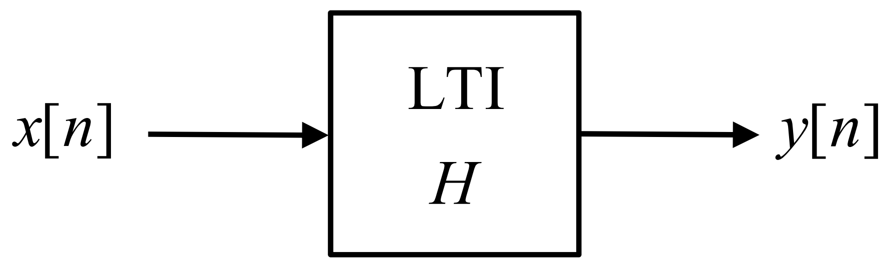
REVIEW OF 1D LTI SYSTEMS



- Operator notation: $y[n] = H\{x[n]\}$

- In English, this is read: "$y[n]$ is the output of the system H when $x[n]$ is the input."

THE MOST IMPORTANT PROPERTIES

- **Impulse response:** when the input is $\delta[n]$, the output is $h[n]$.

$$h[n] = H\left\{\delta[n]\right\}$$

- **Homogeneity:** if $y[n] = H\{x[n]\}$ and c is a constant, then

$$H\left\{cx[n]\right\} = cH\left\{x[n]\right\} = cy[n].$$

  - In other words, the action of the system **commutes** with multiplication by constants.

- **Superposition:** if $y_1[n] = H\{x_1[n]\}$ and $y_2[n] = H\{x_2[n]\}$, then

$$H\left\{x_1[n] + x_2[n]\right\} = H\left\{x_1[n]\right\} + H\left\{x_2[n]\right\} = y_1[n] + y_2[n].$$

  - In other words, the action of the system **commutes** with sums.

IMPORTANT LTI PROPERTIES

- Together, homogeneity and superposition are called **linearity**.
- Together, they imply that the action of a **linear** system commutes with linear combinations:

$$H\{c_1x_1[n]+c_2x_2[n]\}=c_1y_1[n]+c_2y_2[n].$$

- **Translation Invariance:** if $y[n]=H\{x[n]\}$ and n_0 is an integer constant, then

$$H\{x[n-n_0]\}=y[n-n_0].$$

  - In other words, the action of the system **commutes** with (time) shifts.
  - Also called **time invariance** or **shift invariance**.

1D Linear Convolution

- As we have seen, any 1D discrete-time signal $x[n]$ can be written as a linear combination of the translates of $\delta[n]$:

$$x[n] = \cdots + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + \cdots$$

- Here, it is **important** to realize that $x[-1]$, $x[0]$, $x[1]$, etc., are constants; they are **numbers**.
- So, if $x[n]$ is the input to an LTI system H, the output is

$$\begin{aligned}
y[n] &= H\{x[n]\} \\
&= H\{\cdots + x[-1]\delta[n+1] + x[0]\delta[n] + x[1]\delta[n-1] + \cdots\} \\
&= \cdots + x[-1]H\{\delta[n+1]\} + x[0]H\{\delta[n]\} + x[1]H\{\delta[n-1]\} + \cdots \\
&= \cdots + x[-1]h[n+1] + x[0]h[n] + x[1]h[n-1] + \cdots
\end{aligned}$$

Σ Notation

- To save time and paper, we can write this **exact same thing** using "capital Sigma do-loops":

$$x[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n-k]$$

$$\begin{aligned} y[n] &= H\{x[n]\} \\ &= H\left\{ \sum_{k=-\infty}^{\infty} x[k]\delta[n-k] \right\} \\ &= \sum_{k=-\infty}^{\infty} x[k] H\{\delta[n-k]\} \\ &= \sum_{k=-\infty}^{\infty} x[k] h[n-k] \end{aligned}$$

- This is called **linear convolution**; written $y[n] = x[n] * h[n]$.

Interpretation

- For each n, the output signal $y[n]$ is a number.
- This number is given by the dot product of the input $x[n]$ with a **flipped-and-shifted version** of the impulse response:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] = \left\langle x[k], h[-k-(-n)] \right\rangle$$

- Another way to think of it:
  - Let the input be $x[n] = 2\delta[n] + 3\delta[n\text{-}1] + 4\delta[n\text{-}2]$.
  - We can think of this as a sum of three input signals.
  - For $2\delta[n]$, the output is $2h[n]$.
  - For $3\delta[n\text{-}1]$, the output is $3h[n\text{-}1]$.
  - For $4\delta[n\text{-}2]$, the output is $4h[n\text{-}2]$.
  - The total output is the sum of these: that's **convolution**.

More About 1D Linear Convolution

- Continuous-time version:

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\theta)h(t-\theta)d\theta.$$

- Computing 1D linear convolution in the transform domain:

$$Y\left(e^{j\omega}\right) = X\left(e^{j\omega}\right)H\left(e^{j\omega}\right)$$

$$Y(z) = X(z)H(z)$$

discrete time

$$Y(\Omega) = X(\Omega)H(\Omega)$$

$$Y(s) = X(s)H(s)$$

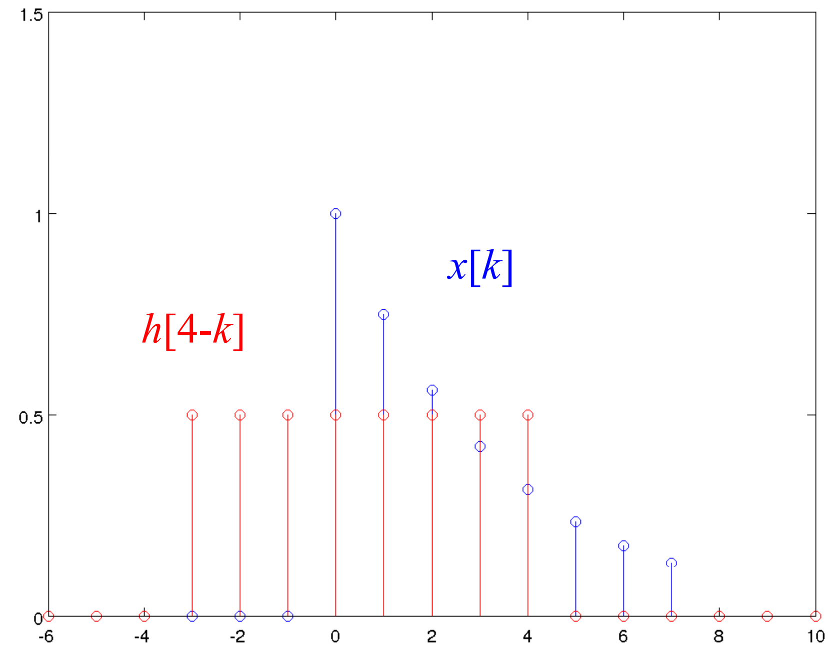
continuous time

An Important Idea

- Suppose $x[n] = (3/4)^n$, $0 \le n \le 7$, and zero otherwise.
- Let $h[n] = 1/2$, $0 \le n \le 7$, and zero otherwise.
- Then, according to the convolution formula,

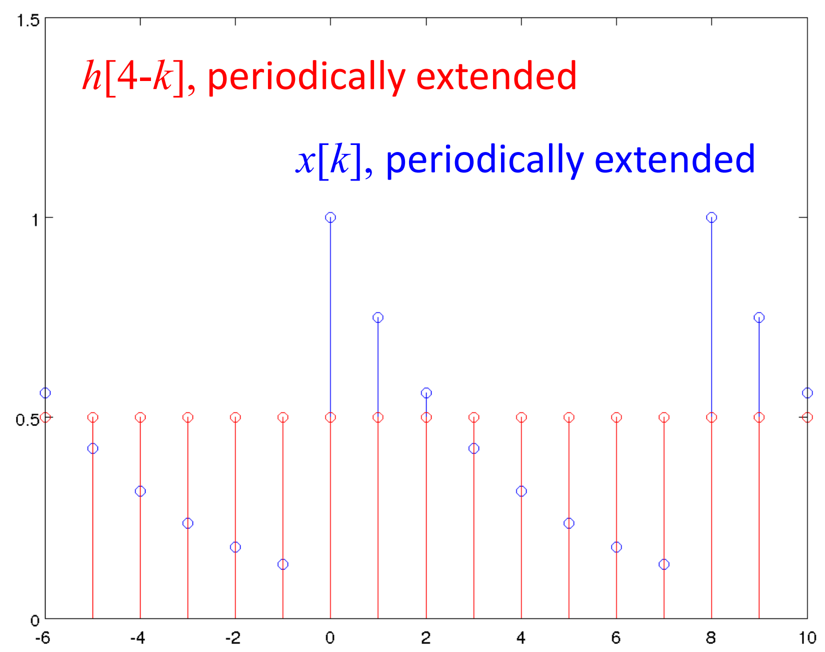
$$y[4] = \sum_{k=-\infty}^{\infty} x[k]h[4-k]$$

- Notice that the product $x[k]h[4\text{-}k]$ is zero for $k < 0$ because $x[k]$ is zero there.
- Similarly, the product is zero for $k > 4$ because $h[4\text{-}k]$ is zero there.
- In the linear convolution sum, we get **zero** for the product in places k where one of the signals "hangs over."



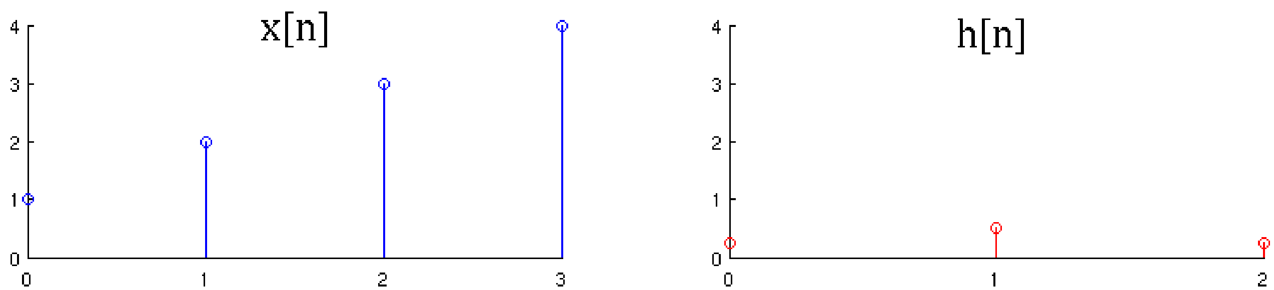
- Now suppose we try to compute this same convolution by multiplying the 8-point DFT's $X[k]$ and $H[k]$.
- Recall that, to the DFT, a finite-length signal is **one period** of a **periodic** signal.
- So the picture will be different this time!

- Because the signals are now **periodically extended**, there will no longer be zeros in the sum at places where one of the signals "hangs over."
- The number we get for $y[4]$ this way will **not** be the same as what we got by linear convolution on the last page.
- It is something **different** – it is called **wraparound convolution**.
- More on this in a minute…

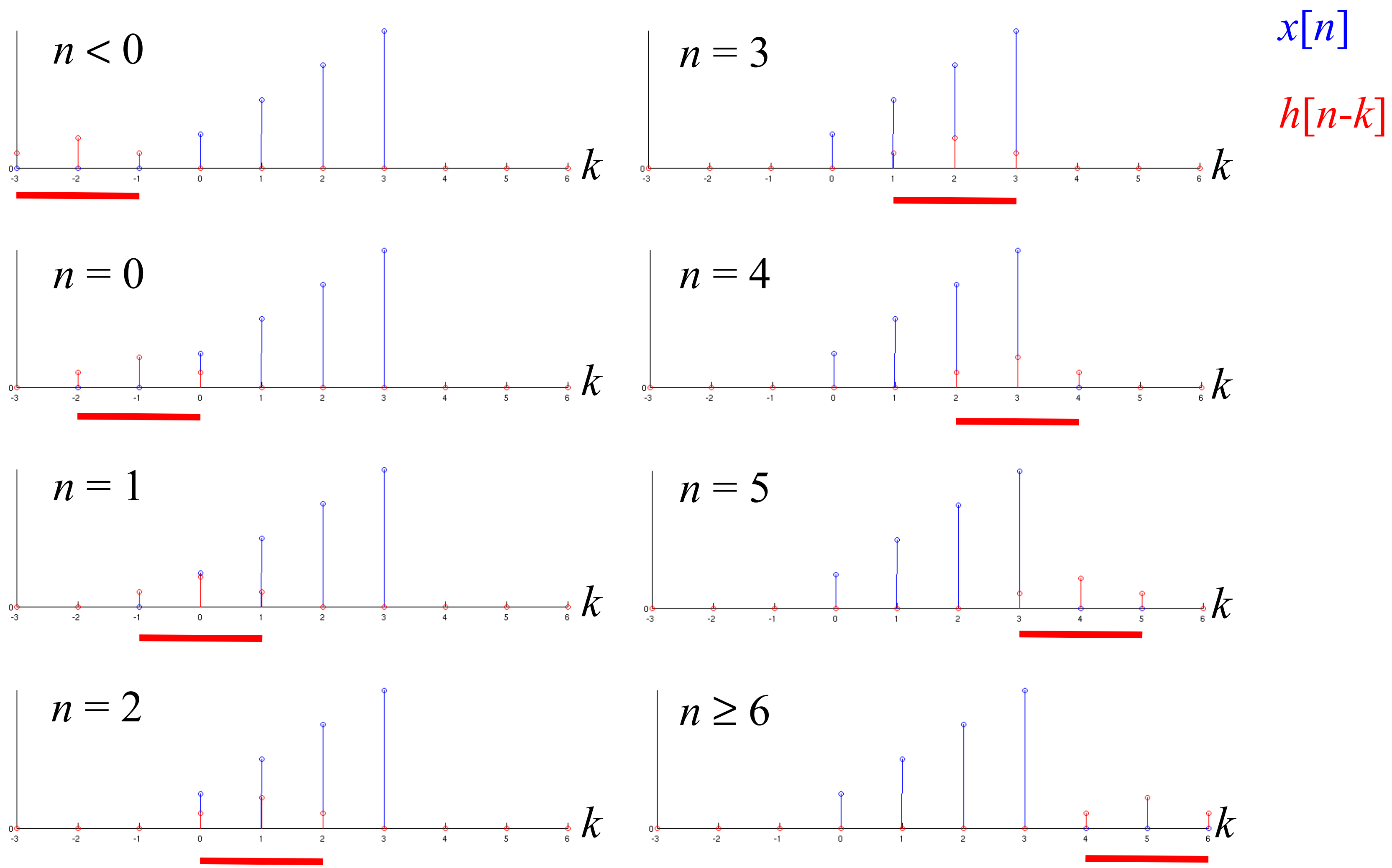


Finite-Length 1D Linear Convolution

- Let H be a 1D LTI causal 3-point weighted average filter with
$$h[n] = \tfrac{1}{4}\delta[n] + \tfrac{1}{2}\delta[n-1] + \tfrac{1}{4}\delta[n-2].$$
- Let the input be the 4-point signal
$$x[n] = 1\delta[n] + 2\delta[n-1] + 3\delta[n-2] + 4\delta[n-3].$$



- Let N_1 = length($x[n]$) = 4 and N_2 = length($h[n]$) = 3.
- The system output is the 1D linear convolution
$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k].$$



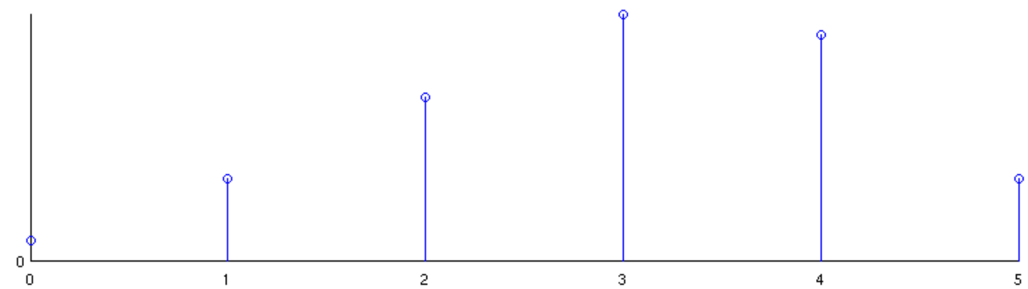
- There is nonzero overlap for $n = 0, 1, 2, 3$; i.e., once for each sample in $x[n]$, and for $n = 4, 5$; i.e., once for each sample in $h[n]$ **but** the last.
- After that there is **no** overlap.
- So the length of the convolution is $N_1 + N_2 - 1 = 6$.

- More precisely, computing $y[n]$ means computing the dot product of $x[k]$ and $h[n\text{-}k]$ for each n.
- Consider values of n starting at the far left ($-\infty$) and going right:
  - for $n < 0$, the rightmost sample of $h[n\text{-}k]$ does not yet reach the leftmost sample of $x[k]$. So the dot product is **zero**.
  - starting at $n = 0$, the rightmost sample of $h[n\text{-}k]$ starts to overlap with the graph of $x[k]$, so we get **nonzero** in general.
  - this situation continues for N_1 values of n, as the rightmost sample of $h[n\text{-}k]$ progresses to overlap each sample in $x[k]$. Thus, in general we get a nonzero dot product for $0 \leq n \leq N_1\text{-}1$; that is, for exactly N_1 values of n.
  - then, at $n = N_1$, the rightmost sample of $h[n\text{-}k]$ hangs over the right edge of the graph of $x[k]$. But we still get a nonzero dot product in general.
  - at $n = N_1+1$, the rightmost **two** samples of $h[n\text{-}k]$ hang over the right edge of the graph of $x[k]$, but the dot product is still nonzero in general.
  - this situation continues until **all but one sample** of $h[n\text{-}k]$ hang over. After that, the graph of $h[n\text{-}k]$ is entirely past the graph of $x[k]$ and the dot product is again zero.
- So, counting this up, we see that in general the dot product can be nonzero one time for each sample in $x[k]$ and one time for each sample in $h[n\text{-}k]$ **except the last one**… because once the last one hangs over there is no overlap.

- So, the convolution of two finite-length sequences with lengths N_1 and N_2 has a length that is given by $N_1 + N_2 - 1$.

- Matlab:

```
>> xn = [1 2 3 4];
>> hn = [0.25 0.5 0.25];
>> yn = conv(xn,hn);
>> length(yn)
ans = 6
>> stem([0:5],yn);
```



- To re-compute this same example using the 1D DFT and circular convolution, we need to zero pad both sequences to a length of at least $N_1 + N_2 - 1 = 6$:

```
>> xprime = [xn zeros(1,2)];
>> hprime = [hn zeros(1,3)];
>> yprime = ifft(fft(xprime).*fft(hprime));
>> max(abs(yn-yprime))
ans = 2.2204e-16
>> % output signal is the same as before
```

Notes on this 1D Example

- You can zero pad to a length $> N_1 + N_2 - 1$. Especially in 2D, the DFT may run faster if the padded length is a power of 2.
  - if you do this, the linear convolution still has length $N_1 + N_2 - 1$. It is contained in the first (leftmost) $N_1 + N_2 - 1$ samples of the result sequence.
- Interpreting $y[n]$ as the weighted 3-point average of $x[n]$:
  - There are **edge effects** on both ends of $y[n]$: zeros are averaged in where the graph of $h[n\text{-}k]$ hangs over the graph of $x[k]$ ($n = 0, 1, 4, 5$).
  - Because the filter is **causal**, it is **not** a centered average. For example, $y[4] = 0.25x[2] + 0.5x[3] + 0.25x[4]$.
  - In other words, there is **delay** (the filter introduces nonzero phase).

Notes 1D Example…

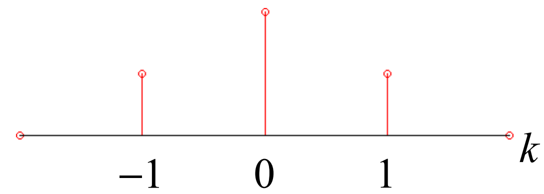
- Often, the edge effects are not a concern in 1D applications.
  - the impulse response $h[n]$ is often short compared to the signal $x[n]$.
  - for example, applying a 13-point average to one minute of digital audio at 44 kHz: $x[n]$ has length 2.64×10^6. But the edge effects impact only the first 7 samples and the last 7.
- A reasonable time delay is also okay in many 1D applications:
  - audio CD player, MP3 player
- A reasonable time delay **may** be of no concern in some image/video applications:
  - DVD/Blu ray player, cable set top box, youtube…

1D Linear Convolution Again

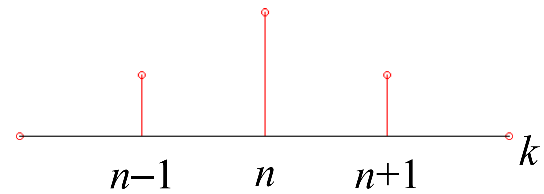
- Back on page 5.38, we had the 1D LTI causal 3-point weighted average filter with
$$h[n] = \tfrac{1}{4}\delta[n] + \tfrac{1}{2}\delta[n-1] + \tfrac{1}{4}\delta[n-2].$$
- By shifting $h[n]$ in time, we can turn this into a non-causal 3-point weighted average that **is centered**, so that it has zero phase and introduces no phase shift:
$$g[n] = h[n+1] = \tfrac{1}{4}\delta[n+1] + \tfrac{1}{2}\delta[n] + \tfrac{1}{4}\delta[n-1].$$
- Because $g[n]$ is real and even, it's DTFT is also **real** and even.
  - This means that the phase $\angle G(e^{j\omega})$ is **identically zero**.
  - So the filter G is not causal, but it introduces no phase shift between the input signal and the output signal.

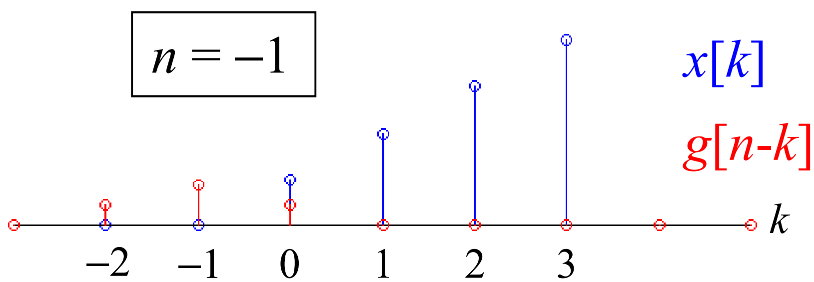
- Recall: $x[n] = 1\delta[n] + 2\delta[n-1] + 3\delta[n-2] + 4\delta[n-3]$.

Graph of $g[k]$



Graph of $g[n-k]$

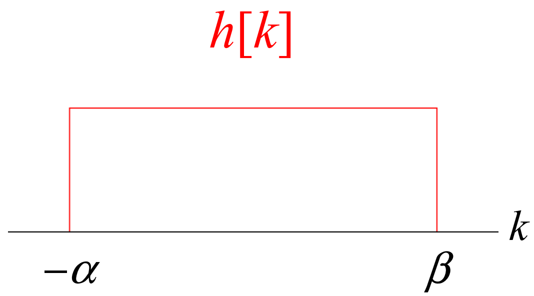


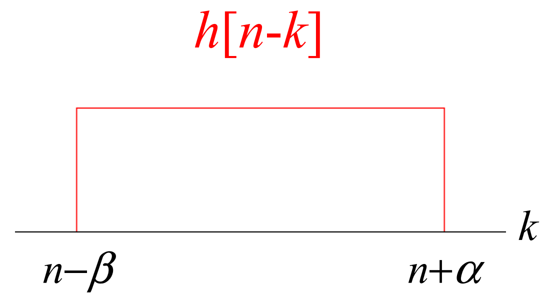


- Now, the first nonzero overlap occurs when $n = -1$ instead of $n = 0$.
- The linear convolution $y[n] = x[n] * g[n]$ still has length $N_1 + N_2 - 1 = 6$.
- But now this corresponds to times $n = -1$ to 4 **instead** of n=0 to 5.
- In Matlab, **everything is still exactly the same as before**.

- So even though everything is exactly the same in Matlab, we have to remember that this time the first array element of `yn=conv(xn,hn)` is for $n = -1$, not $n=0$!
- The length 4 weighted 3-point average signal corresponding to $x[n]$ is obtained by taking `yn` for $n=0$ to 3 **only**.
- In Matlab, this is `yn(2:5)`.
- Why?
  - The convolution $y[n]$ has length 6 and goes from $n = -1$ to $n=4$.
  - So, in Matlab, the first element of `yn` is for $n = -1$, the second element is for $n=0$, and so on…
- Notice that `yn(2:5)` is the same size as $x[n]$ and is not shifted relative to $x[n]$.

- More generally, suppose we have a 1D LTI filter H with an impulse response $h[n]$ that is nonzero from $n = -\alpha$ to $n = +\beta$:





- If $x[n]$ starts at n=0, then the **first** nonzero overlap in the linear convolution $y[n] = x[n] * h[n]$ will occur at n = $-\alpha$.
- So the α+1'st nonzero sample of the convolution will be the one that corresponds to n=0.
- Thus, in the Matlab array `yn=conv(xn,hn)`, it is the element `yn(α+1)` that corresponds to n=0.
- To obtain an output sequence the same length as the input that is **not** shifted, we keep length($x[n]$) samples from `yn` starting at index `α+1` .
- In the example on pages 5.56-5.57, $\alpha = 1$ and length($x[n]$) = 4, so we kept `yn(2:5)`.