ESE 531: Digital Signal Processing

Lec 19: March 29, 2018

Discrete Fourier Transform, Pt 2

Penn ESE 531 Spring 2018 – Khanna
Adapted from M. Lustig, EECS Berkeley

Today

- Review:
  - Discrete Fourier Transform (DFT)
  - Circular Convolution
- Fast Convolution Methods

Discrete Fourier Transform

- The DFT

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]W_N^{-kn} \quad \text{Inverse DFT, synthesis}$$

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

- It is understood that,

$$x[n] = 0 \quad \text{outside } 0 \le n \le N-1$$

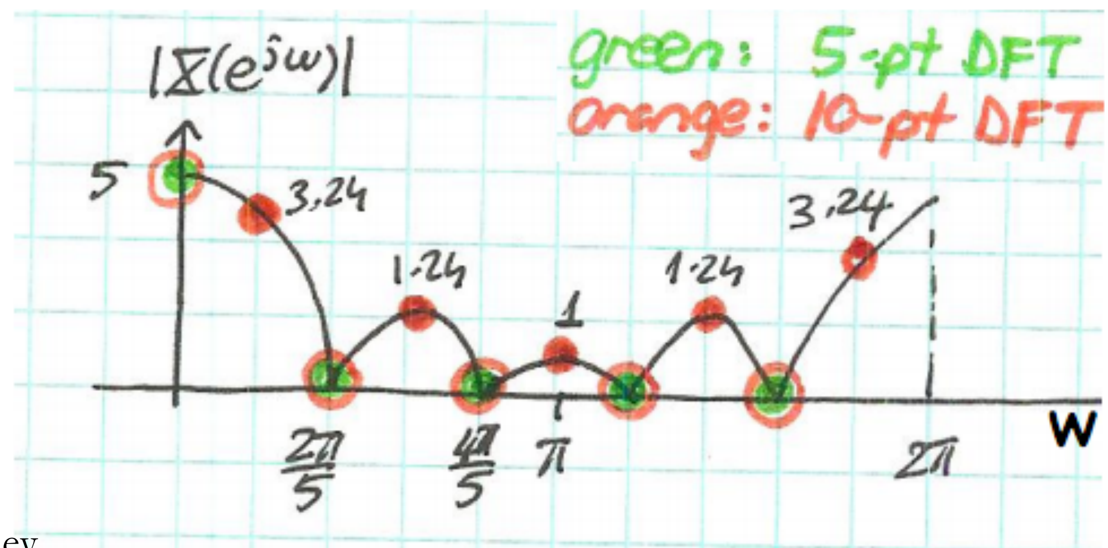
$$X[k] = 0 \quad \text{outside } 0 \le k \le N-1$$

DFT vs DTFT

- Back to example

$$X[k] = \sum_{n=0}^{4} W_{10}^{nk}$$

$$= e^{-j\frac{4\pi}{10}k} \frac{\sin(\frac{\pi}{2}k)}{\sin(\frac{\pi}{10}k)}$$



Properties of the DFS/DFT

Discrete Fourier Series			Discrete Fourier Transform		
Property	***N*-periodic sequence**	***N*-periodic DFS**	**Property**	***N*-point sequence**	***N*-point DFT**
	$\tilde{x}[n]$ $\tilde{x}_1[n],\ \tilde{x}_2[n]$	$\tilde{X}[k]$ $\tilde{X}_1[k],\ \tilde{X}_2[k]$		$x[n]$ $x_1[n],\ x_2[n]$	$X[k]$ $X_1[k],\ X_2[k]$
Linearity	$a\tilde{x}_1[n]+b\tilde{x}_2[n]$	$a\tilde{X}_1[k]+b\tilde{X}_2[k]$	Linearity	$ax_1[n]+bx_2[n]$	$aX_1[k]+bX_2[k]$
Duality	$\tilde{X}[n]$	$N\,\tilde{x}[-k]$	Duality	$X[n]$	$N\,x[((-k))_N]$
Time Shift	$\tilde{x}[n-m]$	$W_N^{km}\tilde{X}[k]$	Circular Time Shift	$x[((n-m))_N]$	$W_N^{km}X[k]$
Frequency Shift	$W_N^{-ln}\tilde{x}[n]$	$\tilde{X}[k-l]$	Circular Frequency Shift	$W_N^{-ln}x[n]$	$X[((k-l))_N]$
Periodic Convolution	$\sum_{m=0}^{N-1}\tilde{x}_1[m]\tilde{x}_2[n-m]$	$\tilde{X}_1[k]\tilde{X}_2[k]$	Circular Convolution	$\sum_{m=0}^{N-1}x_1[m]x_2[((n-m))_N]$	$X_1[k]X_2[k]$
Multiplication	$\tilde{x}_1[n]\tilde{x}_2[n]$	$\frac{1}{N}\sum_{l=0}^{N-1}\tilde{X}_1[l]\tilde{X}_2[k-l]$	Multiplication	$x_1[n]x_2[n]$	$\frac{1}{N}\sum_{l=0}^{N-1}X_1[l]X_2[((k-l))_N]$
Complex Conjugation	$\tilde{x}^*[n]$	$\tilde{X}^*[-k]$	Complex Conjugation	$x^*[n]$	$X^*[((-k))_N]$

Properties (Continued)

Time-Reversal and Complex Conjugation	$\tilde{x}^*[-n]$	$\tilde{X}^*[k]$	Time-Reversal and Complex Conjugation	$x^*[((-n))_N]$	$X^*[k]$
Real Part	$\mathrm{Re}\{\tilde{x}[n]\}$	$\tilde{X}_{ep}[k]=\frac{1}{2}\left(\tilde{X}[k]+\tilde{X}^*[-k]\right)$	Real Part	$\mathrm{Re}\{x[n]\}$	$X_{ep}[k]=\frac{1}{2}\left(X[k]+X^*[((-k))_N]\right)$
Imaginary Part	$j\,\mathrm{Im}\{\tilde{x}[n]\}$	$\tilde{X}_{op}[k]=\frac{1}{2}\left(\tilde{X}[k]-\tilde{X}^*[-k]\right)$	Imaginary Part	$j\,\mathrm{Im}\{x[n]\}$	$X_{op}[k]=\frac{1}{2}\left(X[k]-X^*[((-k))_N]\right)$
Even Part	$\tilde{x}_{ep}[n]=\frac{1}{2}\left(\tilde{x}[n]+\tilde{x}^*[-n]\right)$	$\mathrm{Re}\{\tilde{X}[k]\}$	Even Part	$x_{ep}[n]=\frac{1}{2}\left(x[n]+x^*[((-n))_N]\right)$	$\mathrm{Re}\{X[k]\}$
Odd Part	$\tilde{x}_{op}[n]=\frac{1}{2}\left(\tilde{x}[n]-\tilde{x}^*[-n]\right)$	$j\,\mathrm{Im}\{\tilde{X}[k]\}$	Odd Part	$x_{op}[n]=\frac{1}{2}\left(x[n]-x^*[((-n))_N]\right)$	$j\,\mathrm{Im}\{X[k]\}$
Symmetry for Real Sequence	$\tilde{x}[n]=\tilde{x}^*[n]$	$\tilde{X}[k]=\tilde{X}^*[-k]$ $\begin{cases}\mathrm{Re}\{\tilde{X}[k]\}=\mathrm{Re}\{\tilde{X}[-k]\}\\ \mathrm{Im}\{\tilde{X}[k]\}=-\mathrm{Im}\{\tilde{X}[-k]\}\end{cases}$ $\begin{cases}\lvert\tilde{X}[k]\rvert=\lvert\tilde{X}[-k]\rvert\\ \angle\tilde{X}[k]=-\angle\tilde{X}[-k]\end{cases}$	Symmetry for Real Sequence	$x[n]=x^*[n]$	$X[k]=X^*[((-k))_N]$ $\begin{cases}\mathrm{Re}\{X[k]\}=\mathrm{Re}\{X[((-k))_N]\}\\ \mathrm{Im}\{X[k]\}=-\mathrm{Im}\{X[((-k))_N]\}\end{cases}$ $\begin{cases}\lvert X[k]\rvert=\lvert X[((-k))_N]\rvert\\ \angle X[k]=-\angle X[((-k))_N]\end{cases}$
Parseval's Identity	$\sum_{n=0}^{N-1}\tilde{x}_1[n]\tilde{x}_2^*[n]=\frac{1}{N}\sum_{k=0}^{N-1}\tilde{X}_1[k]\tilde{X}_2^*[k]$ $\sum_{n=0}^{N-1}\lvert\tilde{x}[n]\rvert^2=\frac{1}{N}\sum_{k=0}^{N-1}\lvert\tilde{X}[k]\rvert^2$		Parseval's Identity	$\sum_{n=0}^{N-1}x_1[n]x_2^*[n]=\frac{1}{N}\sum_{k=0}^{N-1}X_1[k]X_2^*[k]$ $\sum_{n=0}^{N-1}\lvert x[n]\rvert^2=\frac{1}{N}\sum_{k=0}^{N-1}\lvert X[k]\rvert^2$	

Duality

If $x \xrightarrow{DFT} X$, then $\{X[n]\}_{n=0}^{N-1} \xrightarrow{DFT} N\{x[((-k))_N]\}_{k=0}^{N-1}$

Duality

$$\text{If } x \xrightarrow{DFT} X, \text{ then } \{X[n]\}_{n=0}^{N-1} \xrightarrow{DFT} N\{x[((-k))_N]\}_{k=0}^{N-1}$$

$$\tilde{x}[n] \stackrel{\mathcal{DFS}}{\longleftrightarrow} \tilde{X}[k],$$

$$\tilde{X}[n] \stackrel{\mathcal{DFS}}{\longleftrightarrow} N\tilde{x}[-k].$$

Proof of Duality

DFT of $\{x[n]\}_{n=0}^{N-1}$ is $X[k]=\sum_{p=0}^{N-1} x[p]e^{-j\frac{2\pi}{N}kp}$; $k \le 0 \le N-1$

Proof of Duality

DFT of $\{x[n]\}_{n=0}^{N-1}$ is $\quad X[k]=\sum_{p=0}^{N-1} x[p]e^{-j\frac{2\pi}{N}kp}; \qquad k \le 0 \le N-1$

DFT of $\{X[n]\}_{n=0}^{N-1}$ is $\quad \sum_{n=0}^{N-1} \underbrace{\sum_{p=0}^{N-1} x[p]e^{-j\frac{2\pi}{N}pn}}_{X[n]} e^{-j\frac{2\pi}{N}kn}, \qquad k \le 0 \le N-1$

Proof of Duality

DFT of $\{x[n]\}_{n=0}^{N-1}$ is $\quad X[k] = \sum_{p=0}^{N-1} x[p] e^{-j\frac{2\pi}{N}kp}; \qquad k \le 0 \le N-1$

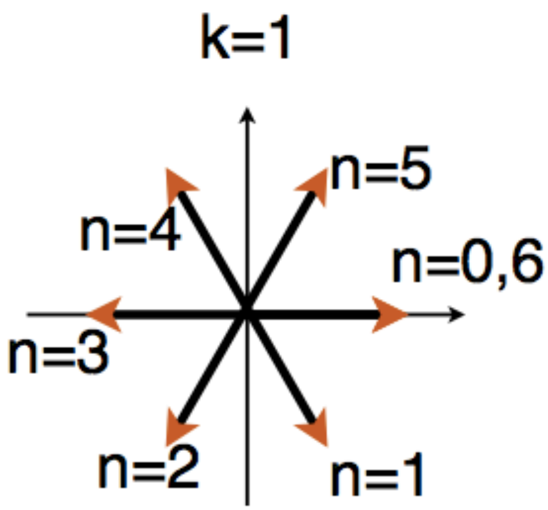
DFT of $\{X[n]\}_{n=0}^{N-1}$ is $\quad \sum_{n=0}^{N-1} \underbrace{\sum_{p=0}^{N-1} x[p] e^{-j\frac{2\pi}{N}pn}}_{X[n]} e^{-j\frac{2\pi}{N}kn}, \qquad k \le 0 \le N-1$

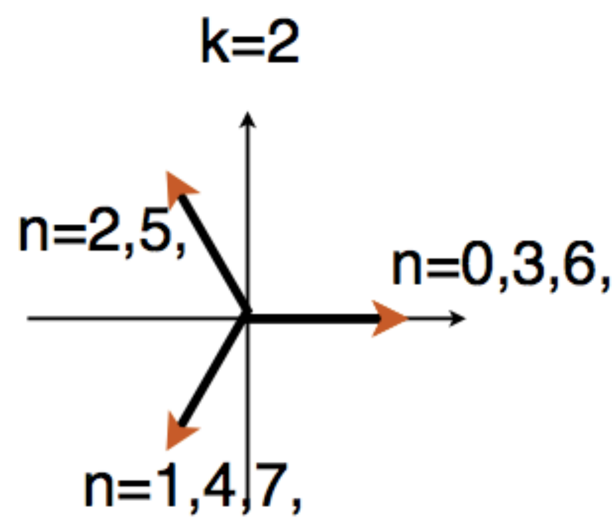
$$= \sum_{p=0}^{N-1} x[p] \underbrace{\sum_{n=0}^{N-1} e^{-j\frac{2\pi}{N}(p+k)n}}_{\substack{N \text{ for } ((p+k))_N = 0, \\ 0 \text{ otherwise}}}$$

Discrete Fourier Series

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

- Properties of WN:
  - $W_N^0 = W_N^N = W_N^{2N} = \ldots = 1$
  - $W_N^{k+r} = W_N^k W_N^r$ and, $W_N^{k+N} = W_N^k$
- Example: W_N^{kn} (N=6)





Proof of Duality

DFT of $\{x[n]\}_{n=0}^{N-1}$ is $X[k]=\sum_{p=0}^{N-1} x[p]e^{-j\frac{2\pi}{N}kp}$; $\quad k \le 0 \le N-1$

DFT of $\{X[n]\}_{n=0}^{N-1}$ is $\sum_{n=0}^{N-1} \underbrace{\sum_{p=0}^{N-1} x[p]e^{-j\frac{2\pi}{N}pn}}_{X[n]} e^{-j\frac{2\pi}{N}kn}$, $\quad k \le 0 \le N-1$

$$= \sum_{p=0}^{N-1} x[p] \underbrace{\sum_{n=0}^{N-1} e^{-j\frac{2\pi}{N}(p+k)n}}_{\substack{N \text{ for } ((p+k))_N=0, \\ 0 \text{ otherwise}}}$$

$((p+k))_N = 0$ for $0 \le p$ & $k \le N-1 \Rightarrow p = ((-k))_N$

$\therefore$ DFT of $\{X[n]\}_{n=0}^{N-1}$ is $N\{x[((-k))_N]\}_{k=0}^{N-1}$

Circular Convolution

- Circular Convolution:

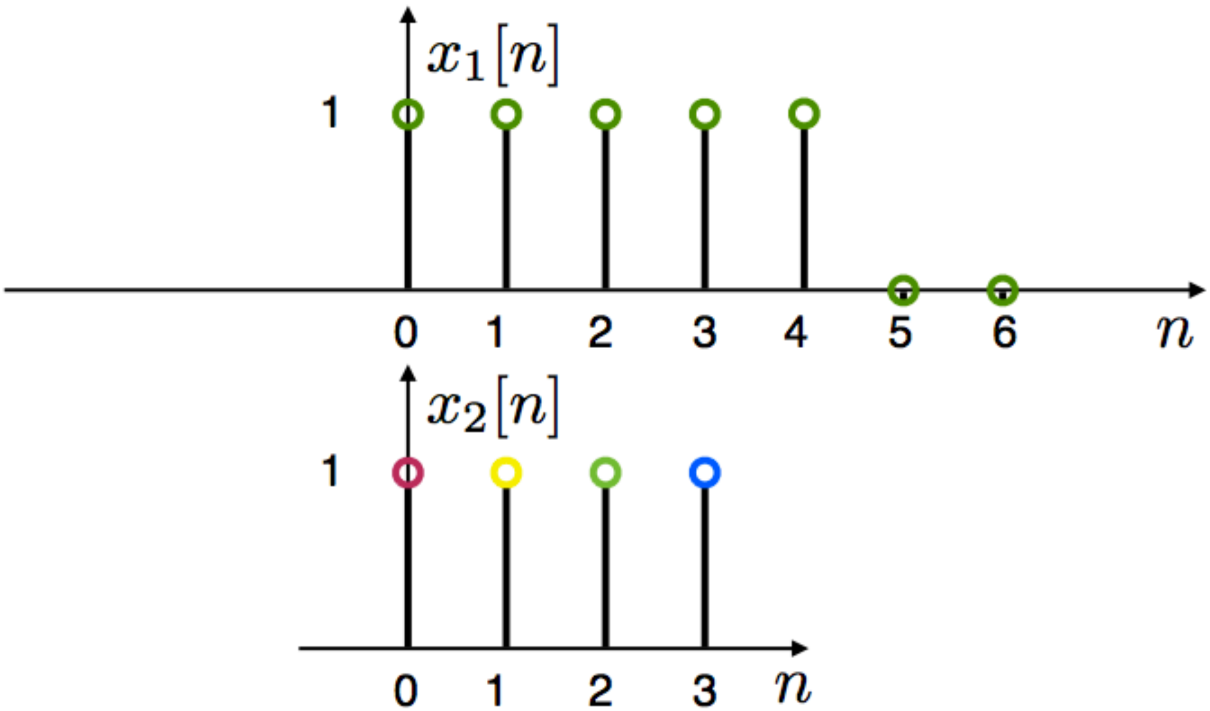
$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

For two signals of length N

Note: Circular convolution is commutative

$$x_2[n] \circledN x_1[n] = x_1[n] \circledN x_2[n]$$

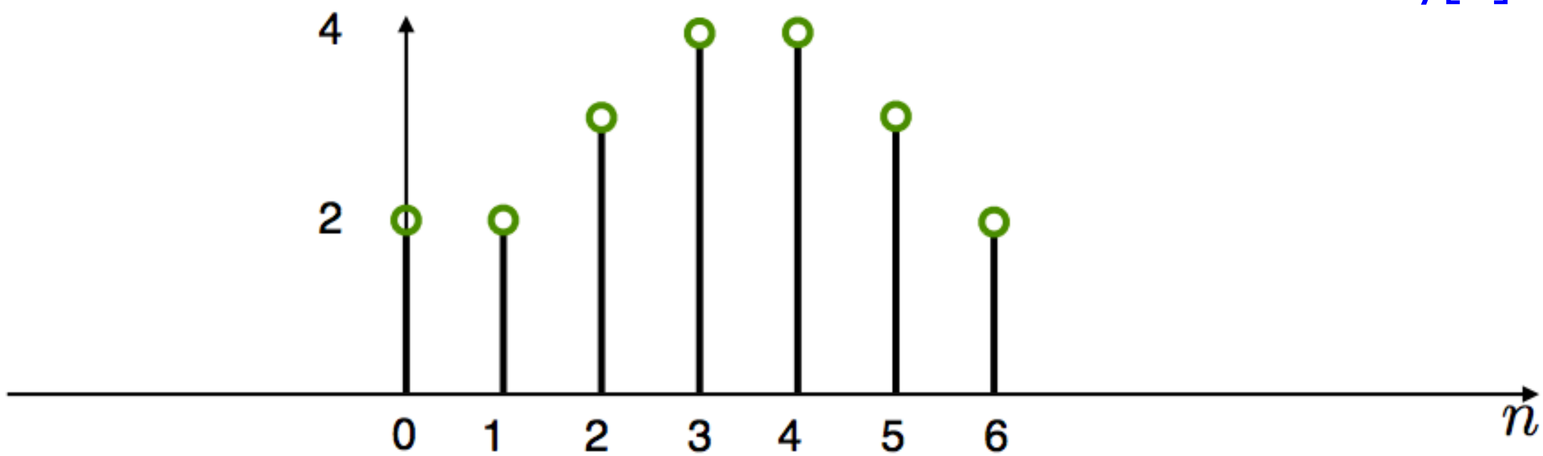
Compute Circular Convolution Sum



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

Result

y[0]=2
y[1]=2
y[2]=3
y[3]=4



$$x_1[n] \circledN x_2[n] \triangleq \sum_{m=0}^{N-1} x_1[m] x_2[((n-m))_N]$$

Circular Convolution

- For $x_1[n]$ and $x_2[n]$ with length N

$$x_1[n] \circledN x_2[n] \leftrightarrow X_1[k] \cdot X_2[k]$$

  - Very useful!! (for linear convolutions with DFT)

Multiplication

- For $x_1[n]$ and $x_2[n]$ with length N

$$x_1[n] \cdot x_2[n] \leftrightarrow \frac{1}{N} X_1[k] \circledN X_2[k]$$

Linear Convolution

- Next….
  - Using DFT, circular convolution is easy
    - Matrix multiplication… more later
  - But, linear convolution is useful, not circular
  - So, show how to perform linear convolution with circular convolution
  - Use DFT to do linear convolution (via circular convolution)

Linear Convolution

- We start with two non-periodic sequences:

$$x[n] \quad 0 \le n \le L-1$$
$$h[n] \quad 0 \le n \le P-1$$

  - E.g. x[n] is a signal and h[n] a filter's impulse response

Linear Convolution

- We start with two non-periodic sequences:

$$x[n] \quad 0 \leq n \leq L-1$$
$$h[n] \quad 0 \leq n \leq P-1$$

  - E.g. x[n] is a signal and h[n] a filter's impulse response

- We want to compute the linear convolution:

$$y[n] = x[n] * h[n] = \sum_{m=0}^{L-1} x[m]h[n-m]$$

  - y[n] is nonzero for $0 \leq n \leq L+P-2$ (ie. length M=L+P-1)

Requires L*P multiplications

Linear Convolution via Circular Convolution

- Zero-pad x[n] by P-1 zeros

$$x_{\mathrm{zp}}[n] = \begin{cases} x[n] & 0 \le n \le L-1 \\ 0 & L \le n \le L+P-2 \end{cases}$$

- Zero-pad h[n] by L-1 zeros

$$h_{\mathrm{zp}}[n] = \begin{cases} h[n] & 0 \le n \le P-1 \\ 0 & P \le n \le L+P-2 \end{cases}$$

- Now, both sequences are length M=L+P-1

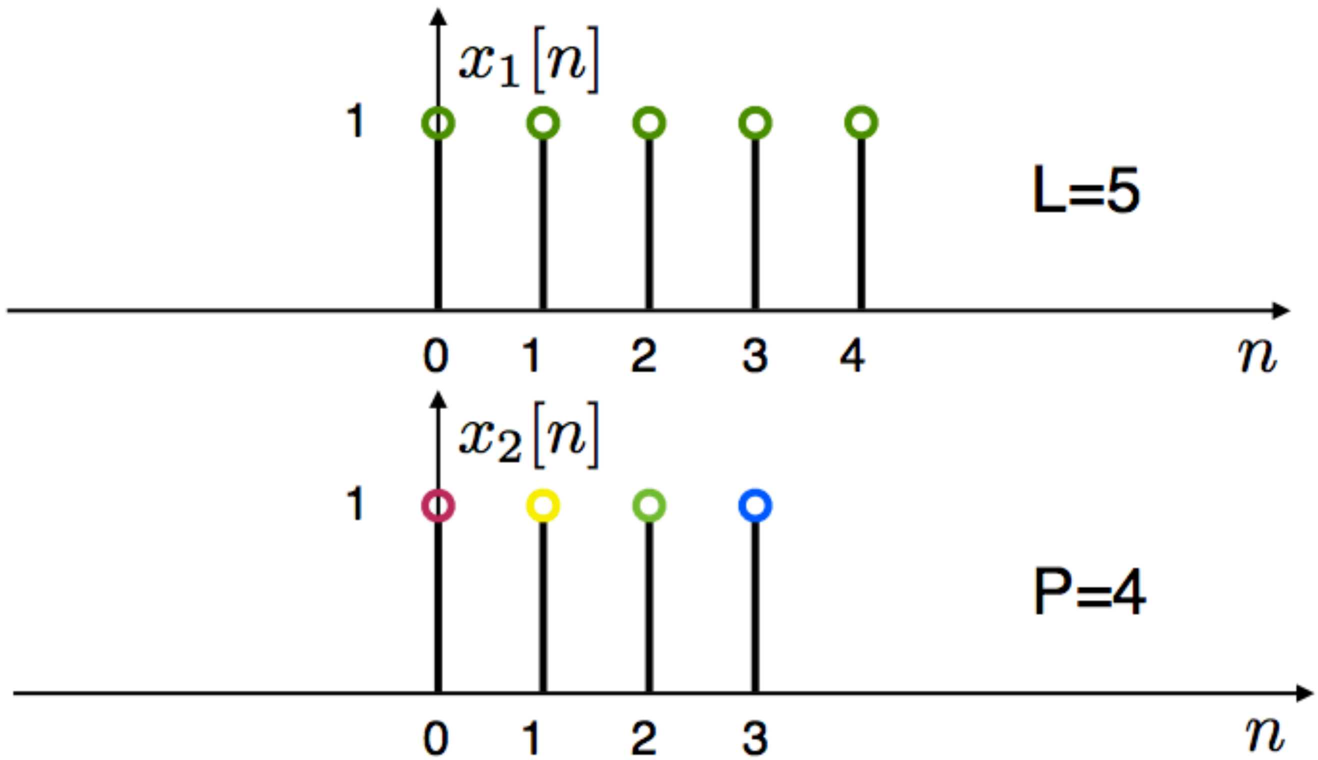
Linear Convolution via Circular Convolution

- Now, both sequences are length M=L+P-1
- We can now compute the linear convolution using a circular one with length M=L+P-1

Linear convolution via circular

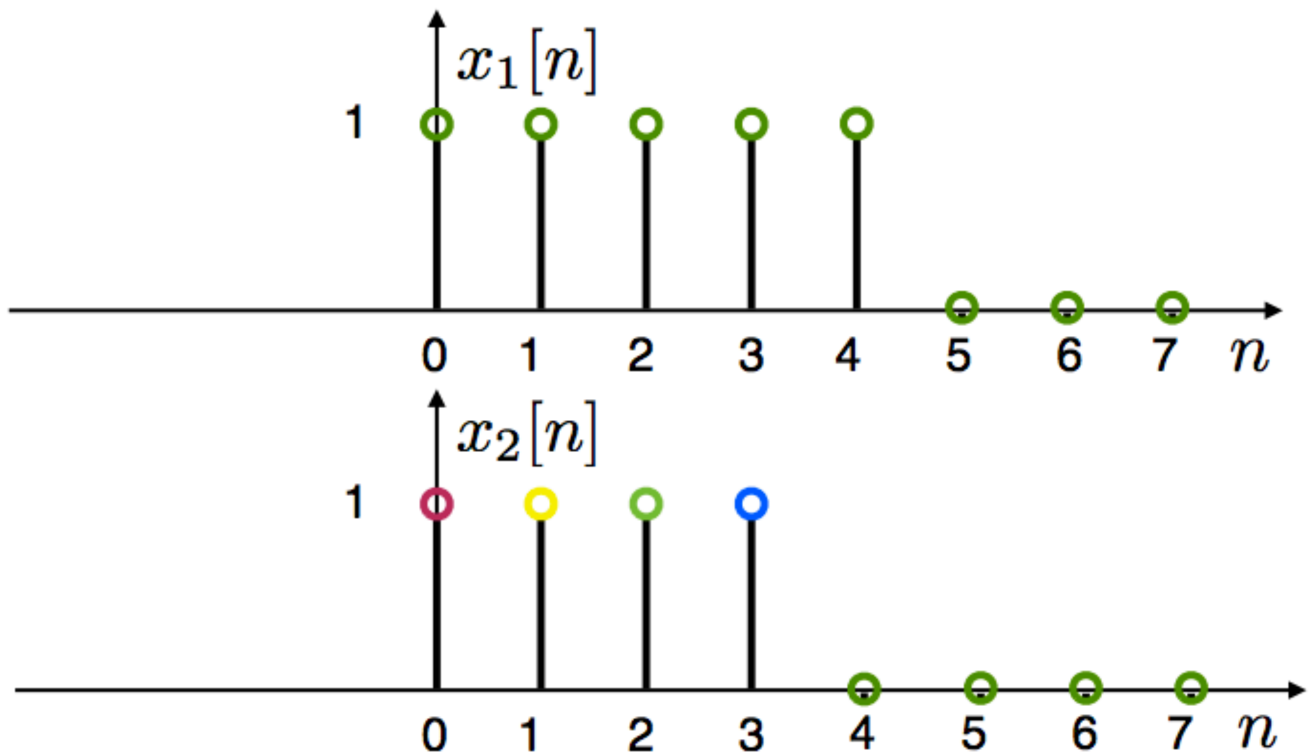
$$y[n] = x[n] * y[n] = \begin{cases} x_{zp}[n] \circledM h_{zp}[n] & 0 \leq n \leq M-1 \\ 0 & \text{otherwise} \end{cases}$$

Example



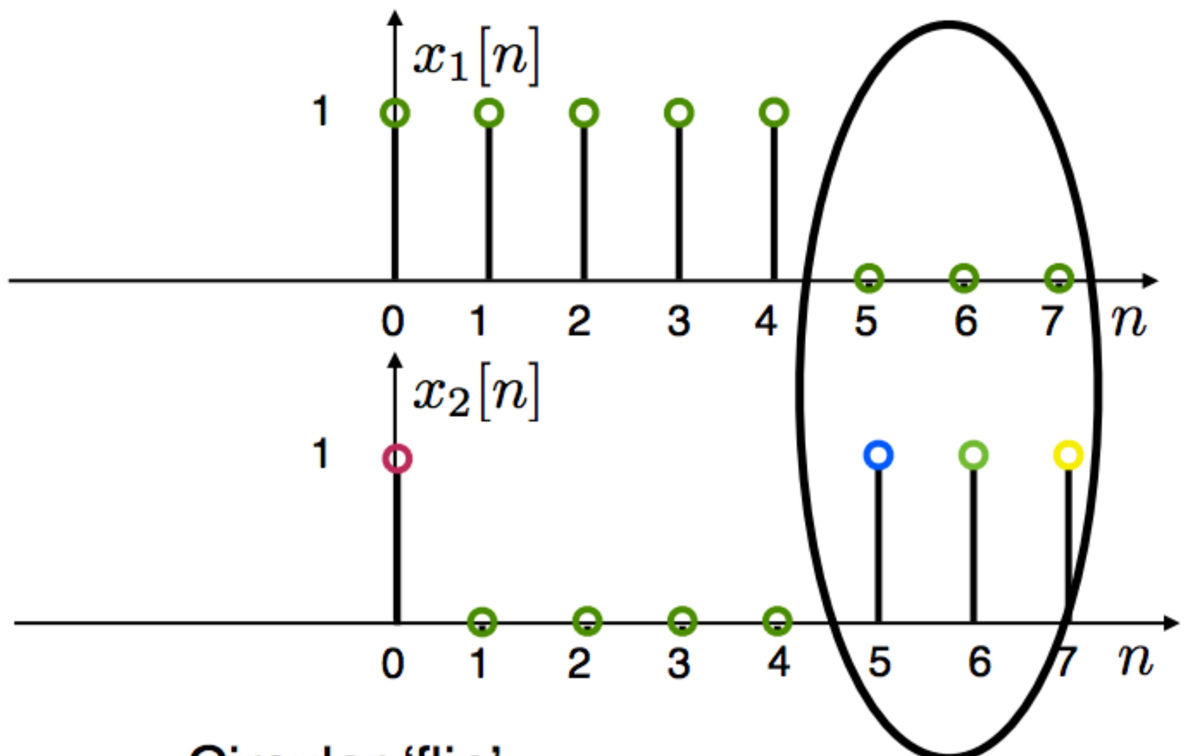
M = L + P - 1 = 8

Example



M = L + P - 1 = 8

Example



Circular 'flip'

M = L + P - 1 = 8

$$y[n] = x_1[n] \circledast_8 x_2[n] = x_1[n] * x_2[n]$$

Linear Convolution with DFT

- In practice we can implement a circulant convolution using the DFT property:

$$
\begin{aligned}
x[n] * h[n] &= x_{\mathrm{zp}}[n] \circledM h_{\mathrm{zp}}[n] \\
&= \mathcal{DFT}^{-1}\{\mathcal{DFT}\{x_{\mathrm{zp}}[n]\} \cdot \mathcal{DFT}\{h_{\mathrm{zp}}[n]\}\}
\end{aligned}
$$

for $0 \le n \le M-1$, $M=L+P-1$

Linear Convolution with DFT

- In practice we can implement a circulant convolution using the DFT property:

$$\begin{aligned} x[n] * h[n] &= x_{\text{zp}}[n] \circledM h_{\text{zp}}[n] \\ &= \mathcal{DFT}^{-1}\{\mathcal{DFT}\{x_{\text{zp}}[n]\} \cdot \mathcal{DFT}\{h_{\text{zp}}[n]\}\} \end{aligned}$$

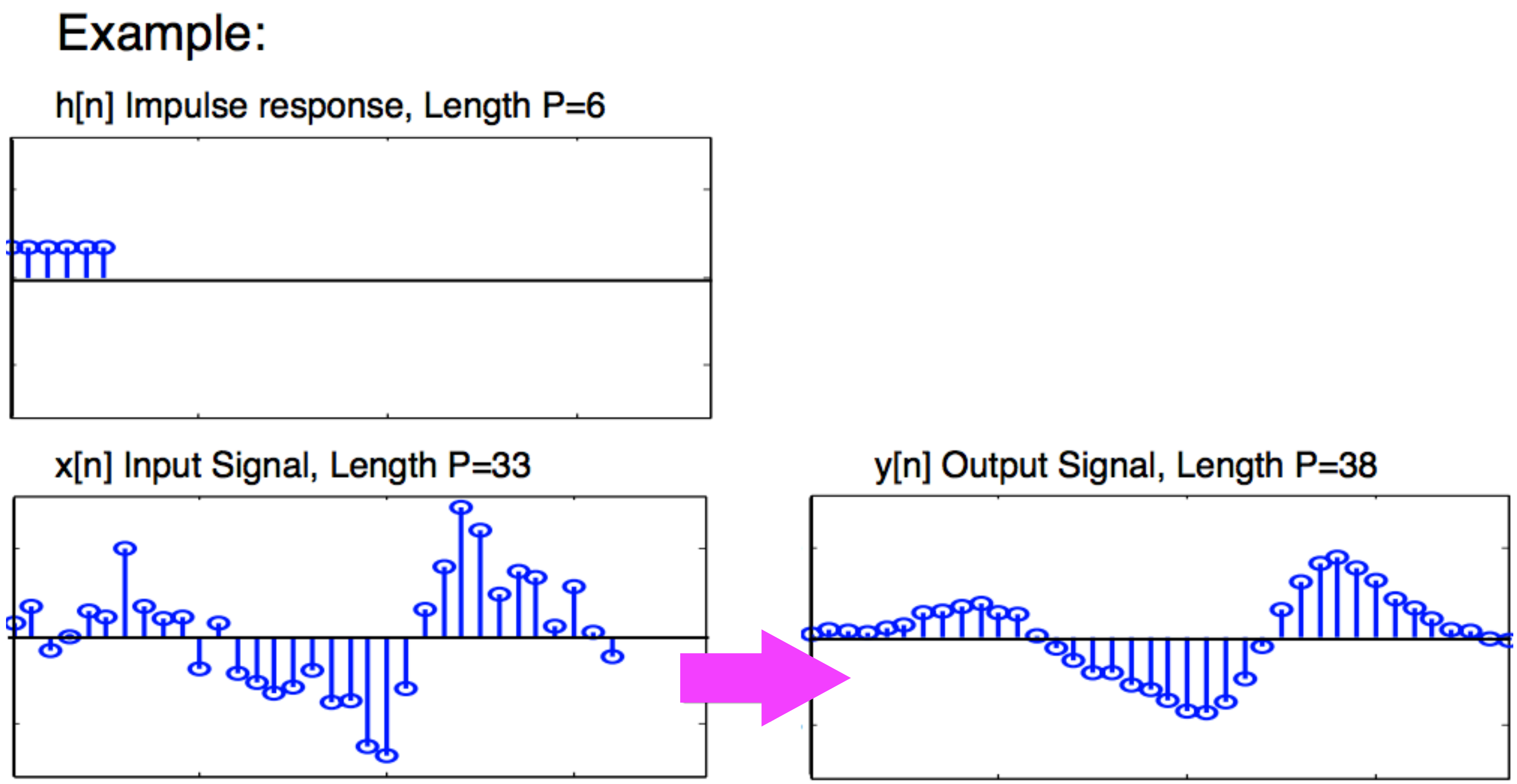
for $0 \leq n \leq M-1$, $M=L+P-1$

- Advantage: DFT can be computed with $N\log_2 N$ complexity (FFT algorithm later!)
- Drawback: Must wait for all the samples -- huge delay -- incompatible with real-time filtering

Block Convolution

- Problem:
  - An input signal x[n], has very long length (could be considered infinite)
  - An impulse response h[n] has length P
  - We want to take advantage of DFT/FFT and compute convolutions in blocks that are shorter than the signal
- Approach:
  - Break the signal into small blocks
  - Compute convolutions
  - Combine the results

Block Convolution



Overlap-Add Method

- Decompose into non-overlapping segments

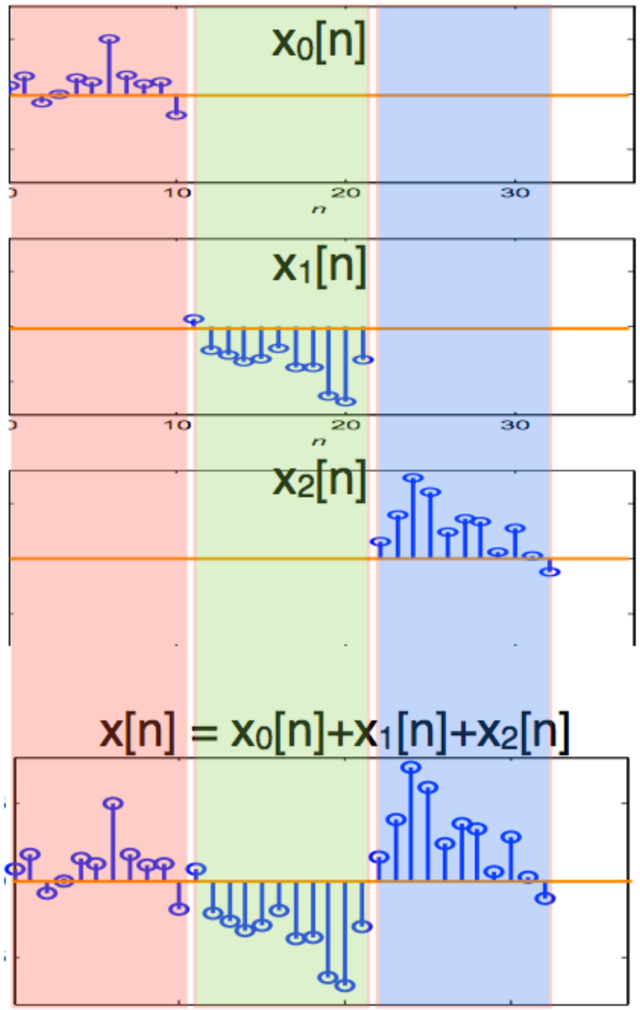
$$x_r[n] = \begin{cases} x[n] & rL \leq n < (r+1)L \\ 0 & \text{otherwise} \end{cases}$$

- The input signal is the sum of segments

$$x[n] = \sum_{r=0}^{\infty} x_r[n]$$

Example

L=11



Overlap-Add Method

$$x[n] = \sum_{r=0}^{\infty} x_r[n]$$

- The output is:

$$y[n] = x[n] * h[n] = \sum_{r=0}^{\infty} x_r[n] * h[n]$$

- Each output segment $x_r[n]*h[n]$ is length N=L+P-1
  - h[n] has length P
  - x_r[n] has length L

Overlap-Add Method

- We can compute $x_r[n]*h[n]$ using circular convolution with the DFT
- Using the DFT:
  - Zero-pad $x_r[n]$ to length N
  - Zero-pad h[n] to length N and compute $DFT_N\{h_{zp}[n]\}$
    - Only need to do once

Overlap-Add Method

- We can compute $x_r[n]*h[n]$ using circular convolution with the DFT
- Using the DFT:
  - Zero-pad $x_r[n]$ to length N
  - Zero-pad h[n] to length N and compute $DFT_N\{h_{zp}[n]\}$
    - Only need to do once
  - Compute:

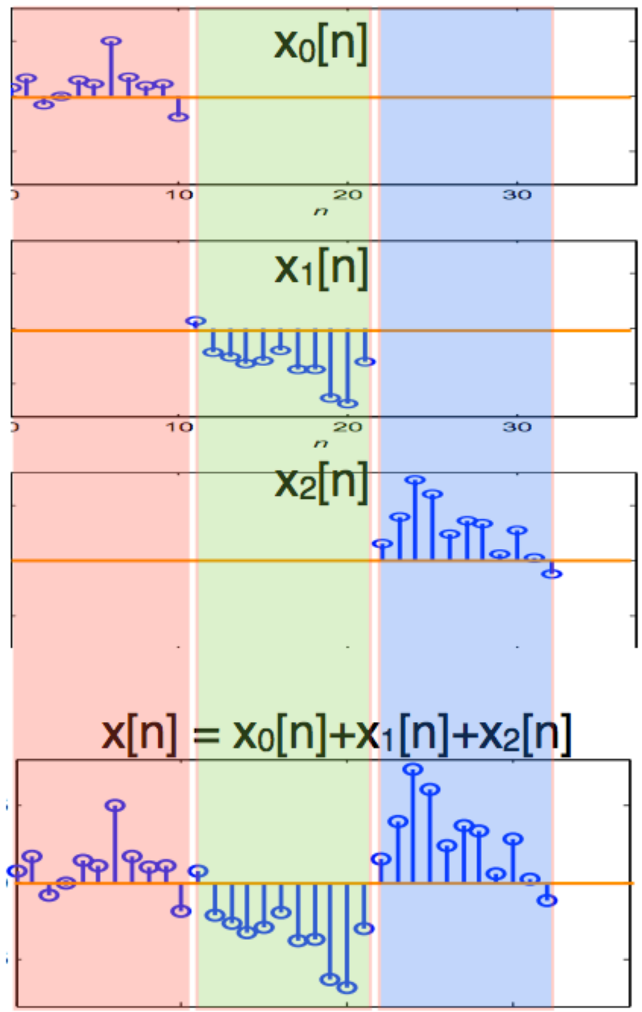
$$x_r[n] * h[n] = \mathrm{DFT}^{-1}\left\{\mathrm{DFT}\{x_{r,zp}[n]\} \cdot \mathrm{DFT}\{h_{zp}[n]\}\right\}$$

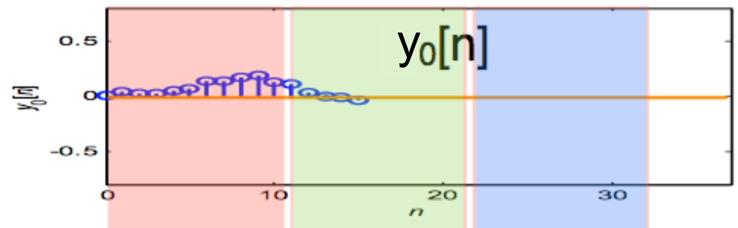
- Results are of length N=L+P-1
  - Neighboring results overlap by P-1
  - Add overlaps to get final sequence

Example of Overlap-Add

L+P-1=16

L=11





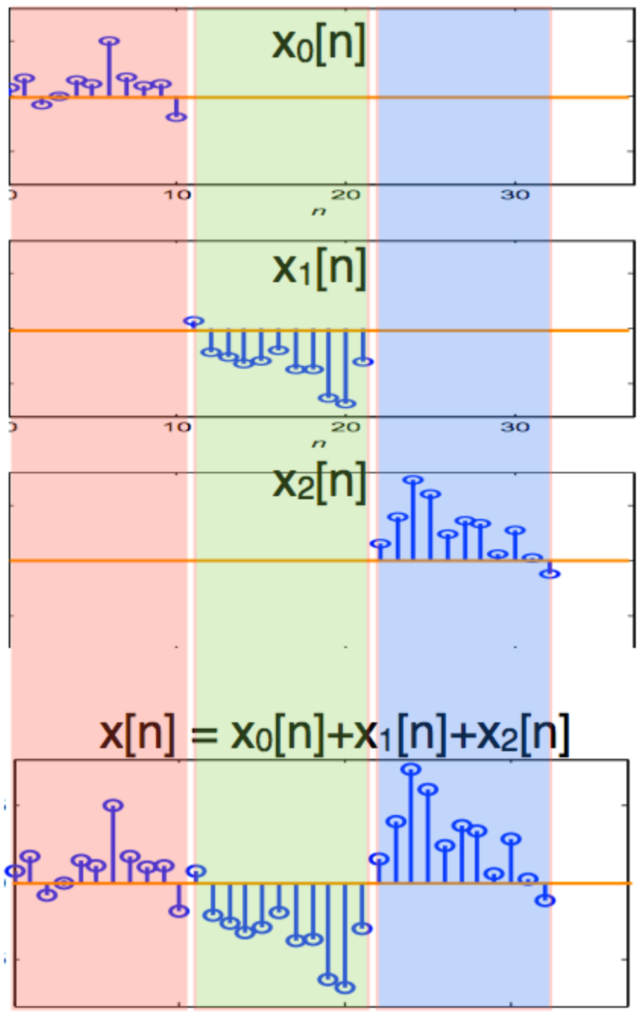
Example:

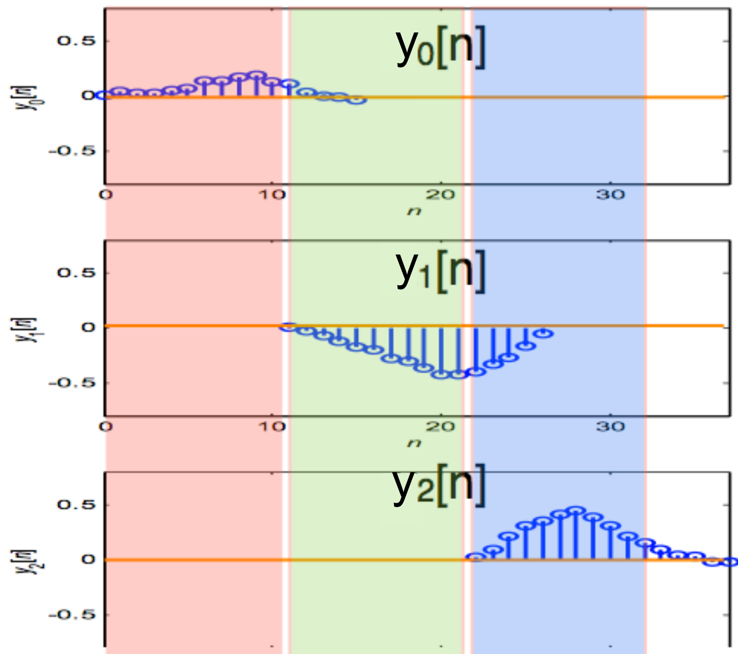
h[n] Impulse response, Length P=6

Example of Overlap-Add

L+P-1=16

L=11

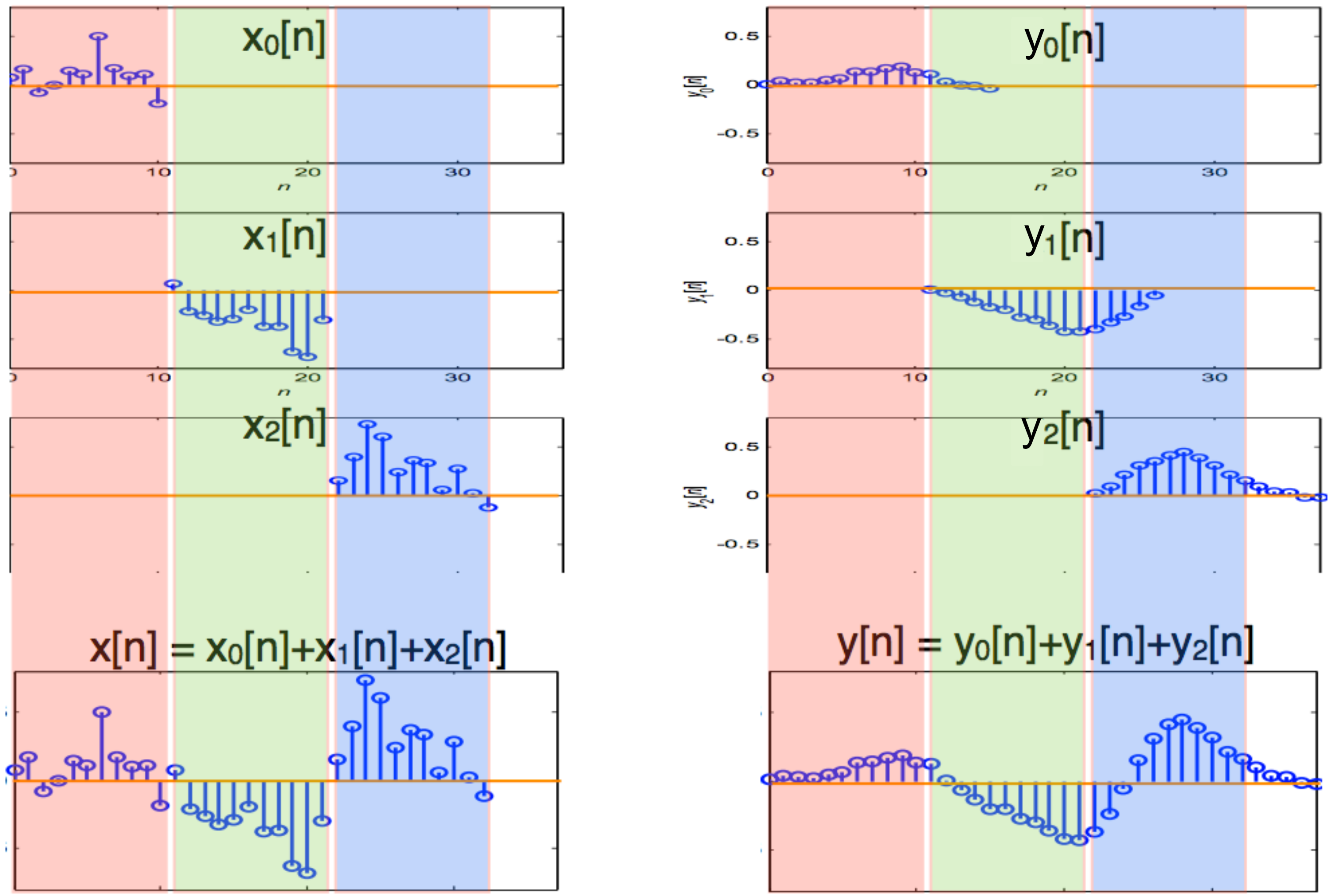




Example of Overlap-Add

L+P-1=16

L=11



Overlap-Save Method

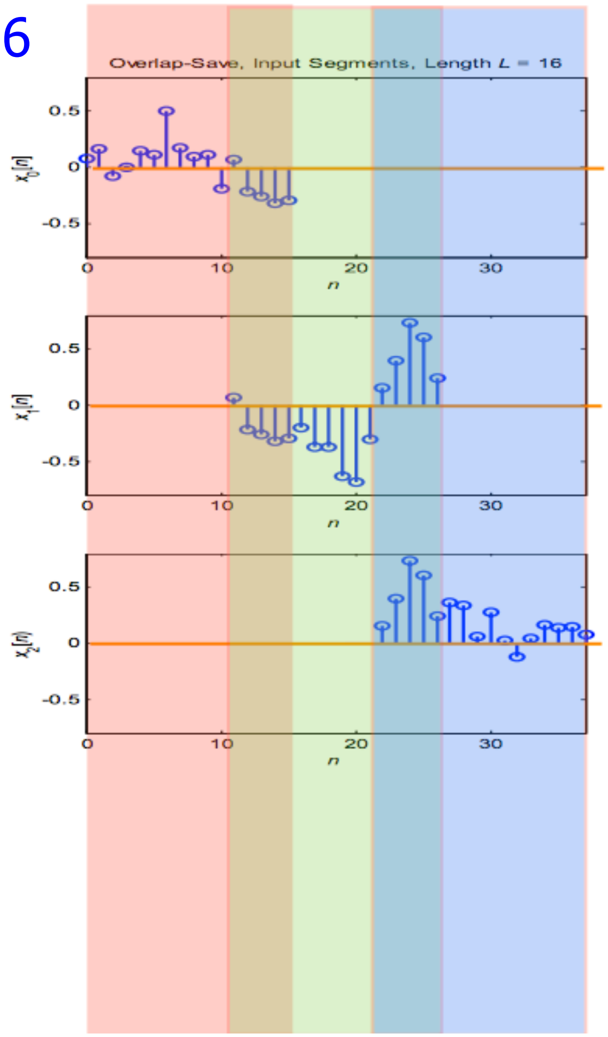
- Basic idea:
- Split input into overlapping segments with length L+P-1
  - P-1 sample overlap

$$x_r[n] = \begin{cases} x[n] & rL \le n < (r+1)L + P \\ 0 & \text{otherwise} \end{cases}$$

- Perform circular convolution in each segment, and keep the L sample portion which is a valid linear convolution

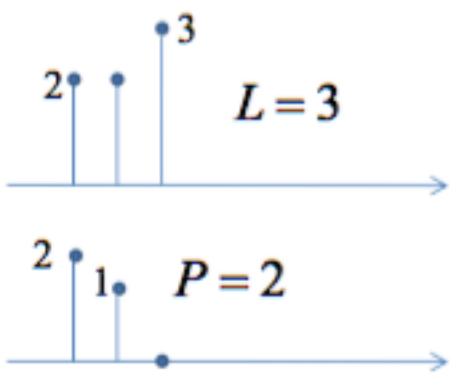
Example of Overlap-Save

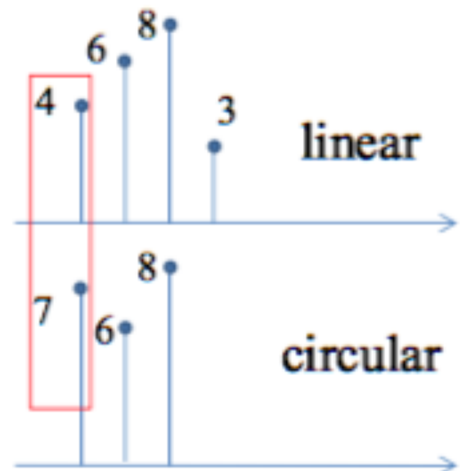
L+P-1=16



Circular to Linear Convolution

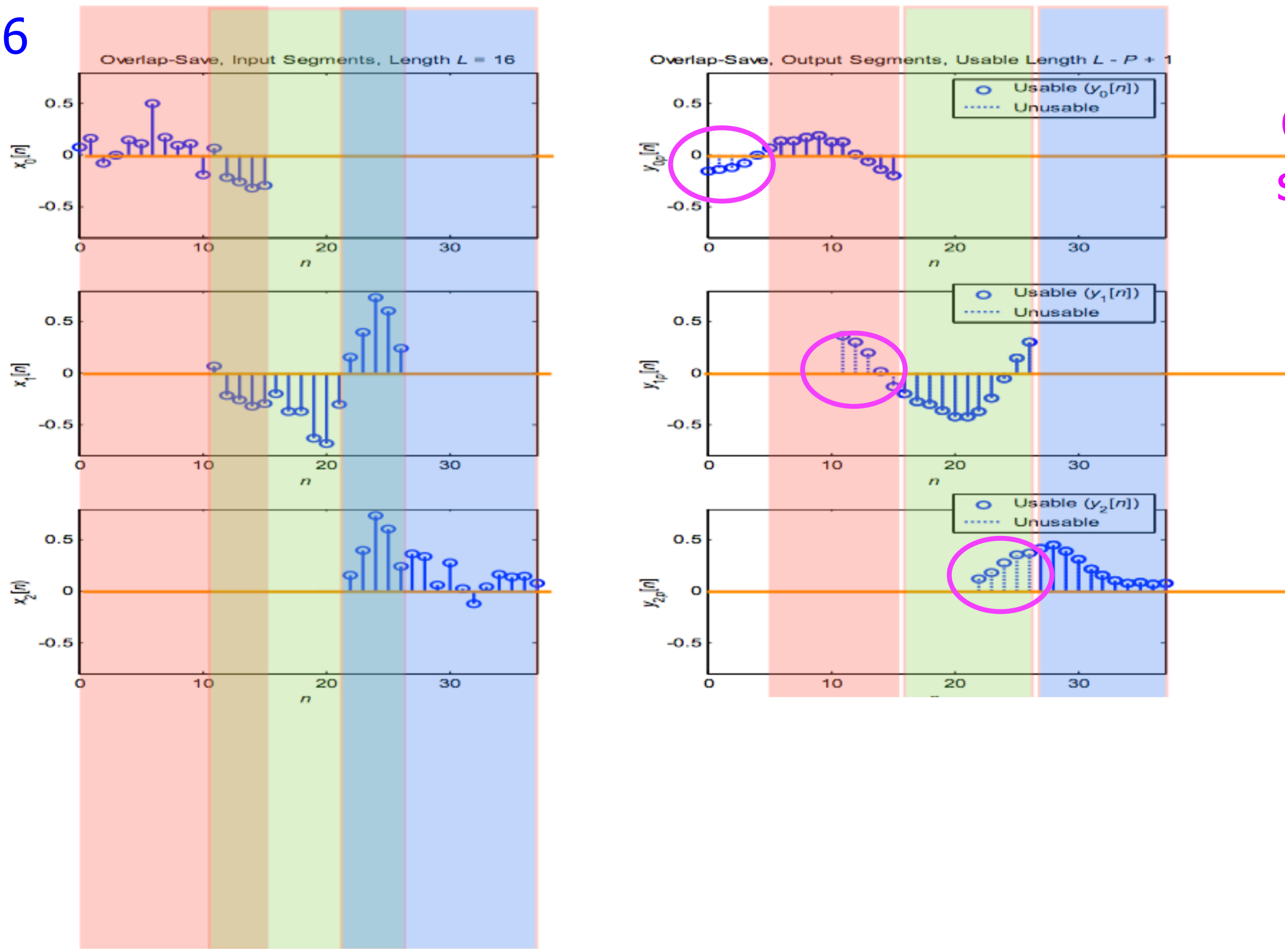
- An L-point sequence circularly convolved with a P-point sequence
  - with $L - P$ zeros padded, $P < L$
- gives an L-point result with
  - the first $P - 1$ values *incorrect* and
  - the next $L - P + 1$ the *correct* linear convolution result





Example of Overlap-Save

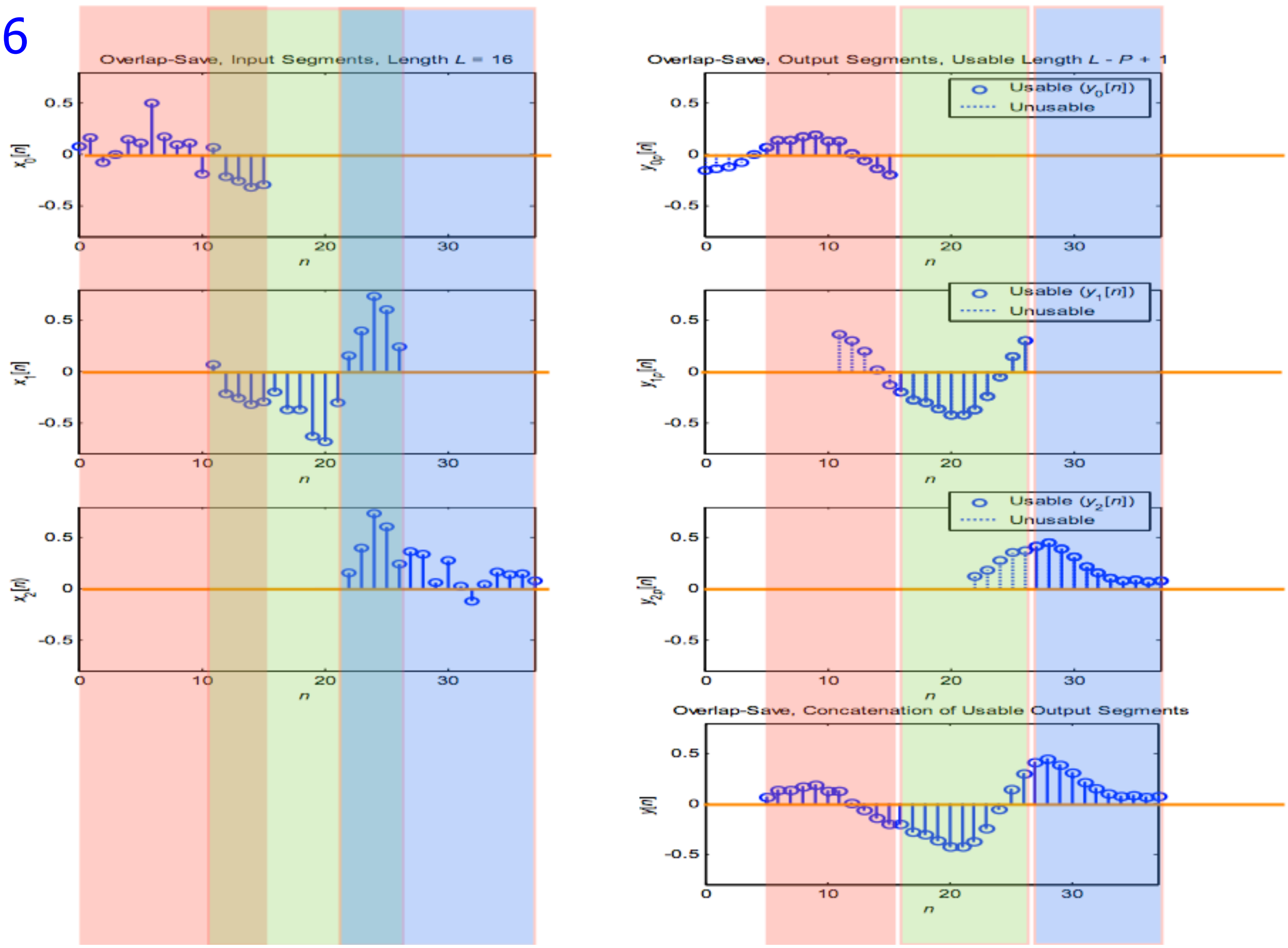
L+P-1=16



P-1=5
Overlap
samples

Example of Overlap-Save

L+P-1=16

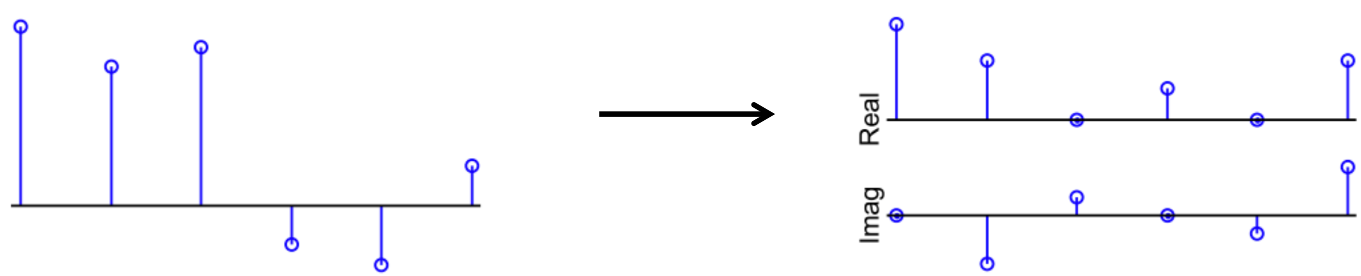


Discrete Cosine Transform

- Similar to the discrete Fourier transform (DFT), but using only real numbers
- Widely used in lossy compression applications (eg. Mp3, JPEG)
- Why use it?

DFT Problems

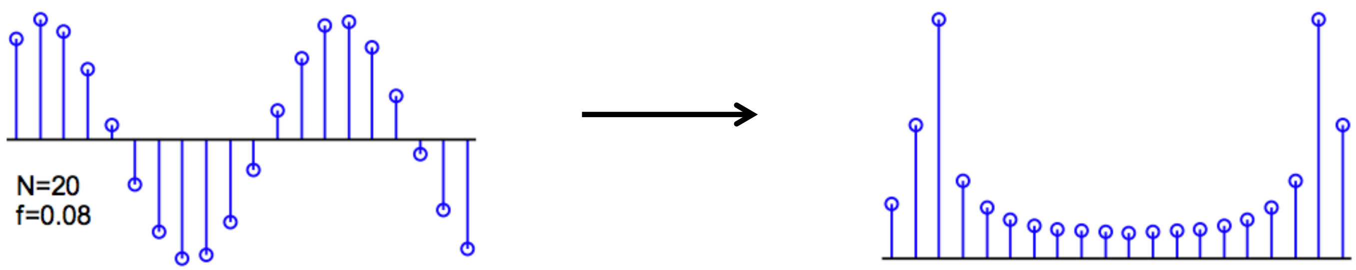
- For processing 1-D or 2-D signals (especially coding), a common method is to divide the signal into "frames" and then apply an invertible transform to each frame that compresses the information into few coefficients.
- The DFT has some problems when used for this purpose:
  - N real x[n] $\leftrightarrow$ N complex X[k] : 2 real, N/2 − 1 conjugate pairs
  - DFT is of the periodic signal formed by replicating x[n]



DFT Problems

- For processing 1-D or 2-D signals (especially coding), a common method is to divide the signal into “frames” and then apply an invertible transform to each frame that compresses the information into few coefficients.
- The DFT has some problems when used for this purpose:
  - N real x[n] ↔ N complex X[k] : 2 real, N/2 − 1 conjugate pairs
  - DFT is of the periodic signal formed by replicating x[n]

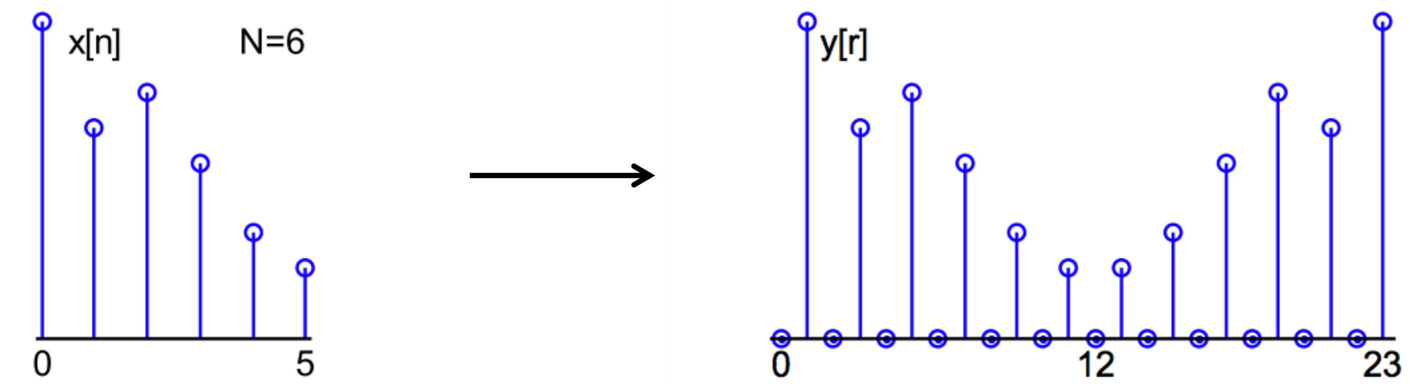
    ⇒ Spurious frequency components from boundary discontinuity



The Discrete Cosine Transform (DCT) overcomes these problems.

Discrete Cosine Transform

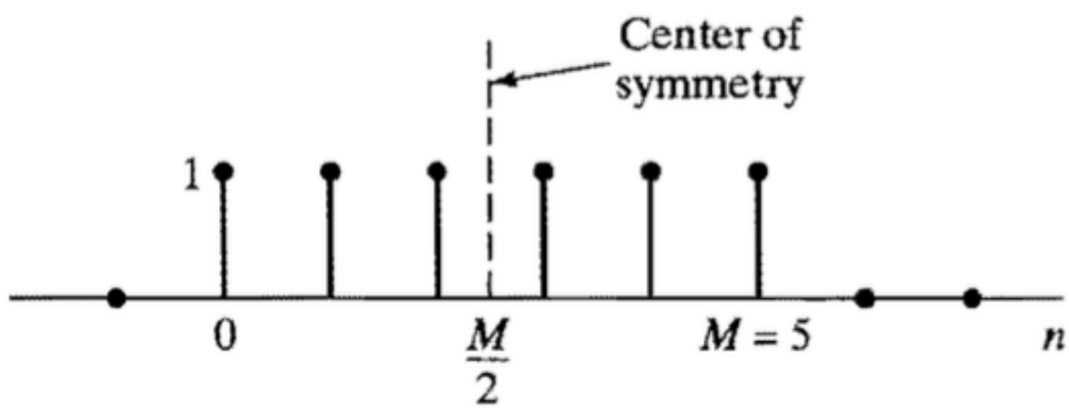
- To form the Discrete Cosine Transform (DCT), replicate $x[0 : N-1]$ but in reverse order and insert a zero between each pair of samples:



FIR GLP: Type II

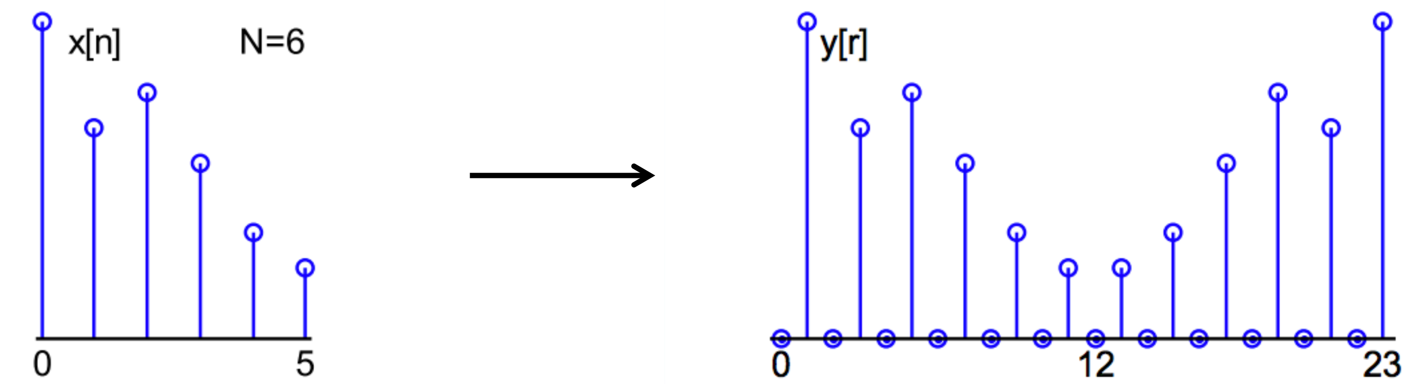
Type II Even Symmetry, M odd

$$h[n] = h[M-n], \quad n = 0,1,\ldots,M$$



Discrete Cosine Transform

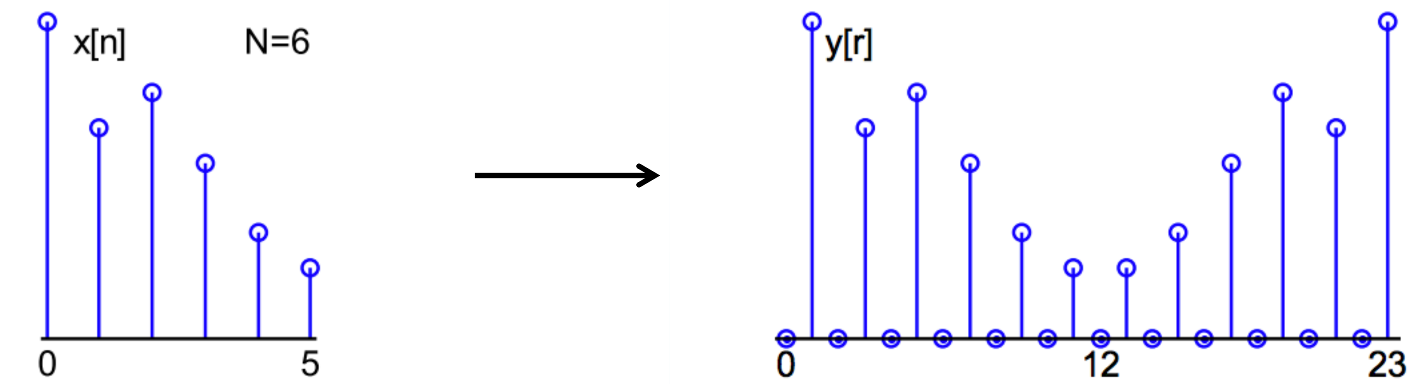
- To form the Discrete Cosine Transform (DCT), replicate $x[0 : N-1]$ but in reverse order and insert a zero between each pair of samples:



- Take the DFT of length 4N real, symmetric, odd-sample-only sequence

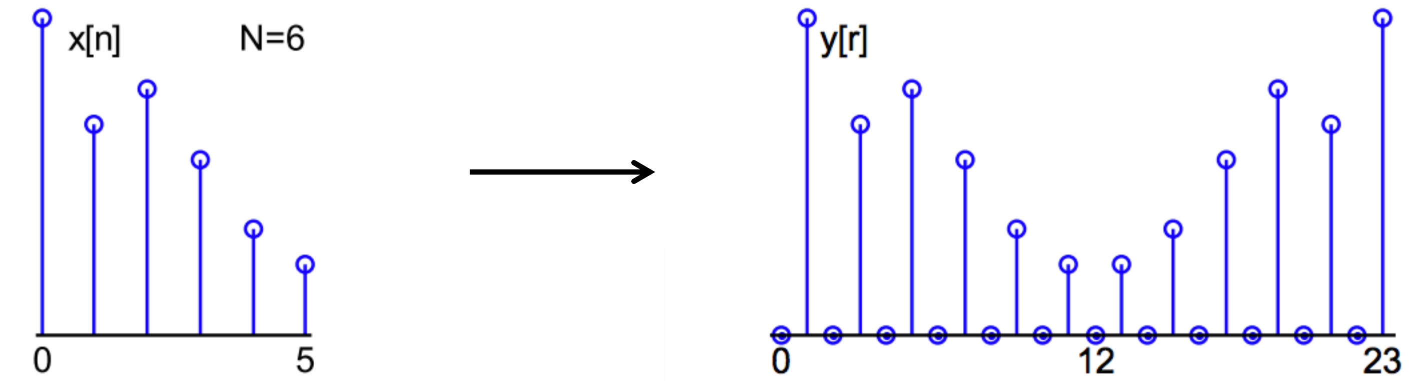
Discrete Cosine Transform

- To form the Discrete Cosine Transform (DCT), replicate $x[0:N-1]$ but in reverse order and insert a zero between each pair of samples:

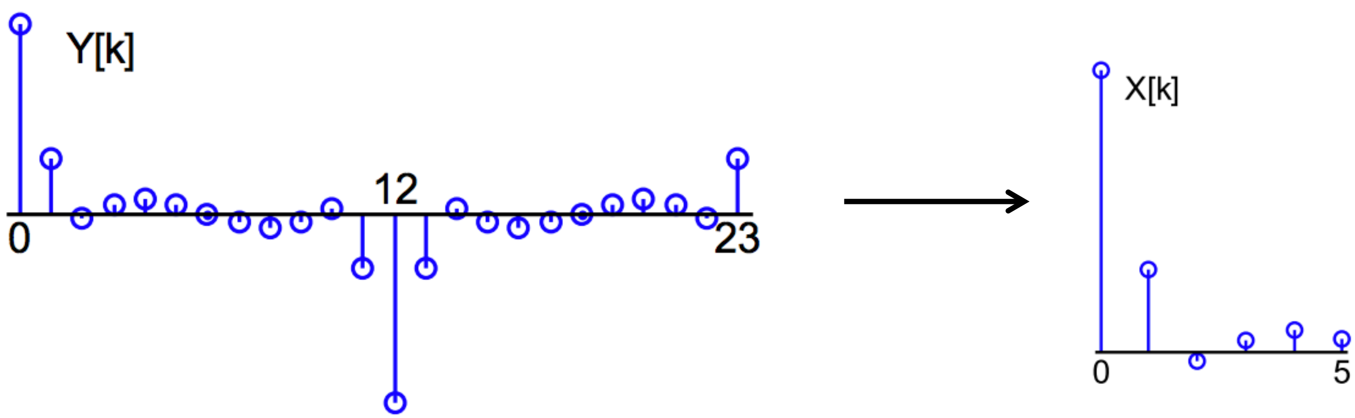


- Take the DFT of length 4N real, symmetric, odd-sample-only sequence
- Result is real, symmetric and anti-periodic: only need first N values

Discrete Cosine Transform



- Result is real, symmetric and anti-periodic: only need first N values



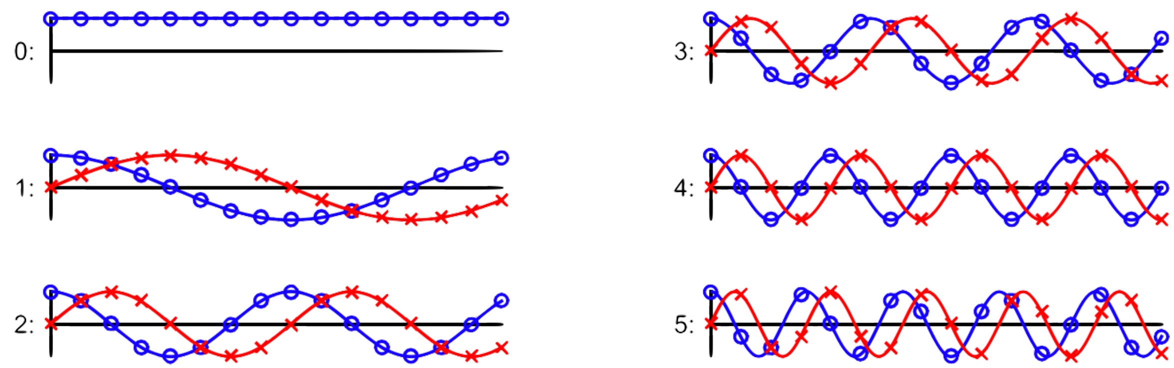
Discrete Cosine Transform

Forward DCT: $X_C[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$ for $k = 0 : N-1$

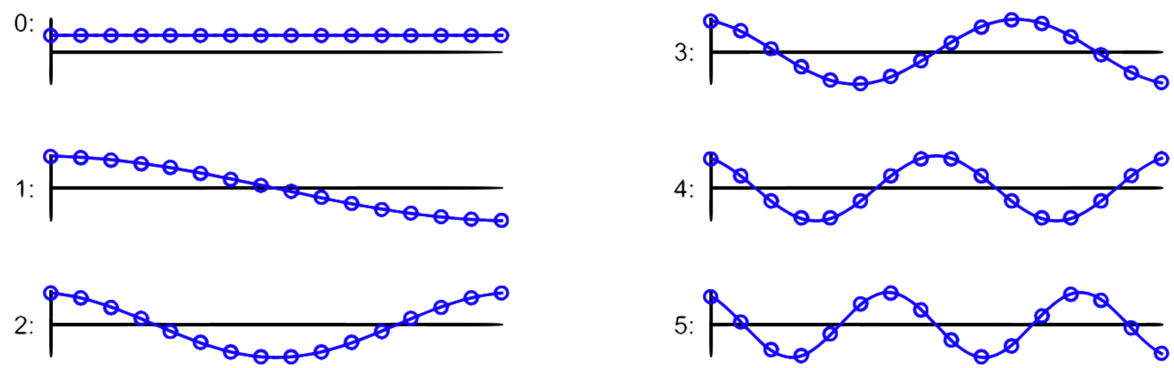
Inverse DCT: $x[n] = \frac{1}{N} X[0] + \frac{2}{N} \sum_{k=1}^{N-1} X[k] \cos \frac{2\pi(2n+1)k}{4N}$

Basis Functions

DFT basis functions: $x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{j2\pi\frac{kn}{N}}$



DCT basis functions: $x[n] = \frac{1}{N}X[0] + \frac{2}{N}\sum_{k=1}^{N-1} X[k]\cos\frac{2\pi(2n+1)k}{4N}$

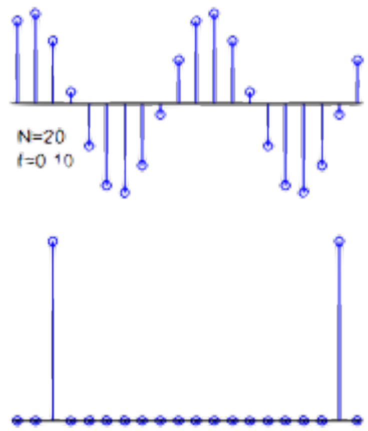


DFT of Sine Wave

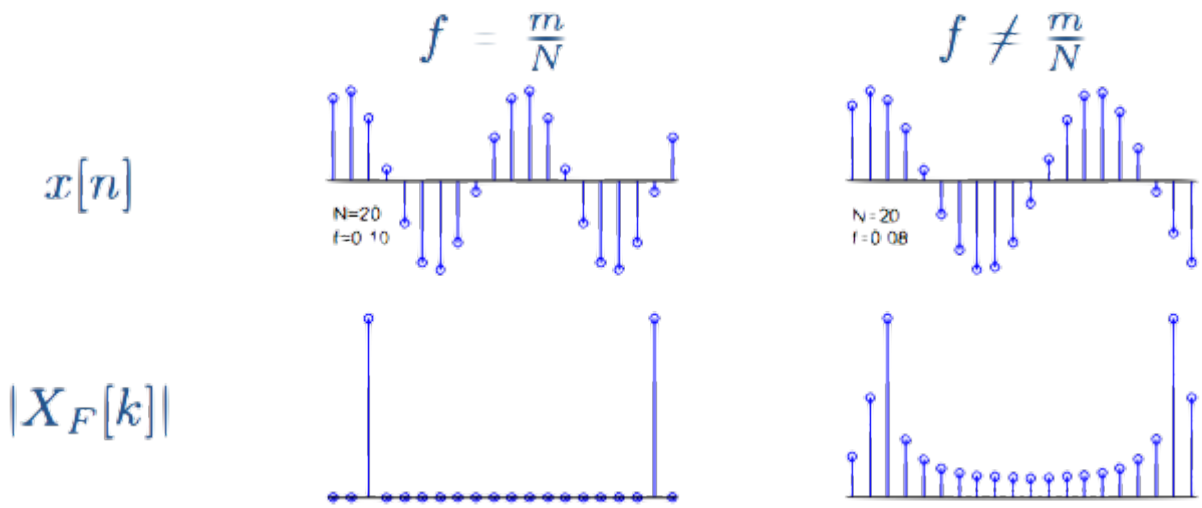
$$f = \frac{m}{N}$$

$x[n]$

$|X_F[k]|$



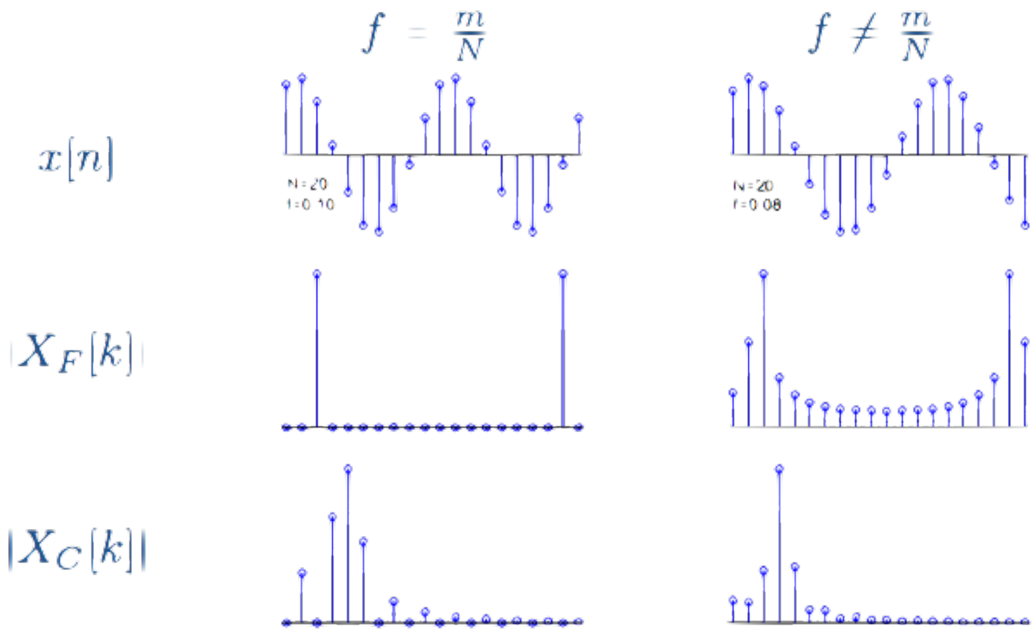
DFT of Sine Wave



DFT: Real→Complex; Freq range $[0, 1]$; Poorly localized unless $f = \frac{m}{N}$; $|X_F[k]| \propto k^{-1}$ for $Nf < k \ll \frac{N}{2}$

DCT of Sine Wave

DCT: $X_C[k] = \sum_{n=0}^{N-1} x[n] \cos \frac{2\pi(2n+1)k}{4N}$



DFT: Real→Complex; Freq range $[0, 1]$; Poorly localized unless $f = \frac{m}{N}$; $|X_F[k]| \propto k^{-1}$ for $Nf < k \ll \frac{N}{2}$

DCT: Real→Real; Freq range $[0, 0.5]$; Well localized $\forall f$; $|X_C[k]| \propto k^{-2}$ for $2Nf < k < N$

Big Ideas

- Discrete Fourier Transform (DFT)
  - For finite signals assumed to be zero outside of defined length
  - N-point DFT is sampled DTFT at N points
  - DFT properties inherited from DFS, but circular operations!
- Fast Convolution Methods
  - Use circular convolution (i.e DFT) to perform fast linear convolution
    - Overlap-Add, Overlap-Save
- DCT useful for frame rate compression of large signals

Admin

- HW 8 due tomorrow
- Project
  - Work in groups of up to 2
    - Can work alone if you want
    - Use Piazza to find partners