CMPE323 Signals and Systems

Lecture 21: Decimation in Time FFT

The DFT and the FT

- **Now consider the DFT as an approximation of the FT**

Let $x(t)$ be an energy signal, $\int_{-\infty}^{\infty} |x(t)|^2 \, dt < \infty$ (basically not periodic)

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt \approx \sum_{n=-\infty}^{\infty} x(n\Delta t) e^{-j2\pi fn\Delta t} \Delta t, \ \Delta t = \frac{1}{f_S}$$

$$\approx \frac{1}{f_S} \sum_{n=-\infty}^{\infty} x(n\Delta t) e^{-j2\pi n(f/f_S)} = \frac{1}{f_S} \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi nF} = \frac{1}{f_S} X_{DTFT}(F)$$

We make the *time* resolution $\Delta t = t_s$ better by increasing f_S.

We make the *frequency* resolution Δf better by increasing T.

$$\text{Truncate } x(t) \text{ at } T,\ x_T(t) = \begin{cases} x(t) & 0 \le t < T \\ 0 & \text{otherwise} \end{cases}$$

$$X_T(f) \approx \frac{1}{f_S}\sum_{n=0}^{N-1} x[n]e^{-j2\pi nF},\ N = \frac{T}{t_s} = Tf_S,\ f_S = \frac{N}{T}$$

$$X_T(f) \approx \frac{T}{N}\sum_{n=0}^{N-1} x[n]e^{-j2\pi nF},$$

$$\text{so } X_T(k\Delta f) \approx \frac{T}{N}\sum_{n=0}^{N-1} x[n]e^{-j2\pi nk/N} = \frac{T}{N}X_{DFT}[k]$$

If we make $x_T(t)$ periodic with period T, the coefficients of the DFS

$$c_k = \frac{1}{T}\int_0^T x_T(t)e^{-j2\pi kt/T}dt \approx \frac{\Delta t}{T}\sum_{n=0}^{N-1} x_T(n\Delta t)e^{-j2\pi k\Delta tn/T}$$

$$= \frac{1}{f_S T}\sum_{n=0}^{N-1} x_T(n\Delta t)e^{-j2\pi kn/N} = \frac{1}{N}\sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N} = \frac{1}{N}X_{DFT}[k]$$

Back to the DFT

- **We have the DFT equation**

$$X_{DFT}[k] = X\left(\frac{k}{T}\right) = \sum_{n=0}^{N-1} x(n\Delta t)e^{-j2\pi kn\Delta t/(N\Delta t)} = \sum_{n=0}^{N-1} x[n]e^{-j2\pi nk/N} = \sum_{n=0}^{N-1} x[n]W_N^{nk}$$

- **...and the DFT is sampled in both time and frequency!**
- **To compute this requires**

$$\left(N \text{ complex multiplications } \left(x[n]W_N^{nk}\right) + N \text{ complex additions}\right)$$

$$\times N \text{ values of } k \ \approx N^2 \text{ complex "multiply and accumulate" operations}$$

- **The FFT is an algorithm to drastically reduce this computation load to** $N\log_2 N$

$$N = 1024,\ N^2 = 1{,}048{,}576,\ N\log_2 N = 1024 \times 10 = 10{,}240$$

$$N^2/(N\log_2 N) = 102.4$$

The FFT algorithm

- First published and documented by Cooley & Tukey in 1965 (others had similar ideas)
- The algorithm has multiple forms…
- …but all the forms make use of the same ideas
- …including the symmetry and periodicity of W_N
- …and DFTs of simple sequences

$$W_{N/2} = e^{-j2\pi/(N/2)} = e^{-j2(2\pi/N)} = W_N^2$$

$$W_N^{k+(N/2)} = e^{-j2\pi(k+N/2)/N} = e^{-j2\pi k/N} e^{-j2\pi N/(2N)} = e^{-j2\pi k/N} e^{-j\pi}$$

$$= -e^{-j2\pi k/N} = -W_N^k$$

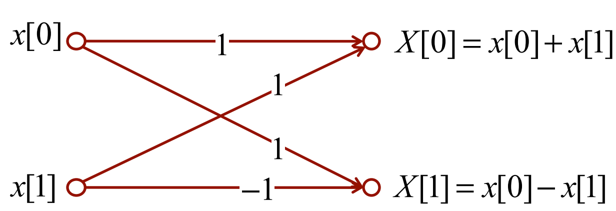
$$\text{If } N = 1,\; X(0) = x[0]W_1^0 = x[0]$$

$$\text{If } N = 2,\; X[0] = x[0]W_2^{0\times 0} + x[1]W_2^{1\times 0} = x[0] + x[1]$$

$$X[1] = x[0]W_2^{0\times 1} + x[1]W_2^{1\times 1} = x[0] + x[1]W_N^{0+(N/2)} = x[0] - x[1]$$

The simple FFT "butterfly"

- The SFG of an FFT operation has a simple shape known as a butterfly…
- …with the simplest butterfly being an $N = 2$ DFT/FFT



The Radix-2 Decimation in Time FFT

- **"Radix-2" means we group the time-domain points in groups of two…and that we restrict** $N = 2^m$
- **"Decimation in time" means that we "reorder" the time sequence in preparation for performing the FFT algorithm**
- **We'll get to that in a few minutes**
- **First, we start with our defining relationship**

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

- **Second, break the sum into sums on the even and odd indexed terms (not the even and odd parts!)**

$$X_{DFT}[k] = \underbrace{\sum_{n=0}^{(N/2)-1} x[2n] W_N^{2nk}}_{\substack{\text{even indexed terms} \\ 0,2,4,\ldots,N-2}} + \underbrace{\sum_{n=0}^{(N/2)-1} x[2n+1] W_N^{(2n+1)k}}_{\substack{\text{odd indexed terms} \\ 1,3,5\ldots N-1}}$$

Radix-2 DIT FFT, cont

- **Third, apply our relationship on the kernel**

$$X_{DFT}[k] = \sum_{n=0}^{(N/2)-1} x[2n] \underbrace{W_{N/2}^{nk}}_{W_{N/2}=W_N^2} + W_N^k \sum_{n=0}^{(N/2)-1} x[2n+1] \underbrace{W_{N/2}^{nk}}_{\substack{W_N^{(2n+1)k}=W_N^{2nk}W_N^k \\ W_{N/2}=W_N^2}}$$

- **Let** $G[k] = DFT(x[2n]), H[k] = DFT(x[2n+1])$**. These DFTs are** $N/2$ **point DFTs, but there are two of them**

$$X_{DFT}[k] = G[k] + W_N^k H[k],\ k = 0,1,\ldots \frac{N}{2} - 1$$

- **Now,** $G[k], H[k]$ are periodic with period $N/2$

$$X_{DFT}\left[k + \frac{N}{2}\right] = G[k + N/2] + W_N^{(k+N/2)} H[k + N/2]$$

$$= G[k] \underbrace{-W_N^k}_{W_N^{k+N/2}=-W_N^k} H[k]$$

Radix-2 DIT FFT, cont

- **So, I can do 2 N/2 point DFTs and combine them to get an N-point DFT**
- **If I've already done the N/2 point DFT computation (we'll get to that in a minute), then this computation takes**

N complex additions (one for each value of k)

$\frac{N}{2}$ complex multiplications to compute $W_N^k H[k]$, one for each

$k = 0,1,2,....\frac{N}{2}-1$

...and this includes special simple cases ($W_N^k = 1,-1,j,-j$)

as full complex multiplications

Radix-2 DIT FFT, cont

- **So now I need two N/2 point DFTs**
- **...each of which can be divided into 2 N/4 point DFTs**
- **...using the same algorithm requires a total of**

$$\underbrace{2}_{\substack{\text{one for each}\\ N/2 \text{ point DFT}}} \times \underbrace{\frac{N}{2}}_{\substack{\text{total \# points}\\ \text{in } N/2 \text{ pt. DFT}}} = \underbrace{N}_{\substack{\text{total we}\\ \text{started}\\ \text{with}}} \text{ complex adds}$$

$$\text{and} \quad \underbrace{2}_{\substack{\text{one for each}\\ N/2 \text{ pt DFT}}} \times \underbrace{\frac{N}{4}}_{\substack{\text{one multiply}\\ \text{for every odd}\\ \text{point}}} = \underbrace{\frac{N}{2}}_{\substack{\text{same number}\\ \text{as for the } N \text{ pt}\\ \text{DFT}}} \text{ complex mults.}$$

- **...and then 4 N/4 point DFTs become 8 N/8 point DFTs, all the way down to N/2 2-point DFTs.**

Radix-2 DIT FFT, cont

- **It works the same way at each step of the process**
- **Assume I'm on the** r-th **step, where** $1 \le r \le \log_2 N$
- **I decompose the** $N/2^{r-1}$ **point DFT that I'm trying to compute into two** $N/2^r$ **point DFTs…**
- **…each of which requires**

$$N/2^r \text{ complex adds and } N/2^{r+1} \text{ complex multiplies}$$

- **…so there are a total of**

$$2\times\frac{N}{2^r}=\frac{N}{2^{r-1}} \text{ complex adds and } 2\times\frac{N}{2^{r+1}}=\frac{N}{2^r} \text{ complex mults}$$

- **…per** $N/2^{r-1}$-point **DFT, and there are a total of** 2^r **such DFTs to compute, so the** r-th step total is

$$2^{r-1}\times\frac{N}{2^{r-1}}=N \text{ adds and } 2^{r-1}\times\frac{N}{2^r}=\frac{N}{2} \text{ multiplies}$$

Radix-2 DIT FFT, cont

- **But** $1 \le r \le \log_2 N$, **so the total over all of the steps is**

$$\log_2 N \times N \text{ adds and } \log_2 N\times\frac{N}{2} \text{ multiplies}$$

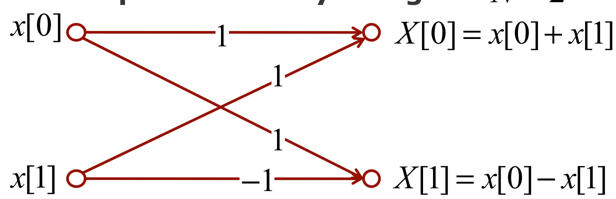
- **…or approximately** $N\log_2 N$ **operations!**

N	log2(N)	DFT	Nlog2(N)	FFT	FFT/DFT
2	1	4	2	3	0.75
4	2	16	8	12	0.75
8	3	64	24	36	0.5625
16	4	256	64	96	0.375
32	5	1,024	160	240	0.234375
64	6	4,096	384	576	0.140625
128	7	16,384	896	1,344	0.08203125
256	8	65,536	2,048	3,072	0.046875
512	9	262,144	4,608	6,912	0.02636719
1024	10	1,048,576	10,240	15,360	0.01464844
4096	12	16,777,216	49,152	73,728	0.00439453
65536	16	4,294,967,296	1,048,576	1,572,864	0.00036621
1048576	20	1,099,511,627,776	20,971,520	31,457,280	2.861E-05

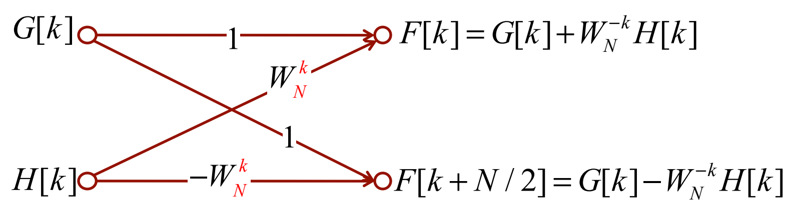
Extra complication may not be worth the effort, minimal "bang for the buck"

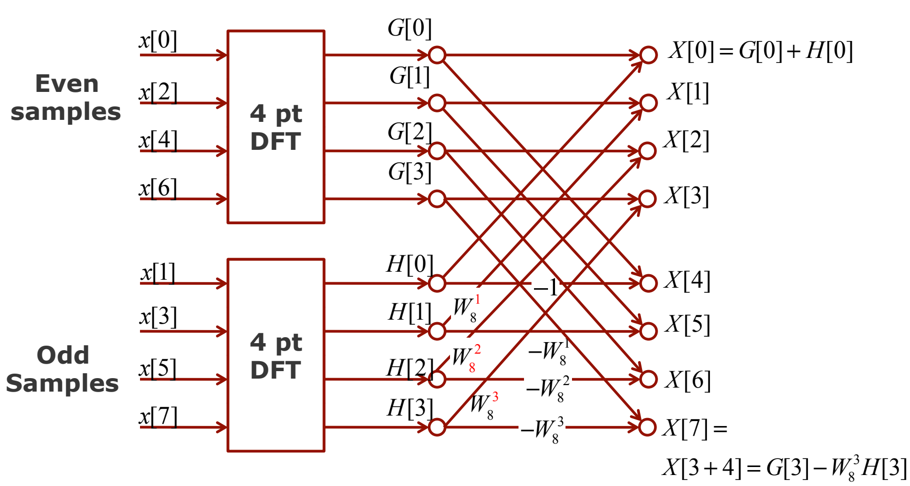
The simple FFT "butterfly"

- **The SFG of an FFT operation has a simple shape known as a butterfly…**
- **…with the simplest butterfly being an $N=2$ DFT/FFT**



- **The first stage of our DIT FFT looks like this**

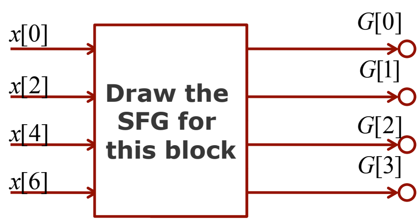




path transmittances assumed $=1$ unless otherwise noted

You try:

- Draw SFG to implement the “upper” 4-point DFT
- Include any branches necessary for re-ordering…
- …and the combination branches to generate the 4 point DFT



Click to remove

Click to remove

Full 8 point FFT butterfly

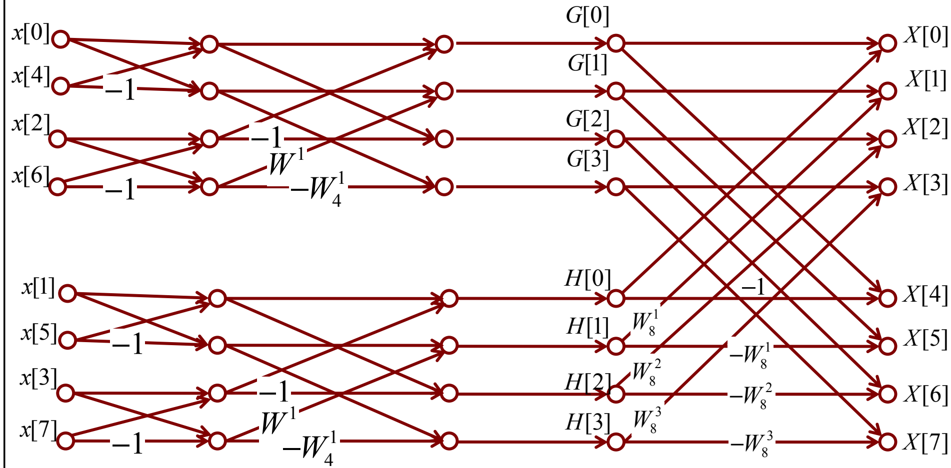
Click to remove

Work the 8-pt DIT algorithm

$$x[n] = \{\overset{\Downarrow}{1}, 1, -1, -1, 1, 1, -1, -1\}$$

$$W_8^1 = e^{-j2\pi/8} = e^{-j\pi/4} = 0.707 - j0.707$$

$$W_4^1 = W_8^2 = e^{-j2\pi/4} = e^{-j\pi/2} = -j$$



23-19

Bit reversed order

- **From this we have that the DIT FFT algorithm must reorder the inputs…**
- **…so that the outputs show up in the correct order.**
- **The decimation in frequency (DIF) FFT algorithm keeps the inputs in their natural (index) order…**
- **…and reorders the outputs *in exactly the same way!***
- **We call this reindexing *bit reversed order***

Binary	LSB on rt	LSB on LF
000	0	0
001	1	4
010	2	2
011	3	6
100	4	1
101	5	5
110	6	3
111	7	7

23-20

The other radix 2 algorithm

- **The other "famous" radix 2 algorithm is "decimation in frequency"**

$$X_{DFT}[k] = \underbrace{\sum_{n=0}^{N/2-1} x[n]W_N^{nk}}_{\text{first half of input data}} + \underbrace{\sum_{n=0}^{N/2-1} x[n+N/2]W_N^{(n+N/2)k}}_{\text{second half of input data}}$$

$$= \sum_{n=-}^{N/2-1} \left(x[n] + W_N^{(N/2)k} x[n+N/2]\right) W_N^{nk}$$

$$= \sum_{n=0}^{N/2-1} \left(x[n] + (-1)^k x[n+N/2]\right) W_N^{nk}$$

$$k \text{ even: } X[k] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+N/2]\right) W_{N/2}^{nk}$$

$$k \text{ odd: } X[k] = \sum_{n=0}^{N/2-1} \left(x[n] - x[n+N/2]\right) W_N^n W_{N/2}^{nk}$$

Now the dependence is on the *output index* being even or odd

...and we still have a nice $A \pm W_N^p B$ form

$$X[2k] = \sum_{n=0}^{N/2-1} \left(x[n] + x[n+N/2]\right) W_{N/2}^{nk}$$

$$= DFT\left(x[n] + x[n+N/2]\right), k = 0,1,2,...N/2-1$$

$$X[2k+1] = \sum_{n=0}^{N/2-1} \left(x[n] - x[n+N/2]W_N^k\right) W_{N/2}^{nk}$$

$$= DFT\left(\left(x[n] - x[n+N/2]\right)W_N^k\right), k = 0,1,2,...N/2-1$$

Let $g[n] = x[n] + x[n+N/2]$

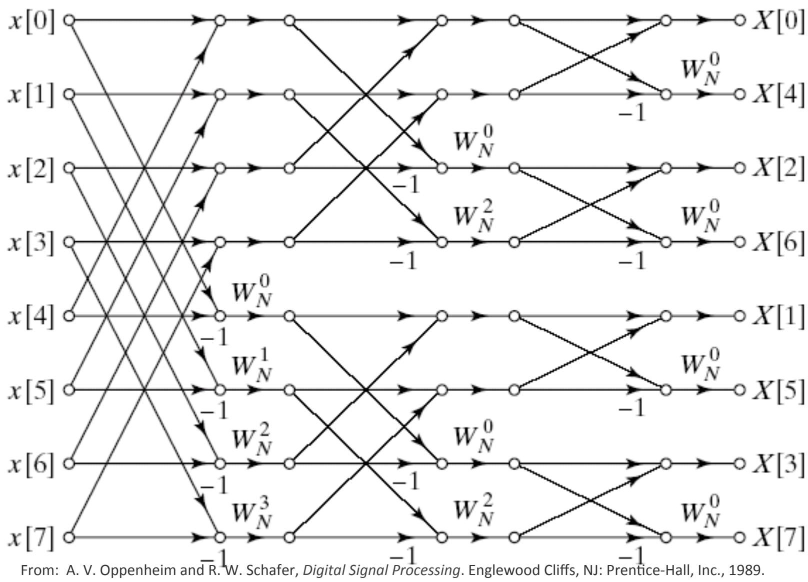
Let $h[n] = \left(x[n] - x[n+N/2]\right)W_N^k$

$X[2k] = G[k]$

$X[2k+1] = H[k]$

- **...and this, too has a butterfly structure**

DIF Butterfly structure



From: A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1989.

Wednesday 11/23

- **11 AM Discussion (questions & answers, nothing prepared)**
- **2:30 Lecture is your "test" period. We had the 2nd exam as take home, but I continued to lecture, so this is the time when I would have given the exam!**
- **There is homework, but it is short.**