Lecture 12: Circular Convolution

Mark Hasegawa-Johnson

ECE 401: Signal and Image Analysis

Outline

Review: DFT

The DFT (discrete Fourier transform) of any signal is $X[k]$, given by

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi kn}{N}}$$

$$x[n] = \frac{1}{N} \sum_{0}^{N-1} X[k] e^{j\frac{2\pi kn}{N}}$$

Particular useful examples include:

$$f[n] = \delta[n] \leftrightarrow F[k] = 1$$

$$g[n] = \delta\left[((n - n_0))_N\right] \leftrightarrow G[k] = e^{-j\frac{2\pi kn_0}{N}}$$

Properties of the DTFT

Properties worth knowing include:

0. Periodicity: $X[k+N] = X[k]$
1. Conjugate symmetric: $x[n]$ Real $\leftrightarrow X[k] = X^*[-k]$
2. Linearity:

$$z[n] = ax[n] + by[n] \leftrightarrow Z[k] = aX[k] + bY[k]$$

3. Time Shift: $x[n-n_0] \leftrightarrow e^{-j\frac{2\pi k n_0}{N}} X[k]$, paying attention to the fact that $x[n]$ is periodic in time
4. Frequency Shift: $e^{j\frac{2\pi k_0 n}{N}} x[n] \leftrightarrow X[k-k_0]$, paying attention to the fact that $X[k]$ is periodic in frequency

Convolution

Convolution (finite impulse response filtering) is a generalization of weighted local averaging:

$$y[n] = h[n] * x[n] \equiv \sum_m x[m]h[n-m] = \sum_m x[n-m]h[m]$$

- If all samples of $h[n]$ are positive, then it's a weighted local averaging filter
- If the samples of $h[n]$ are positive for $n > 0$ and negative for $n < 0$ (or vice versa), then it's a weighted local differencing filter

Frequency Response

- **Tones in $\rightarrow$ Tones out**

$$x[n] = e^{j\omega n} \rightarrow y[n] = H(\omega)e^{j\omega n}$$
$$x[n] = \cos(\omega n) \rightarrow y[n] = |H(\omega)|\cos(\omega n + \angle H(\omega))$$
$$x[n] = A\cos(\omega n + \theta) \rightarrow y[n] = A|H(\omega)|\cos(\omega n + \theta + \angle H(\omega))$$

- where the **Frequency Response** is given by

$$H(\omega) = \sum_m h[m]e^{-j\omega m}$$

Putting it all together

If the input signal is

$$x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] e^{j \frac{2\pi kn}{N}}$$

. . . then the output of $y[n] = h[n] * x[n]$ is. . .

$$y[n] = \frac{1}{N} \sum_{n=0}^{N-1} H[k] X[k] e^{j \frac{2\pi kn}{N}}$$

$$H[k] \equiv H\left(\frac{2\pi k}{N}\right)$$

Outline

Convolution is an $\mathcal{O}\left\{N^2\right\}$ Operation

Suppose we convolve an N-sample $h[n]$ with an N-sample $x[n]$. Consider carefully where the nonzero terms are:

$$y[n] = \sum_{m=n}^{N-1} h[m]x[n-m], \quad 0 \leq n \leq 2N-1$$

Let's do a computational complexity analysis:

- There are $2N = \mathcal{O}\{N\}$ output samples.
- The m^{th} output sample results from a sum of length $N - m = \mathcal{O}\{N\}$
- The total complexity is therefore $\mathcal{O}\left\{N^2\right\}$

Fast Fourier Transform (FFT)

- The regular DFT formula, $X[k] = \sum x[n] e^{-j\frac{2\pi kn}{N}}$, is also $\mathcal{O}\left\{N^2\right\}$ (N terms in the sum, N output coefficients).
- In 1965, Cooley and Tukey found a way to re-arrange the terms of the DFT so it can be computed in $\mathcal{O}\left\{N \log_2 N\right\}$ multiply-add operations. Their algorithm is called the "fast Fourier transform" (FFT).
- For example, suppose $N = 1024$, then
  - $N^2 = 1,048,576$ operations
  - $N \log_2 N = 10,240$ operations (a factor of 100 savings)

Can we use the FFT to speed up convolution?

Remember this formula:

$$y[n] = \frac{1}{N} \sum_{n=0}^{N-1} H[k]X[k]e^{j\frac{2\pi kn}{N}}$$

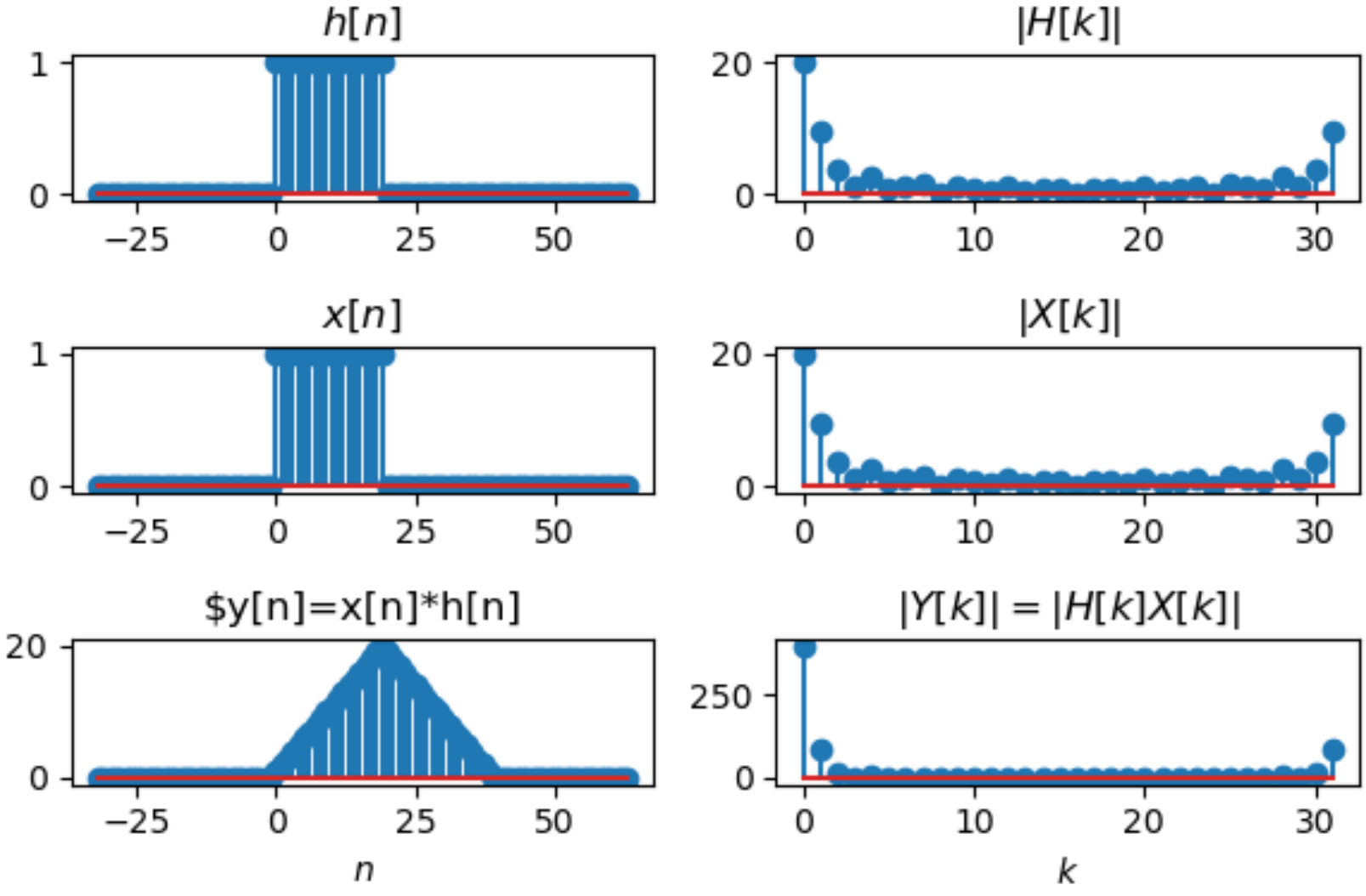
How about the following?

1. Take the FFT of $x[n]$ to find $X[k]$ $(\mathcal{O}\{N \log N\})$
2. Take the FFT of $h[n]$ to find $H[k]$ $(\mathcal{O}\{N \log N\})$
3. Multiply them to find $Y[k]$ $(\mathcal{O}\{N\})$
4. Take the inverse FFT to find $y[n]$ $(\mathcal{O}\{N \log N\})$

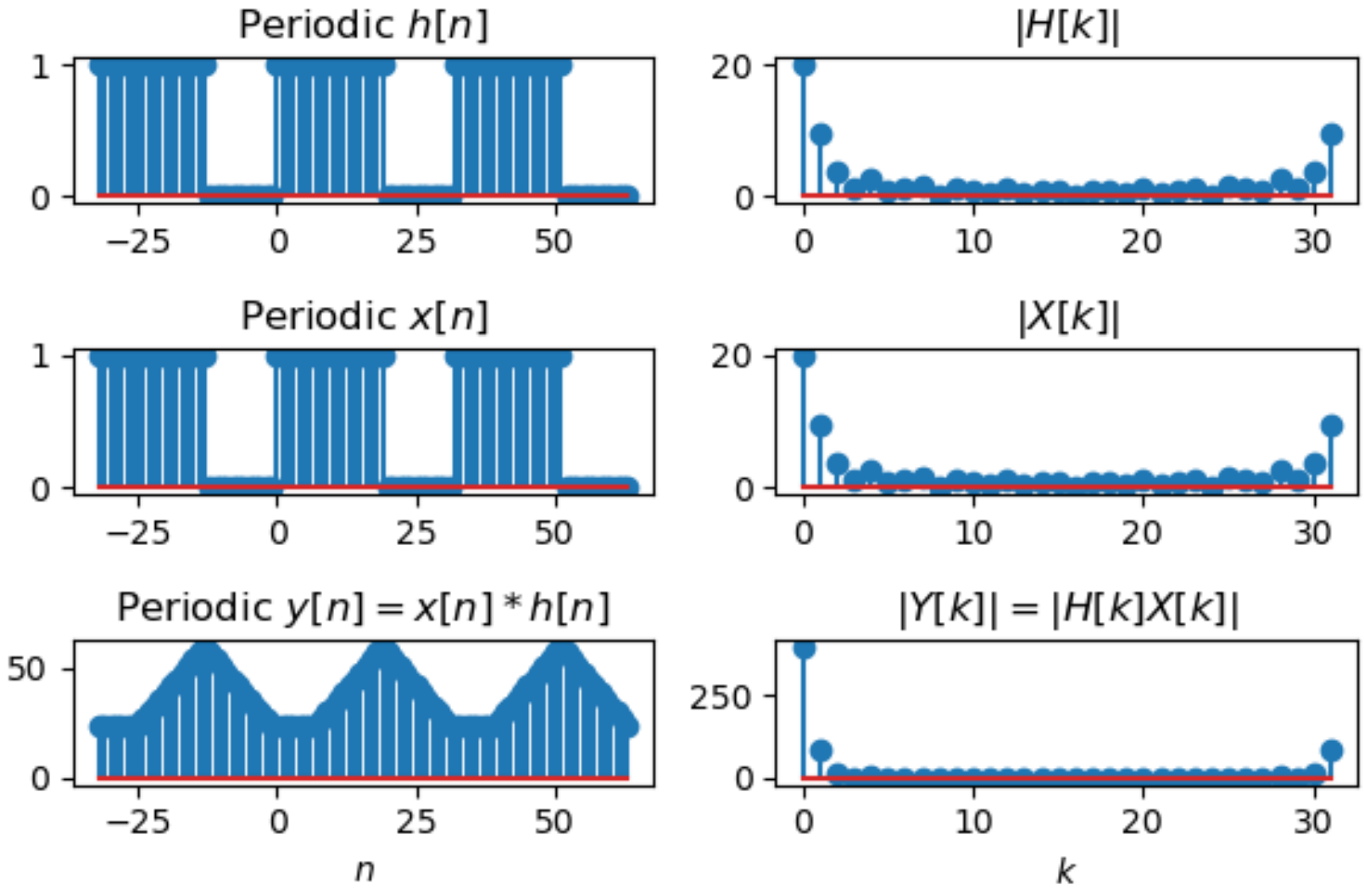
Is the total complexity $\mathcal{O}\{N \log N\}$, much faster than regular convolution? YES!! Does it compute the desired answer? NO!!! Why not, and how can we fix it?

Outline

Multiplying two DFTs: what we think we're doing



Multiplying two DFTs: what we're actually doing



Circular convolution

Suppose $Y[k] = H[k]X[k]$, then

$$\begin{aligned} y[n] &= \frac{1}{N}\sum_{k=0}^{N-1} H[k]X[k]e^{j\frac{2\pi kn}{N}} \\ &= \frac{1}{N}\sum_{k=0}^{N-1} H[k]\left(\sum_{m=0}^{N-1} x[m]e^{-j\frac{2\pi km}{N}}\right)e^{j\frac{2\pi kn}{N}} \\ &= \sum_{m=0}^{N-1} x[m]\left(\frac{1}{N}\sum_{k=0}^{N-1} H[k]e^{-j\frac{2\pi k(n-m)}{N}}\right) \\ &= \sum_{m=0}^{N-1} x[m]h\left[((n-m))_N\right] \end{aligned}$$

. . . where the notation $((n-m))_N$ means "modulo N," i.e., we need to pay attention to the periodicity.

Circular convolution

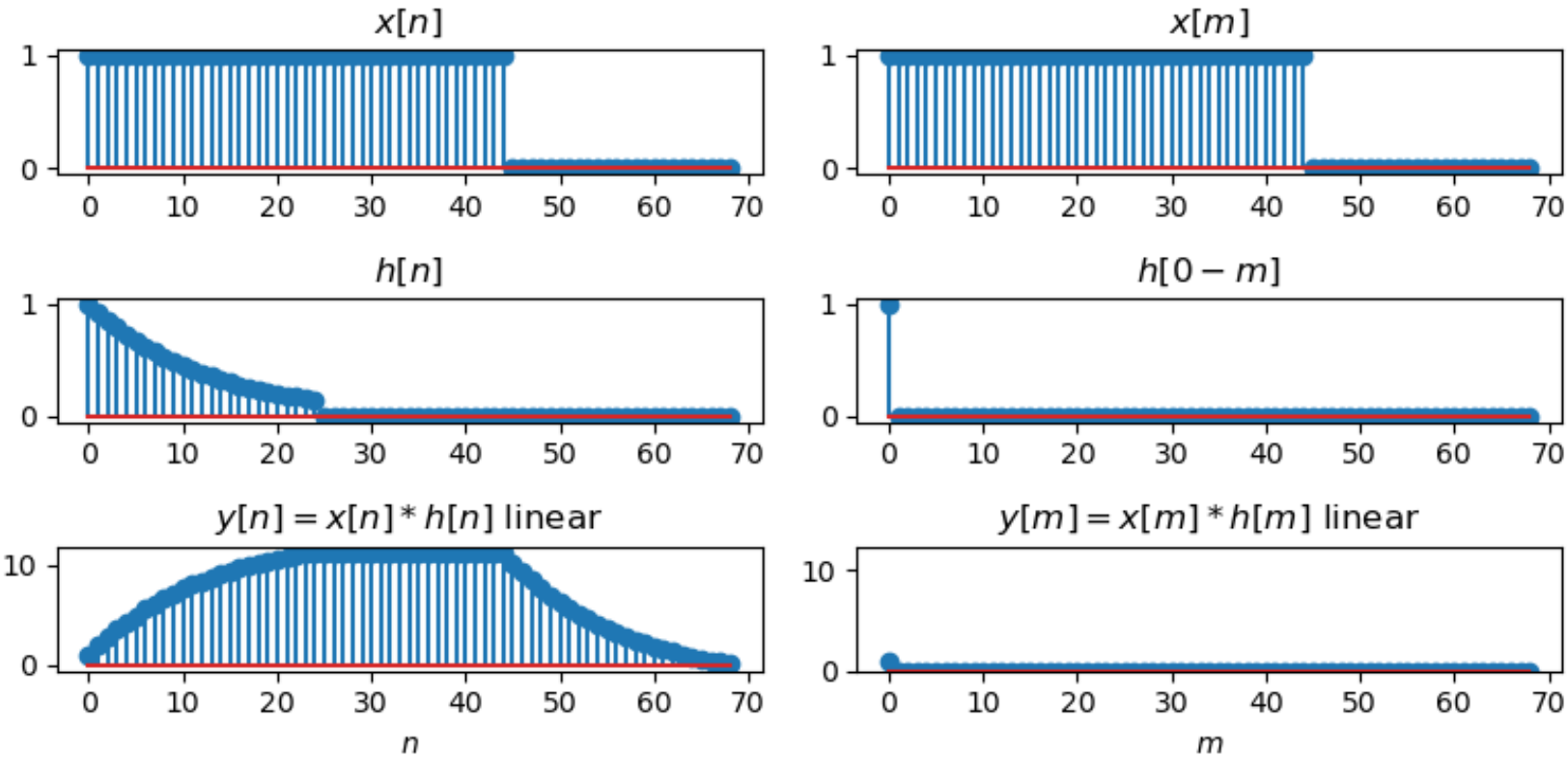
Multiplying the DFT means **circular convolution** of the time-domain signals:

$$y[n] = h[n] \circledast x[n] \leftrightarrow Y[k] = H[k]X[k],$$

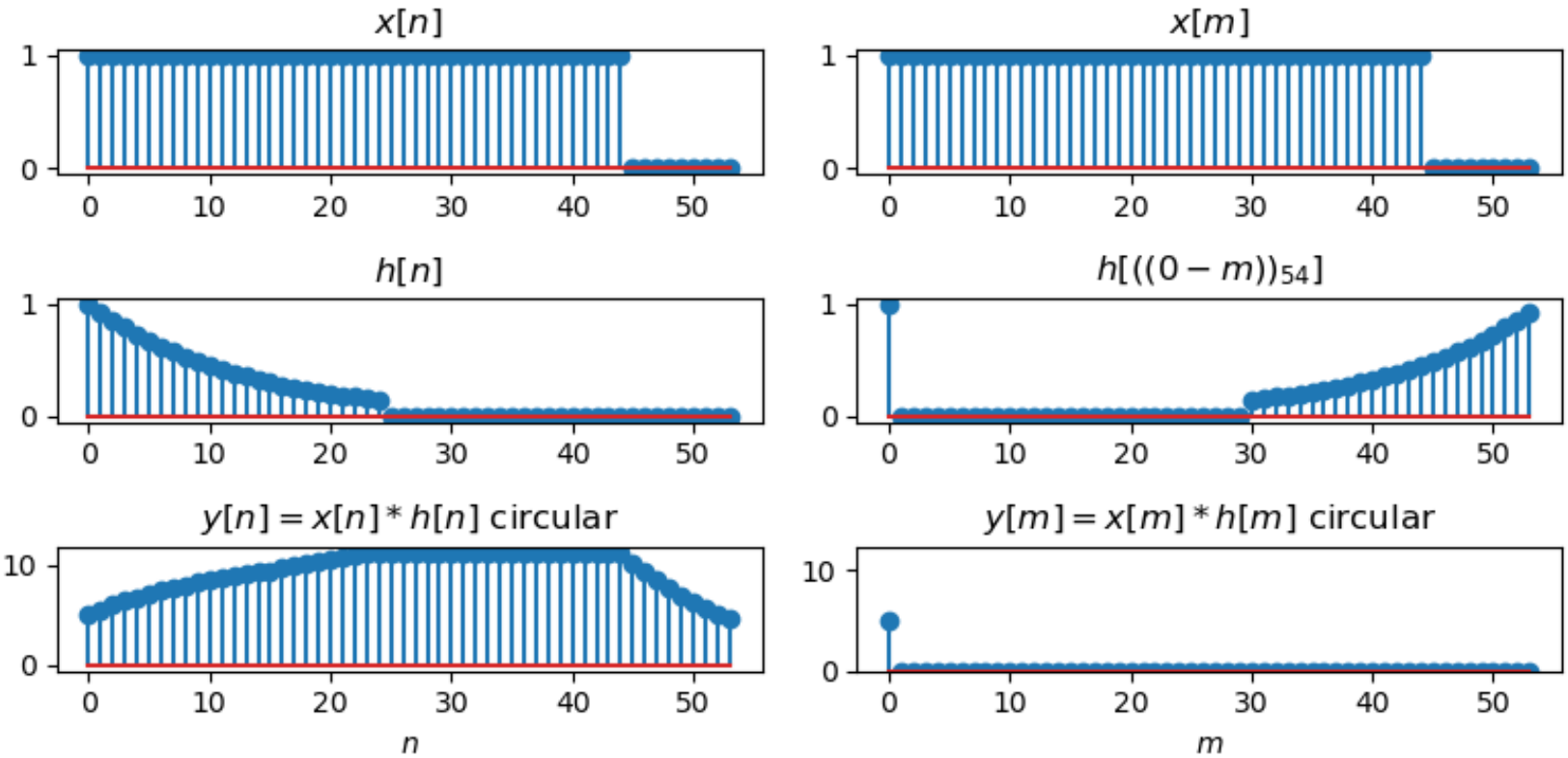
Circular convolution ($h[n] \circledast x[n]$) is defined like this:

$$h[n] \circledast x[n] = \sum_{m=0}^{N-1} x[m]h\left[((n-m))_N\right] = \sum_{m=0}^{N-1} h[m]x\left[((n-m))_N\right]$$

Linear convolution example



Circular convolution example



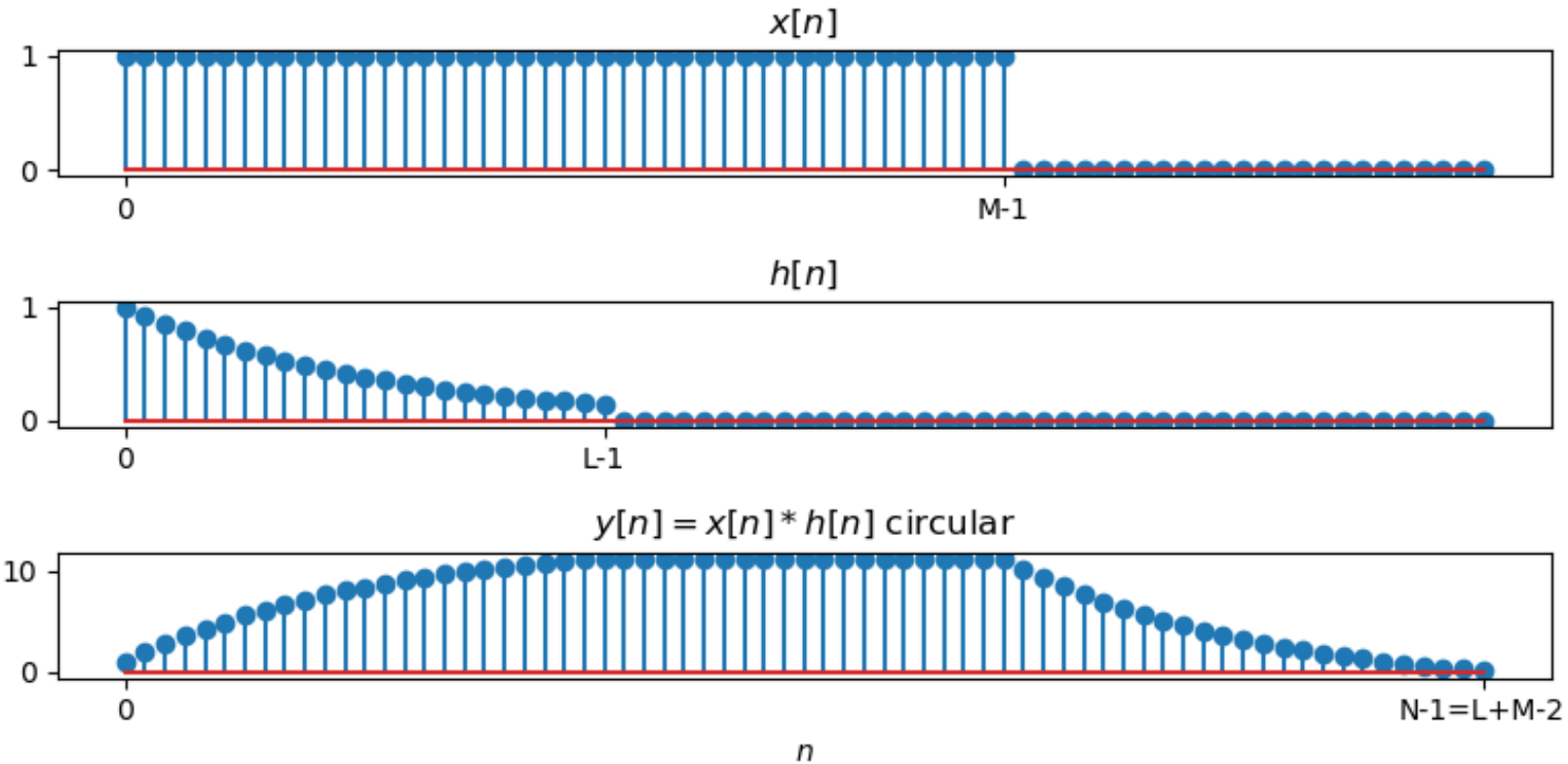
Quiz!

Go to the course webpage, try the quiz!

Outline

How long is $h[n] * x[n]$?

If $x[n]$ is M samples long, and $h[n]$ is L samples long, then their linear convolution, $y[n] = x[n] * h[n]$, is $M + L - 1$ samples long.



Zero-padding turns circular convolution into linear convolution

How it works:

- $h[n]$ is length-L
- $x[n]$ is length-M
- As long as they are both zero-padded to length $N \geq L + M - 1$, then
- $y[n] = h[n] \circledast x[n]$ is the same as $h[n] * x[n]$.

Zero-padding turns circular convolution into linear convolution

Why it works: Either...

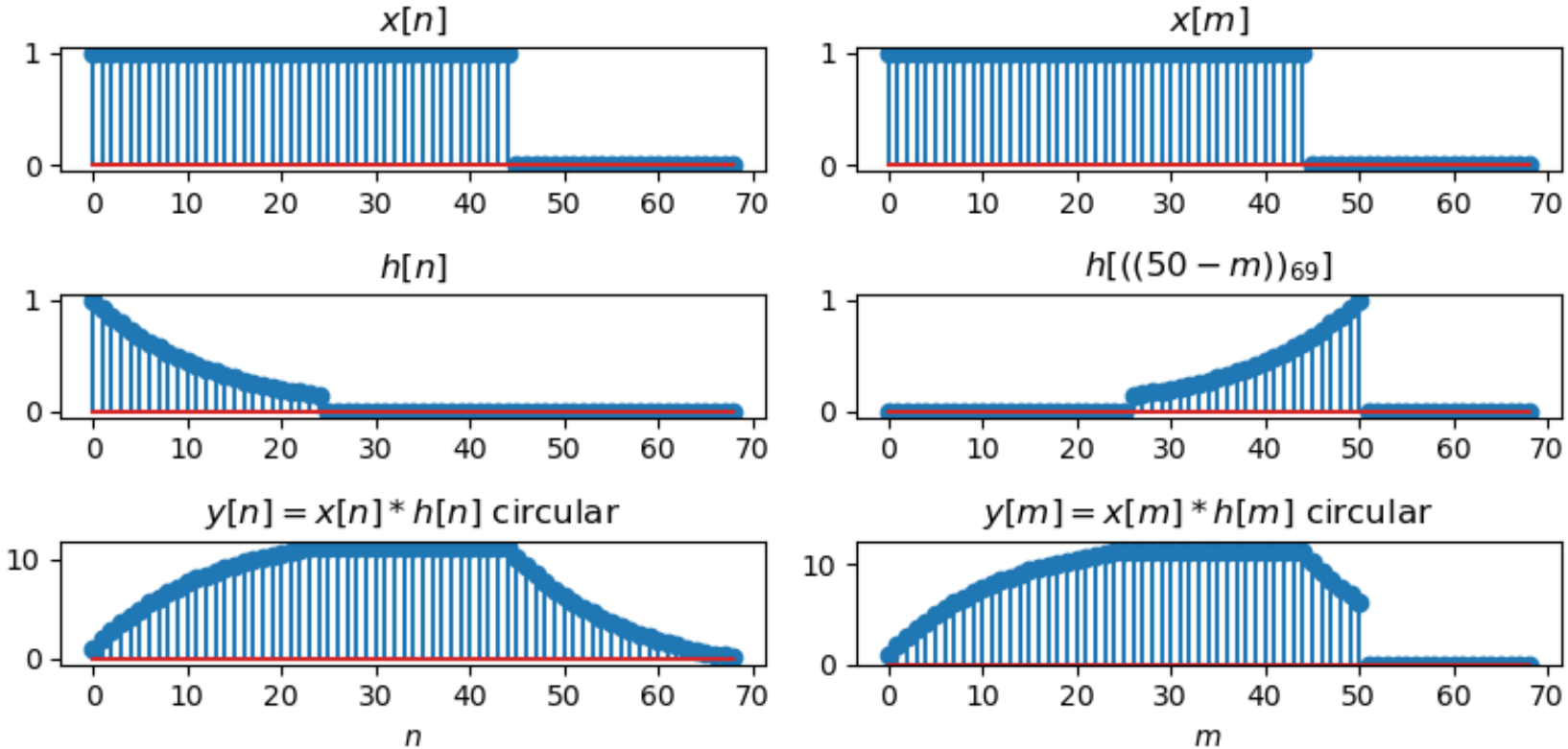
- $n - m$ is a positive number, between 0 and $N - 1$. Then $((n - m))_N = n - m$, and therefore

$$x[m]h\left[((n - m))_N\right] = x[m]h[n - m]$$

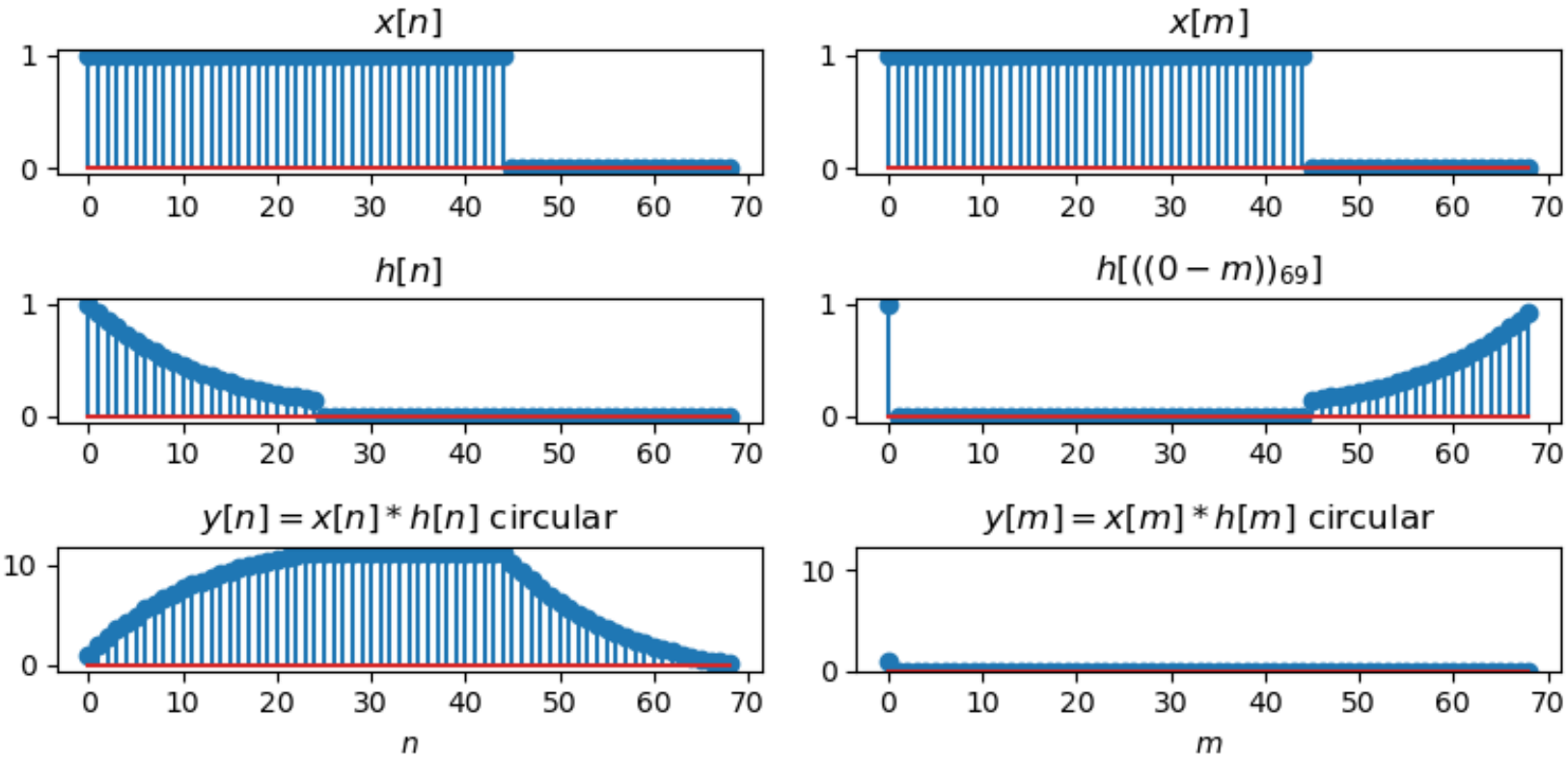
- $n - m$ is a negative number, between 0 and $-(L - 1)$. Then $((n - m))_N = N + n - m \geq N - (L - 1) > M - 1$, so

$$x[m]h\left[((n - m))_N\right] = 0$$

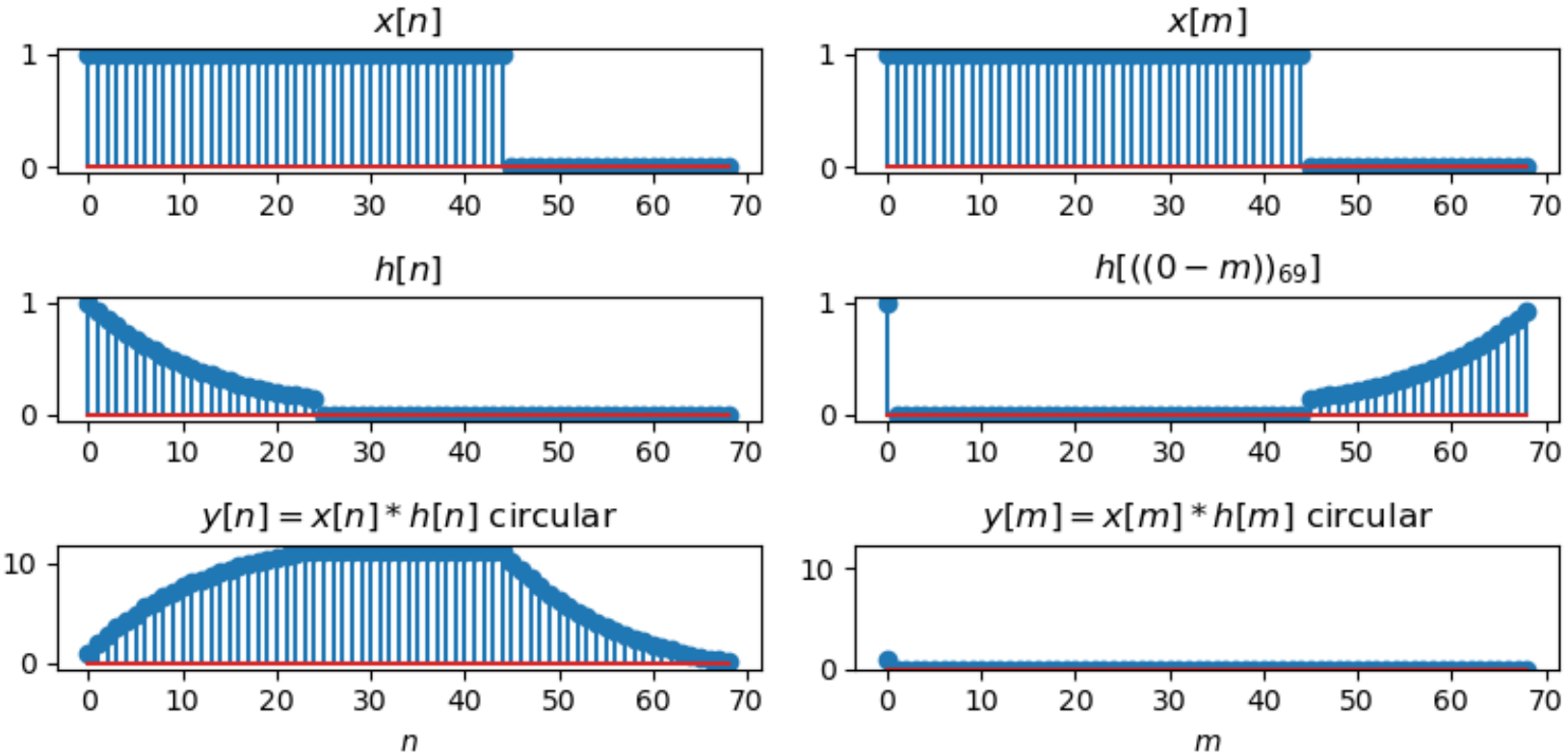
Case #1: $n - m$ is positive, so circular convolution is the same as linear convolution



Case #2: $n - m$ is negative, so it wraps around, but N is long enough so that the wrapped part of $h[((n-m))_N]$ doesn't overlap with $x[m]$



Zero-padding turns circular convolution into linear convolution



Outline

Circular convolution

Multiplying the DFTs, instead of implementing convolution directly, can save 100-fold computation or better. However, multiplying the DFT means **circular convolution** of the time-domain signals:

$$y[n] = h[n] \circledast x[n] \leftrightarrow Y[k] = H[k]X[k],$$

Circular convolution ($h[n] \circledast x[n]$) is defined like this:

$$h[n] \circledast x[n] = \sum_{m=0}^{N-1} x[m]h\left[((n-m))_N\right] = \sum_{m=0}^{N-1} h[m]x\left[((n-m))_N\right]$$

Circular convolution is the same as linear convolution if and only if $N \geq L + M - 1$.