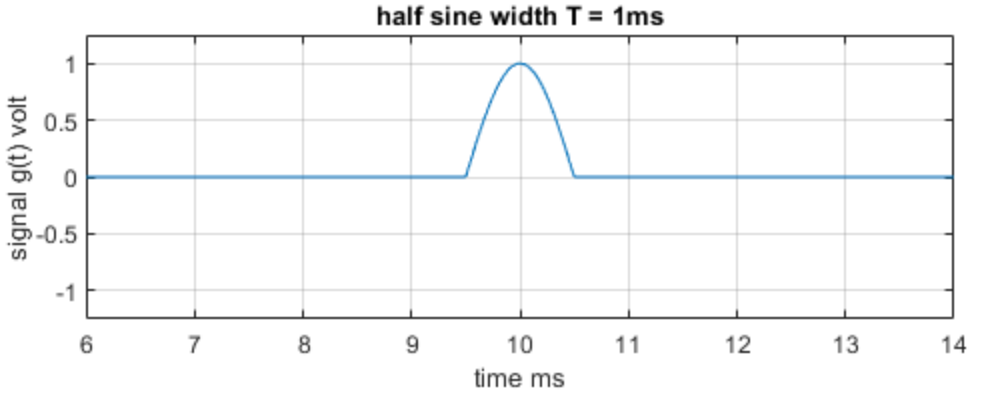
Kevin Mahabeer | ECET 480 | Homework 2

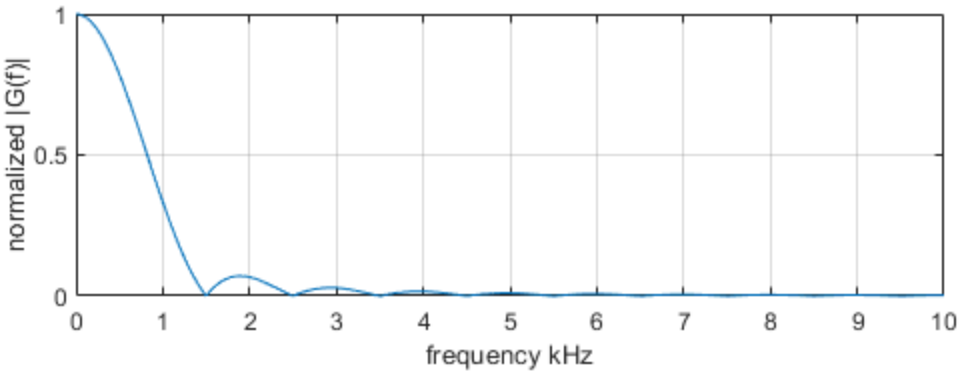
Table of Contents

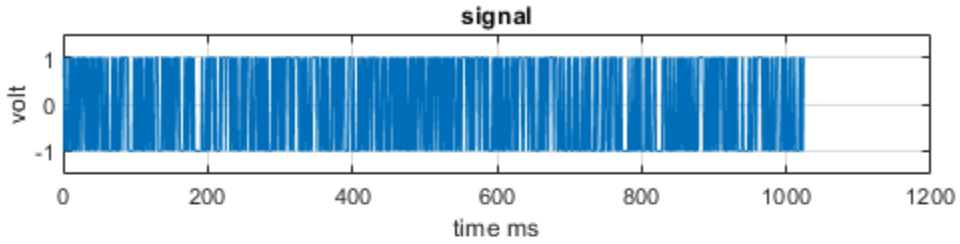
1.

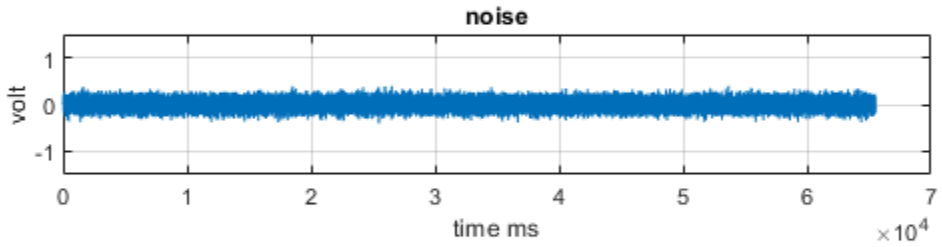
```
clear; close all; clc

%{
Generate a binary random signal (+-1) with 1024 bits.
%}

Nb = 1024;
binarySequence = randi([0, 1], 1, Nb);
binarySequence(binarySequence == 0) = -1;

% Pulse shape (Page 6 from handout)
%{
The basic pulse shape is given (you will be provided with pulse
 shapes).
%}

T = 1e-3;           % pulse duration
fs = 64/T;          % sampling frequency
Ts = 1/fs;          % sampling period
N = 4096*4;         % number of points used for N-pt DFT
t = (0:N-1)*Ts;     % N time samples

T0 = 2*T;           % fundamental period of sine wave
f0 = 1/T0;          % fundamental frequency of sine wave

x = sin((2*pi*f0*t)+(0.5*pi)).*rectpuls(t-10*T,T);

figure('Name','Pulse Shape');
subplot(211)
plot(t/T,x)
grid on; xlim([6 14]); ylim([-1.25 1.25]);
title('half sine width T = 1ms');
xlabel('time ms'); ylabel('signal g(t) volt');

%{
Obtain its PSD.
%}
```

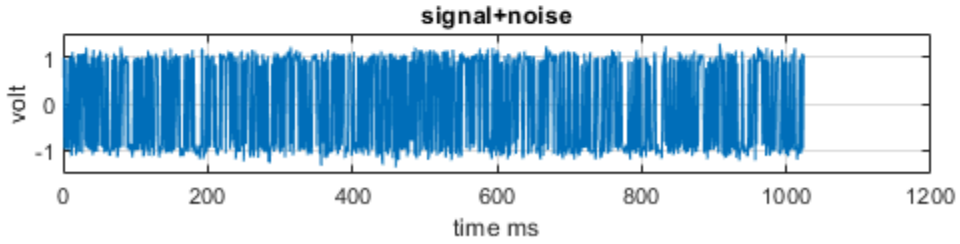
```
Ws = 2.*pi/Ts;
FB = fft(x);
FBP = FB(1:N/2+1)*Ts;
WW = Ws*(0:N/2.)/N;
WF = (1/(2*pi))*WW;
FB = FBP/max(abs(FBP));

subplot(212)
plot(WF*1e-3, abs(FB))
grid on; xlim([0 10]);
xlabel('frequency kHz'); ylabel('normalized |G(f)|');

% Noise to signal
%{
Add white Gaussian noise.
%}
noiseSample = linspace(0,N,N*4);
nt = random('normal',0,0.1,[1, length(noiseSample)]);

xNoise = size(binarySequence);

for i=1:length(binarySequence)
%     binaryVector = binarySequence.*x;
    xNoise(i) = binarySequence(i)+nt(i);
%     xNoise = binarySequence + nt;
end

figure;

subplot(311)
plot(binarySequence)
grid on; ylim([-1.5 1.5]);
xlabel('time ms'); ylabel('volt');
title('signal');

subplot(312)
plot(nt);
grid on; ylim([-1.5 1.5]);
xlabel('time ms'); ylabel('volt');
title('noise');

subplot(313)
plot(xNoise);
grid on; ylim([-1.5 1.5]);
xlabel('time ms'); ylabel('volt');
title('signal+noise');
```


half sine width T = 1ms
signal g(t) volt
time ms


normalized |G(f)|
frequency kHz

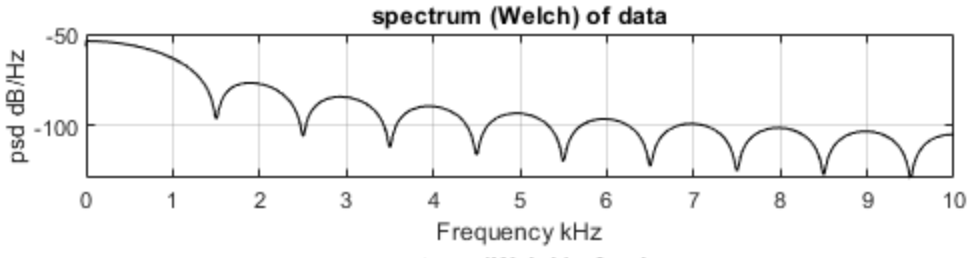

signal
volt
time ms

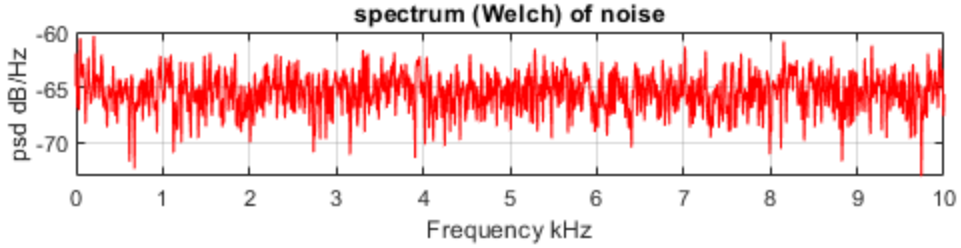

noise
volt
time ms
×10^4

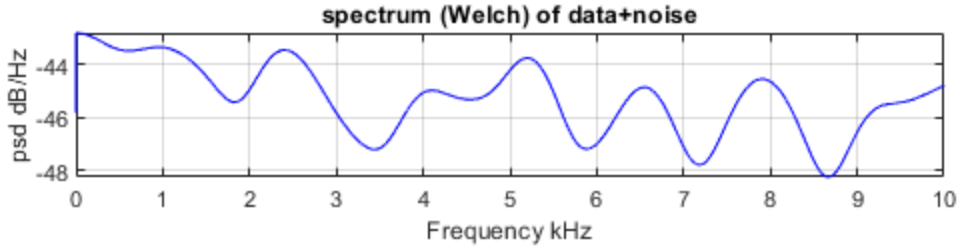

signal+noise
volt
time ms

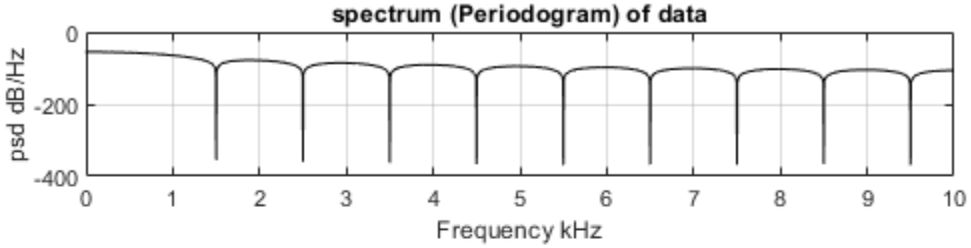
1. Welch

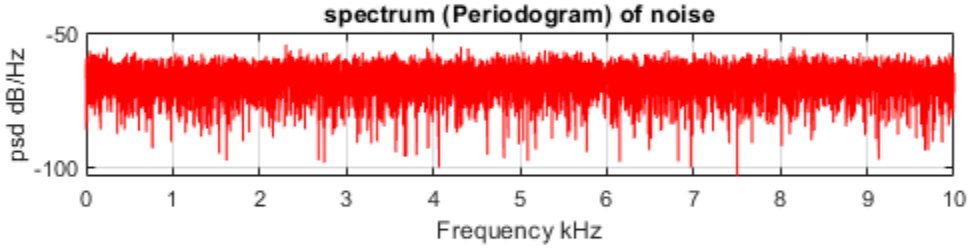
```
[pxxn1, W1] = pwelch(x,[],0,length(x),fs);
[pxxn2, W2] = pwelch(nt,[],0,length(nt),fs);
[pxxn3, W3] = pwelch(xNoise,[],0,length(nt),fs);

figure;

subplot(311)
plot(W1*1e-3,10*log10(pxxn1),'k')
title('spectrum (Welch) of data');
grid on; xlim([0 10]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');

subplot(312)
plot(W2*1e-3,10*log10(pxxn2),'r')
title('spectrum (Welch) of noise');
grid on; xlim([0 10]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');

subplot(313)
plot(W3*1e-3,10*log10(pxxn3),'b')
title('spectrum (Welch) of data+noise');
grid on; xlim([0 10]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');

% 1. Periodogram
[pxxn4, W4] = periodogram(x,[],[],fs);
[pxxn5, W5] = periodogram(nt,[],[],fs);
[pxxn6, W6] = periodogram(xNoise,[],[],fs);

figure;
subplot(311)
plot(W4*1e-3,10*log10(pxxn4),'k')
title('spectrum (Periodogram) of data');
grid on; xlim([0 10]); %ylim([-100, -50]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');

subplot(312)
plot(W5*1e-3,10*log10(pxxn5),'r')
title('spectrum (Periodogram) of noise');
grid on; xlim([0 10]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');

subplot(313)
plot(W6*1e-3,10*log10(pxxn6),'b')
title('spectrum (Periodogram) of data+noise');
grid on; xlim([0 10]); %ylim([-100 0]);
xlabel('Frequency kHz'); ylabel('psd dB/Hz');
```

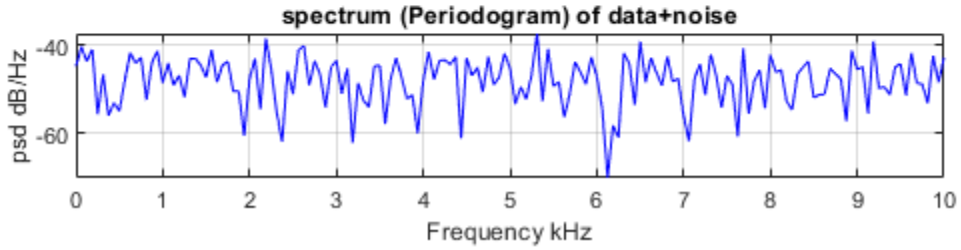

spectrum (Welch) of data
psd dB/Hz
-50
-100
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz


spectrum (Welch) of noise
psd dB/Hz
-60
-65
-70
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz


spectrum (Welch) of data+noise
psd dB/Hz
-44
-46
-48
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz


spectrum (Periodogram) of data
psd dB/Hz
0
-200
-400
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz


spectrum (Periodogram) of noise
psd dB/Hz
-50
-100
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz


spectrum (Periodogram) of data+noise
psd dB/Hz
-40
-60
0 1 2 3 4 5 6 7 8 9 10
Frequency kHz

noise = 0.1 * randn(size(x)); xNoise = x + noise;

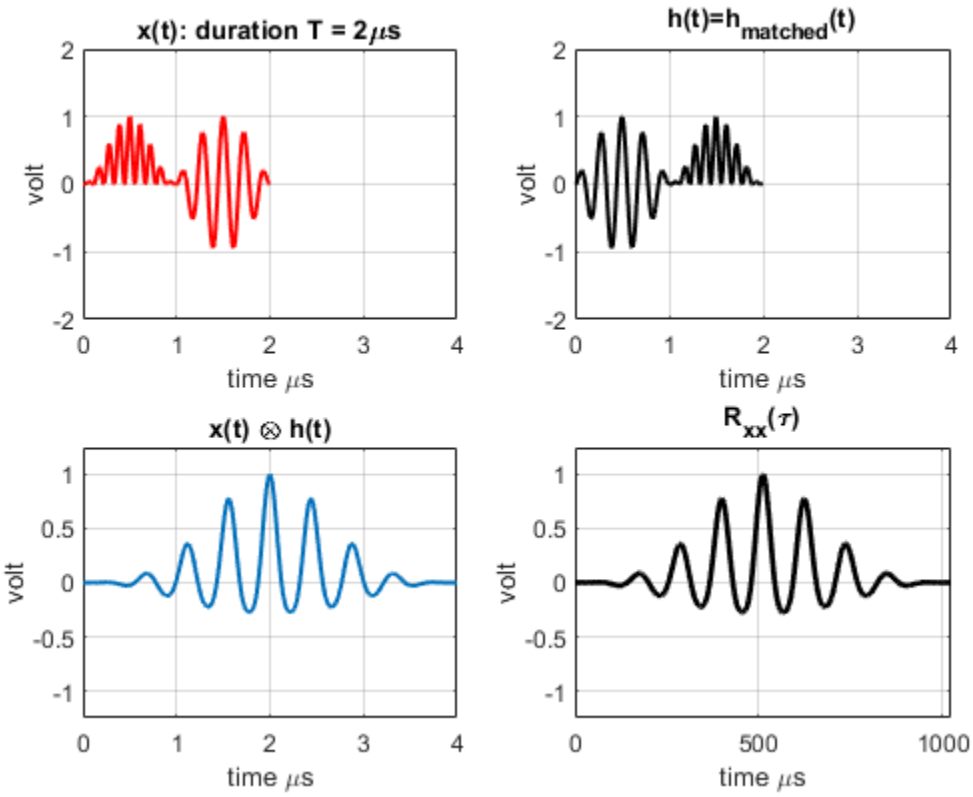
```
% figure('Name','Pulse Shape with Noise');
% subplot(211)
% plot(xNoise)
% grid on; xlim([6 14]); ylim([-1.25 1.25]);
% title('half sine width T = 1ms');
% xlabel('time ms'); ylabel('signal g(t) volt');


%{
Obtain the PSD of the noise and the signal plus noise.
%}

% Ws = 2.*pi/Ts;
% FB = fft(xNoise);
% FBP = FB(1:N/2+1)*Ts;
% WW = Ws*(0:N/2.)/N;
% WF = (1/(2*pi))*WW;
% FB = FBP/max(abs(FBP));
%
% subplot(212)
% plot(WF*1e-3, abs(FB))
% grid on; xlim([0 10]);
% xlabel('frequency kHz'); ylabel('normalized |G(f)|');
%
% %
%
% bits = length(binarySequence);
% x2 = sin((2*pi*f0*t)+(0.5*pi));
%
% for i = 1:length(binarySequence)
%     if i ~= length(binarySequence)
%         bits(i) = x2(i+1)*binarySequence(i);
%     else
%         continue
%     end
%
% end
```

2.

```
clear; close all; clc;

%{
You are given a message signal. Obtain a matched filter corresponding
 to
this signal. Obtain the output of the matched filter with the message
 as
the input. Obtain the autocorrelation of the input and compare the
autocorrelation with the output of the matched filter.
%}
```

```
% T = 1e-6;       % pulse duration
% N = 1024;       % number of points used for N-pt DFT
% fs = 64/T;      % sampling frequency
% Ts = 1/fs;      % sampling period
% t = (0:N-1)*Ts; % N time samples
% t2 = linspace(0,2,N).*Ts;
%
% T0 = 2*T;       % fundamental period of sine wave
% f0 = 1/T0;      % fundamental frequency of sine wave

T = 1e-6;
fs = 100e3;
Ts = 1/fs;
N = 1024;

t = linspace(0,2*T,N);

T0 = 2*T;
f0 = 1/T0;

x1 =
 abs(((sin(18*pi*f0*t).*sin((2*pi*f0*t))).*rectpuls(t-0.5*T,T)).^2);
x2 = (sin(18*pi*f0*t).*sin((2*pi*f0*t))).*rectpuls(t-1.5*T,T);
x3 = x1+x2;


figure;

% Original Symbol
subplot(221)
plot(t/T,x3,'Linewidth',1.5,'color','r')
grid on; xlim([0 4]); ylim([-2 2]);
title('x(t): duration T = 2\mus');
xlabel('time \mus'); ylabel('volt');


% Matched Filter
h = flip(x3);
subplot(222)
plot(t/T,h,'Linewidth',1.5,'color','k')
grid on; xlim([0 4]); ylim([-2 2]);
title('h(t)=h_{matched}(t)');
xlabel('time \mus'); ylabel('volt');

% Convolution
w = conv(x3, h);
% w = filter(h,1,x3);
w2 = w(N/2+1:N+N/2);

subplot(223)
plot(2*t/T,(1/200*w2),'Linewidth',1.5)
% plot(t/T,1/200*w((N/2-1):(N/2-1)+(N-1)),'Linewidth',1.5)
grid on; ylim([-1.25 1.25]); xlim([0 4]); %ylim([-1.25 1.25]);
title('x(t) \otimes h(t)');
```

```
xlabel('time \mus'); ylabel('volt');

% Correlation
[corx, lags] = xcorr(x3);
corx2 = corx(N/2+1:N/2+N);

subplot(224)
plot((1/200*corx2),'Linewidth',2,'color','k')
%plot(t/T,(1/200)*(corx),'Linewidth',2,'color','k')
grid on; ylim([-1.25 1.25]); %xlim([-N/2 N/2]);
title('R_{xx}(\tau)');
xlabel('time \mus'); ylabel('volt');
```



Published with MATLAB® R2019b