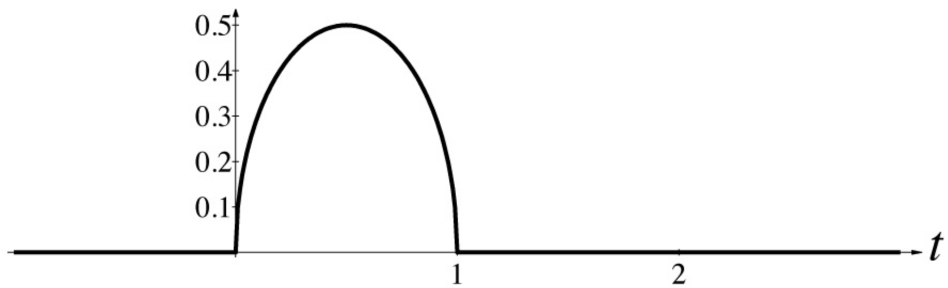
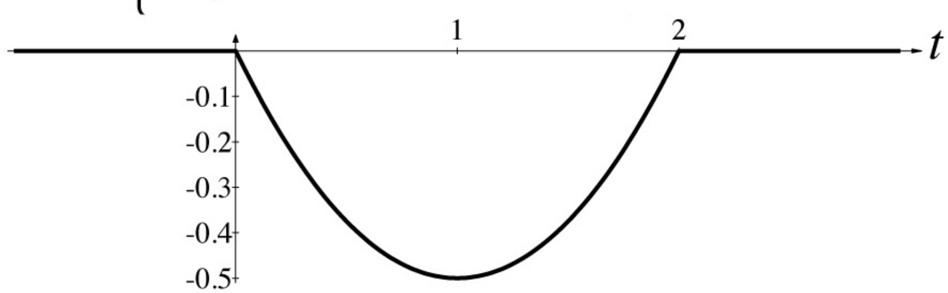
Convolve the two signals pictured below. This is difficult, if not impossible, analytically. So let's do it numerically using the DFT.

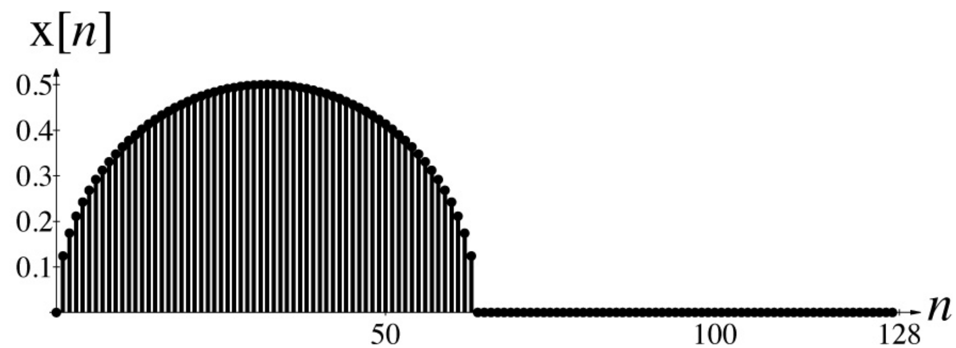
$$\mathrm{x}(t) = \begin{cases} \sqrt{0.25 - (t-1)^2} & , \ 0 < t < 1 \\ 0 & , \ \text{otherwise} \end{cases}$$



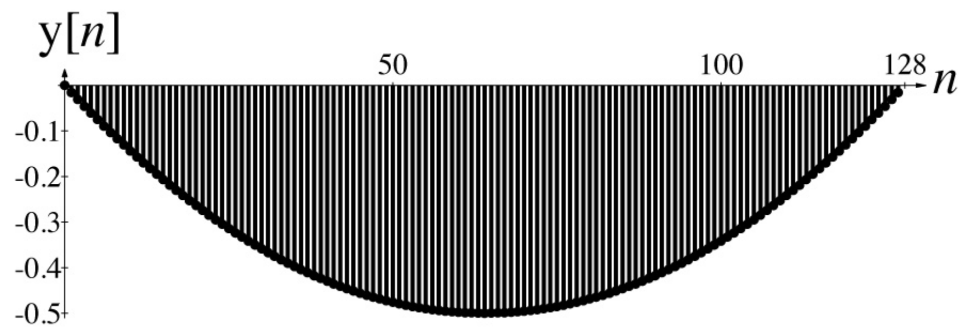
$$\mathrm{y}(t) = \begin{cases} -t(1 - t/2) & , \ 0 < t < 2 \\ 0 & , \ \text{otherwise} \end{cases}$$



The first step is to sample the two signals. They are not bandlimited so we cannot actually obey the sampling theorem but we can make a reasonable approximation. Let's sample both 128 times in the time range $0 < t < 2$.

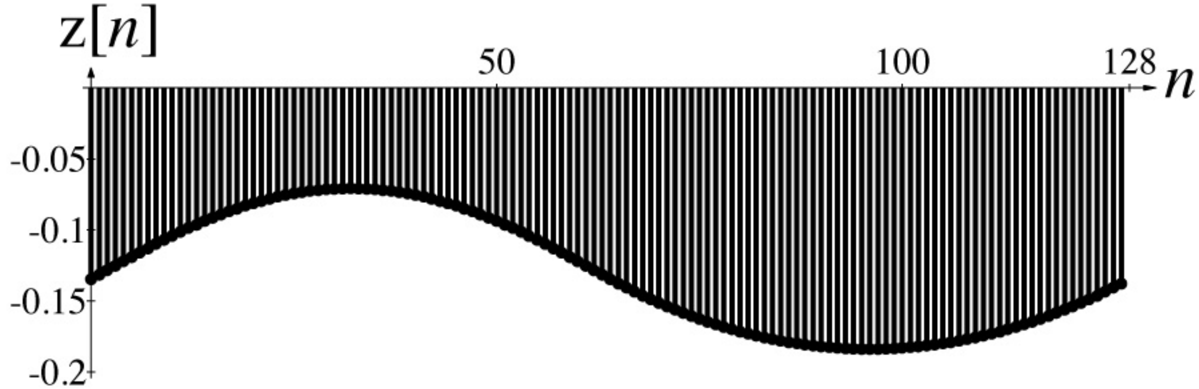


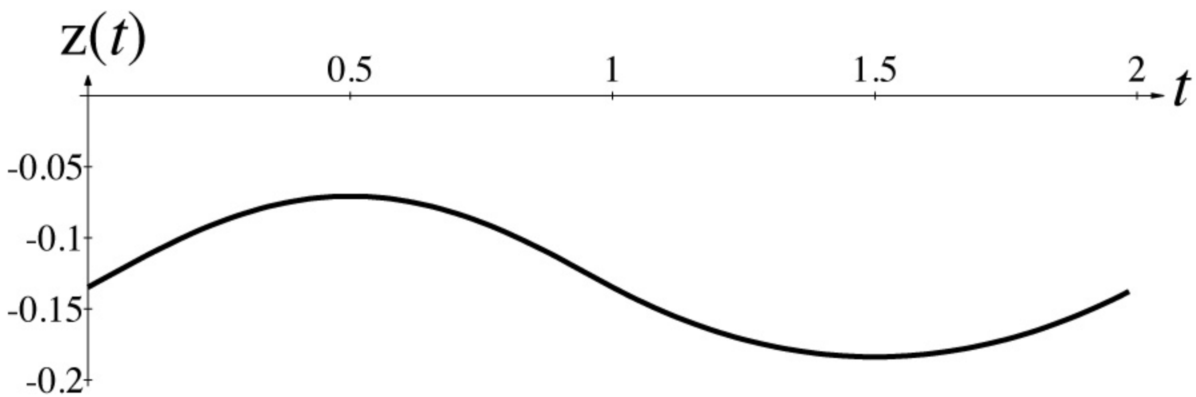

```
N = 128 ;      %   Number of samples
T = 2 ;        %   Total sampling time
Ts = T/N ;     %   Time between samples
n = [0:N-1]' ; %   Vector of discrete times covering time range 0<t<2
%   Compute samples of x(t) as xs[n]
xs = sqrt(0.25 - (n*Ts-0.5).^2) ; I = find(n*Ts>1) ; xs(I) = 0 ;
%   Compute samples of y(t) as ys[n]
ys = -n*Ts.*(1-n*Ts/2) ;
```



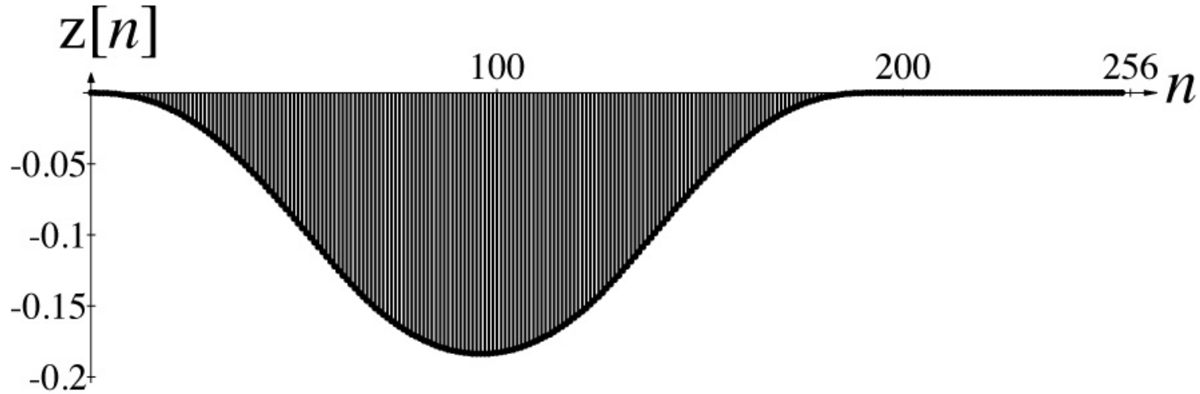
Now convolve the two signals by finding their DFT's, multiplying them and then finding the inverse DFT of that product.

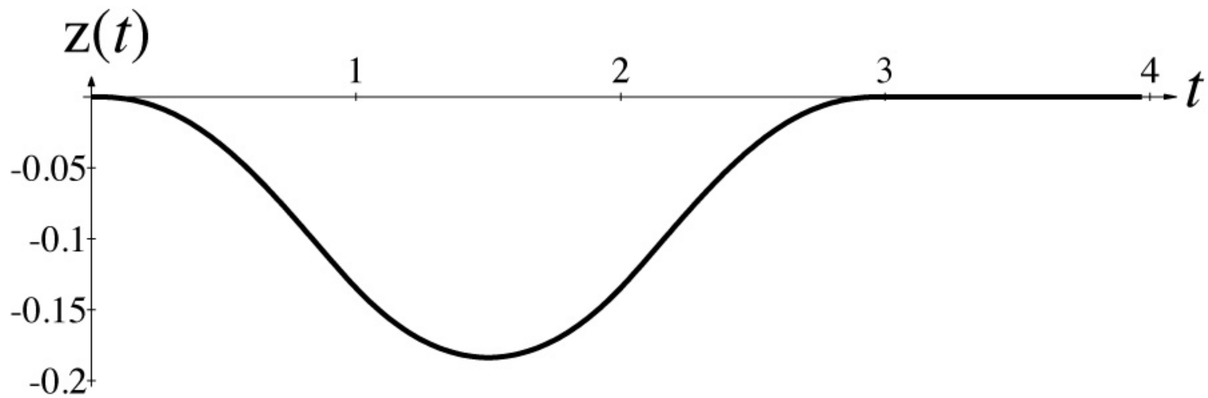
```
Xs = fft(xs) ;      %   DFT of xs
Ys = fft(ys) ;      %   DFT of ys
Zs = Xs.*Ys ;       %   DFT of zs = xs convolved with ys
zs = Ts*real(ifft(Zs)) ;    %   Sampled version of z(t) as zs[n]
```





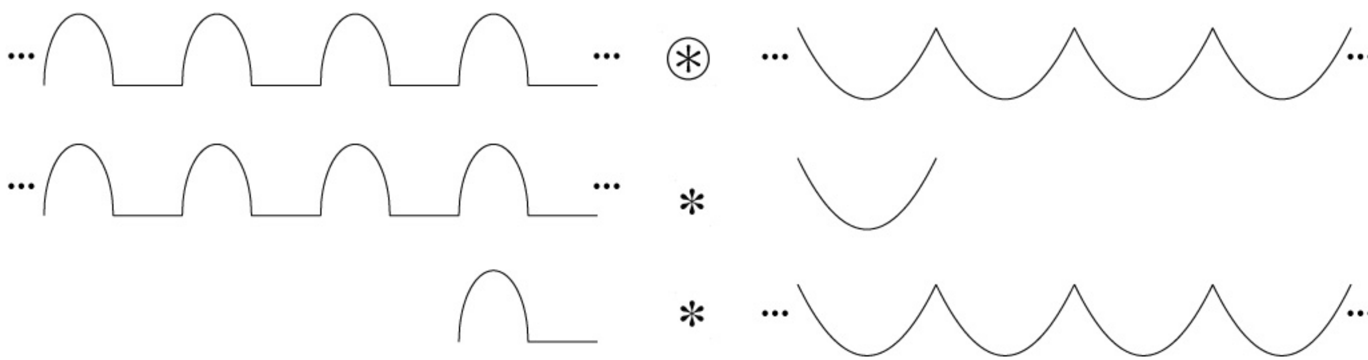
Let's check this result by using the "conv" function in MATLAB.





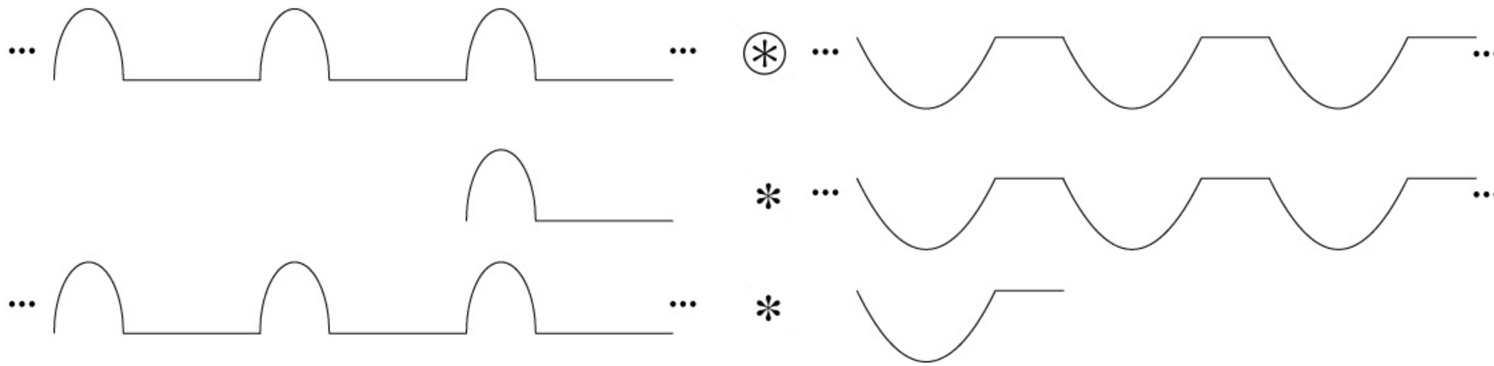
Obviously something is wrong. The two convolution results do not look the same.

The problem is that when we use the DFT to approximate aperiodic convolution we are actually doing periodic convolution as illustrated here.



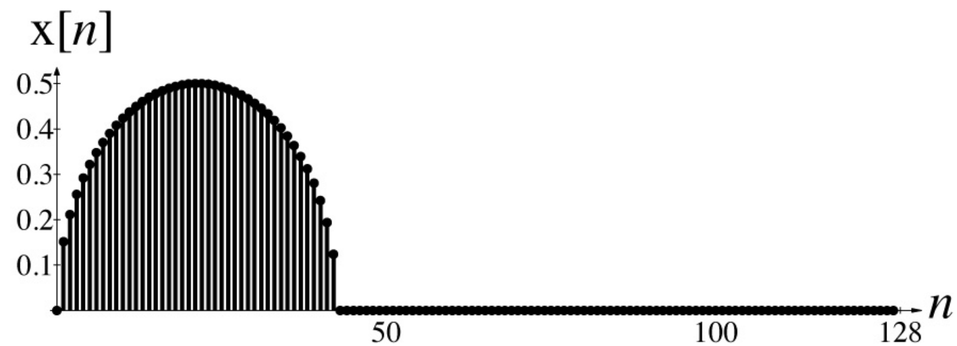
To make periodic convolution a good approximation to aperiodic convolution we need to avoid the "wrap-around" effect by extending the signals with enough zeros that the non-zero part of the periodic extension of one signal never overlaps the non-zero part of the other signal. In this case one signal's non-zero part has a width of one and the other signal's non-zero part has a width of two. So we need to extend both signals with zeros to a width of at least three.

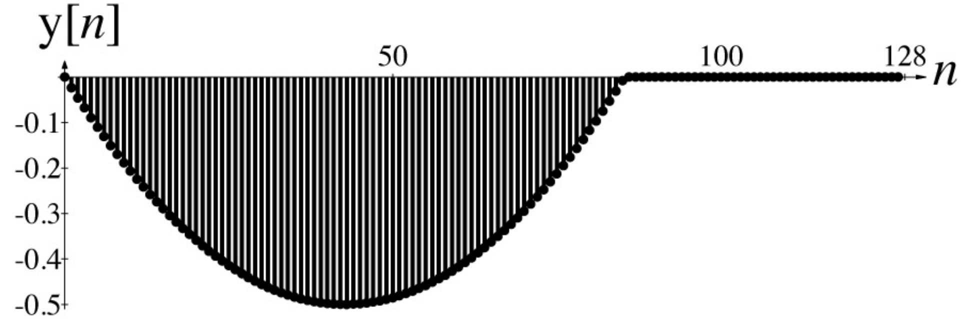
Then the convolution process will look like this.



Second

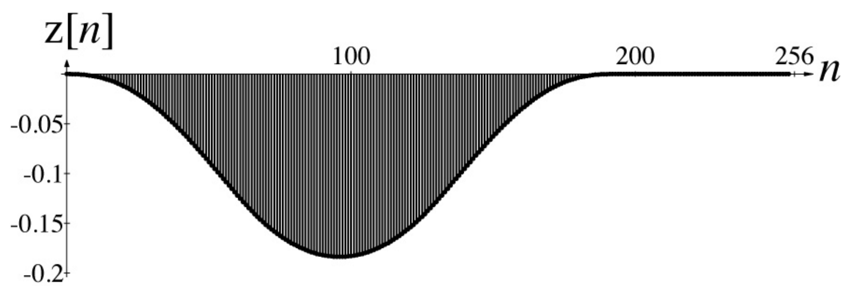
Sampling

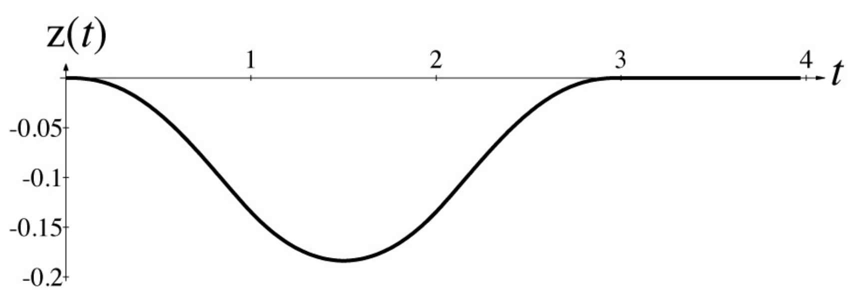




Now the two convolutions are practically identical

Using the MATLAB "conv" Function





Using the DFT