Lecture – 21

Monday, October 26, 2020 2:01 PM

Last class: Discrete Fourier Transform (DFT)

- Concept of zero padding a signal → more samples of DTFT
- DFT expressions: periodic in time & periodic in frequency
  i.e. $x[n+N] = x[n]$ i.e. $X[k+N] = X[k]$
- Properties of DFT: circular shift
- Circular / periodic convolution: $X_1[k]\, X_2[k] = x_1[n] \circledast x_2[n] = \sum_{m=0}^{N-1} x_1[m]\, x_2[n-m]$
- Linear convolution using circular convolution (write convolutions in matrix form)

Today's class: linear convolution: $x[n]$, $n = 0, 1, \cdots L-1$ length L

$h[n]$, $n = 0, 1, \cdots M-1$ length M

$y[n] = x[n] * h[n]$, $n = 0, 1, \cdots L+M-2$ length $L+M-1$

To perform linear convolution using circular convolution:

(a) zero padding: add $(M-1)$ zeros to $x[n] \rightarrow \tilde{x}[n]$
add $(L-1)$ zeros to $h[n] \rightarrow \tilde{h}[n]$
both of length $(L+M-1)$

(b) circular convolution: $\tilde{x}[n] \circledast \tilde{h}[n] = x[n] * h[n]$ length $(L+M-1)$.

perform circular convolution using DFT.

$\tilde{x}[n] \longleftrightarrow \tilde{X}[k]$

$\tilde{h}[n] \longleftrightarrow \tilde{H}[k]$

multiply → $\tilde{X}[k] \cdot \tilde{H}[k]$

↓ Take Inverse DFT

$\tilde{x}[n] \circledast \tilde{h}[k] = x[n] * h[n] = y[n]$

This process can be made efficient.

Fast Fourier Transform (FFT): collection of algorithms to compute DFT efficiently.

DFT: $$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}kn}, \quad k = 0, 1, \cdots N-1 \quad \text{N-point DFT.}$$

a) direct DFT computation :

\# complex multiplication : $N(N) = N^2$

\# complex additions : $N(N-1) = N^2 - N$

both are $O(N^2)$

1 complex multiplication ≡ 4 real multiplications & 2 real additions

1 — " — addition ≡ 2 real additions

$$W_N^p = e^{-j\frac{2\pi}{N}p}$$ & F_N is made up of W_N powers

→ twiddle factors or phase factors

b) efficient DFT computation

some properties of W_N

① $W_N^{(k+N)} = W_N^k$ ② $W_N^{kN} = 1$ k is any integer

N is even

③ $W_N^{2p} = e^{-j\frac{2\pi}{N}(2p)} = e^{-j\frac{2\pi}{(N/2)}p} = W_{N/2}^p$ ...(N even)

④ $W_N^{k+N/2} = e^{-j\frac{2\pi}{N}(k+\frac{N}{2})} = e^{-j\frac{2\pi}{N}k} e^{-j\pi} = -W_N^k$ ...(N even)

* Radix - 2 FFT * assume $N = 2^m$

$$X[k] = \sum_{0}^{N-1} x[n] W_N^{kn} \quad (N\text{-DFT})$$

Decimation in time.

$$= \sum_{n-\text{even}} x[n] W_N^{kn} + \sum_{n-\text{odd}} x[n] W_N^{kn}$$

$\downarrow n=2r$ $\qquad \downarrow n=2r+1$

$$= \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_N^{2kr} + \sum_{r=0}^{N/2-1} x[2r+1] W_N^{k(2r+1)}$$

→ $W_N^{2kr} W_N^{k}$

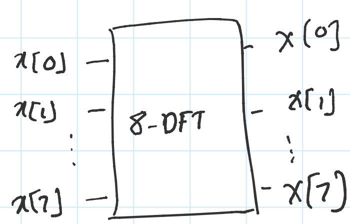
$$= \sum_{r=0} x[2r]\, W_N + \sum_{r=0} x[2r+1]\, W_N \quad \hookrightarrow W_N^{2kr} W_N^{k}$$

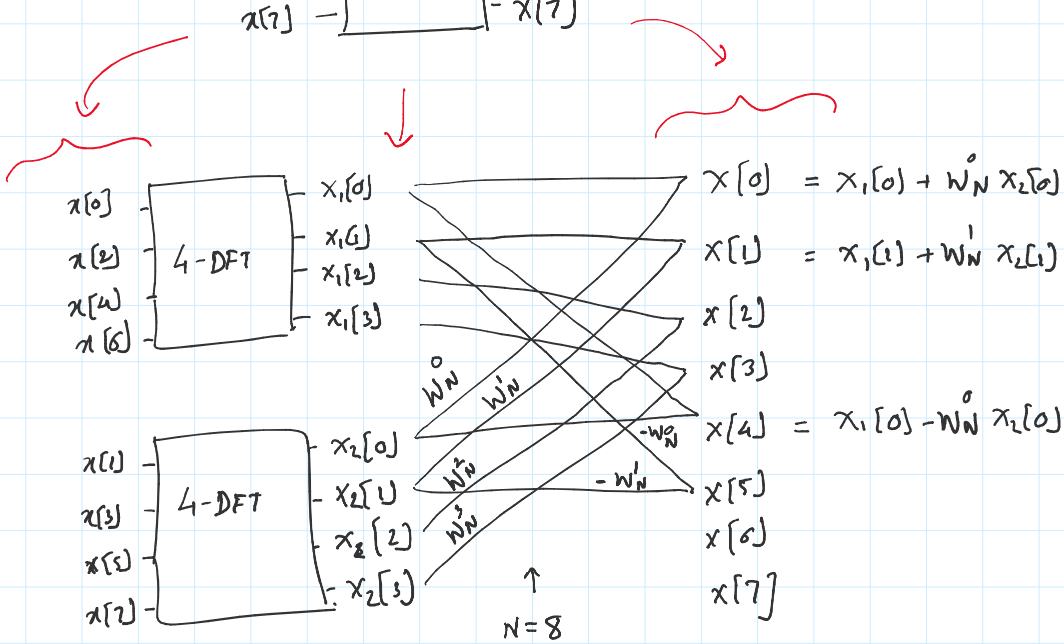
$$X[k] = \underbrace{\sum_{r=0}^{\frac{N}{2}-1} x[2r]\, W_{N/2}^{kr}}_{\frac{N}{2}\text{-point DFT}} + W_N^{k} \underbrace{\sum_{r=0}^{N/2-1} x[2r+1]\, W_{N/2}^{kr}}_{\frac{N}{2}\text{-point DFT}}$$

$$X[k] = X_1[k] + W_N^{k} X_2[k], \quad k = 0, 1, \cdots N-1.$$

$$\left.\begin{aligned} * \; X[k] &= X_1[k] + W_N^{k} X_2[k], & k &= 0, 1, \cdots \tfrac{N}{2}-1 \\ * \; X[k + \tfrac{N}{2}] &= X_1[k] + W_N^{k+N/2} X_2[k], & k &= 0, 1, \cdots \tfrac{N}{2}-1 \\ &= X_1[k] - W_N^{k} X_2[k], & k &= 0, \cdots \tfrac{N}{2}-1 \end{aligned}\right\} \text{A}$$

Ex. $N = 8$.



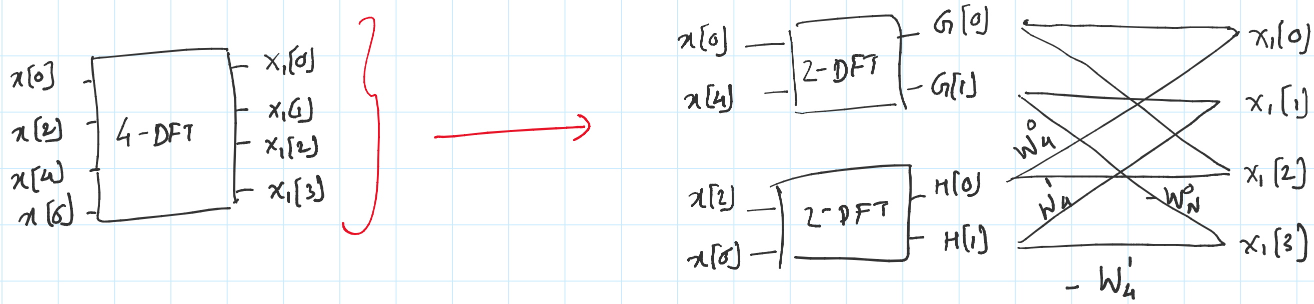


For this one step: computations from A

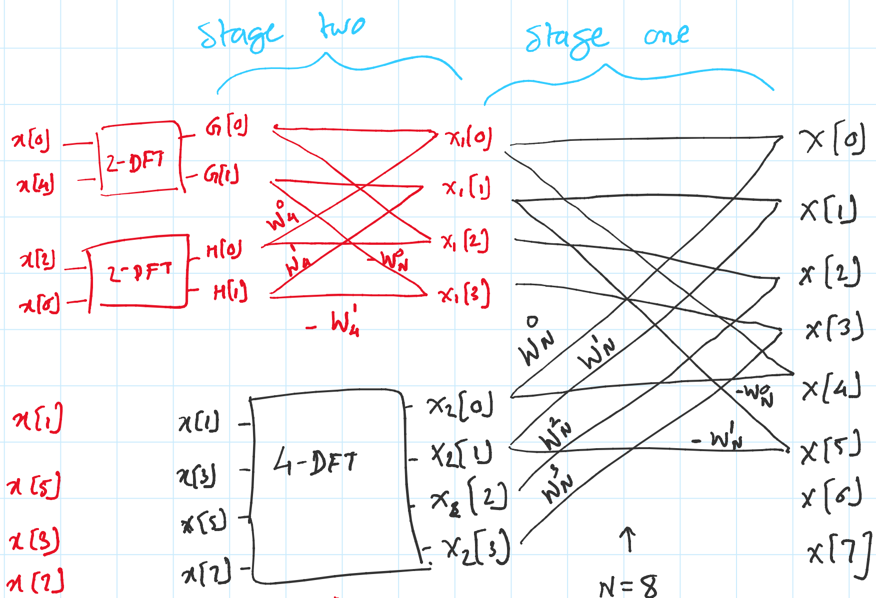
$$\#\text{ complex multiplication} = 2\left(\frac{N}{2}\right)^2 + \frac{N}{2} = \frac{N^2+N}{2}$$

$$\#\text{ complx. additions} = 2\,\frac{N}{2}\left(\frac{N}{2}-1\right) + N = \frac{N^2}{2}$$

In above ex. 4-point DFT will also be implemented using 2-point DFT.



2-DFT is very simple $\rightarrow$ $F_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ $\Rightarrow$ $G[0] = x[0] + x[4]$, $G[1] = x[0] - x[4]$ & $H[0] = x[2] + x[6]$, $H[1] = x[2] - x[6]$



$X[0] = X_1[0] + W_N^0 X_2[0]$

$X[1] = X_1[1] + W_N^1 X_2[1]$

$X[4] = X_1[0] - W_N^0 X_2[0]$

8-DFT using 2-DFT blocks.

Decimation in time.

similarly for this block - we will replace with 2-DFT blocks.

For $N = 2^m$ - we will have $\log_2 N = m$ stages.

each stage has $\left(\frac{N}{2}\right)$ cmplx. multiplication

(N) complx. additions.

Total complexity : $(\frac{N}{2})\log_2 N$ multiplication & $N \log_2 N$ additions.

i.e $O(N \cdot \log_2 N)$ complexity.

Ex. $N = 2^{10} = 1024$

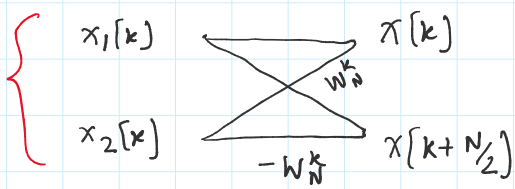
(a) direct DFT : # mult. $= N^2 = 2^{20} \approx 10^6 = 100,000$

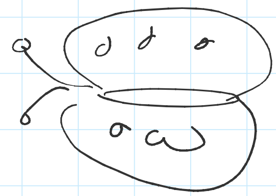
(b) radix-2 FFT : # mult. $= \frac{N}{2}\log_2 N = \frac{2^{10}}{2}(10) \approx \frac{10^4}{2} = 5120$

Note : We can also have decimation in frequency.

Butterfly diagram :

structure





Note on Indexing: we can get input signal order using bit-reversed indexing.

$x[0]$	000	000	$x[0]$
$x[1]$	001	100	$x[4]$
$x[2]$	010	010	$x[2]$
$x[3]$	011	110	$x[6]$
$x[4]$	100	001	$x[1]$
$x[5]$	101	101	$x[5]$
$x[6]$	110	011	$x[3]$
$x[7]$	111	111	$x[7]$
$\underbrace{\quad}_{x[n]}$	$\underbrace{\quad}_{n}$	$\underbrace{\quad}_{n'}$	$\underbrace{\quad}_{x[n']}$

reverse bits

$n \longrightarrow n'$ bit reversal

$x[n']$ ↳ input for radix-2 FFT (decimation in time).