Convolution

- The familiar one:

$$y[n] = \sum_{k=-\infty}^{\infty} x_1[k]\, x_2[n-k]$$

- Leave the first signal $x_1[k]$ unchanged

- For $x_2[k]$:

  - Flip the signal: k becomes $-k$, giving $x_2[-k]$

  - Shift the *flipped* signal to the *right* by n samples:
    k becomes $k-n$
    $x_2[-k] \rightarrow x_2[-(k-n)] = x_2[n-k]$

- Carry out sample-by-sample multiplication and sum the resulting sequence to get the output at time index n, i.e. $y[n]$

What happens to periodic signals?

- Suppose both signals are periodic

$$x_1[n+N] = x_1[n]$$
$$x_2[n+N] = x_2[n]$$

  Then $x_1[k]\,x_2[n_0-k]$ will also be periodic (with period N)

- For each value of n_0 we get a different periodic signal (periodicity is N in all cases)

- $|y[n]|$ will be either 0 or ∞

Circular Convolution

$$y[n] \stackrel{?}{=} \sum_{k=0}^{N-1} \tilde{x}_1[k]\, \tilde{x}_2[n-k]$$

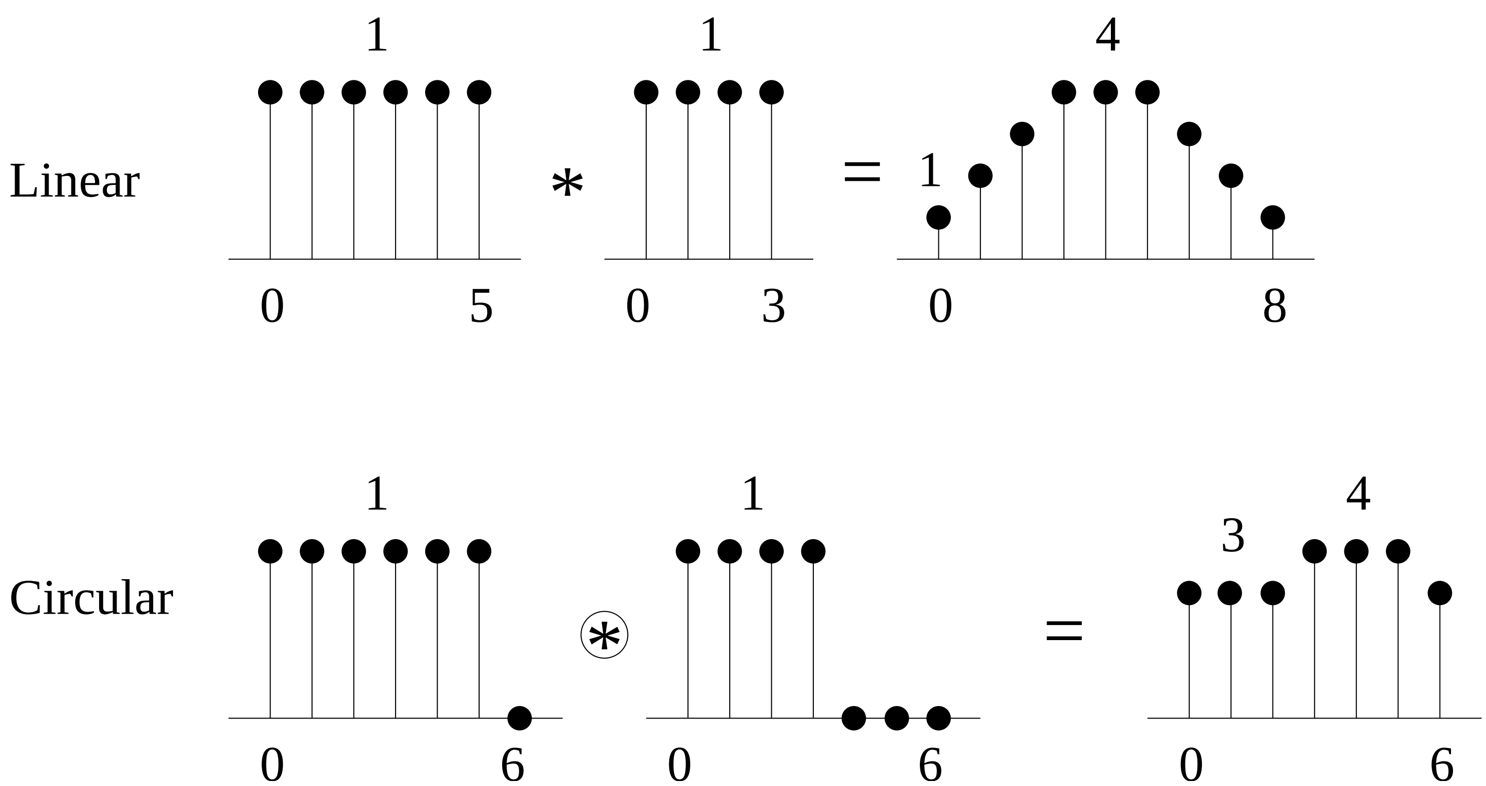
- $y[n]$ is periodic with period N

- $n-k$ can be replaced by $\langle n-k \rangle_N$ ("$n-k$ mod N")

- "Circular" Convolution: $\tilde{y}[n] = \tilde{x}_1[n] \circledast \tilde{x}_2[n]$

$$\tilde{y}[n] \stackrel{\text{def}}{=} \sum_{k=0}^{N-1} \tilde{x}_1[k]\, \tilde{x}_2[\langle n-k \rangle_N] \qquad n = 0, 1, \ldots, N-1$$

Examples



Relationship Between Linear and Circular Convolution

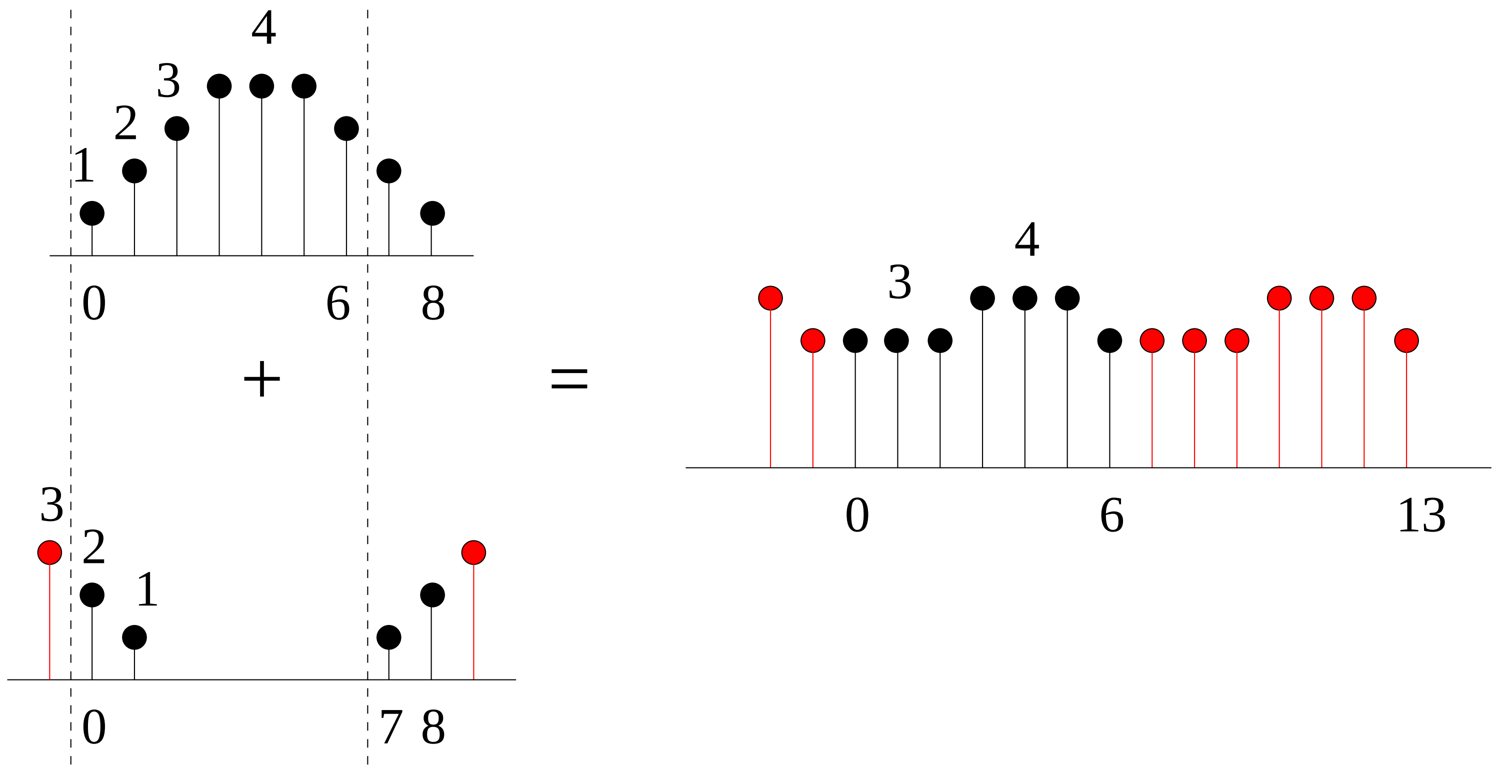
- If $x_1[n]$ has length P and $x_2[n]$ has length Q, then $x_1[n] * x_2[n]$ is $P + Q - 1$ long (e.g., $6 + 4 - 1 = 9$)

- $N \geq \max(P, Q)$. In general

$$\tilde{x}_1[n] \circledast \tilde{x}_2[n] \neq x_1[n] * x_2[n] \qquad n = 0, 1, \ldots, N-1$$

- *Circular convolution can be thought of as repeating the result of linear convolution every N samples and adding the results (over one period)*

Example (cont'd)

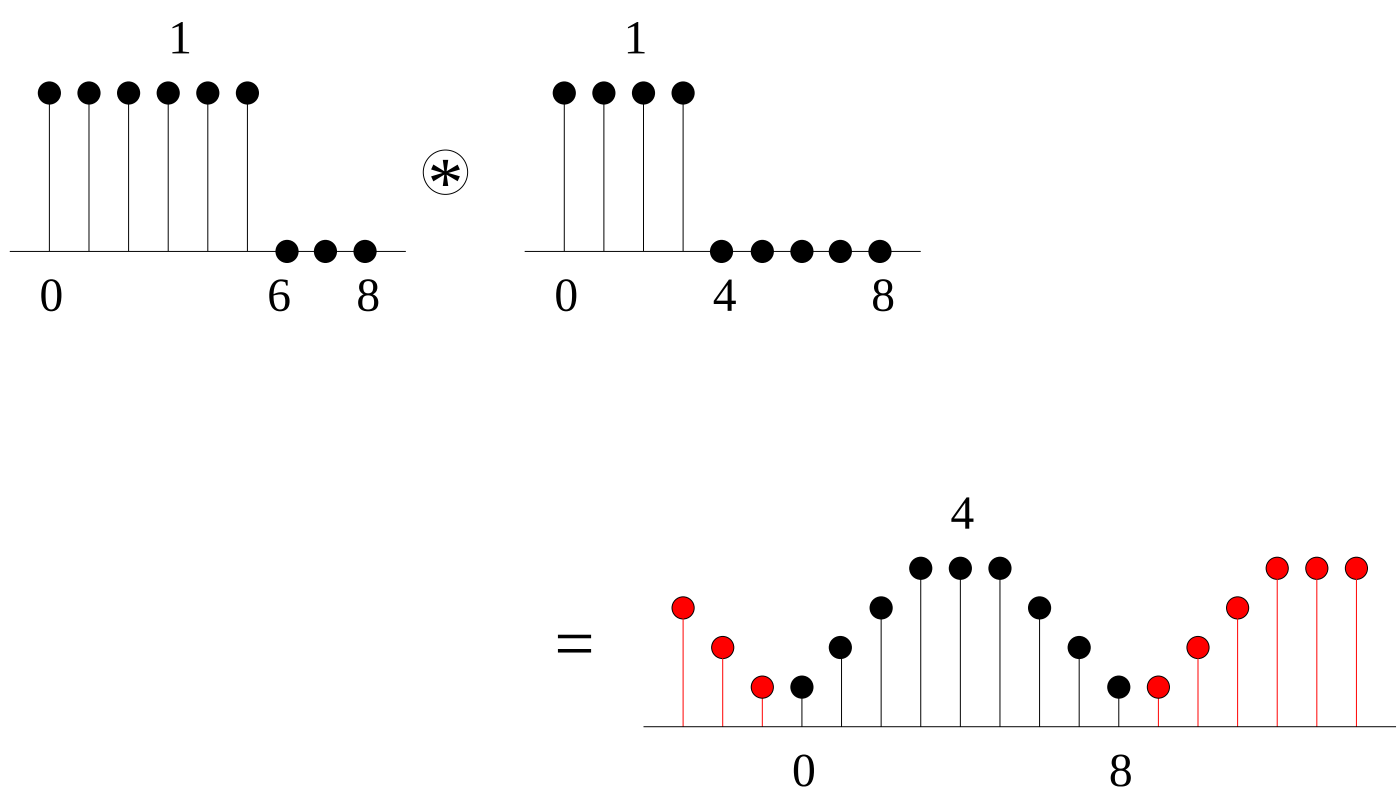


- But if $N \geq P + Q - 1$

$$\tilde{x}_1[n] \circledast \tilde{x}_2[n] = x_1[n] * x_2[n] \qquad n = 0, 1, \ldots, N-1$$

Linear Convolution via Circular Convolution

- If $N \geq 9$ one period of circular convolution will be equal to linear convolution.



Convolution Using the DFT

- A very efficient algorithm, called the Fast Fourier Transform (FFT), exists for computing the DFT

- Since $x_1[n] \circledast x_2[n] \longleftrightarrow X_1[k]\, X_2[k]$, it is more efficient to compute circular convolution using the FFT as follows:

$$y[n] = \text{DFT}^{-1}\left(X_1[k]\, X_2[k] \right)$$