier Transform (FFT)

Moreover,

Assuming that N is a power of 2 (i.e. there are γ stages, where $N = 2^{\gamma}$), 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.

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.

10.2 Fast Fourier Transform (FFT)

The basic computation at the heart of the FFT is known as the butterfly because of its criss-cross appearance.

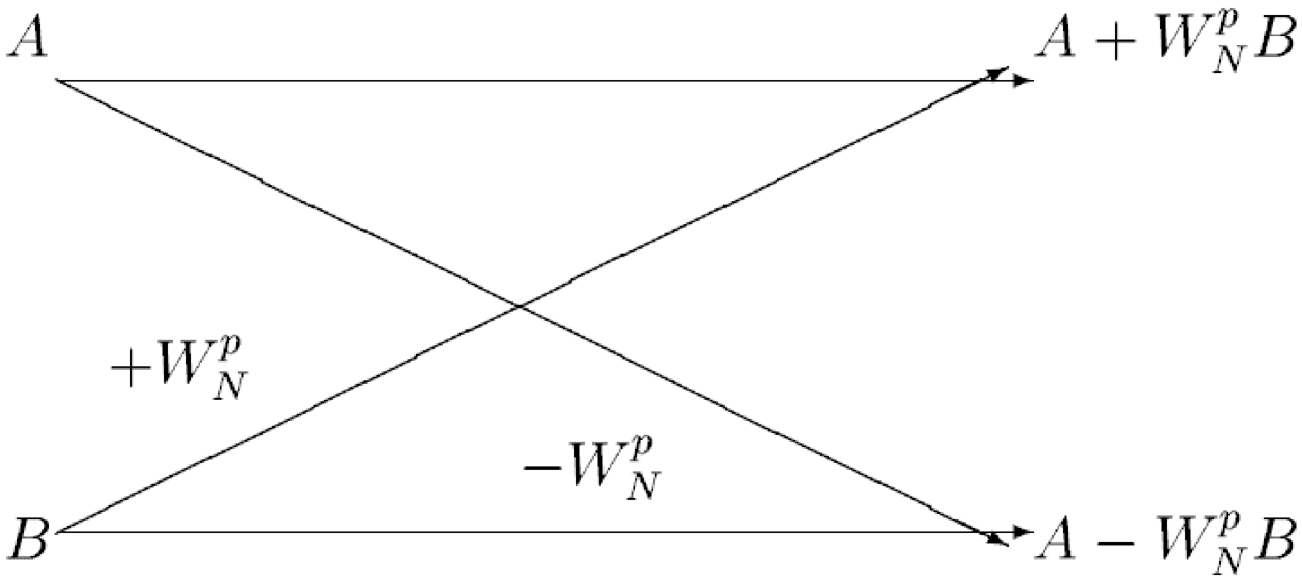


Figure Butterfly operation in FFT.

where A and B are complex numbers.

10.2 Fast Fourier Transform (FFT)

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 $\frac{N}{2}\log_2 N$ (approximately because multiplications by factors such as W_N^0, $W_N^{\frac{N}{2}}$, $W_N^{\frac{N}{4}}$, $W_N^{\frac{3N}{4}}$ and are really just complex additions and subtractions).

N	N^2 (DFT)	$\frac{N}{2}\log_2 N$ (FFT)	Cost Saving
32	1,024	80	92%
256	65,536	1024	98%
1,024	1,048,576	5120	99.5%

10.2 Fast Fourier Transform (FFT)

Q: What if N is not a power of 2?

Usually, implement padding for the data with zeroes.

e.g. Assume we have data with 5 (i.e. $5 = 2^2 + 1$) points, then we perform the padding that include 3 zeroes with the 8 points (i.e. $N = 5 + 3 = 8 = 2^3$) and compute a 8-point FFT.

	$f[n] = \{0,1,0,1,1\}$	$N = 5 = 2^2 + 1$
After zero padding	$f[n] = \{0,1,0,1,1,0,0,0\}$	$N = 8 = 2^3$

Remarks

- FFT requires N to be a power of 2.

- If N is not a power of 2, need zero padding to let N be a power of 2 before FFT.

Review for Lecture 10

- Discrete Fourier Transform (DFT)
- Fast Fourier Transform (FFT)

Exercise

Please Check https://github.com/uoaworks/FourierAnalysisAY2018

Reading materials:

1. Discrete Fourier transform https://en.wikipedia.org/wiki/Discrete_Fourier_transform
2. Discrete Fourier Transform https://brilliant.org/wiki/discrete-fourier-transform
3. Stephen Roberts, http://www.robots.ox.ac.uk/~sjrob/Teaching/SP/l7.pdf
4. Daniel P. K. Lun, http://www.eie.polyu.edu.hk/~enpklun/EIE327/FFT.pdf
5. Section10.1, 10.3, 10.4, Textbook
6. Julius O. Smith III, Mathematics of the Discrete Fourier Transform, 2002

References

[1] Nakhlé H. Asmar, *Partial Differential Equations with Fourier Series and Boundary Value Problems 2nd Edition*, 2004

[2] Stephen Roberts, http://www.robots.ox.ac.uk/~sjrob/Teaching/SP/l7.pdf

[3] Daniel P. K. Lun, http://www.eie.polyu.edu.hk/~enpklun/EIE327/FFT.pdf

[4] Alex Townsend, http://pi.math.cornell.edu/~ajt/presentations/TopTenAlgorithms.pdf

[5] Barry A. Cipra, http://www.uta.edu/faculty/rcli/TopTen/topten.pdf