Noah Choe
179008726

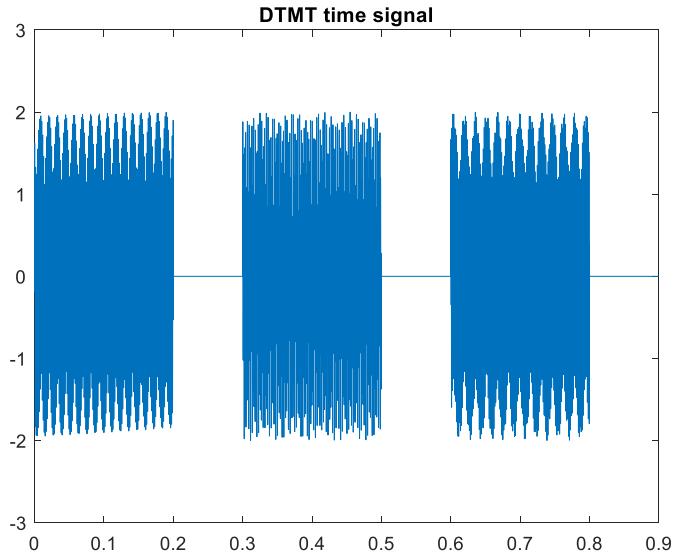
Lab 3

Problem 1:

The purpose of problem 1 is to find the 3 keys/digits corresponding to our RUID. I saw that the blanking interval does not affect the result of decoding a signal. This is because where there is blanking, the values are zero so the frequency does not change. I had to convert between the frequencies from Hz to radians per sample in order to get the correct results for this problem.

Problem 1.1:

```
y = dtmfsig(179008726);
x = linspace(0, 0.9, 7200);

figure;
plot(x, y);
hold on;
xlim([0, 0.9]);
ylim([-3, 3]);
title("DTMT time signal");
hold off;
```



Problem 2.2:

```
fL = [697, 770, 852, 941];
fH = [1209, 1336, 1477];
fs = 8000;

fL = (2*pi*fL)/fs;
fH = (2*pi*fH)/fs;

first = y(1:2400);
second = y(2401:4800);
third = y(4801:7200);

fL1 = abs(freqz(first, 1, fL));
display(fL1);
fH1 = abs(freqz(first, 1, fH));
display(fH1);

fL2 = abs(freqz(second, 1, fL));
display(fL2);
fH2 = abs(freqz(second, 1, fH));
display(fH2);

fL3 = abs(freqz(third, 1, fL));
display(fL3);
fH3 = abs(freqz(third, 1, fH));
display(fH3);

first = y(1:1600);
second = y(2401:4000);
third = y(4801:6400);
```

```
fL1 = abs(freqz(first, 1, fL));
display(fL1);
fH1 = abs(freqz(first, 1, fH));
display(fH1);

fL2 = abs(freqz(second, 1, fL));
display(fL2);
fH2 = abs(freqz(second, 1, fH));
display(fH2);

fL3 = abs(freqz(third, 1, fL));
display(fL3);
fH3 = abs(freqz(third, 1, fH));
display(fH3);
```

With Blanking:

```
fL1 =
  800.2524   14.5731    0.8751    1.6369

fH1 =
    2.2222    4.2074  799.9657

fL2 =
   19.0534  798.8779   11.7836    3.0972

fH2 =
  798.5292   10.4165    5.8453

fL3 =
   17.6748  800.3964   13.7706    4.3757
```

fH3 =

4.2039 5.5850 800.3260

Without Blanking:

fL1 =

800.2524 14.5731 0.8751 1.6369

fH1 =

2.2222 4.2074 799.9657

fL2 =

19.0534 798.8779 11.7836 3.0972

fH2 =

798.5292 10.4165 5.8453

fL3 =

17.6748 800.3964 13.7706 4.3757

fH3 =

4.2039 5.5850 800.3260

We see that with or without blanking does not matter and that the results are the same.

fL1 → The highest number is the first one which corresponds to 697

fH1 → The highest number is the third one which corresponds to 1477

fL2 → The highest number is the second one which corresponds to 770

fH2 → The highest number is the first one which corresponds to 1336

fL3 → The highest number is the second one which corresponds to 770

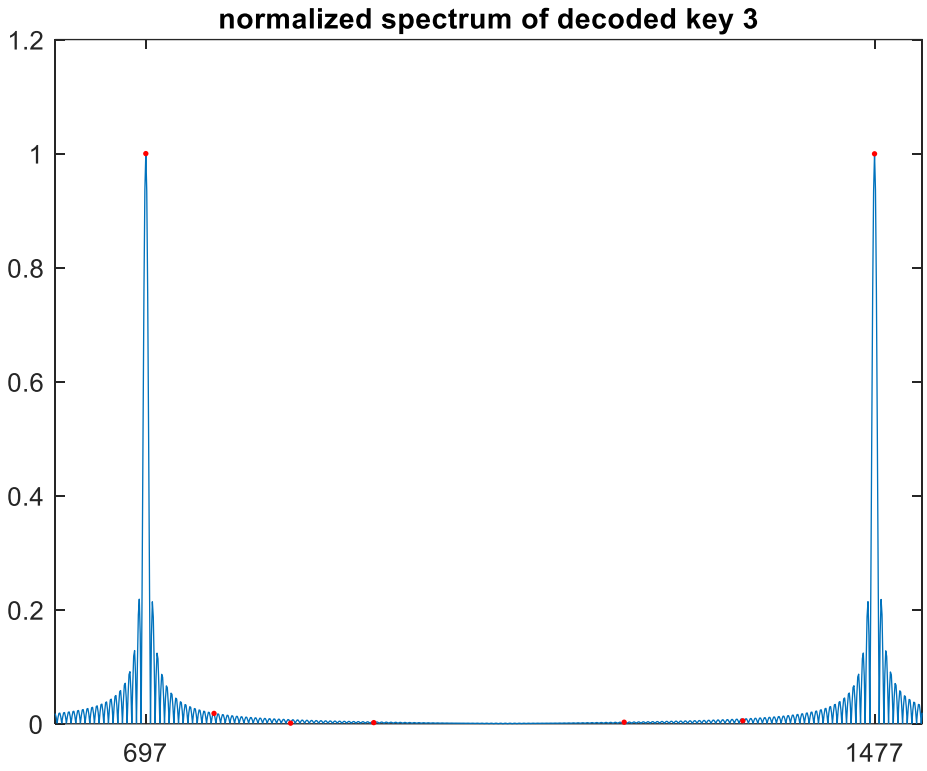
fH3 → The highest number is the third one which corresponds to 1477

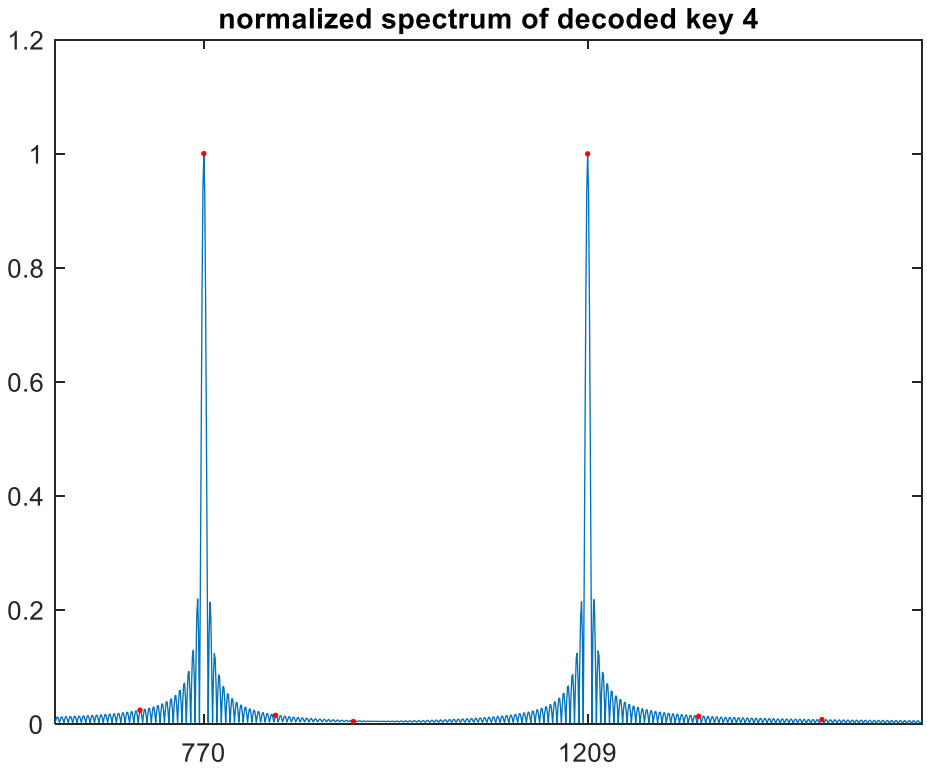
Key1 = 3 Key2 = 5 Key3 = 6

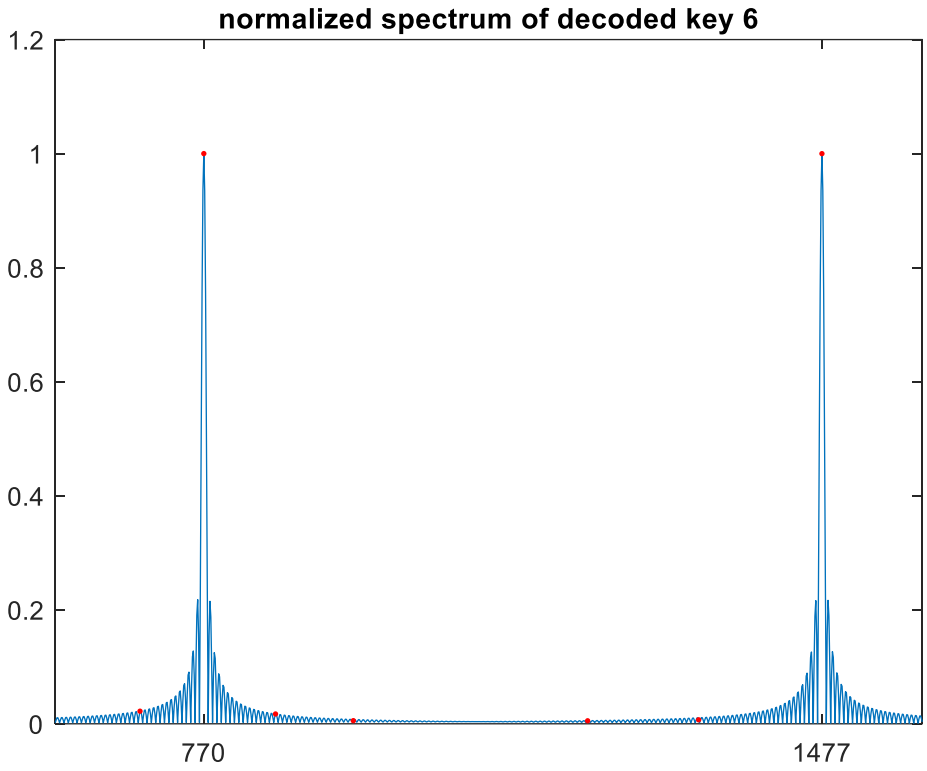
Problem 1.3:

```
f = 600:1600;
f = (2*pi*f)/fs;

%key 3
dtft = abs(freqz(first, 1, f));
dialfreq = [fL, fH];
key3 = [fL1, fH1]/max(dtft);

figure;
plot(f, dtft/max(dtft));
hold on;
plot(dialfreq, key3, 'r.');
title('normalized spectrum of decoded key 3');
ax = gca;
ax.XTick = [697/8000*2*pi 1477/8000*2*pi];
ax.XTickLabel = {'697' , '1477'};
xlim([0.4712 1.2]);
ylim([0 1.2]);
hold off;

%key 4
dtft = abs(freqz(second, 1, f));
dialfreq = [fL, fH];
key4 = [fL2, fH2]/max(dtft);

figure;
plot(f, dtft/max(dtft));
hold on;
```

```
plot(dialfreq, key4, 'r.');
title('normalized spectrum of decoded key 4');
ax = gca;
ax.XTick = [770/8000*2*pi 1209/8000*2*pi];
ax.XTickLabel = {'770' , '1209'};
xlim([0.4712 1.25]);
ylim([0 1.2]);
hold off;

%key 6
dtft = abs(freqz(third, 1, f));
dialfreq = [fL, fH];
key6 = [fL3, fH3]/max(dtft);

figure;
plot(f, dtft/max(dtft));
hold on;
plot(dialfreq, key6, 'r.');
title('normalized spectrum of decoded key 6');
ax = gca;
ax.XTick = [770/8000*2*pi 1477/8000*2*pi];
ax.XTickLabel = {'770' , '1477'};
xlim([0.4712 1.25]);
ylim([0 1.2]);
hold off;
```


normalized spectrum of decoded key 3
1.2
1
0.8
0.6
0.4
0.2
0
697
1477


normalized spectrum of decoded key 4
1.2
1
0.8
0.6
0.4
0.2
0
770
1209



Problem 1.4:

```
f = [697 770 852 941 1209 1336 1477];
Table = [f; key3; key4; key6];
fprintf('   f  |   key 3   key 4   key 6\n');
fprintf('------|---------------------------\n');
fprintf(' %4i | %7.3f %7.3f %7.3f\n', Table);
```

f	key 3	key 4	key 6
697	1.000	0.024	0.022
770	0.018	1.000	1.000
852	0.001	0.015	0.017
941	0.002	0.004	0.005
1209	0.003	1.000	0.005
1336	0.005	0.013	0.007
1477	1.000	0.007	1.000

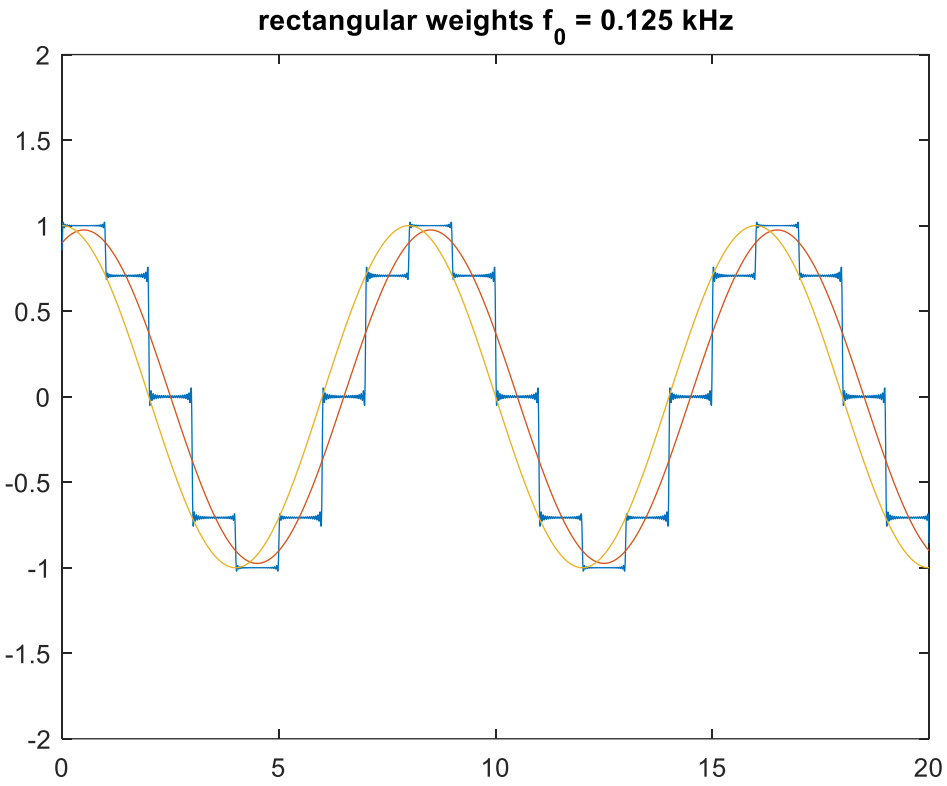
Problem 2:

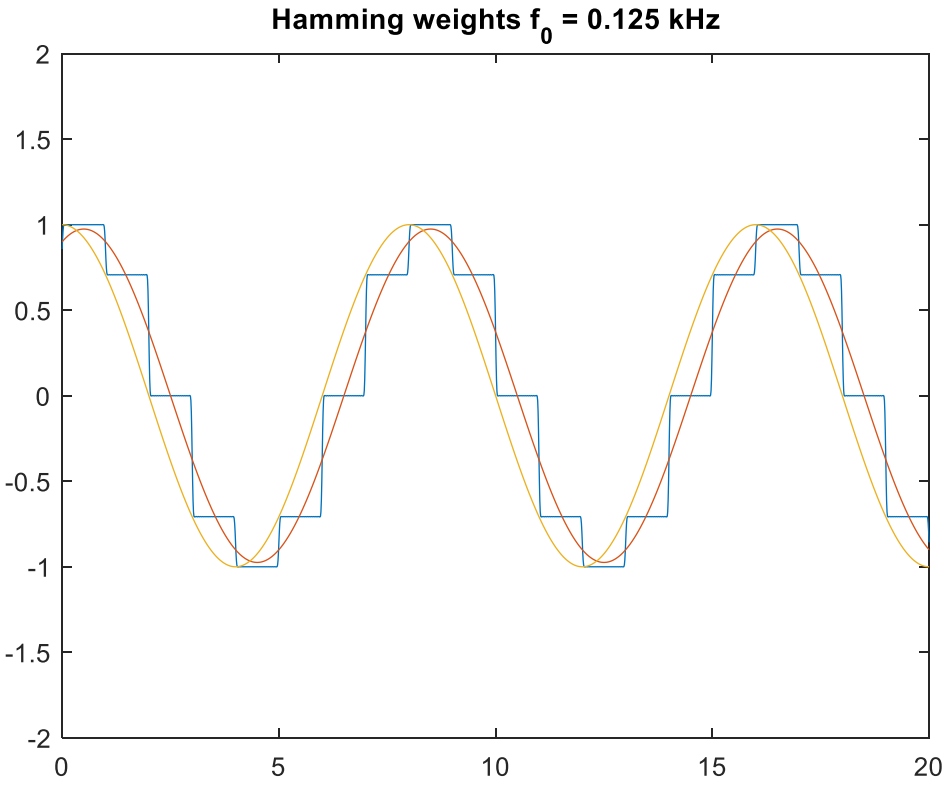
The purpose of problem 2 is to see the change in the fundamental frequency after it has been through the discrete domain. I saw that the hamming weights had less noise when rectangular weight signals were used. Also, I tried to create an ideal reconstruction using the Butterworth analog lowpass filter. I also saw the differences of each frequency when put through different steps. The reconstruction stage showed very similar graphs for both frequencies.

Problem 2.1:

```
f0 = 0.125;
fs = 1;
M = -20:1:20;
fm = f0 + M*fs;
t = 0:.01:20;

wm = 0.54 + 0.46*cos((pi/20)*M);
[tf, F] = meshgrid(t, fm);
[tw, W] = meshgrid(t, wm);
G = @(f) sin((pi*f)/fs)./((pi*f)/fs);

xa = cos(2*pi*f0*t);
xr = sum(G(F).*cos(2*pi*tf.*F-pi*F));
xh = sum(W.*G(F).*cos(2*pi*tw.*F-pi*F));
```

```
xp = G(f0)*cos(2*pi*f0*t-pi*f0);

figure;
plot(t, xr, t, xp, t, xa);
hold on;
title('rectangular weights f_0 = 0.125 kHz');
xlim([0 20]);
ylim([-2 2]);
hold off;

figure;
plot(t, xh, t, xp, t, xa);
hold on;
title('Hamming weights f_0 = 0.125 kHz');
xlim([0 20]);
ylim([-2 2]);
hold off;

att = xp/xa;
attdB = -10*log10(att);
display(attdB);

[c, i] = max(xp);
[d, j] = max(xa);
phase = i-j;
display(phase);
```


rectangular weights f0 = 0.125 kHz
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
5
10
15
20


Hamming weights f0 = 0.125 kHz
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
5
10
15
20

Attenuation in dB =

0.4560

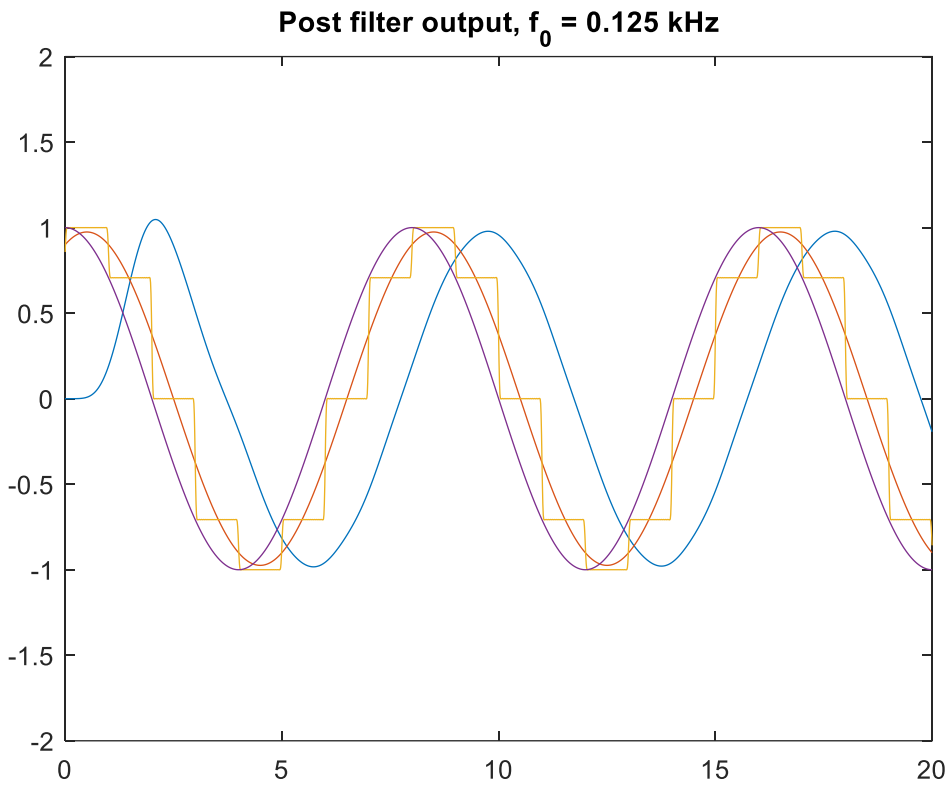
Phase Delay =

50

We see that the attenuation and the phase delay is consistent with the graphs.

Problem 2.2:

```
f3dB = fs/2;
[b, a] = butter(6, 2*pi*f3dB, 's');
xf = lsim(b, a, xh, t);

figure;
plot(t, xf, t, xp, t, xh, t, xa);
hold on;
title('Post filter output, f_0 = 0.125 kHz')
xlim([0 20]);
ylim([-2 2]);
hold off;

[maxxp, xpindex] = max(xp(800:1000));
[maxxf, xfindex] = max(xf(800:1000));
delay = (xfindex-xpindex)/100;
display(delay);

Hpost = @(g) polyval(b, 2*pi*1i*g)./polyval(a, 2*pi*1i*g);
exactdelay = (-atan2(imag(Hpost(f0)),real(Hpost(f0))))/(2*pi*f0);
display(exactdelay);
```



delay =

1.2600

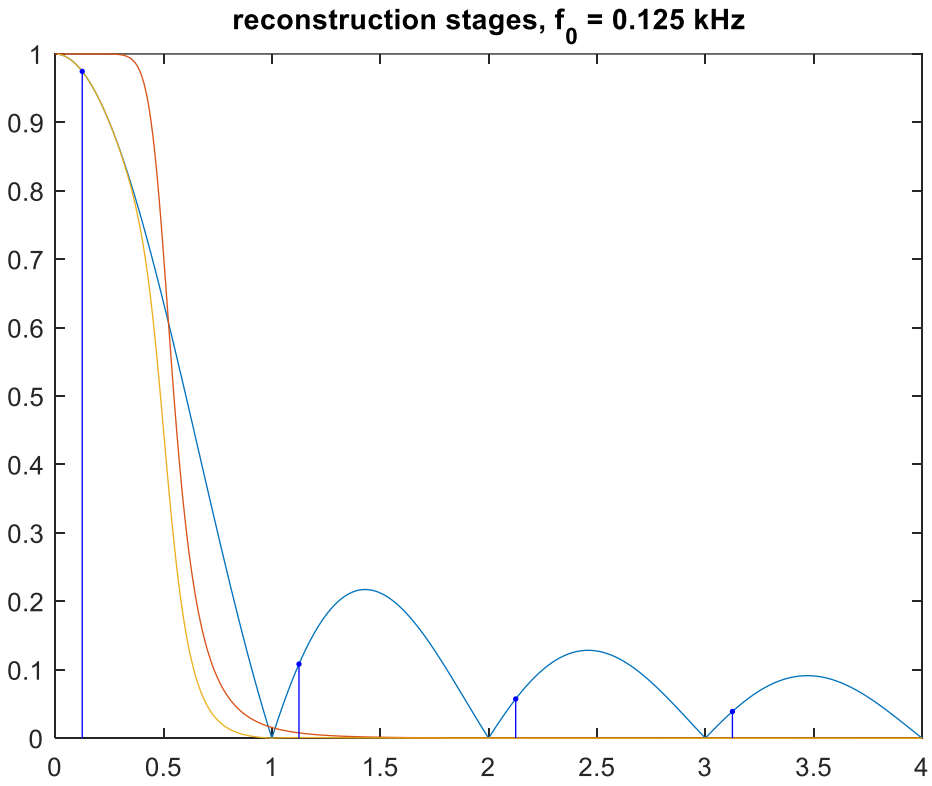
Exact delay =

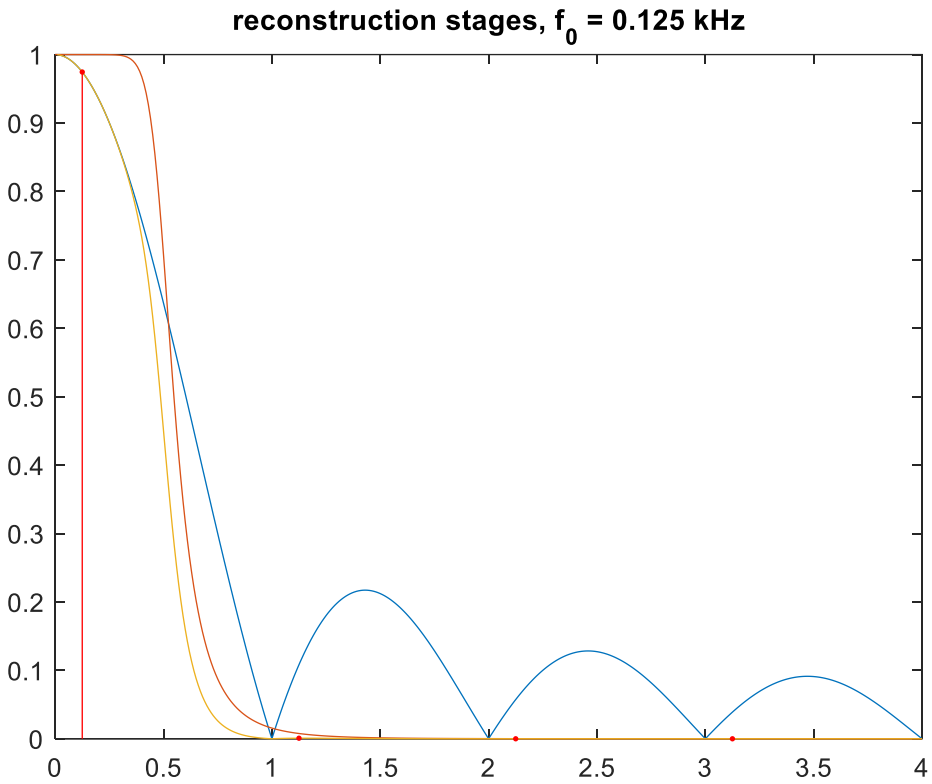
1.2395

Problem 2.3:

```
M = 0:1:3;
f = 0:0.01:4;
fm = f0 + M*fs;

figure;
plot(f, abs(G(f)), f, abs(Hpost(f)), f, abs(G(f)).*abs(Hpost(f)));
hold on;
stem(fm,abs(G(fm)),'b.');
title('reconstruction stages, f_0 = 0.125 kHz')
hold off;
```

```
figure;
plot(f, abs(G(f)), f, abs(Hpost(f)), f, abs(G(f)).*abs(Hpost(f)));
hold on;
stem(fm,abs(G(fm)).*abs(Hpost(fm)),'r.');
title('reconstruction stages, f_0 = 0.125 kHz');
hold off;
```



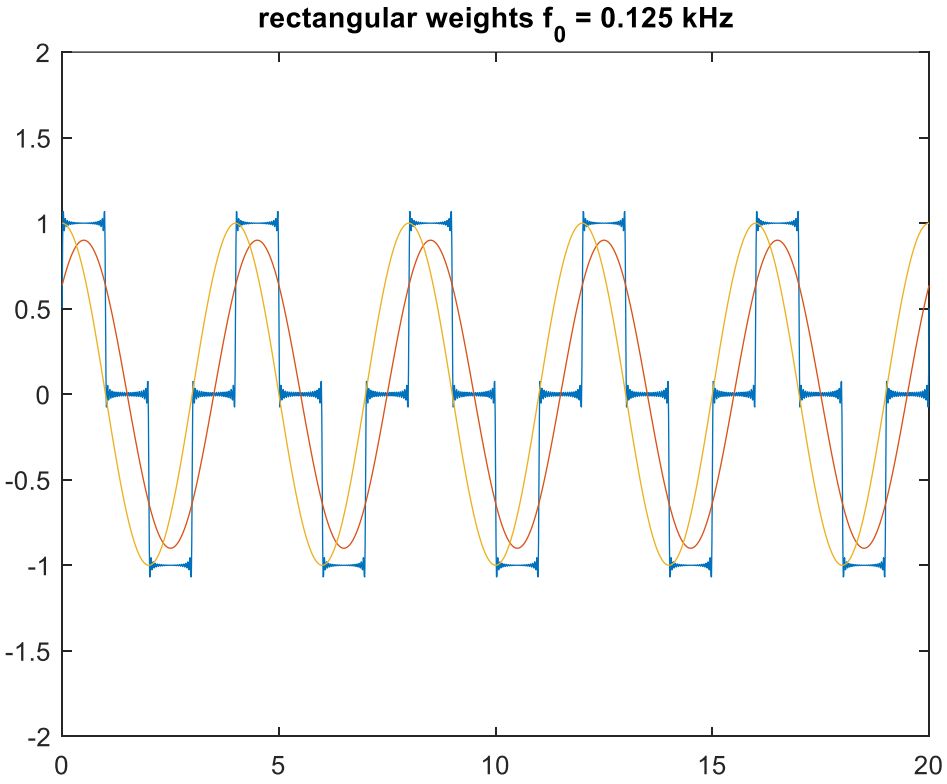


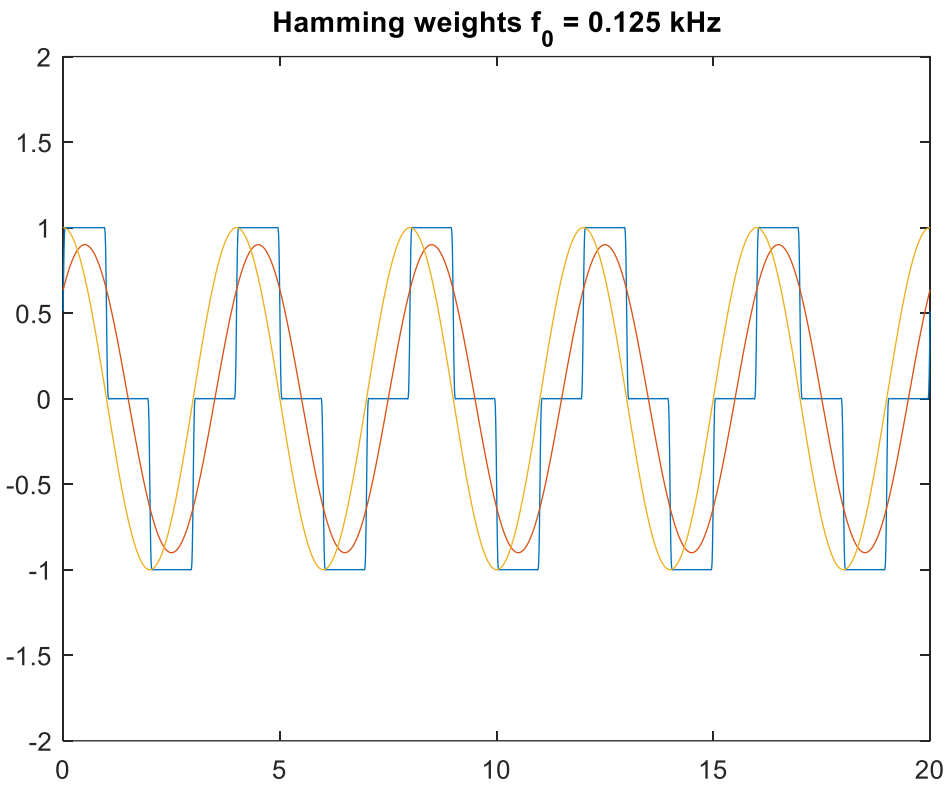
Problem 2.4:

```
f0 = 0.25;
fs = 1;
M = -20:1:20;
fm = f0 + M*fs;
t = 0:.01:20;

wm = 0.54 + 0.46*cos((pi/20)*M);
[tf, F] = meshgrid(t, fm);
[tw, W] = meshgrid(t, wm);
G = @(f) sin((pi*f)/fs)./((pi*f)/fs);

xa = cos(2*pi*f0*t);
xr = sum(G(F).*cos(2*pi*tf.*F-pi*F));
xh = sum(W.*G(F).*cos(2*pi*tw.*F-pi*F));
xp = G(f0)*cos(2*pi*f0*t-pi*f0);
```

```
figure;
plot(t, xr, t, xp, t, xa);
hold on;
title('rectangular weights f_0 = 0.125 kHz');
xlim([0 20]);
ylim([-2 2]);

figure;
plot(t, xh, t, xp, t, xa);
hold on;
title('Hamming weights f_0 = 0.125 kHz');
xlim([0 20]);
ylim([-2 2]);

att = xp/xa;
attdB = -10*log10(att);
display(attdB);

[c, i] = max(xp);
[d, j] = max(xa);
phase = i-j;
display(phase);

f3dB = fs/2;
[b, a] = butter(6, 2*pi*f3dB, 's');
xf = lsim(b, a, xh, t);

figure;
plot(t, xf, t, xp, t, xh, t, xa);
hold on;
```

```
title('Post filter output, f_0 = 0.125 kHz');
xlim([0 20]);
ylim([-2 2]);
hold off;

[maxxp, xpindex] = max(xp(800:1000));
[maxxf, xfindex] = max(xf(800:1000));
delay = (xfindex-xpindex)/100;
display(delay);

Hpost = @(g) polyval(b,2*pi*1i*g)./ polyval(a,2*pi*1i*g);
exactdelay = (-atan2(imag(Hpost(f0)),real(Hpost(f0))))/(2*pi*f0);
display(exactdelay);
```



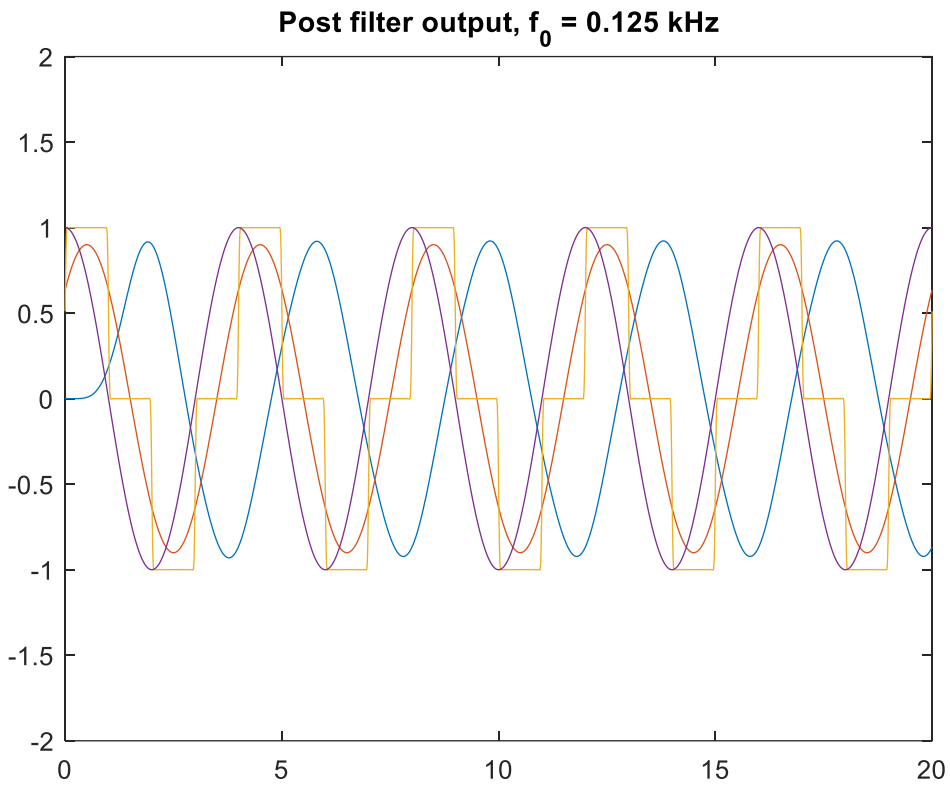


Attenuation in dB =

1.9612

Phase Delay =

50



delay =

1.3000

Exact Delay =

1.2725

Problem 2.5:

```
% For f0 = 0.125
f0 = 0.125;
M =0:3;
fm =@(m) f0 + m.*fs;
fm0 = fm(M);
gm0 = abs(G(fm0));
postf0 = abs(G(fm0).*Hpost(fm0));

% For f0 = 0.25
```

```
f0 = 0.25;
fm =@(m) f0 + m.*fs;
fm1 = fm(M);
gm1 = abs(G(fm1));
postf1 = abs(G(fm1).*Hpost(fm1));
Table = [fm0; fm1; gm0; gm1; postf0; postf1];

fprintf(' fm = f0 + m*fs |      |G(fm)|      |  |G(fm)Hpost(fm)|\n');
fprintf('----------------|------------------|--------------------\n');
fprintf('%7.4f %7.4f | %8.6f %8.6f | %8.6f %8.6f\n', Table);
```

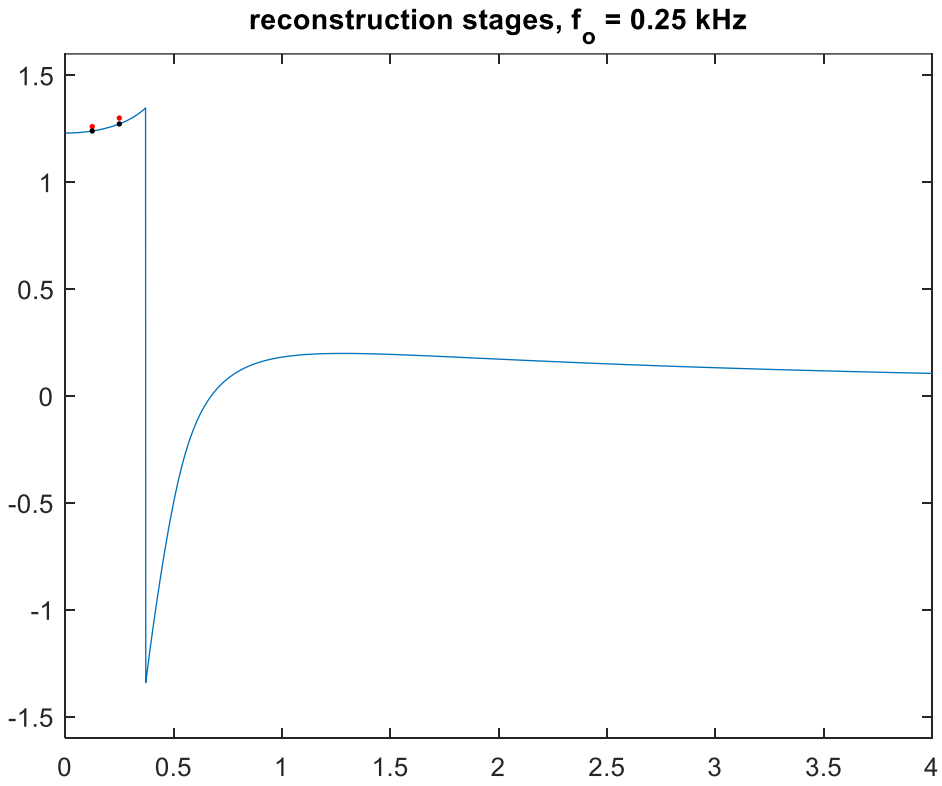
```
fm = f0 + m*fs |      |G(fm)|      |  |G(fm)Hpost(fm)|
----------------|------------------|--------------------
 0.1250  0.2500 | 0.974495 0.900316 | 0.974495 0.900206
 1.1250  1.2500 | 0.108277 0.180063 | 0.000835 0.000738
 2.1250  2.2500 | 0.057323 0.100035 | 0.000010 0.000012
 3.1250  3.2500 | 0.038980 0.069255 | 0.000001 0.000001
```

Problem 2.6:

```
fo = [0.125 .250];
td = [1.26, 1.30];
T0 = [1.2395, 1.2725];
T1 = -angle/(.7854);
T2 = -angle/(1.5708);
f = 0:.001:4;

angle = -atan2(imag(Hpost(f)),real(Hpost(f)))./(2*pi*f);
fprintf('  phase delay exact estimated\n');
fprintf('--------------------------------\n');
fprintf('fO =%6.3f | %6.4f | %6.4f \n', [fo(1:2);T0(1:2); td(1:2)]);
```

```
figure;
plot(f, angle, fo, td, 'r.', fo, T0, 'k.');
hold on;
title('reconstruction stages, f_o = 0.25 kHz');
xlim([0 4]);
ylim([-1.6 1.6]);
```

```
  phase delay exact estimated
--------------------------------
fO = 0.125 | 1.2395 | 1.2600
fO = 0.250 | 1.2725 | 1.3000
```



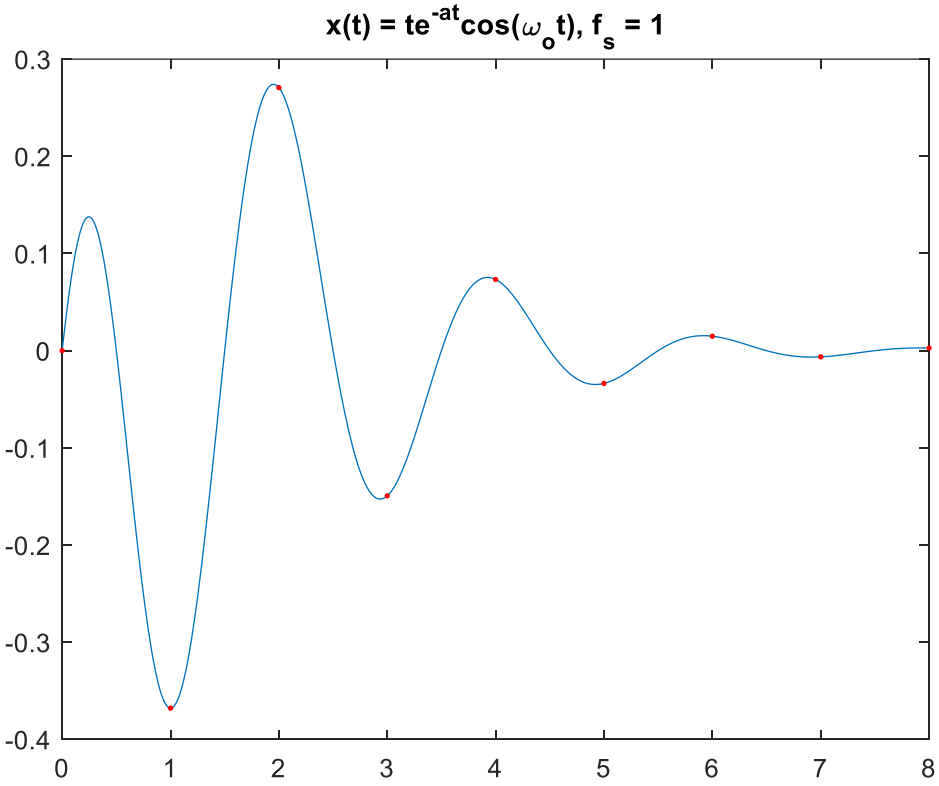
Problem 3:

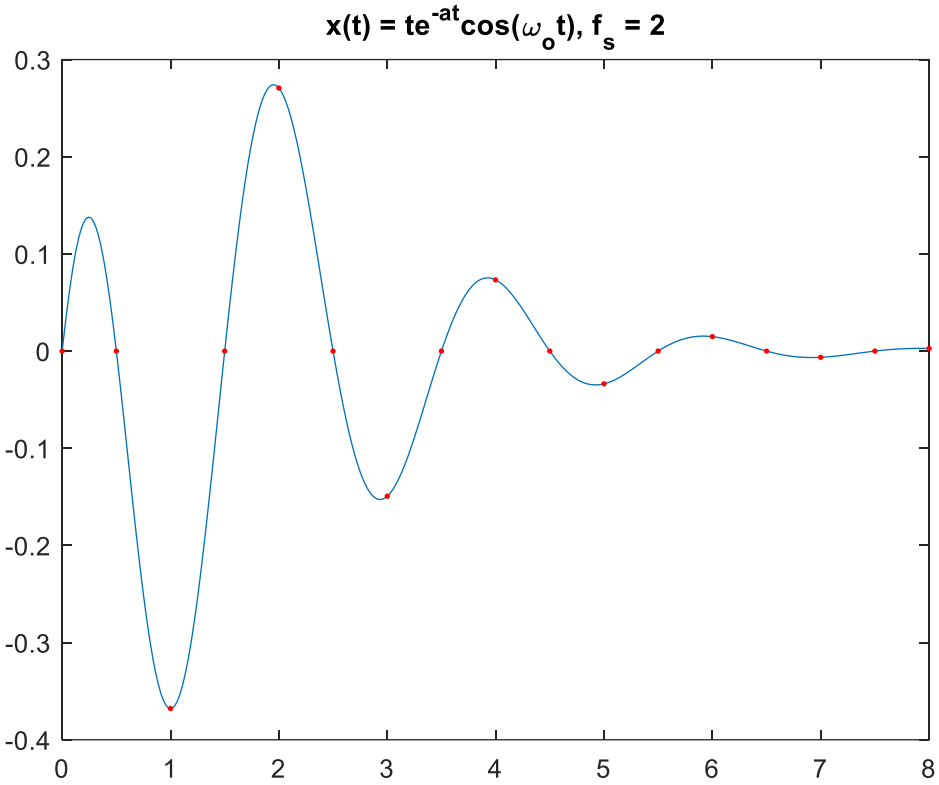
The purpose of problem 3 is to compare X(Ω), T Xd(Ω), and T XM(Ω) for various values of T and M. For the first part of the problem we compare the difference when fs is 1Hz and 2Hz. The graphs look similar, but there are more sample points in fs = 2Hz. For the second part, we compare when the value of fs and M are changing. When fs is increased, the max and min becomes more like that of the analog signal. When M is increased, the replica signal is extended.

Problem 3.1:

```
a = 1;
fo = .5;
fs = 1;
wo = 2*pi*fo;
X = @(t) t.*exp((-a*t)+1i*wo*t);
t = 0:.001:8;
ts = 0:(1/fs):8;

figure;
plot(t, real(X(t)), ts, real(X(ts)), 'r.');
hold on;
title('x(t) = te^{-at}cos(\omega_ot), f_s = 1');
xlim([0 8]);
ylim([-0.4 0.3]);
hold off;

fs = 2;
ts = 0:(1/fs):8;
figure;
plot(t, real(X(t)), ts, real(X(ts)), 'r.');
hold on;
title('x(t) = te^{-at}cos(\omega_ot), f_s = 2');
xlim([0 8]);
```

```
ylim([-0.4 0.3]);
hold off;
```





Problem 3.2:

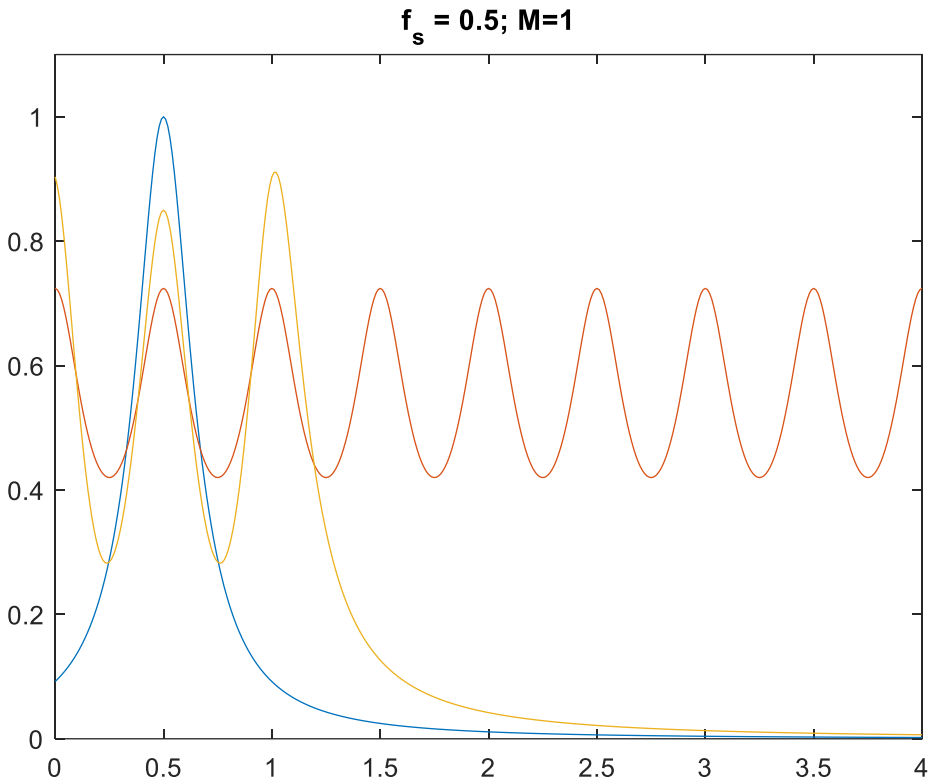
```
f0 = 0.5;
M = -1:1:1;
fs = 0.5;
f = 0:0.01:4;
omega = 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
```

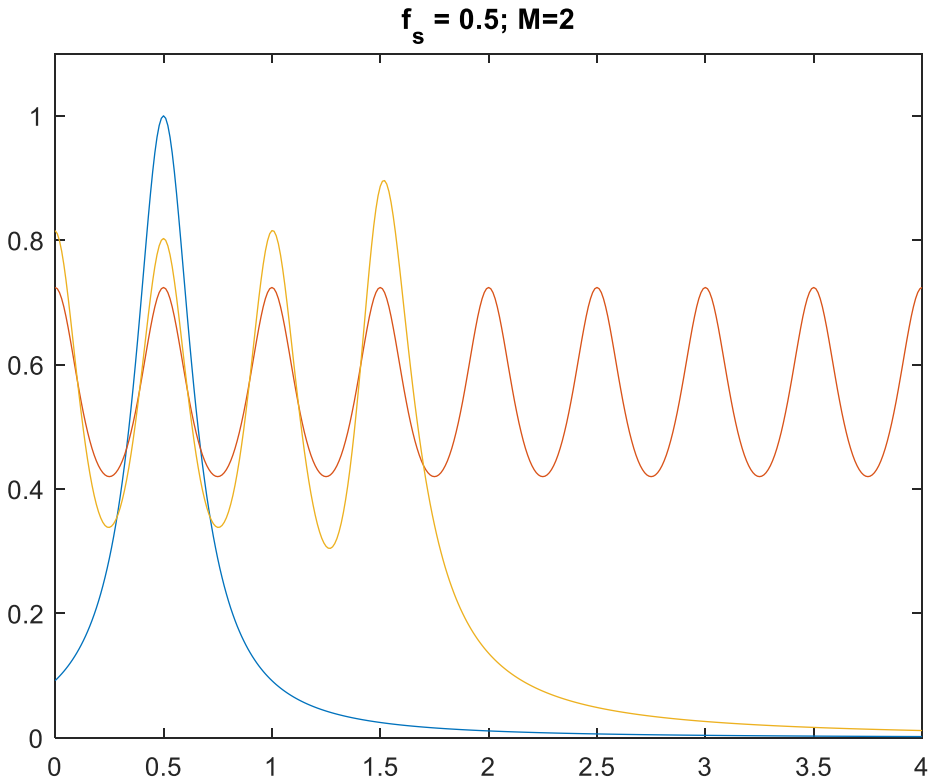
```
hold on;
title('f_s = 0.5; M=1');
xlim([0 4]);
ylim([0 1.1]);
hold off;

%fs = .5, M = 2
f0 = 0.5;
M = -2:1:2;
fs = 0.5;
f = (0:0.01:4);
omega= 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
hold on;
title('f_s = 0.5; M=2');
xlim([0 4]);
ylim([0 1.1]);
hold off;

%fs = 1, M = 1
f0 = 0.5;
M = -1:1:1;
```

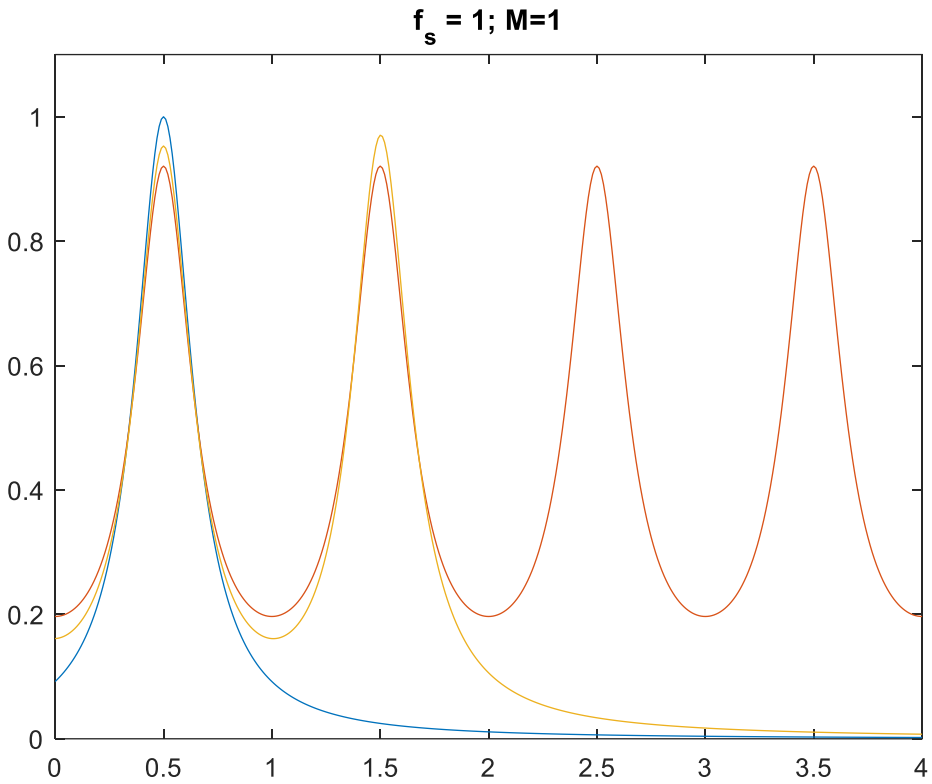
```
fs = 1;
f = (0:0.01:4);
omega= 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
hold on;
title('f_s = 1; M=1');
xlim([0 4]);
ylim([0 1.1]);
hold off;

%fs = 1, M = 2
f0 = 0.5;
M = -2:1:2;
fs = 1;
f = (0:0.01:4);
omega= 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
```

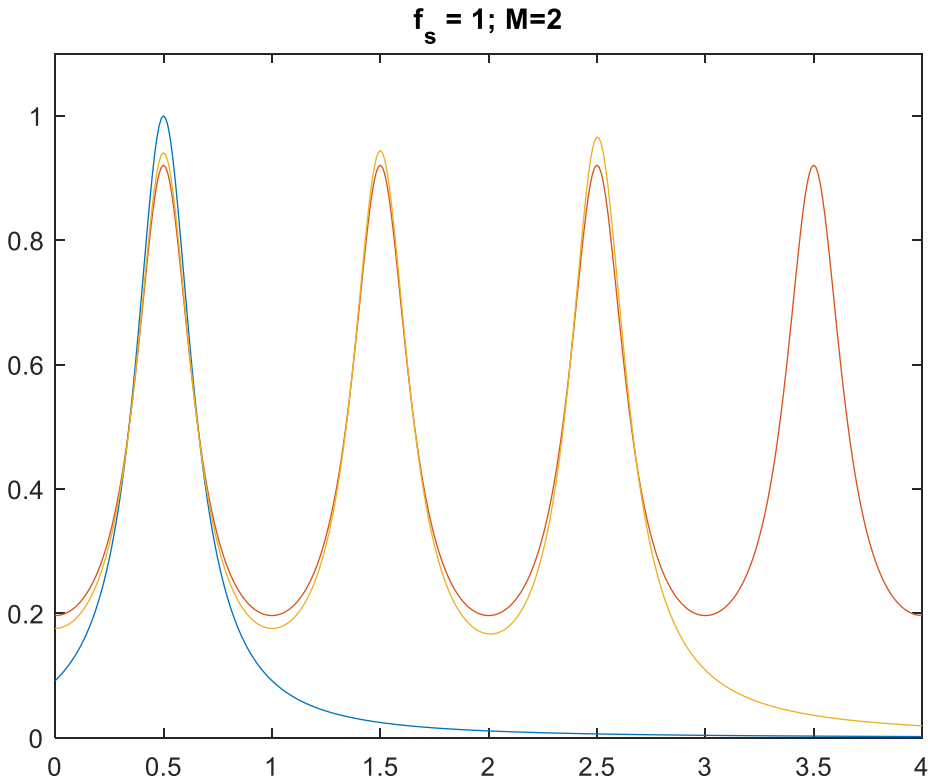
```
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
hold on;
title('f_s = 1; M=2');
xlim([0 4]);
ylim([0 1.1]);
hold off;

%fs = 2, M = 1
f0 = 0.5;
M = -1:1:1;
fs = 2;
f = (0:0.01:4);
omega= 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
hold on;
title('f_s = 2; M=1');
xlim([0 4]);
ylim([0 1.1]);
hold off;
```

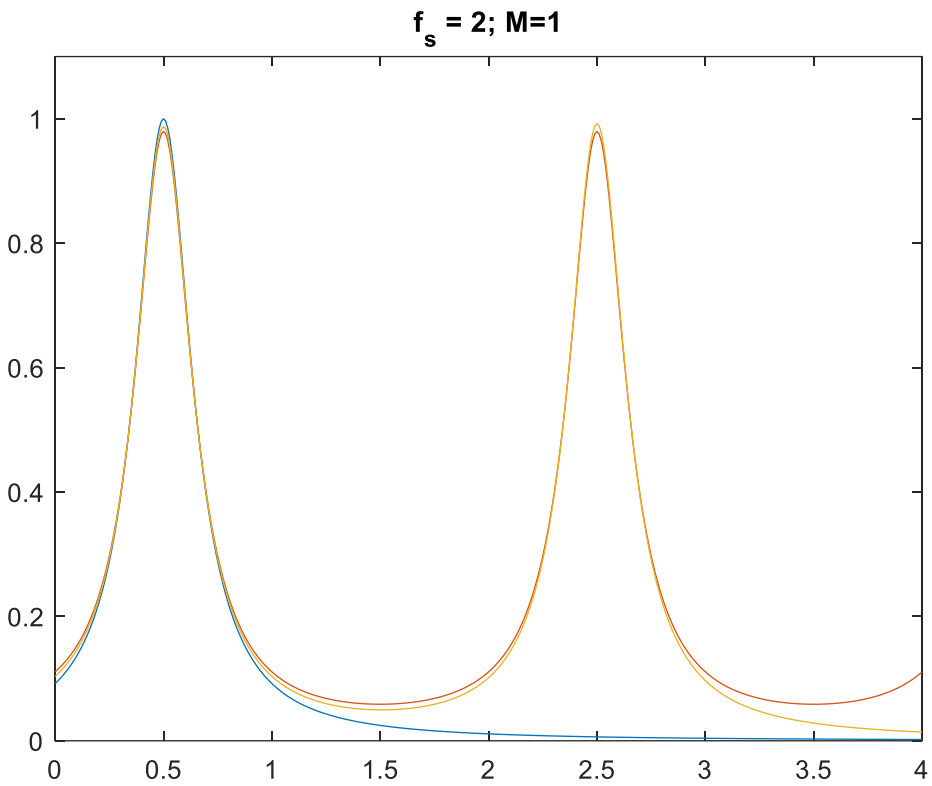
```
% fs = 2, M = 2
f0 = 0.5;
M = -2:1:2;
fs = 2;
f = (0:0.01:4);
omega= 2*pi*f;

X = 1./(a + 1i*(omega - 2*pi*f0)).^2;
[F, R] = meshgrid(f, M);

Xd = (1/fs)^2.*exp(a/fs).*exp(-1i*(omega-2*pi*f0)/fs)./(1-exp(a/fs).*exp(-1i*(omega-
2*pi*f0)/fs)).^2;
Xm = sum(1./(a+1i.*(2*pi.*F-(2*pi*f0)-(R.*(2*pi*fs)))).^2);

figure;
plot(f, abs(X/max(X)), f, abs(Xd), f, abs(Xm));
hold on;
title('f_s = 2; M=2');
xlim([0 4]);
ylim([0 1.1]);
hold off;
```

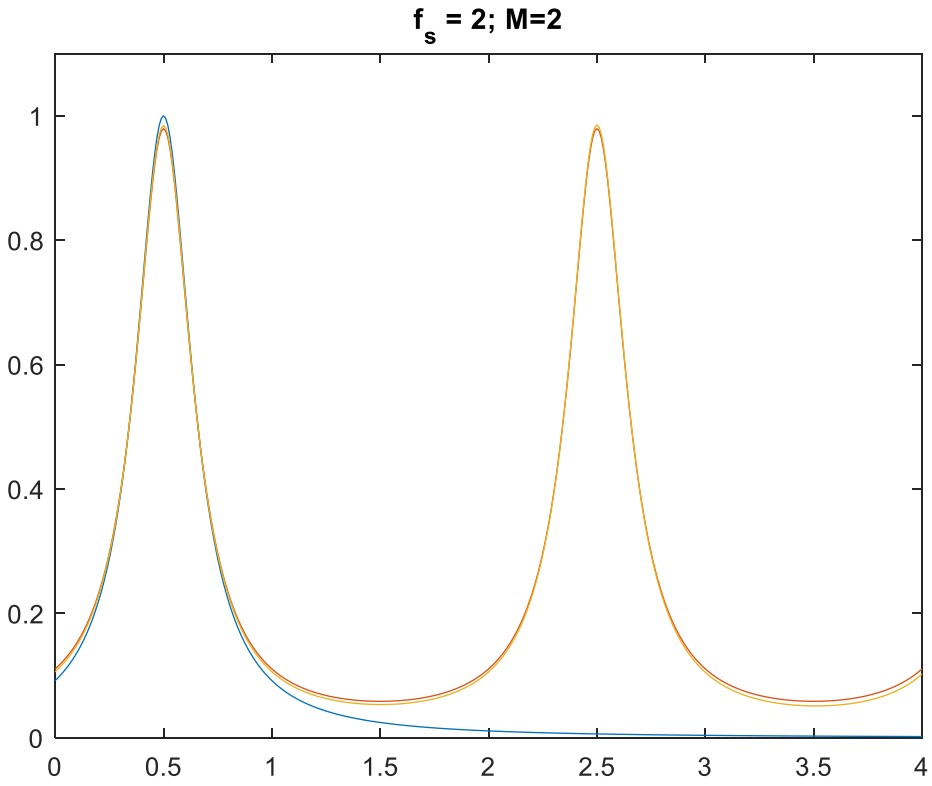

f_s = 0.5; M=1
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4


f_s = 0.5; M=2
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4


f_s = 1; M=1
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4


f_s = 1; M=2
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4


f_s = 2; M=1
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4


f_s = 2; M=2
1
0.8
0.6
0.4
0.2
0
0
0.5
1
1.5
2
2.5
3
3.5
4