Complexity of Filtering and the FFT

Dr. Deepa Kundur

University of Toronto

Digital Filtering in the Time Domain

Let $x(n)$ and $h(n)$ be real signals.

Let the support of $x(n)$ be $n = 0, 1, \ldots, N-1$. We are interested in determining $y(n)$ for $n = 0, 1, \ldots, N-1$.

$$\begin{aligned} y(n) &= x(n) * h(n) \\ &= \sum_{k=-\infty}^{\infty} x(k)h(n-k) \\ &= \sum_{k=0}^{N-1} x(k)h(n-k) \qquad n = 0, 1, \ldots, N-1 \end{aligned}$$

Digital Filtering in the Time Domain

Complexity of doing a brute-force convolution is given by:

- For fixed n:

$$y(n) = \sum_{k=0}^{N-1} x(k) \bullet h(n-k)$$

  - N real multiplications
  - $N-1$ real additions
- For all n (n=0, 1, . . . , N-1):
  - $N \cdot N = N^2 = O(N^2)$ real multiplications
  - $(N-1) \cdot N = N(N-1) = O(N^2)$ real additions

Complexity of Digital Filtering in the Time Domain

- Is $O(N^2)$ high?
  - Yes.
- **Idea:** Maybe filtering in the frequency domain can reduce complexity.

Discrete Fourier Transform (DFT)

- Frequency analysis of discrete-time signals is conveniently performed on a DSP.
- Therefore, both time-domain and frequency-domain signals must be discrete.
  - $x(t) \xrightarrow{sampling} x(n)$
  - $X(\omega) \xrightarrow{sampling} X(\frac{2\pi k}{N})$ or $X(k)$
- What happens when we sample in the frequency domain?

Fourier Duality

Time Domain	Frequency Domain	
sinc	rectangle	
rectangle	sinc	
sinc^2	triangle	
triangle	sinc^2	
ringing	truncation	
truncation	ringing	
discrete	periodic	
periodic	discrete	
continuous	aperiodic	
aperiodic	continuous	among others ...

DFT Intuition

aperiodic + dst in time $\overset{\mathcal{F}}{\longleftrightarrow}$ cts + periodic in freq

$\downarrow$ sampling

periodic + dst in time $\overset{\mathcal{F}}{\longleftrightarrow}$ dst + periodic in freq

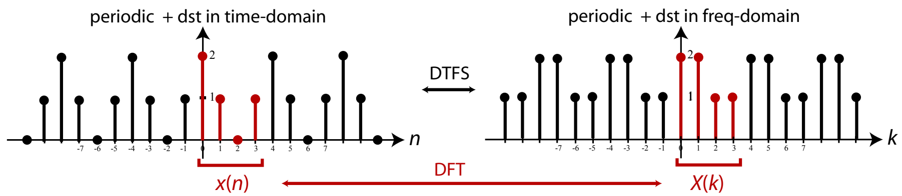
periodic + discrete $\overset{\text{DTFS}}{\longleftrightarrow}$ periodic + discrete

one period of dst samples $\overset{\text{DFT}}{\longleftrightarrow}$ one period of dst samples

$n = 0, 1, \ldots, N-1$ $\quad$ $k = 0, 1, \ldots, N-1$

DFT Intuition

Example



note: signal examples are artificial

Frequency Domain Sampling

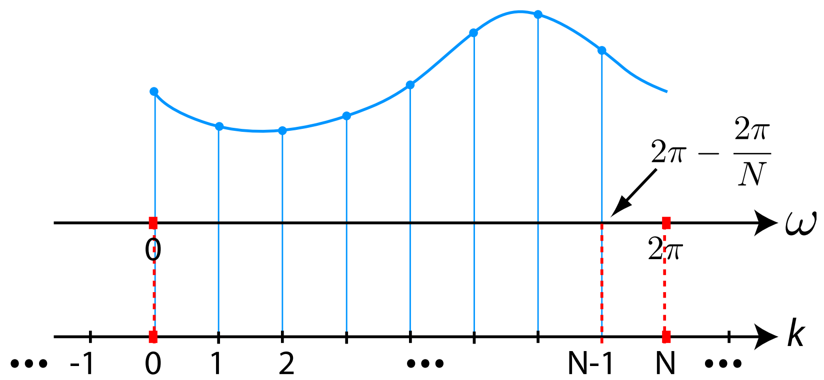
- Recall, sampling in time results in a periodic repetition in frequency.

$$x(n) = x_a(t)|_{t=nT} \quad \stackrel{\mathcal{F}}{\longleftrightarrow} \quad X(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} X_a(\omega + \frac{2\pi}{T}k)$$

- Similarly, sampling in frequency results in periodic repetition in time.

$$x_p(n) = \sum_{l=-\infty}^{\infty} x(n+lN) \quad \stackrel{\mathcal{F}}{\longleftrightarrow} \quad X(k) = X(\omega)|_{\omega=\frac{2\pi}{N}k}$$

Frequency Domain Sampling



Note: N is directly proportional to the sampling rate of ω; there are N samples per 2π

Frequency Domain Sampling and Reconstruction

- Therefore,

$$x(n) \stackrel{\mathcal{F}}{\longleftrightarrow} X(\omega)$$
$$x_p(n) \stackrel{\mathcal{F}}{\longleftrightarrow} X(k)$$

- Implications:
  - The samples of $X(\omega)$ can be used to reconstruct $x_p(n)$.

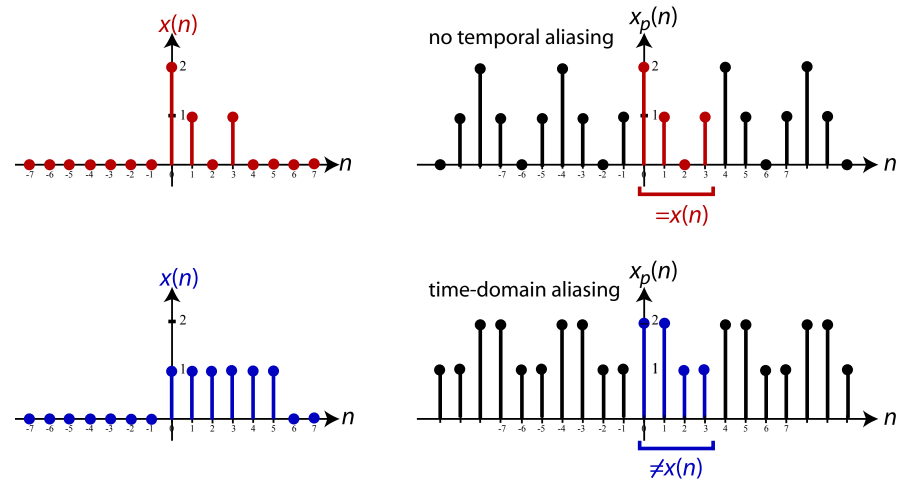
Frequency Domain Sampling and Reconstruction

- **Q:** Can we reconstruct $x(n)$ from the samples of $X(\omega)$?
  - Can we reconstruct $x(n)$ from $x_p(n)$?
- **A:** Maybe.

$$x_p(n) = \left[\sum_{l=-\infty}^{\infty} x(n+lN)\right]$$

Frequency Domain Sampling and Reconstruction

$N = 4$



Frequency Domain Sampling and Reconstruction

- $x(n)$ can be recovered from $x_p(n)$ if there is no overlap when taking the periodic repetition.
- If $x(n)$ is finite duration and non-zero in the interval $0 \leq n \leq L-1$, then

$$x(n) = x_p(n), \quad 0 \leq n \leq N-1 \quad \text{when } N \geq L$$

- If $N < L$ then, $x(n)$ cannot be recovered from $x_p(n)$.
  - or equivalently $X(\omega)$ cannot be recovered from its samples $X\left(\frac{2\pi}{N}k\right)$ due to time-domain aliasing

DTFT, DTFS and DFT

$$x(n) \text{ for all } n \quad \overset{\text{DTFT}}{\longleftrightarrow} \quad X(\omega) \text{ for all } \omega$$
$$x_p(n) \text{ for all } n \quad \overset{\text{DTFS}}{\longleftrightarrow} \quad X(k) \text{ for all } k$$
$$\hat{x}(n) \quad \overset{\text{DFT}}{\longleftrightarrow} \quad \hat{X}(k)$$

where

$$\hat{x}(n) = \begin{cases} x_p(n) & \text{for } n = 0, \ldots, N-1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\hat{X}(k) = \begin{cases} X(k) & \text{for } n = 0, \ldots, N-1 \\ 0 & \text{otherwise} \end{cases}$$

The Discrete Fourier Transform Pair

- DFT and inverse-DFT (IDFT):

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi k \frac{n}{N}}, \quad k = 0, 1, \ldots, N-1$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi k \frac{n}{N}}, \quad n = 0, 1, \ldots, N-1$$

Note: we drop the $\hat{\ }$ notation from now on.

Important DFT Properties

Property	Time Domain	Frequency Domain
Notation:	$x(n)$	$X(k)$
Periodicity:	$x(n) = x(n+N)$	$X(k) = X(k+N)$
Linearity:	$a_1x_1(n) + a_2x_2(n)$	$a_1X_1(k) + a_2X_2(k)$
Time reversal	$x(N-n)$	$X(N-k)$
Circular time shift:	$x((n-l))_N$	$X(k)e^{-j2\pi kl/N}$
Circular frequency shift:	$x(n)e^{j2\pi ln/N}$	$X((k-l))_N$
Complex conjugate:	$x^*(n)$	$X^*(N-k)$
Circular convolution:	$x_1(n) \otimes x_2(n)$	$X_1(k)X_2(k)$
Multiplication:	$x_1(n)x_2(n)$	$\frac{1}{N}X_1(k) \otimes X_2(k)$
Parseval's theorem:	$\sum_{n=0}^{N-1} x(n)y^*(n)$	$\frac{1}{N}\sum_{k=0}^{N-1} X(k)Y^*(k)$

Complexity of the DFT (and IDFT)

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi k\frac{n}{N}}, \quad k = 0, 1, \ldots, N-1$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi k\frac{n}{N}}, \quad n = 0, 1, \ldots, N-1$$

New notation: $W_N = e^{-j\frac{2\pi}{N}}$

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn}, \quad k = 0, 1, \ldots, N-1$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-kn}, \quad n = 0, 1, \ldots, N-1$$

Complexity of the DFT

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn}, \quad k = 0, 1, \ldots, N-1$$

Straightforward implementation of DFT to compute $X(k)$ for $k = 0, 1, \ldots, N-1$ requires:

- N^2 complex multiplications
  - 1 complex mult = $(a_R + ja_I)\times(b_R + jb_I) = (a_R \times b_R - a_I \times b_I) + j(a_R \times b_I + a_I \times b_R)$ = 4 real mult + 2 real add
  - $4N^2 = O(N^2)$ real multiplications

Complexity of the DFT

$$X(k) = \sum_{n=0}^{N-1} x(n)W_N^{kn}, \quad k = 0, 1, \ldots, N-1$$

Straightforward implementation of DFT to compute $X(k)$ for $k = 0, 1, \ldots, N-1$ requires:

- $N(N-1)$ complex additions
  - 1 complex add = $(a_R + ja_I) + (b_R + jb_I) = (a_R + b_R) + j(a_I + b_I)$ = 2 real add
  - $2N(N-1) + 2N^2$ (from complex mult) real additions $= 2N(2N-1) = O(N^2)$ real additions.

Complexity of the DFT

- Is $O(N^2)$ high?
  - Yes. A linear increase in the length of the DFT increases the complexity by a power of two.
  - Given the multitude of applications where Fourier analysis is employed (linear filtering, correlation analysis, spectrum analysis), a method of efficient computation is needed.

Complexity of the DFT

- How can we reduce complexity?
  - Exploit symmetry of the complex exponential.

$$
\begin{aligned}
W_N^{k+\frac{N}{2}} &= -W_N^k \\
\text{LHS} = W_N^{k+\frac{N}{2}} &= e^{-j2\pi\frac{k+N/2}{N}} = e^{-j2\pi\frac{k}{N}} e^{-j2\pi\frac{N/2}{N}} \\
&= e^{-j2\pi\frac{k}{N}} e^{-j\pi} \\
&= e^{-j2\pi\frac{k}{N}} \cdot (\cos(-\pi) + j\sin(-\pi)) \\
&= e^{-j2\pi\frac{k}{N}} (-1) \\
&= -e^{-j2\pi\frac{k}{N}} = -W_N^k = \text{RHS}
\end{aligned}
$$

Complexity of the DFT

- How can we reduce complexity?
  - Exploit periodicity of the complex exponential.

$$
\begin{aligned}
W_N^{k+N} &= W_N^k \\
\text{LHS} = W_N^{k+N} &= e^{-j2\pi\frac{k+N}{N}} = e^{-j2\pi\frac{k}{N}} e^{-j2\pi\frac{N}{N}} \\
&= e^{-j2\pi\frac{k}{N}} e^{-j2\pi} \\
&= e^{-j2\pi\frac{k}{N}} \cdot (\cos(-2\pi) + j\sin(-2\pi)) \\
&= e^{-j2\pi\frac{k}{N}} (1) \\
&= e^{-j2\pi\frac{k}{N}} = W_N^k = \text{RHS}
\end{aligned}
$$

Radix-2 FFT: Decimation-in-time

$$
\begin{aligned}
X(k) &= \sum_{n=0}^{N-1} x(n) W_N^{kn} \quad k = 0, 1, \ldots, N-1 \\
&= \sum_{n \text{ even}} x(n) W_N^{kn} + \sum_{n \text{ odd}} x(n) W_N^{kn} \\
&= \sum_{m=0}^{(N/2)-1} x(2m) W_N^{k(2m)} + \sum_{m=0}^{(N/2)-1} x(2m+1) W_N^{k(2m+1)} \\
&= \sum_{m=0}^{(N/2)-1} \underbrace{x(2m)}_{\equiv f_1(m)} W_N^{2km} + \sum_{m=0}^{(N/2)-1} \underbrace{x(2m+1)}_{\equiv f_2(m)} W_N^{2km} W_N^k
\end{aligned}
$$

Radix-2 FFT: Decimation-in-time

Note: $W_N^2 = e^{-j\frac{2\pi}{N}\cdot 2} = e^{-j\frac{2\pi}{N/2}} = W_{N/2}$

$$\begin{aligned} X(k) &= \sum_{m=0}^{(N/2)-1} \underbrace{x(2m)}_{\equiv f_1(m)} W_N^{2km} + \sum_{m=0}^{(N/2)-1} \underbrace{x(2m+1)}_{\equiv f_2(m)} W_N^{2km} W_N^k \\ &= \underbrace{\sum_{m=0}^{(N/2)-1} f_1(m) W_{N/2}^{km}}_{\frac{N}{2}-\text{DFT of } f_1(m)} + W_N^k \underbrace{\sum_{m=0}^{(N/2)-1} f_2(m) W_{N/2}^{km}}_{\frac{N}{2}-\text{DFT of } f_2(m)} \\ &= F_1(k) + W_N^k F_2(k), \quad k = 0, 1, \ldots, N-1 \end{aligned}$$

Radix-2 FFT: Decimation-in-time

Note: since $F_1(k)$ and $F_2(k)$ are $\frac{N}{2}$-DFTs:

$$\begin{aligned} F_1(k) &= F_1(k + \frac{N}{2}) \\ F_2(k) &= F_2(k + \frac{N}{2}) \end{aligned}$$

we have,

$$\begin{aligned} X(k) &= F_1(k) + W_N^k F_2(k) \\ X(k + \frac{N}{2}) &= F_1(k + \frac{N}{2}) + W_N^{k+\frac{N}{2}} F_2(k + \frac{N}{2}) \\ &= F_1(k) - W_N^k F_2(k) \end{aligned}$$

since $W_N^{k+\frac{N}{2}} = e^{-j\frac{2\pi}{N}(k+\frac{N}{2})} = e^{-j\frac{2\pi}{N}k} \cdot e^{-j\frac{2\pi}{N}\frac{N}{2}} = e^{-j\frac{2\pi}{N}k}(-1) = -W_N^k$

Radix-2 FFT: Decimation-in-time

Therefore,

$$\begin{aligned} X(k) &= F_1(k) + W_N^k F_2(k) \quad k = 0, 1, \ldots, \frac{N}{2} - 1 \\ X(k + \frac{N}{2}) &= F_1(k) - W_N^k F_2(k) \quad k = 0, 1, \ldots, \frac{N}{2} - 1 \end{aligned}$$

Radix-2 FFT: Decimation-in-time

Repeating the decimation-in-time for $f_1(n)$ and $f_2(n)$, we obtain:

$$\begin{aligned} v_{11}(n) &= f_1(2n) \quad n = 0, 1, \ldots, N/4 - 1 \\ v_{12}(n) &= f_1(2n+1) \quad n = 0, 1, \ldots, N/4 - 1 \\ v_{21}(n) &= f_2(2n) \quad n = 0, 1, \ldots, N/4 - 1 \\ v_{22}(n) &= f_2(2n+1) \quad n = 0, 1, \ldots, N/4 - 1 \end{aligned}$$

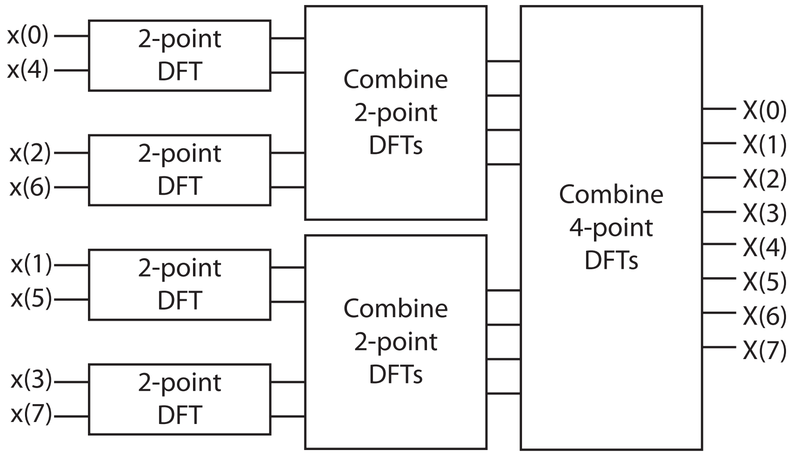
and

$$\begin{aligned} F_1(k) &= V_{11}(k) + W_{N/2}^k V_{12}(k) \quad k = 0, 1, \ldots, N/4 - 1 \\ F_1(k + N/4) &= V_{11}(k) - W_{N/2}^k V_{12}(k) \quad k = 0, 1, \ldots, N/4 - 1 \\ F_2(k) &= V_{21}(k) + W_{N/2}^k V_{22}(k) \quad k = 0, 1, \ldots, N/4 - 1 \\ F_2(k + N/4) &= V_{21}(k) - W_{N/2}^k V_{22}(k) \quad k = 0, 1, \ldots, N/4 - 1 \end{aligned}$$

consisting of $N/4$-DFTs.

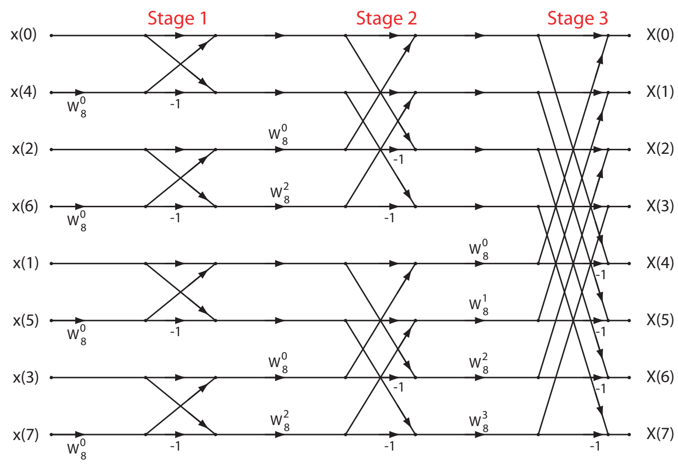
Radix-2 FFT: Decimation-in-time

For $N = 8$.

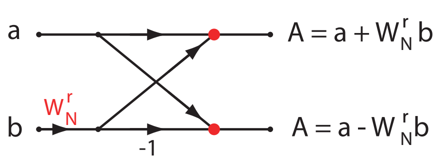


Radix-2 FFT: Decimation-in-time

For $N = 8$.



FFT Complexity



- Each butterfly requires:
  - one complex multiplication
  - two complex additions
- In total, there are:
  - $\frac{N}{2}$ butterflies per stage
  - $\log N$ stages
- Order of the overall DFT computation is: $O(N \log N)$

Convolution using FFT

To compute the convolution of $x(n)$ (support: $n = 0, 1, \ldots, L-1$) and $h(n)$ (support: $n = 0, 1, \ldots, M-1$):

1. Assign N to be the smallest power of 2 such that $N = 2^r \geq M + L - 1$.
2. Zero pad both $x(n)$ and $h(n)$ to have support $n = 0, 1, \ldots, N-1$.
3. Take the N-FFT of $x(n)$ to give $X(k)$, $k = 0, 1, \ldots, N-1$.
4. Take the N-FFT of $h(n)$ to give $H(k)$, $k = 0, 1, \ldots, N-1$.
5. Produce $Y(k) = X(k) \cdot H(k)$, $k = 0, 1, \ldots, N-1$.
6. Take the N-IFFT of $Y(k)$ to give $y(n)$, $n = 0, 1, \ldots, N-1$.

Convolution using FFT

To compute the convolution of $x(n)$ (support: $n = 0, 1, \ldots, L-1$) and $h(n)$ (support: $n = 0, 1, \ldots, M-1$):

1. Assign N to be the smallest power of 2 such that $N = 2^r \geq M + L - 1$.
2. Zero pad both $x(n)$ and $h(n)$ to have support $n = 0, 1, \ldots, N-1$.
   $O(1)$
3. Take the N-FFT of $x(n)$ to give $X(k)$, $k = 0, 1, \ldots, N-1$.
4. Take the N-FFT of $h(n)$ to give $H(k)$, $k = 0, 1, \ldots, N-1$.
   $O(N \log N)$
5. Produce $Y(k) = X(k) \cdot H(k)$, $k = 0, 1, \ldots, N-1$.
   $O(N)$
6. Take the N-IFFT of $Y(k)$ to give $y(n)$, $n = 0, 1, \ldots, N-1$.
   $O(N \log N)$

Complexity of Convolution using FFT

Therefore, the overall complexity of conducting convolution via the FFT is:

$$O(N \log N)$$

which is lower than $O(N^2)$ through direction computation of convolution in the time-domain. ■