Real-Time Digital Signal Processing
Lecture 9 - Fast Fourier Transform

Electrical Engineering and Computer Science
University of Tennessee, Knoxville

March 10, 2015

Overview

Recap

- Week 1: Background
- Week 2: DSK and Lab
- Week 3: I/O - Sampling and Reconstruction
- Week 4: The z-transform and Design Structure
- Week 5-6: The FIR filter with linear phase
- Week 7-8: FIR filter design techniques
- Week 9: Fast Fourier Transform

Review - FIR filter with linear phase

- Linear-phase FIR filter
  - Generalized linear phase, $\sphericalangle H(e^{j\omega}) = \beta - \omega\alpha, 0 < \omega < \pi$
  - Group delay, α
  - Phase shift, β
- Four types of causal FIR filters with generalized linear phase, $h[n] = \pm h[M-n], n = 0, \cdots, M$
  - Definition and derivation (the type of symmetry, with or w/o phase shift (β), integer or non-integer group delay (α), even or odd number of filter coefficients)
  - Design structure (halved # of multiplication)
  - Zero patterns (the set of four reciprocal zeros)
  - $z = \pm 1$ being zero?
- When to use what?
  - Constraints on the zeros
  - Integer/non-integer group delay
  - Phase shift

Review - FIR filter design techniques

- Windows - Kaiser
  - Why using window?
  - The two essential parameters: width of the main lobe and height of the side lobe
  - Pros and cons of windowing techniques
- Optimal methods - Park-McClellan equiripple
  - How does PM overcome issues with Kaiser window?

Review - IIR

- CT -> DT
  - Impuse invariance
    - $\omega = \Omega T$
    - preserve good time-domain characteristics
    - linear scale of the frequency axis + aliasing
  - Bilinear transformation
    - $s = \frac{2}{T}(\frac{1-z^{-1}}{1+z^{-1}})$ or $\Omega = \frac{2}{T}\tan\frac{\omega}{2}$
    - distortion in frequency characteristics
    - no aliasing

Review - DFT

- DTFT
  - $x[n]$ needs to be either absolutely summable or square summable
  - $X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$, $x[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega$
- DFS: sampled version of $X(e^{j\omega})$ at frequencies $w_k = 2\pi k/N$
  - $\tilde{X}[k] = \sum_{n=0}^{N-1} \tilde{x}[n]W_N^{kn}$, $\tilde{x}[n] = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{X}[k]W_N^{-kn}$ where $W_N = e^{-j(2\pi/N)}$.
- DFT: one period of $\tilde{X}[k]$
  - $X[k] = \sum_{n=0}^{N-1} x[n]W_N^{kn}, 0 \leq k \leq N-1$, $x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]W_N^{-kn}, 0 \leq n \leq N-1$
  - $\tilde{x}[n] = x[((n))_N]$, $\tilde{X}[k] = X[((k))_N]$

Discrete Fourier Transform (DFT) - Direct computation

- $X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn}, 0 \leq k \leq N-1,$
  $x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}, 0 \leq n \leq N-1$
- $x[n] W_N^{kn}$ results in 1 complex multiplication
- $\sum_{n=0}^{N-1} x[n] W_N^{kn}$ results in N complex multiplications and $N-1$ complex additions for fixed k
- With $k = 0, \cdots, N-1$, there are N^2 complex multiplications and $N(N-1)$ complex additions
- Therefore, the complexity of direct computation is $O(N^2)$

Rationale of FFT

- By decomposing the original sequence into subsequences, we can reduce the N-point DFT to M-point DFT where $M \ll N$, such that the computational complexity is $O(N \log N)$ instead of $O(N^2)$
- Different ways to decompose a sequence
  - Decimation-in-time FFT
  - Decimation-in-frequency FFT

Decimation-in-time FFT

- Decompose the sequence into a set of even number points and odd number points

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} \tag{1}$$

$$= \sum_{n \text{ even}} x[n] W_N^{nk} + \sum_{n \text{ odd}} x[n] W_N^{nk} \tag{2}$$

$$= \sum_{r=0}^{N/2-1} x[2r] W_N^{2rk} + \sum_{r=0}^{N/2-1} x[2r+1] W_N^{(2r+1)k} \tag{3}$$

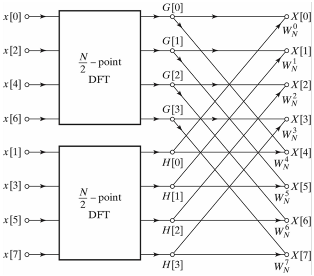
$$= \sum_{r=0}^{N/2-1} x[2r] W_{\frac{N}{2}}^{rk} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_{\frac{N}{2}}^{(rk} \tag{4}$$

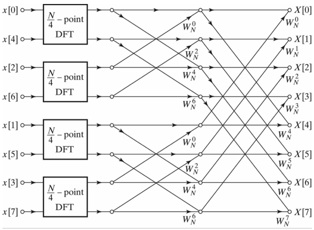
$$= G[k] + W_N^k H[k] \tag{5}$$

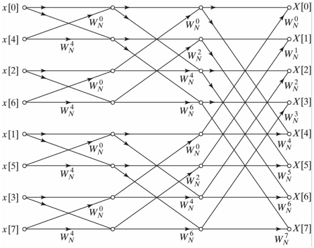
DIT FFT (cont'd)

- The above operations change from an N-point DFT to $\frac{N}{2}$-point DFT. The computational complexity of $G[k]$ and $H[k]$ is $O((\frac{N}{2})^2$, so the complexity of the entire process is $2(\frac{N}{2})^2 + N = N + \frac{N^2}{2} < N^2$

Illustration - Butterfly computation







Computational complexity

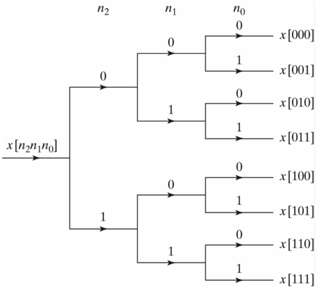
$$
\begin{array}{l|l|l}
N\text{-point DFT} & N^2 & \\
2\frac{N}{2}\text{-point DFT} & 2(\frac{N}{2})^2 + N & (\frac{N}{2})^2 \rightarrow 2(\frac{N}{4})^2 + \frac{N}{2} \\
4\frac{N}{4}\text{-point DFT} & 4(\frac{N}{4})^2 + N + N & (\frac{N}{4})^2 \rightarrow 2(\frac{N}{8})^2 + \frac{N}{4} \\
8\frac{N}{8}\text{-point DFT} & 8(\frac{N}{8})^2 + N + N + N & (\frac{N}{4})^2 \rightarrow 2(\frac{N}{8})^2 + \frac{N}{4}
\end{array}
$$

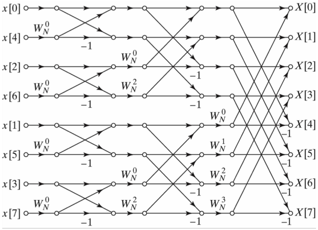
Eventually, when it comes down to just 2-point DFT, there would be $\log_2^N$ number of N's summing up together. Therefore, the complexity of FFT is $O(N \log_2^N)$

In-place computation

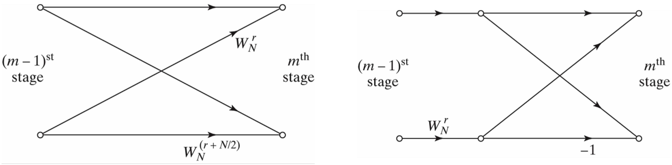
- Each level of output overwrites the original memory. I.e., store results back to the original memory location

Bit-reversed order - input sequence got rearranged



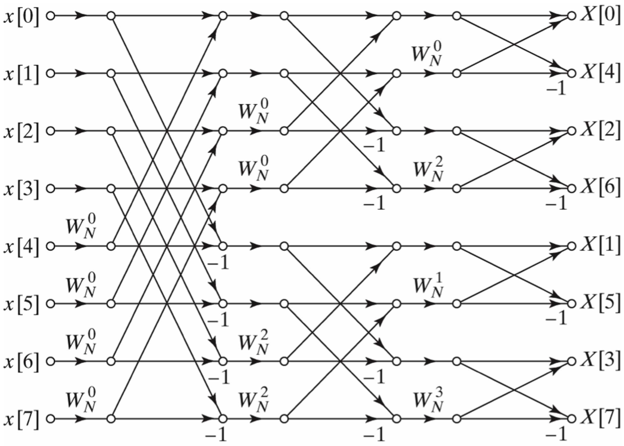


Modifications 1 - from inside BF to outside BF



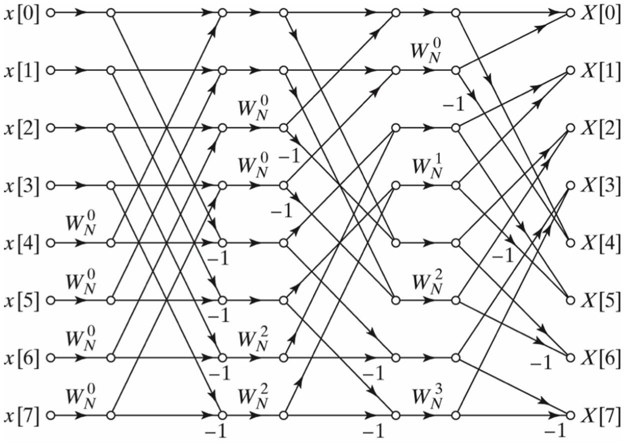
$$W_N^{(r+\frac{N}{2})} = W_N^r . W_N^{\frac{N}{2}}$$

Modifications 2 - rearrange input order (output bit-reversed order)



Modifications 3 - Input in order and output in order

- Distorted butterfly
- Not in-place computation



Decimation-in-frequency

- Decompose the sequence into the first half and the second half

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk} \tag{6}$$

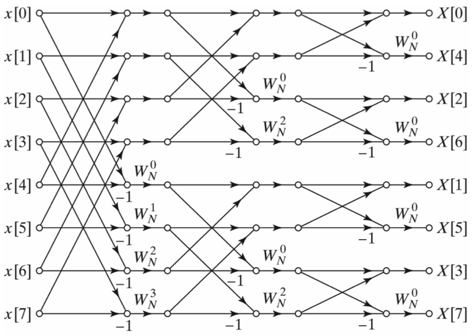
$$= \sum_{n=0}^{\frac{N}{2}-1} x[n] W_N^{nk} + \sum_{n=\frac{N}{2}}^{N-1} x[n] W_N^{nk} \tag{7}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} [x[n] + (-1)^k x[n + \frac{N}{2}]] W_N^{nk} \tag{8}$$

DIF - cont'd

- When k is even, $x[2r] = \sum_{n=0}^{\frac{N}{2}-1} [x[n] + x[n + \frac{N}{2}]] W_N^{2rn}$, i.e., $x[2r] = \sum_{n=0}^{\frac{N}{2}-1} g[n] W_{\frac{N}{2}}^{rn}$, becoming an $\frac{N}{2}$-point DFT
- When k is odd, $X[2r] = \sum_{n=0}^{\frac{N}{2}-1} [x[n] + x[n + \frac{N}{2}]] W_N^{2rn}$, i.e., $X[2r+1] = \sum_{n=0}^{\frac{N}{2}-1} [h[n] W_N^n] W_{\frac{N}{2}}^{rn}$, where $h[n] = x[n] - x[n + \frac{N}{2}]$, again becoming an $\frac{N}{2}$-point DFT

DIF - flow graph



- Compared to DIT, the multiplication occurs after the butterfly while DIT occurs before the butterfly