Example-

In this example, we directly convolve the discrete-time (DT) functions,

$$x[n] = p_5[n-2] = u[n] - u[n-5] \text{ with 5 points}$$

and

$$v[n] = r[n]\ p_7[n-3] = r[n]\big(u[n] - u[n-7]\big) \text{ with 7 points,}$$

both directly (use conv() function) and using the FFT & IFFT. The result of the direct convolution has {length(x) + length(v) – 1} = 5 + 7 – 1 = **11 points**.

For the FFT & IFFT method, we will do an L-point FFTs on $x[n]$ and $v[n]$, where

$$\text{length}(x) + \text{length}(v) = 5 + 7 = 12 \ \leq \ \text{choose } L = 2^4 = 16,$$

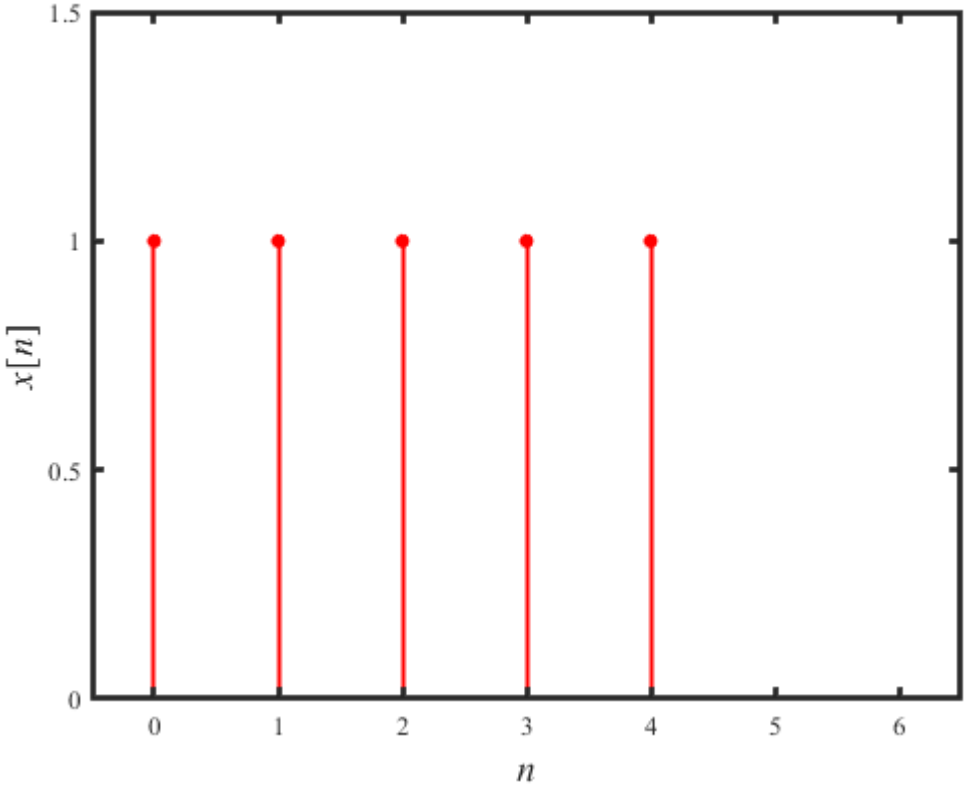
to compute X_k and V_k using the FFT algorithm. MATLAB will add extra zeros to the end of $x[n]$ and $v[n]$, i.e., zero-padding.

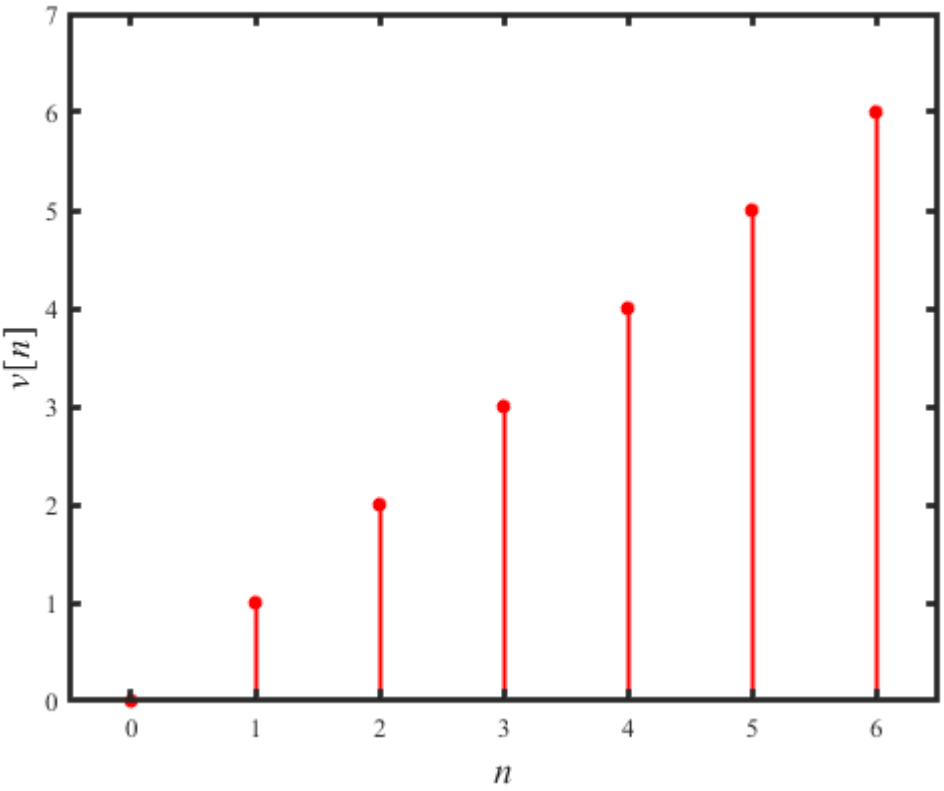
Then, we will compute the frequency-domain product $Y_k = X_k V_k$.

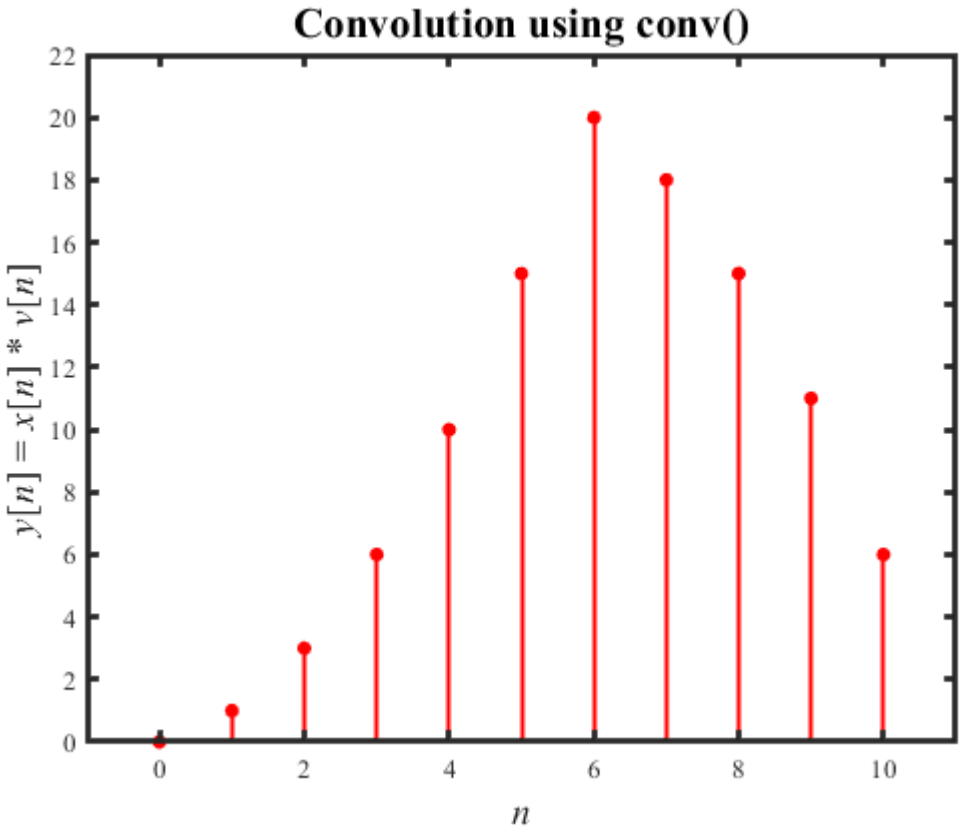
Finally, we will do an L-point IFFT on Y_k to get $y[n] = x[n] * v[n]$. Note that result using the FFT & IFFT has L = **16 points**, i.e., there are 5 extra zeros at the end.

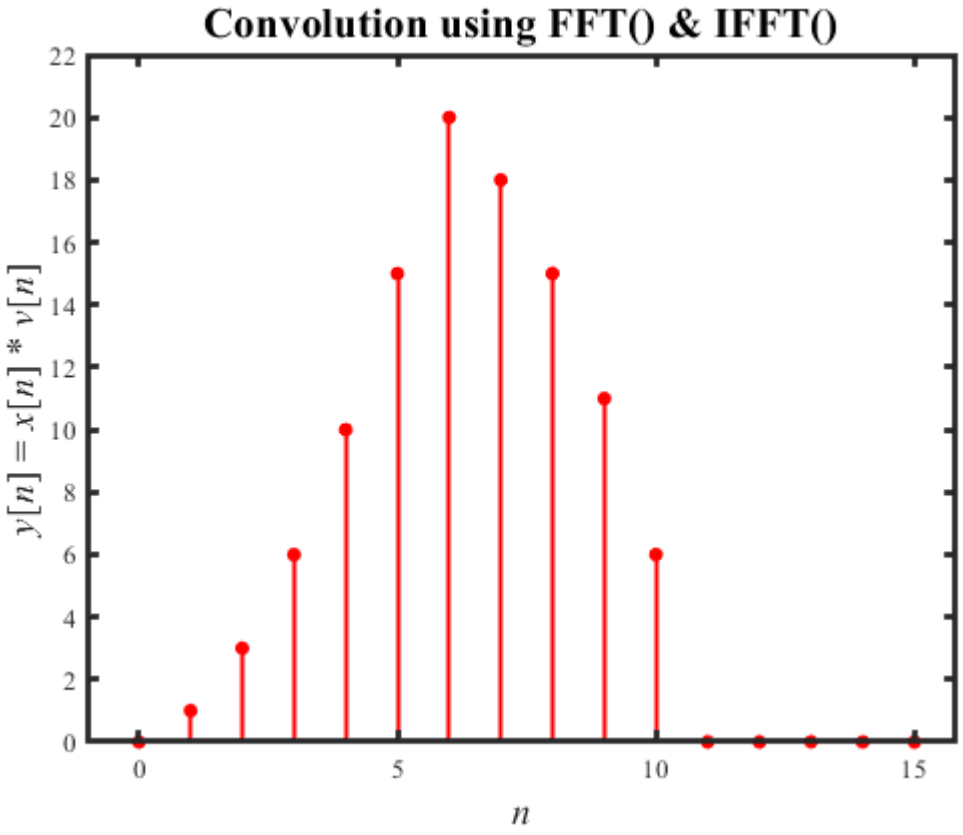
```
% Chapter 4 FFT Example 3 (chap_04_fft_example_3.m)
% Calculate the convolution of two DT signals-
% x[n] = p5[n-2] and v[n] = r[n]*p7[n-3].
% Use FFT & IFFT as well as the conv() function.
% Compare the results>
clear;clc;close all;
Q = 0; P = 0;   % starting indices for x[] & v[]
x = [1,1,1,1,1];     % Define input vector x[n]
v = [0,1,2,3,4,5,6]; % Define input vector v[n]
nx = Q:1:(Q+length(x)-1); % Compute indices for x[n]
nv = P:1:(P+length(v)-1); % Compute indices for v[n]
% plot input x[n]
stem(nx,x,'r.','linewidth',1.5,'markersize',18),
axis([-0.5 6.5 0 1.5]),
ylabel('{\itx}[{\itn}]','fontsize',14,'fontname','times'),
xlabel('{\itn}','fontsize',14,'fontname','times'),
% plot input v[n]
figure, stem(nv,v,'r.','linewidth',1.5,'markersize',18),
axis([-0.5 6.5 0 7]),
ylabel('{\itv}[{\itn}]','fontsize',14,'fontname','times'),
xlabel('{\itn}','fontsize',14,'fontname','times'),
% Compute the output y[n] using conv.m and plot
yconv = conv(x,v);
nconv=(P+Q):1:(P+Q+length(yconv)-1); % Compute indices for yconv[n]
figure, stem(nconv,yconv,'r.','linewidth',1.5,'markersize',18),
axis([P+Q-1 max(nconv)+1 0 22]),
ylabel('{\ity}[{\itn}] = {\itx}[{\itn}] * {\itv}[{\itn}]',...
    'fontsize',14,'fontname','times'),
xlabel('{\itn}','fontsize',14,'fontname','times'),
title('Convolution using conv()','fontsize',16,'fontname','times'),
% Compute the output y[n] using FFT and plot
L = 16; % Choose FFT length >= length(x) + length(v)
Xk = fft(x,L); Vk = fft(v,L); % L-point FFTs of x[n] & v[n]
Yk = Xk.*Vk; % Multiply Xk and Vk (point by point)
yfft = ifft(Yk,L);
nfft = (P+Q):1:(P+Q+length(yfft)-1); % Compute indices for yfft[n]
figure, stem(nfft,yfft,'r.','linewidth',1.5,'markersize',18),
axis([P+Q-1 max(nfft)+0.8 0 22]),
ylabel('{\ity}[{\itn}] = {\itx}[{\itn}] * {\itv}[{\itn}]',...
    'fontsize',14,'fontname','times'),
xlabel('{\itn}','fontsize',14,'fontname','times'),
title('Convolution using FFT() & IFFT()','fontsize',16,'fontname','times'),
set(findobj('type','line'),'linewidth',1.5,'markersize',14)
set(findobj('type','axes'),'linewidth',2,'fontname','times')
set(findobj('type','text'),'fontname','times')
```

- Plot input signals









- **<u>Same result</u>! Note, from zero-padding, the FFT method has a few extra zeros in the answer.**