

Convolution of Signals in MATLAB

Robert Francis

August 29, 2011



[Telegram.me/matlabonline](https://t.me/matlabonline)

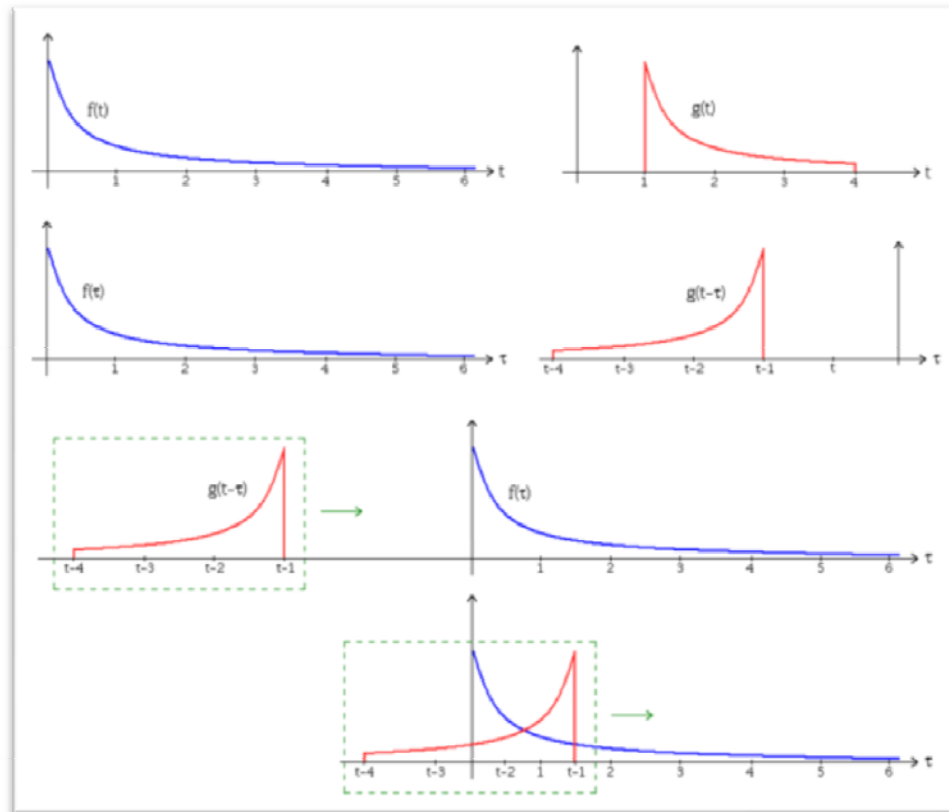
کانال تخصصی
آموزش کاربردی Matlab
دانشگاه تهران

[Telegram.me/expertonline](https://t.me/expertonline)

مرکز شبیه سازی مقالات مرجع و
معتبر

Review of Convolution

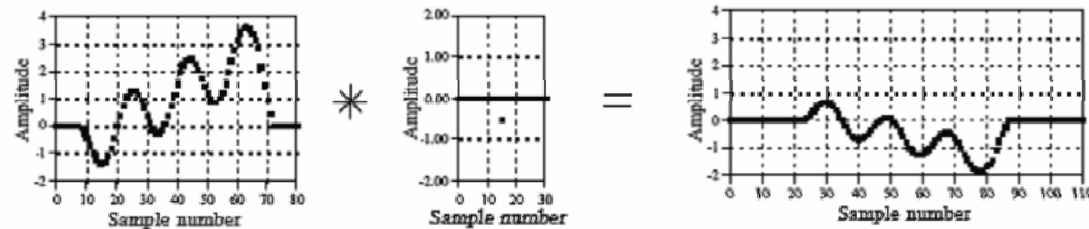
$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau)h(\tau - t)d\tau$$



Visual example copied from Wikipedia

Review of Convolution

a. Inverting Attenuator



b. Discrete Derivative

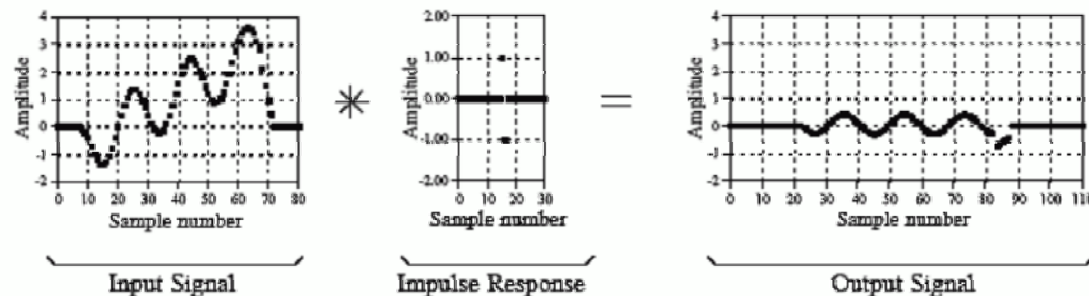


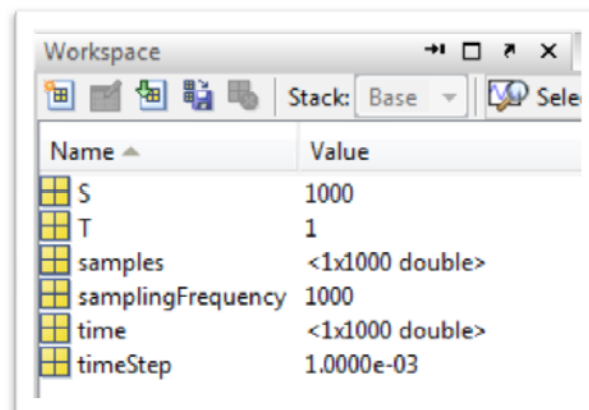
FIGURE 6-4

Examples of signals being processed using convolution. Many signal processing tasks use very simple impulse responses. As shown in these examples, dramatic changes can be achieved with only a few nonzero points.

Smith, <http://www.dspguide.com/ch6/2.htm>

Generating Time Vectors in Matlab

```
%% Define time vector  
samplingFrequency = 1000; %Hz  
timeStep = 1/samplingFrequency; %sec  
T = 1; %sec  
S = T*samplingFrequency; %samples  
samples = 1:S; %samples  
time = samples*timeStep; %sec
```

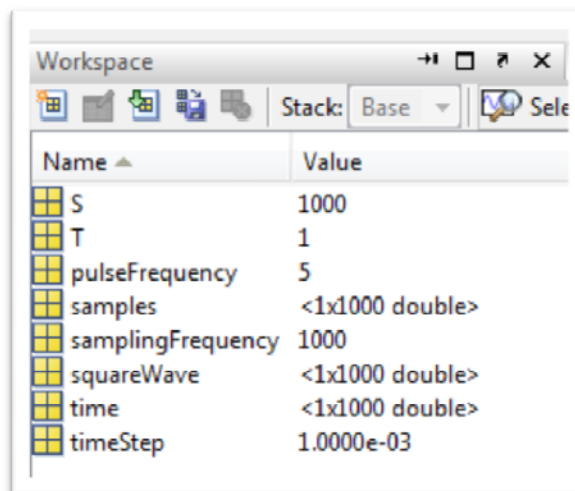


The screenshot shows the MATLAB Workspace window with a table of variables and their values. The variables are S, T, samples, samplingFrequency, time, and timeStep. The values are 1000, 1, <1x1000 double>, 1000, <1x1000 double>, and 1.0000e-03 respectively.

Name	Value
S	1000
T	1
samples	<1x1000 double>
samplingFrequency	1000
time	<1x1000 double>
timeStep	1.0000e-03

Generate Square Wave

```
%% Generate square wave  
pulseFrequency = 5; %Hz  
squareWave = square(time*pulseFrequency*2*pi);
```

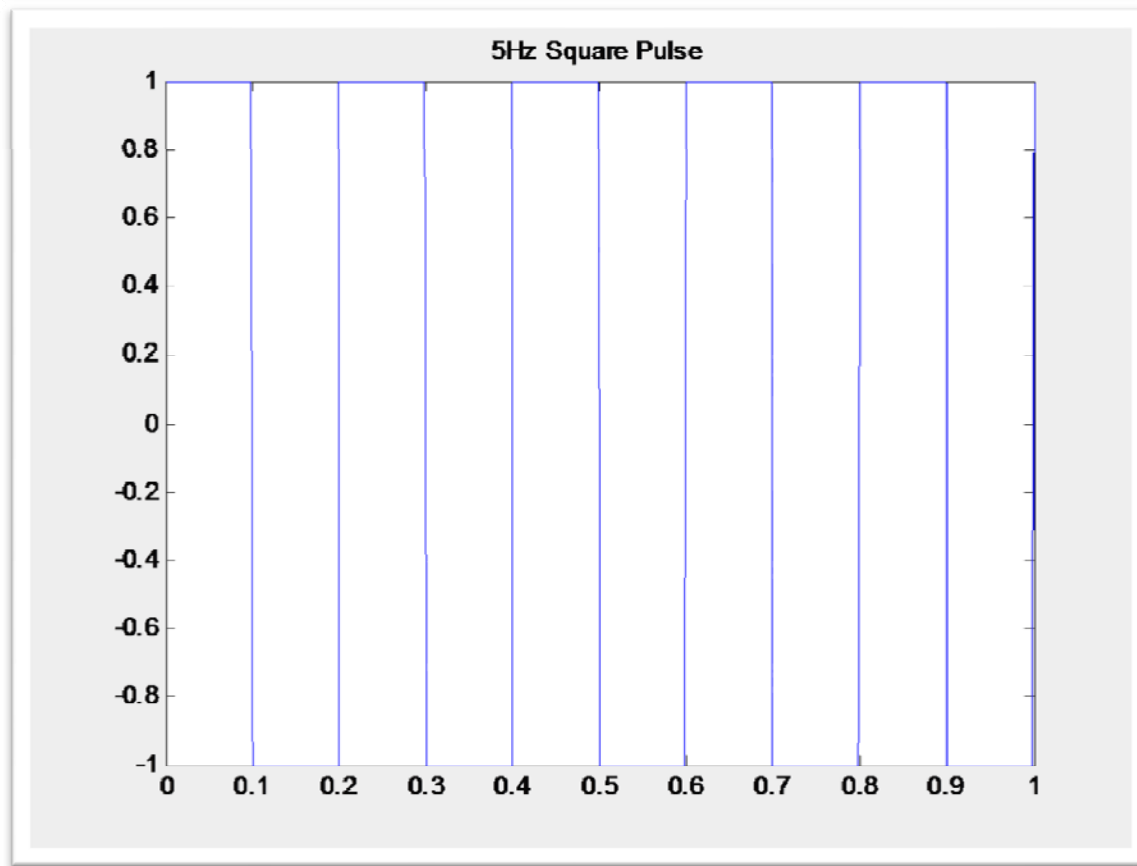


The image shows the MATLAB Workspace window. It contains a table with two columns: 'Name' and 'Value'. The variables listed are S (1000), T (1), pulseFrequency (5), samples (<1x1000 double>), samplingFrequency (1000), squareWave (<1x1000 double>), time (<1x1000 double>), and timeStep (1.0000e-03). The 'Stack' is set to 'Base'.

Name	Value
S	1000
T	1
pulseFrequency	5
samples	<1x1000 double>
samplingFrequency	1000
squareWave	<1x1000 double>
time	<1x1000 double>
timeStep	1.0000e-03

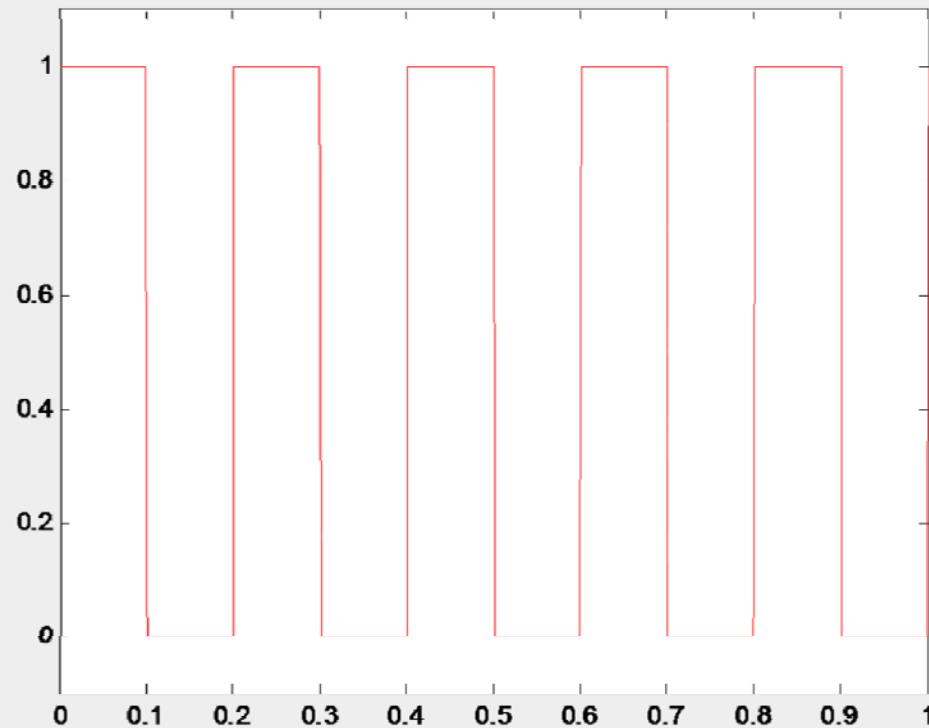
Plot Square Wave

```
%% Plot square wave  
figure(1), plot(time,squareWave,'b')  
title([num2str(pulseFrequency) 'Hz Square Pulse'])
```



Adjust Magnitude and Re-plot

```
%% Adjust amplitude  
squareWavePos = (squareWave+1)/2;  
close(1)  
figure(1), plot(time,squareWavePos,'r')  
set(gca,'YLim',[-0.1 1.1])
```

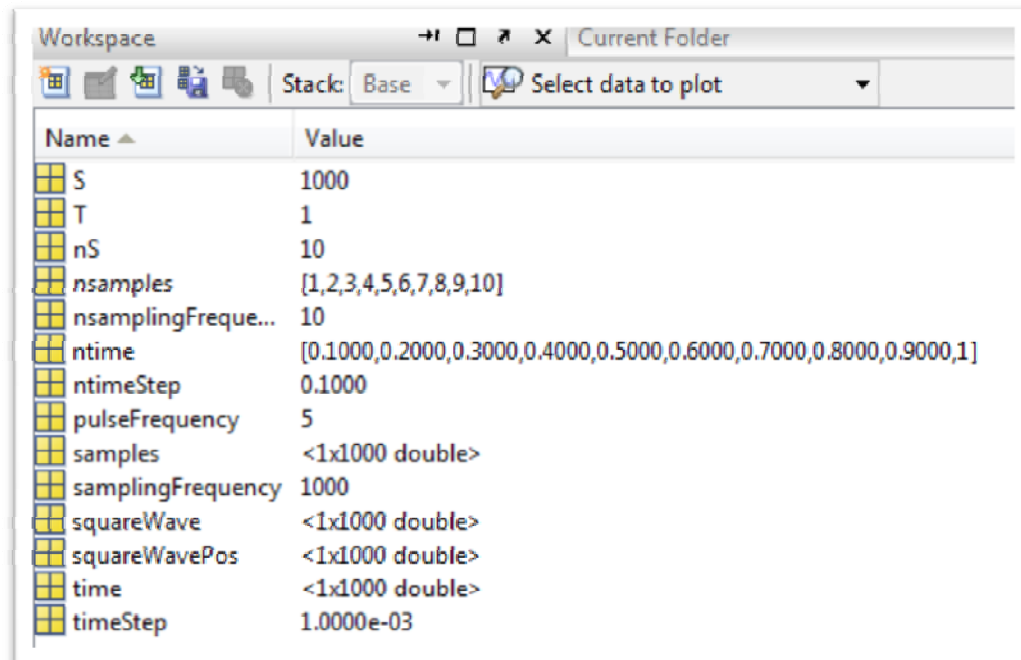


Nyquist Sampling Theorem

- If a continuous time signal has no frequency components above f_h , then it can be specified by a discrete time signal with a sampling frequency greater than twice f_h .

Define Nyquist Sampling Time Vector

```
%% Define Nyquist Sampling Time Vector
nsamplingFrequency = 2*pulseFrequency;
ntimeStep = 1/nsamplingFrequency; %sec
T = 1; %sec
nS = T*nsamplingFrequency; %samples
nsamples = 1:nS; %samples
ntime = nsamples*ntimeStep; %sec
```

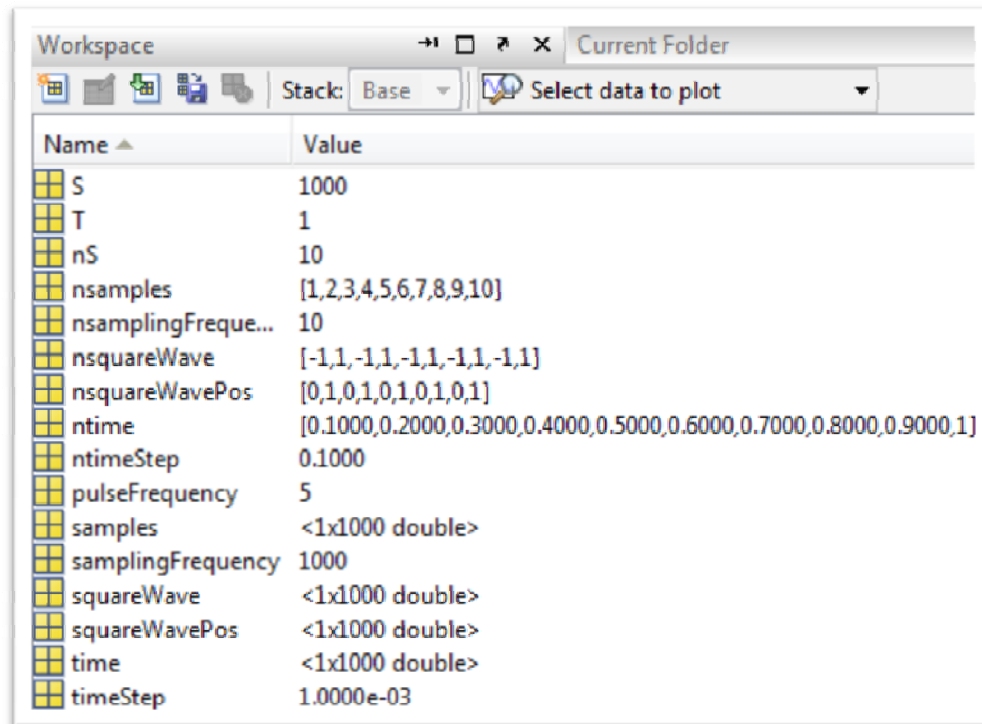


The image shows a screenshot of the MATLAB Workspace window. The window has a title bar with 'Workspace' and standard window controls. Below the title bar is a toolbar with icons for workspace, command window, and current folder. A 'Stack' dropdown menu is set to 'Base'. To the right of the stack menu is a 'Select data to plot' button. The main area of the workspace is a table with two columns: 'Name' and 'Value'. The table lists various variables and their corresponding values.

Name	Value
S	1000
T	1
nS	10
nsamples	[1,2,3,4,5,6,7,8,9,10]
nsamplingFreque...	10
ntime	[0.1000,0.2000,0.3000,0.4000,0.5000,0.6000,0.7000,0.8000,0.9000,1]
ntimeStep	0.1000
pulseFrequency	5
samples	<1x1000 double>
samplingFrequency	1000
squareWave	<1x1000 double>
squareWavePos	<1x1000 double>
time	<1x1000 double>
timeStep	1.0000e-03

Generate Nyquist Sampling Square Wave

```
%% Generate Nyquist Sampling Square Wave  
nsquareWave = square(ntime*pulseFrequency*2*pi);  
nsquareWavePos = (nsquareWave+1)/2;  
|
```

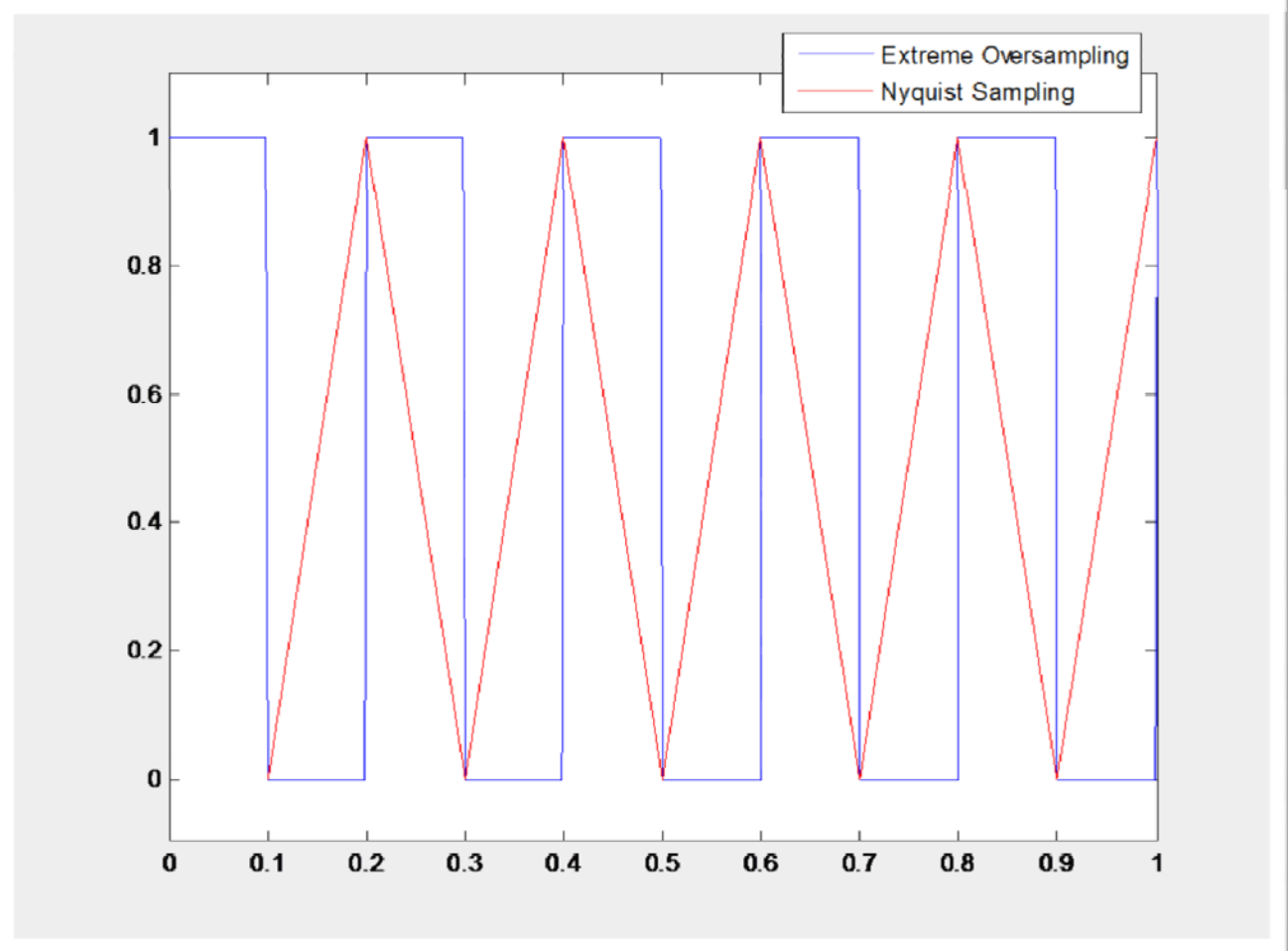


The image shows a screenshot of the MATLAB Workspace window. The window has a title bar with 'Workspace' and a 'Current Folder' tab. Below the title bar is a toolbar with icons for workspace, command window, and other functions. A 'Stack' dropdown menu is set to 'Base'. To the right of the stack menu is a 'Select data to plot' button. The main area of the workspace is a table with two columns: 'Name' and 'Value'. The table lists various variables and their corresponding values.

Name	Value
S	1000
T	1
nS	10
nsamples	[1,2,3,4,5,6,7,8,9,10]
nsamplingFreque...	10
nsquareWave	[-1,1,-1,1,-1,1,-1,1]
nsquareWavePos	[0,1,0,1,0,1,0,1]
ntime	[0.1000,0.2000,0.3000,0.4000,0.5000,0.6000,0.7000,0.8000,0.9000,1]
ntimeStep	0.1000
pulseFrequency	5
samples	<1x1000 double>
samplingFrequency	1000
squareWave	<1x1000 double>
squareWavePos	<1x1000 double>
time	<1x1000 double>
timeStep	1.0000e-03

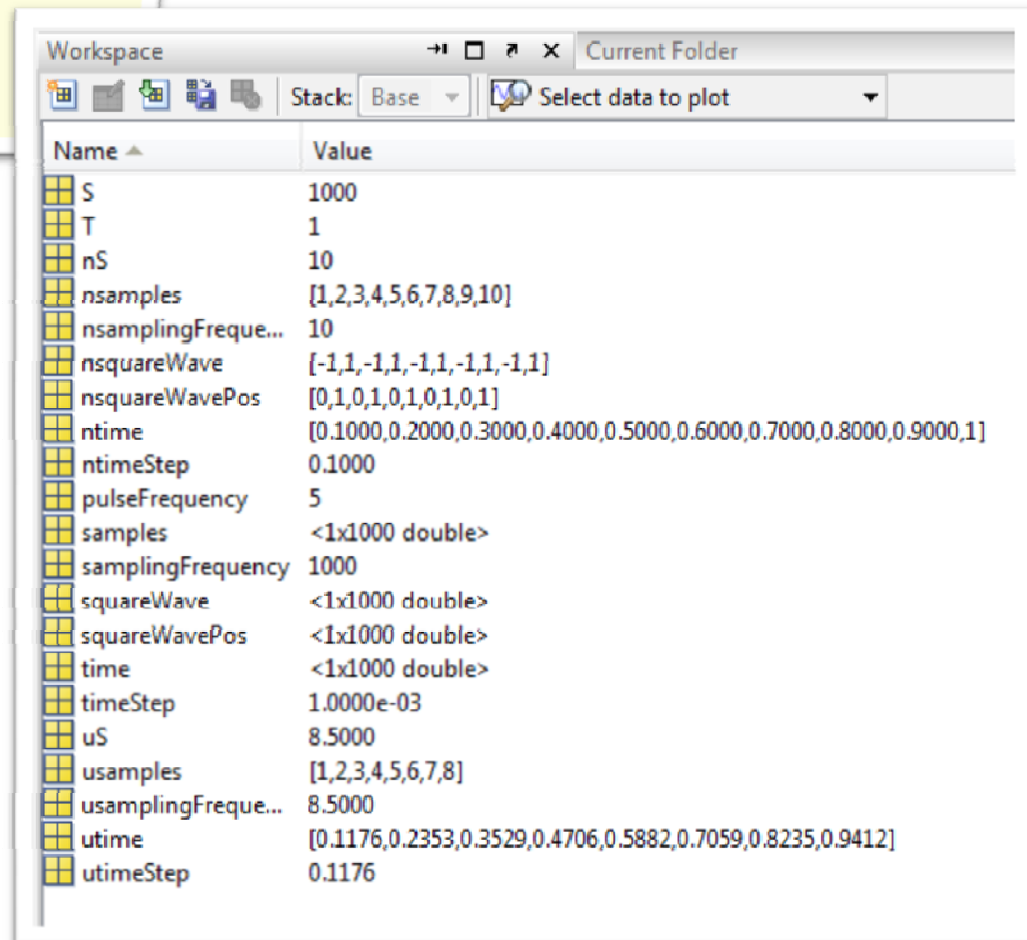
Plot Nyquist Square Wave

```
%% Plot Nyquist Sampling Square Wave Compared to Oversampled Square Wave  
figure(2), plot(time,squareWavePos,'b',ntime,nsquareWavePos,'r')  
legend('Extreme Oversampling','Nyquist Sampling')  
set(gca,'YLim',[-0.1 1.1])
```



Define Undersampled Time Vector

```
%% Define Undersampled Time Vector
usamplingFrequency = 1.7*pulseFrequency;
utimeStep = 1/usamplingFrequency; %sec
T = 1; %sec
uS = T*usamplingFrequency; %samples
usamples = 1:uS; %samples
utime = usamples*utimeStep; %sec
```

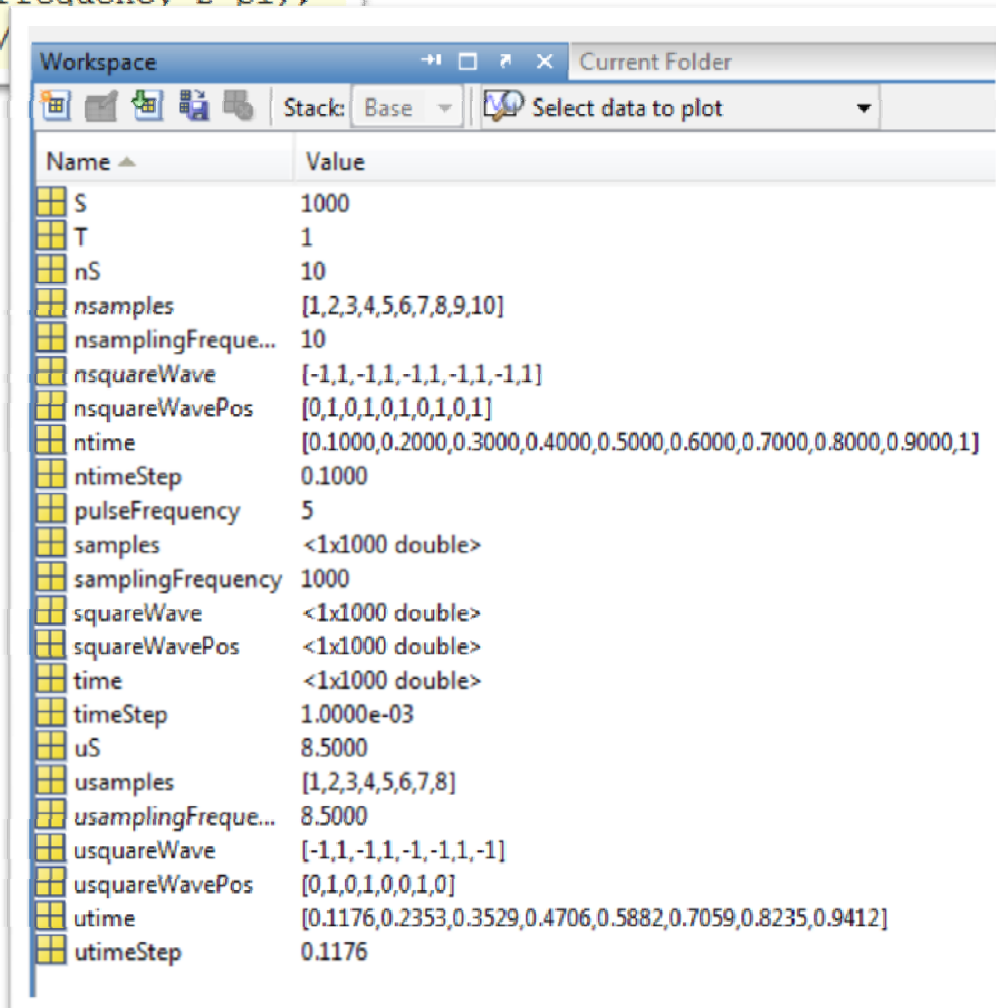


The image shows the MATLAB Workspace window with the following variables and values:

Name	Value
S	1000
T	1
nS	10
nsamples	[1,2,3,4,5,6,7,8,9,10]
nsamplingFreque...	10
nsquareWave	[-1,1,-1,1,-1,1,-1,1,-1,1]
nsquareWavePos	[0,1,0,1,0,1,0,1,0,1]
ntime	[0.1000,0.2000,0.3000,0.4000,0.5000,0.6000,0.7000,0.8000,0.9000,1]
ntimeStep	0.1000
pulseFrequency	5
samples	<1x1000 double>
samplingFrequency	1000
squareWave	<1x1000 double>
squareWavePos	<1x1000 double>
time	<1x1000 double>
timeStep	1.0000e-03
uS	8.5000
usamples	[1,2,3,4,5,6,7,8]
usamplingFreque...	8.5000
utime	[0.1176,0.2353,0.3529,0.4706,0.5882,0.7059,0.8235,0.9412]
utimeStep	0.1176

Generate Undersampled Square Wave

```
%% Generate Undersampled Square Wave
usquareWave = square(utime*pulseFrequency*2*pi);
usquareWavePos = (usquareWave+1)/
```

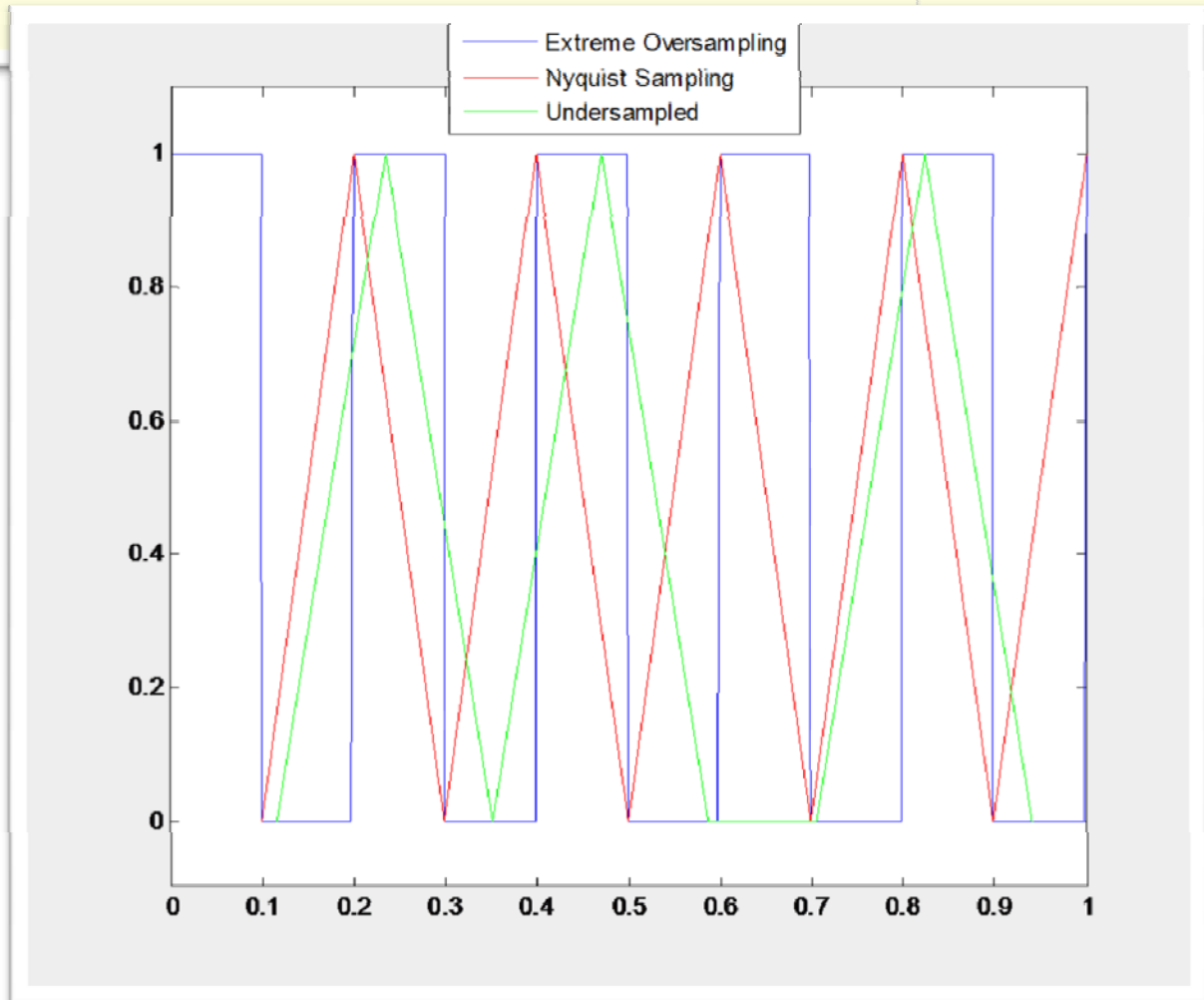


The image shows the MATLAB Workspace window with the following variables and values:

Name	Value
S	1000
T	1
nS	10
nsamples	[1,2,3,4,5,6,7,8,9,10]
nsamplingFreque...	10
nsquareWave	[-1,1,-1,1,-1,1,-1,1,-1,1]
nsquareWavePos	[0,1,0,1,0,1,0,1,0,1]
ntime	[0.1000,0.2000,0.3000,0.4000,0.5000,0.6000,0.7000,0.8000,0.9000,1]
ntimeStep	0.1000
pulseFrequency	5
samples	<1x1000 double>
samplingFrequency	1000
squareWave	<1x1000 double>
squareWavePos	<1x1000 double>
time	<1x1000 double>
timeStep	1.0000e-03
uS	8.5000
usamples	[1,2,3,4,5,6,7,8]
usamplingFreque...	8.5000
usquareWave	[-1,1,-1,1,-1,-1,1,-1]
usquareWavePos	[0,1,0,1,0,0,1,0]
utime	[0.1176,0.2353,0.3529,0.4706,0.5882,0.7059,0.8235,0.9412]
utimeStep	0.1176

Plot Undersampled

```
%% Plot Nyquist Sampling Square Wave Compared to Oversampled Square Wave  
figure(2), plot(time,squareWavePos,'b',ntime,nsquareWavePos,'r',utime,usquareWavePos,'g')  
legend('Extreme Oversampling','Nyquist Sampling','Undersampled')  
set(gca,'YLim',[-0.1 1.1])
```



Convolution With Linear Decay Signal

```
% Convolution with Linear Decay
%Generate Linear Decay Signals
lds = fliplr(time)/sum(time);
nlds = fliplr(ntime)/sum(ntime);
ulds = fliplr(utime)/sum(utime);

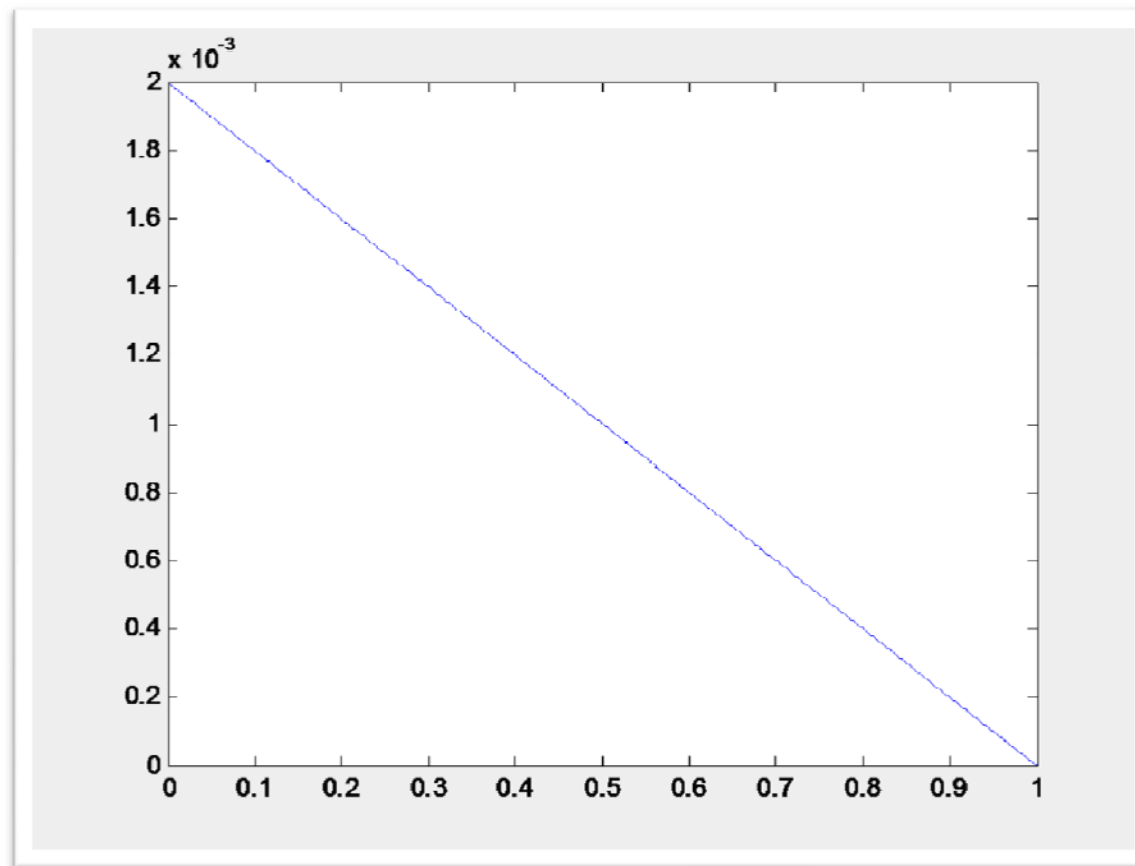
%Convolve Square Wave Signals with Linear Decay Signals
squareWaveLDS = conv(squareWavePos,lds);
nsquareWaveNLDS = conv(nsquareWavePos,nlds);
usquareWaveULDS = conv(usquareWavePos,ulds);

%Extend time vectors to match convolved signals
convtime = (-S:1:S)*timeStep;
nconvtime = (-nS:1:nS)*ntimeStep;
uconvtime = (-uS:1:uS)*utimeStep;

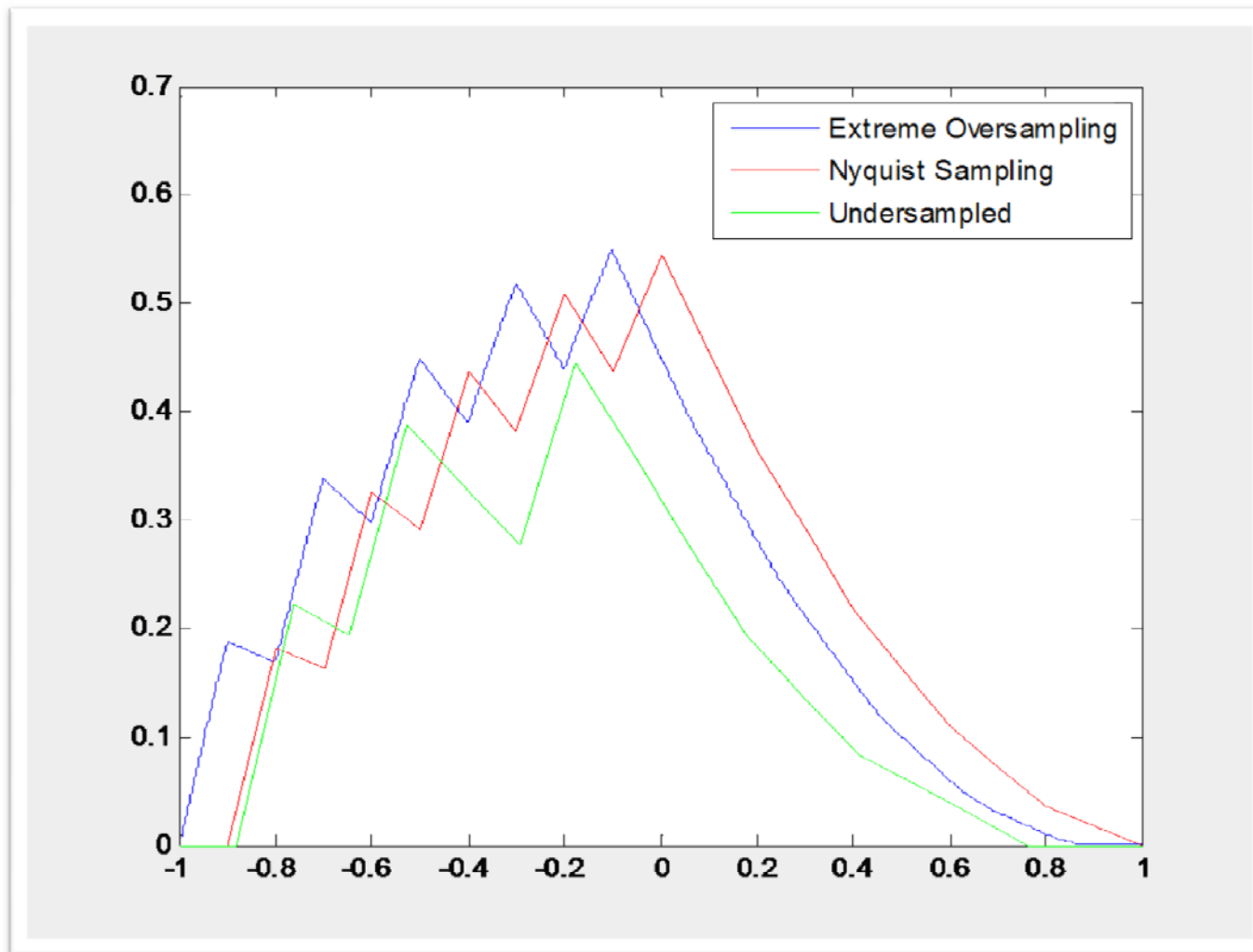
%Pad convolved signals with leading and trailing zero
squareWaveLDS = [0,squareWaveLDS,0];
nsquareWaveNLDS = [0,nsquareWaveNLDS,0];
usquareWaveULDS = [0,usquareWaveULDS,0];

figure(3), plot(convtime,squareWaveLDS,'b',nconvtime,nsquareWaveNLDS,'r',uconvtime,usquareWaveULDS,'g')
legend('Extreme Oversampling','Nyquist Sampling','Undersampled')
```

Linear Decay Signal



Plot of Convolutions



Convolution with Exponential Decay Signal

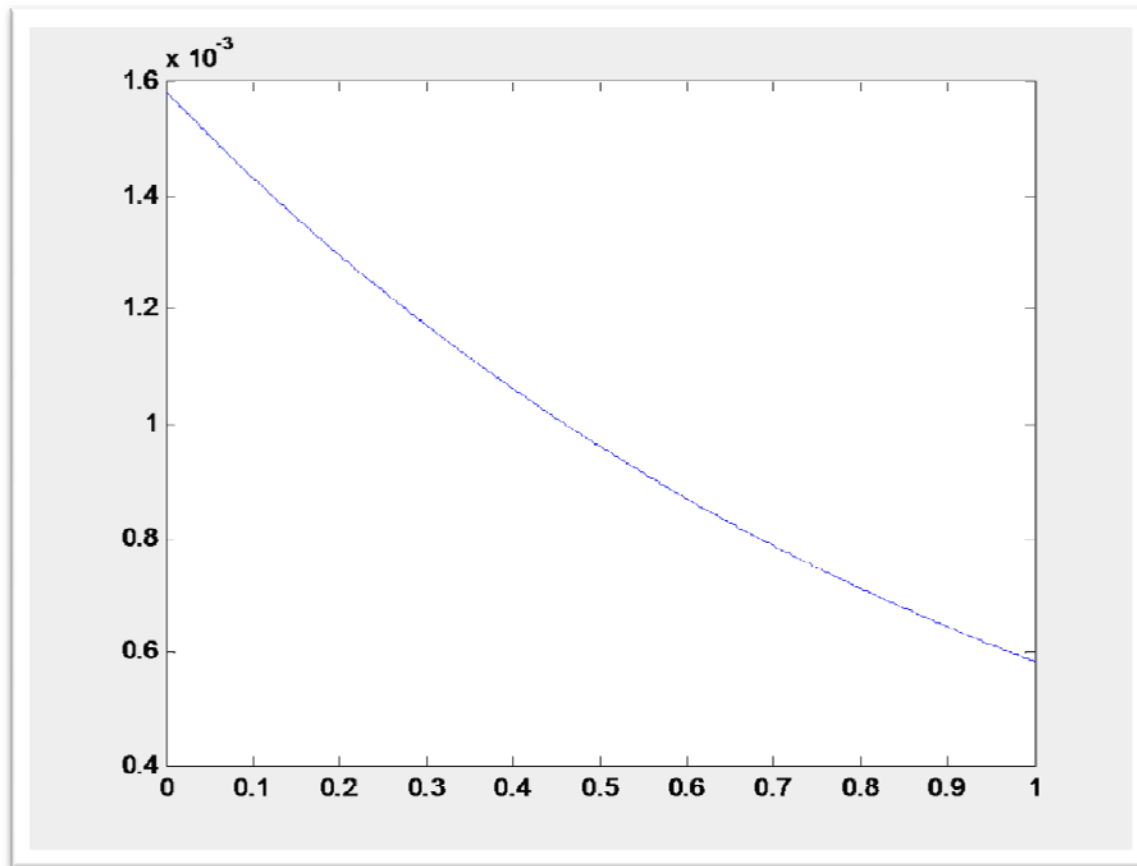
```
%% Convolution with Exponential Decay
%Generate Exponential Decay Signals
eds = exp(-time);
eds = eds/sum(eds);
neds = exp(-ntime);
neds = neds/sum(neds);
ueds = exp(-utime);
ueds = ueds/sum(ueds);

%Convolve Square Wave Signals with Exponential Decay Signals
squareWaveEDS = conv(squareWavePos,eds);
nsquareWaveNEDS = conv(nsquareWavePos,neds);
usquareWaveUEDS = conv(usquareWavePos,ueds);

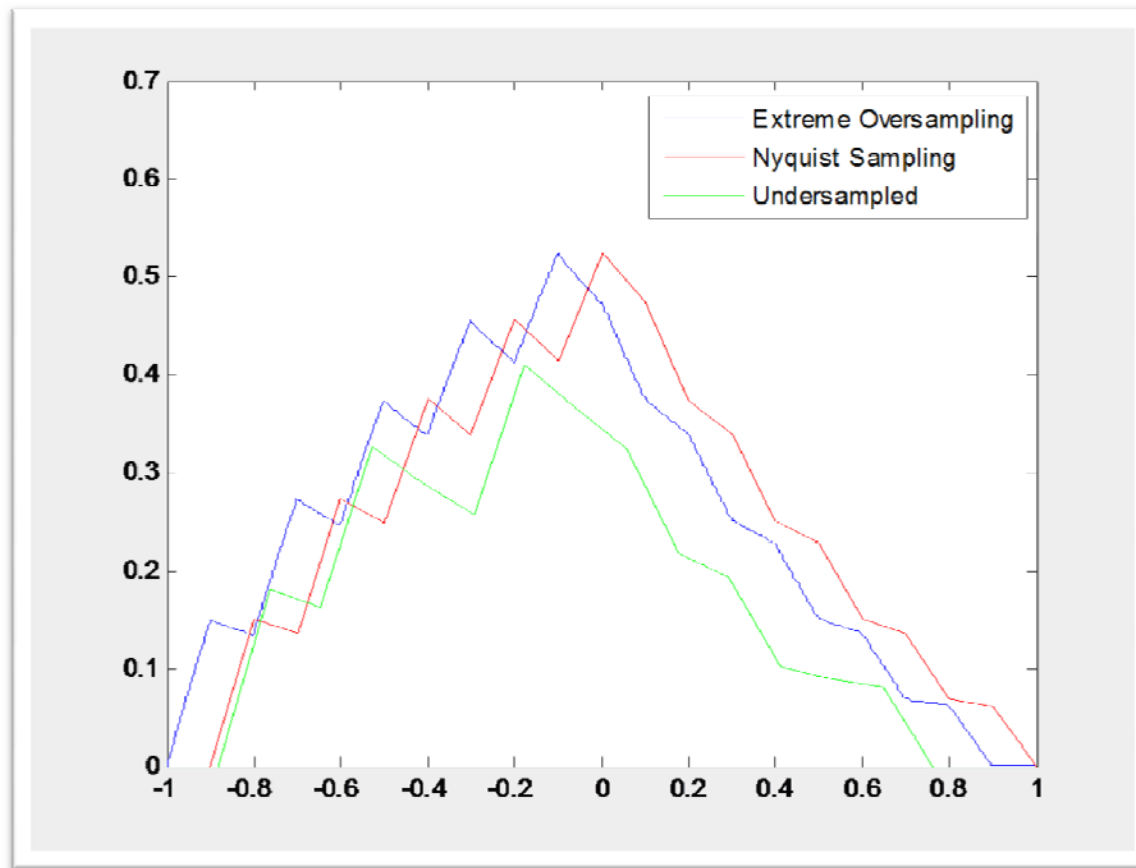
%Pad convolved signals with leading and trailing zero
squareWaveEDS = [0,squareWaveEDS,0];
nsquareWaveNEDS = [0,nsquareWaveNEDS,0];
usquareWaveUEDS = [0,usquareWaveUEDS,0,0];

figure(4), plot(convtime,squareWaveEDS,'b',nconvtime,nsquareWaveNEDS,'r',uconvtime,usquareWaveUEDS,'g')
legend('Extreme Oversampling','Nyquist Sampling','Undersampled')
```

Exponential Decay Signal



Plot of Convolutions



FFT of Signals in MATLAB

Robert Francis

August 31, 2011

Review of Fourier Transform

The Fourier Integral $\rightarrow$
$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

DFT (Discrete Fourier Transform)

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N}, k = 1, 2, \dots, N$$