
Mikhail Grushko - BE110 - PSET 4

Table of Contents

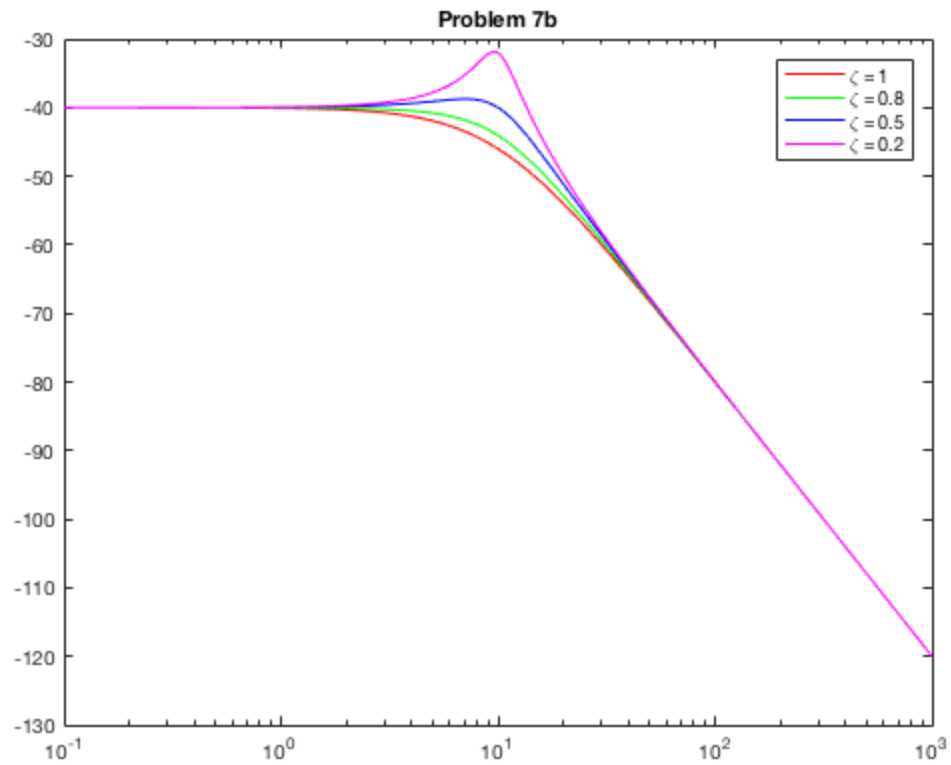
Cleanup	1
Problem 7b	1
Problem 8d	2
Problem 8e	3
Problem 9	5
Problem 10a	7
Problem 10b	8

Cleanup

```
clearvars;  
close all;  
clc;
```

Problem 7b

```
j = sqrt(-1);  
w = 10.^(-1 : 0.01 : 3);  
  
w0 = 10;  
  
zeta = [1, 0.8, 0.5, 0.2];  
  
color = ['r', 'g', 'b', 'm'];  
  
for i = 1 : 4  
    H = 1./((j*w).^2+2*zeta(i)*w0*(j*w)+w0^2);  
    Hdb=20*log10(abs(H));  
    plot(w,Hdb,color(i));  
    set(gca,'xscale','log')  
    hold on;  
end  
title('Problem 7b');  
legend('\zeta = 1', '\zeta = 0.8', '\zeta = 0.5', '\zeta = 0.2');
```



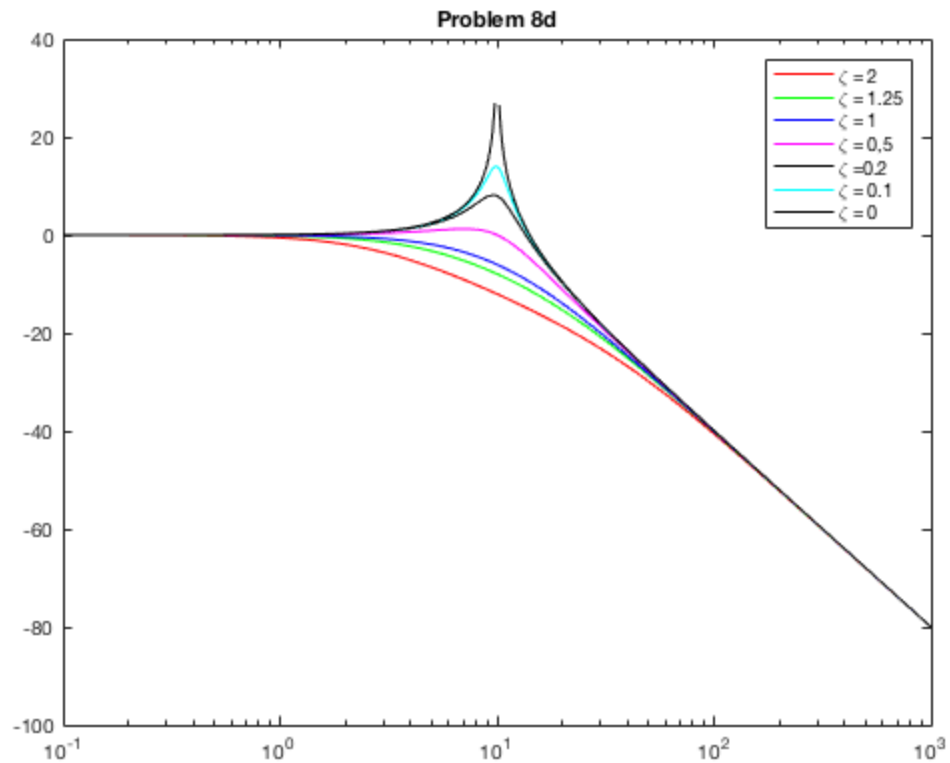
Problem 8d

```

zeta = [2, 1.25, 1, 0.5, 0.2, 0.1, 0];
b = zeta*2*0.1*10;
j = sqrt(-1);
w = 10.^(-1 : 0.01 : 3);
color = ['r', 'g', 'b', 'm', 'k', 'c', 'k'];

figure;
for i = 1 : 7
    H=10./(0.1*(j*w).^2+b(i)*(j*w)+10);
    Hdb=20*log10(abs(H));
    plot(w,Hdb,color(i));
    set(gca, 'xscale', 'log');
    hold on;
end
title('Problem 8d');
legend('\zeta = 2', '\zeta = 1.25', '\zeta = 1', '\zeta = 0,5', '\zeta = 0.2', '\zeta = 0.1', '\zeta = 0');

```



What is going here is that as we decrease ζ from 2 to 0, we get a higher and higher peak values, until eventually @ $\zeta = 0$, we get an infinitely large peak. This is due to the fact that the system's denominator $\rightarrow 0$ as $\zeta \rightarrow 0$

Problem 8e

```
j = sqrt(-1);
w = 10.^(-1 : 0.01 : 3);
zeta = [2, 1.25, 1, 0.5, 0.2, 0.1, 0];
K = 10;
w0 = 10;
color = ['r', 'g', 'b', 'm', 'k', 'c', 'k'];

for i = 1 : 7
    sys=tf(K, [1, 2*zeta(i)*w0, w0^2]);
    Y = stepinfo(sys)
end

Y =

    struct with fields:

        RiseTime: 0.8231
        SettlingTime: 1.4879
        SettlingMin: 0.0902
```

SettlingMax: 0.0999
Overshoot: 0
Undershoot: 0
Peak: 0.0999
PeakTime: 2.7327

Y =

struct with fields:

RiseTime: 0.4624
SettlingTime: 0.8400
SettlingMin: 0.0901
SettlingMax: 0.0999
Overshoot: 0
Undershoot: 0
Peak: 0.0999
PeakTime: 1.4230

Y =

struct with fields:

RiseTime: 0.3359
SettlingTime: 0.5835
SettlingMin: 0.0901
SettlingMax: 0.1000
Overshoot: 0
Undershoot: 0
Peak: 0.1000
PeakTime: 1.1900

Y =

struct with fields:

RiseTime: 0.1639
SettlingTime: 0.8076
SettlingMin: 0.0932
SettlingMax: 0.1163
Overshoot: 16.2929
Undershoot: 0
Peak: 0.1163
PeakTime: 0.3592

Y =

struct with fields:

RiseTime: 0.1206

```
SettlingTime: 1.9596
SettlingMin: 0.0723
SettlingMax: 0.1527
Overshoot: 52.6542
Undershoot: 0
Peak: 0.1527
PeakTime: 0.3224
```

Y =

struct with fields:

```
RiseTime: 0.1127
SettlingTime: 3.8373
SettlingMin: 0.0468
SettlingMax: 0.1729
Overshoot: 72.9156
Undershoot: 0
Peak: 0.1729
PeakTime: 0.3142
```

Y =

struct with fields:

```
RiseTime: NaN
SettlingTime: NaN
SettlingMin: NaN
SettlingMax: NaN
Overshoot: NaN
Undershoot: NaN
Peak: Inf
PeakTime: Inf
```

Problem 9

```
w = 10.^(-1 : 0.01 : 3);

figure;
subplot(2,3,1);
H=1./(j*w/10+1);
Hdb=20*log10(abs(H));
plot(w,Hdb,'r');
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')

subplot(2,3,2);
H=1./(j*w/10+1).^2;
Hdb=20*log10(abs(H));
```

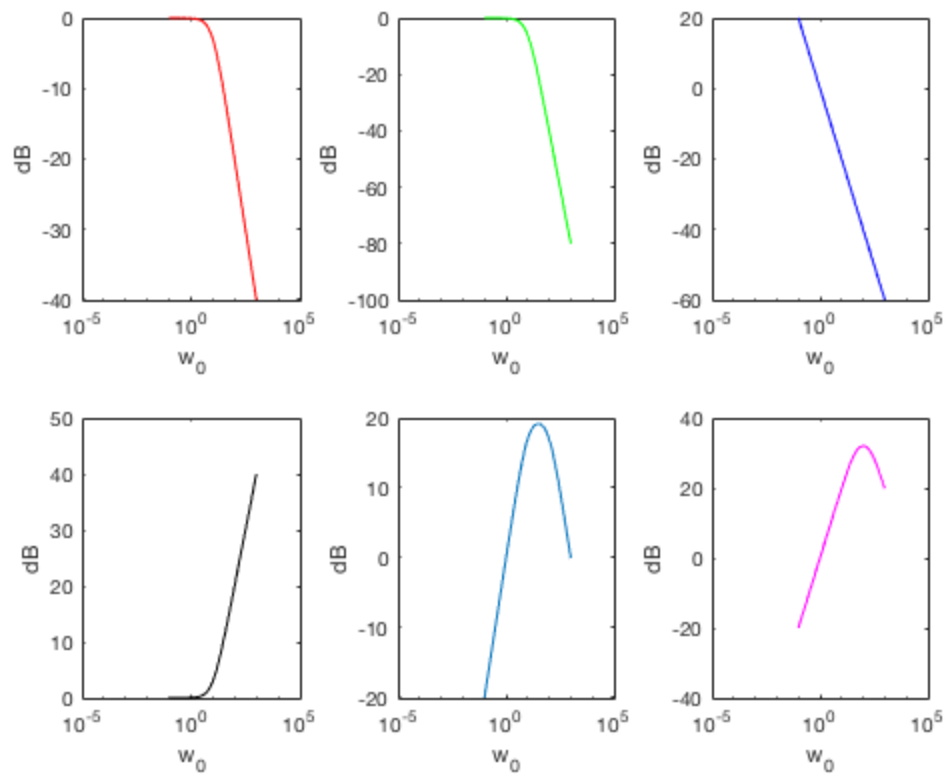
```
plot(w,Hdb,'g');
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')

subplot(2,3,3);
H=1./(j*w);
Hdb=20*log10(abs(H));
plot(w,Hdb,'b');
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')

subplot(2,3,4);
H=((j*w)./10)+1;
Hdb=20*log10(abs(H));
plot(w,Hdb,'k');
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')

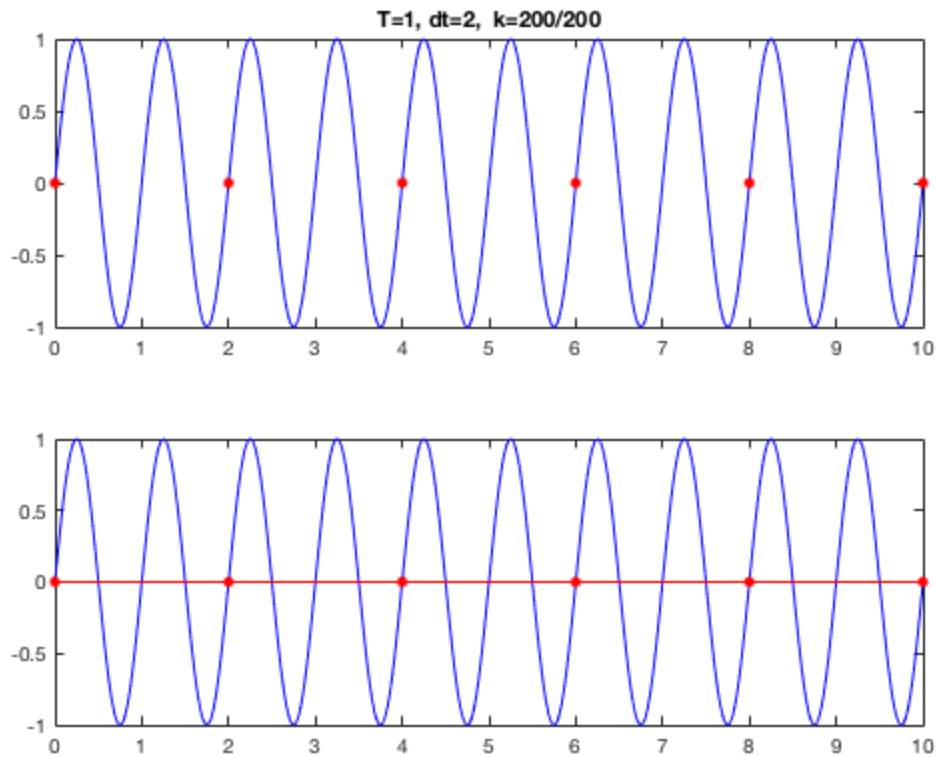
subplot(2,3,5);
H=(j*w)./(((j*w./10) + 1).*((j*w./100) + 1));
Hdb=20*log10(abs(H));
plot(w,Hdb);
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')

subplot(2,3,6);
H=(j*w)./(((j*w./50) + 1).*((j*w./200) + 1));
Hdb=20*log10(abs(H));
plot(w,Hdb,'m');
set(gca,'xscale','log')% define H & make the plot
xlabel('w_0')
ylabel('dB')
```



Problem 10a

```
figure;
t=0:.01:10;
x=sin(2*pi*t);
% define t with an initial sampling interval dt=0.01 and define x(t) to
% have
% period=1
for k=1:200
% for loop that incrementally changes the sampling interval
    is=1:k:length(t);
    ts=t(is);
    xs=x(is);
    %set the new dt to k*dt and then plot the original x (blue) and the
    % sampled x (red)
    subplot(211);
    plot(t,x,'-b',ts,xs,'.r','markersize',16);
    title(['T=1, dt=',num2str(k*0.01), ', k=',num2str(k), '/200']);
    subplot(212); plot(t,x,'-b',ts,xs,'.-r','markersize',16); pause
    % hit the SPACEBAR to advance to next plot
end
% this code flips through sampling rates. The lower plots is the
% same as
% the upper but with the red dots connected
```



10a: As I am flipping through the sampling frequencies, I observe the effect known as aliasing. Aliasing is generally observed in the Discrete-Time Fourier Transform, where only a single fundamental sampling frequency ω_o . At higher ω_o (significantly higher than period T), the DTFT does an accurate job of representing the original wave. However, at $\omega_o = 0.5T$, aliasing effect begins to occur, due to the fact that one cannot unambiguously interpret the samples, creating multiple signals that are aliases of each other (i.e. all of the aliases can produce the sampling obtained).

Problem 10b

```
figure;
dt=0.01;
t=0:dt:10;
N=length(t);
% define t with a sampling interval dt=0.001
for k=1:200
% for loop that incrementally changes the frequency
    x=cos(2*pi*k*t);
    %set the new frequency to k
    subplot(211);
    plot(t,x);
    title(['dt=0.01, Sampling freq= 100, Signal',
    'freq=',num2str(k), '/200']);
    % plot x(t)
    X=fft(x);
    X=X(1:round(N/2)); f=(0:round(N/2)-1)/(N*dt);
```

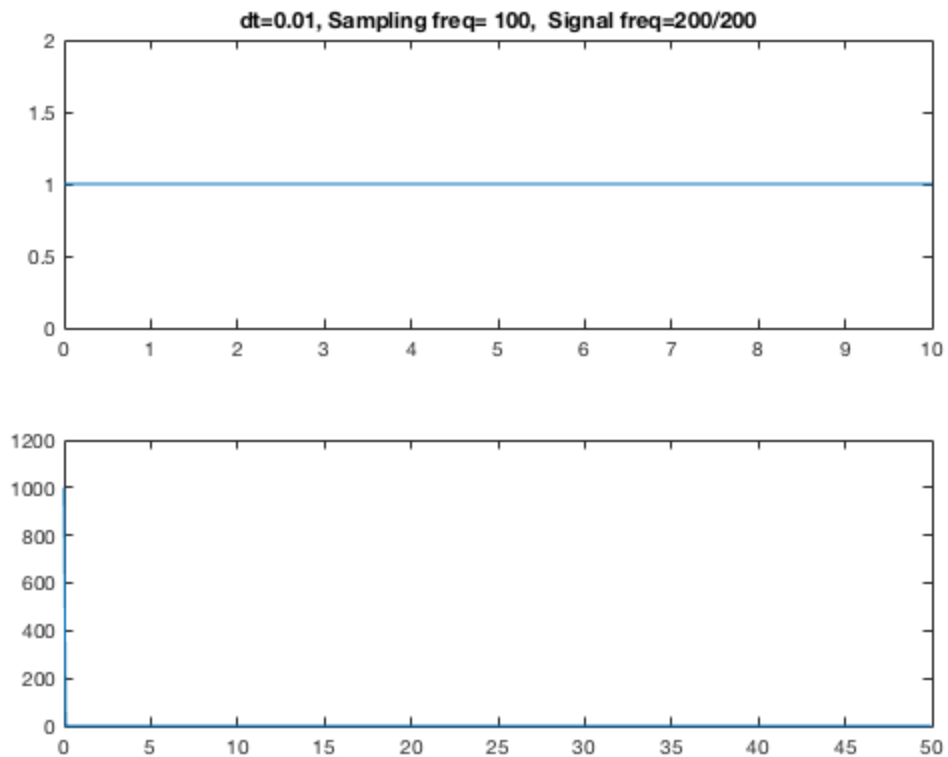


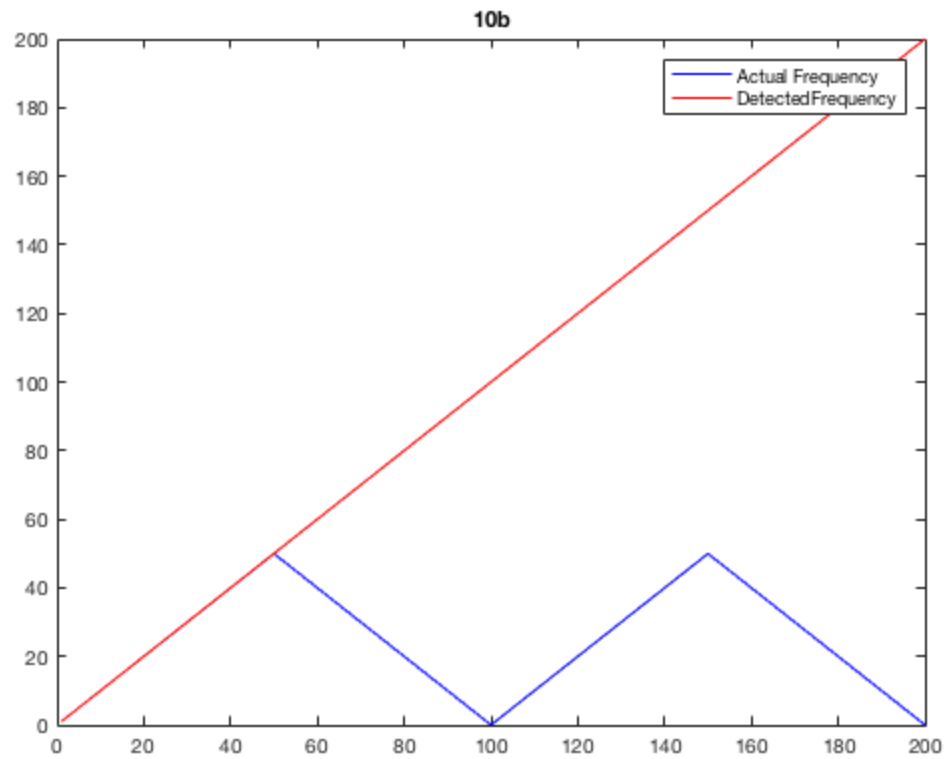
```

% findthe DFT of x(t) thenremove the redundant freqs
subplot(212);
plot(f,abs(X));% plot X(f)in the lowerplot
    [a, b] = max(X);
    f_actual(k)= k;
    f_detected(k)= f(b); %MODIFY THIS LINE
    pause;
    % pressthe SPACEBAR to advance to next frame(or hold it downto
    advance rapidly)\
end

figure;
plot(f_actual, f_detected, '-b',f_actual, f_actual, '-r');
legend({'Actual Frequency','DetectedFrequency'})
title('10b')

```





This is another way of demonstrating the aliasing problem when doing the Discrete-Time Fourier Transform. However, this time we're exploring DTFTs at different signal ω_o , but the same sampling frequency. What happens as we move towards higher signal frequencies, we see repeating frequencies, which is a sign of aliasing. The pattern also follows the Nyquist law, which states that a signal must be sampled at a rate greater twice as ω_o .

Published with MATLAB® R2017b