Chen Ji
SUNetID:jjjjjc

Homework 3

Music424
April 29, 2019

Problem 1

(a) The MATLAB code can be written as below:

```
%%Q = 2^[-4:2]
%constant
wc = 2*pi*1000;
Q = 2.^(-4:2);
N = length(Q);
B = [0 0 1];

for i = 1:N
    A = [1 1/(wc*Q(i)) 1/(wc^2)];
    [H,W] = freqs(B,A);
    subplot(2,1,1);
    loglog(W,abs(H));
    xlabel 'Frequency (rad/s)', ylabel 'Magnitude'
    legend('-1','-3','-2','-1','0','1','2');
    hold all;
    subplot(2,1,2);
    semilogx(W,angle(H));
    xlabel 'Frequency (rad/s)', ylabel 'Phase (degrees)'
    legend('-1','-3','-2','-1','0','1','2');
    hold all;
end
```

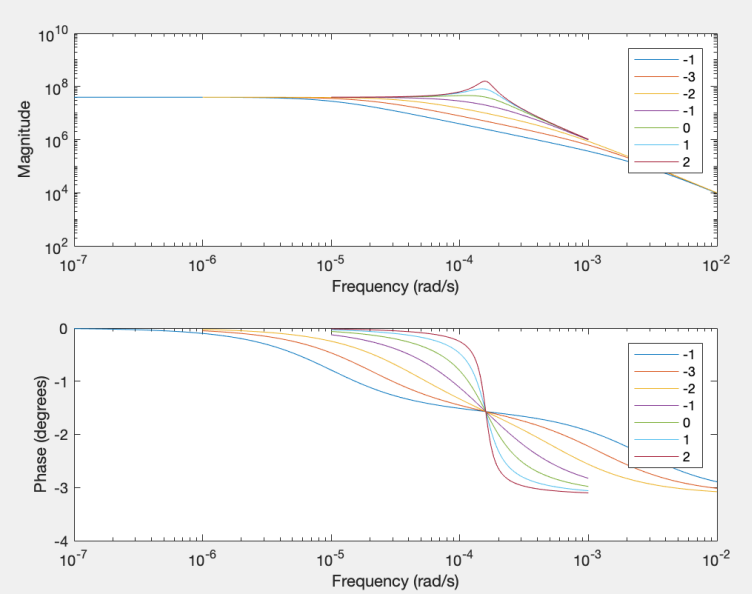

Figure 1: $Q = 2^{[-4:2]}$

Homework 3

```
clear all;
close all;

%constant
wc = 2*pi*1000*2.^(-2:2);
Q = 2;
N = length(wc);
B = 1;

for i = 1:N
    A = [1/(wc(i)^2) 1/(wc(i)*Q) 1];
    [H,W] = freqs(B,A);
    subplot(2,1,1);
    loglog(W,(abs(H)/max(abs(H))));
    xlabel ('Frequency (rad/s)'), ylabel ('Magnitude');
    ylim([-20 20]);
    legend('-2','-1','0','1','2');
    hold all;
    subplot(2,1,2);
    semilogx(W,angle(H));
    xlabel ('Frequency (rad/s)'), ylabel ('Phase (degrees)');
    legend('-2','-1','0','1','2');
    hold all;
end
```

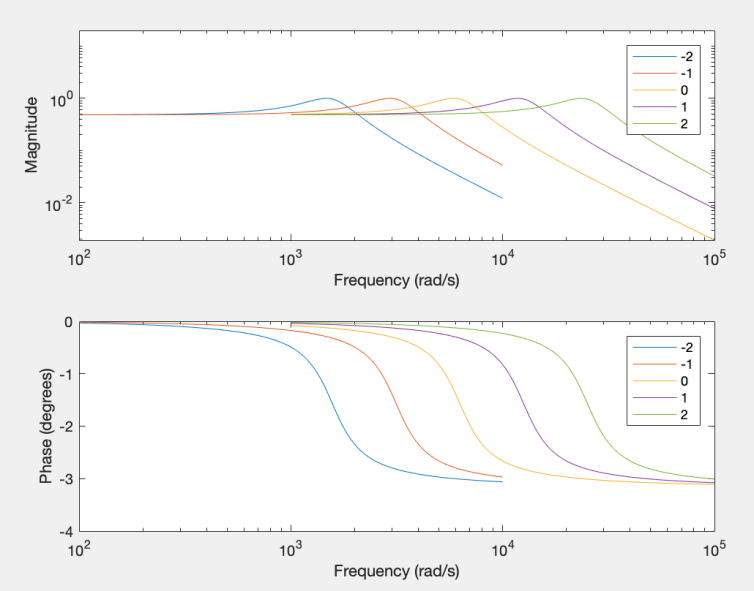

Figure 2: wc = 2*pi*1000*$2^{[-2:2]}$

The following figure shows the varying Q:

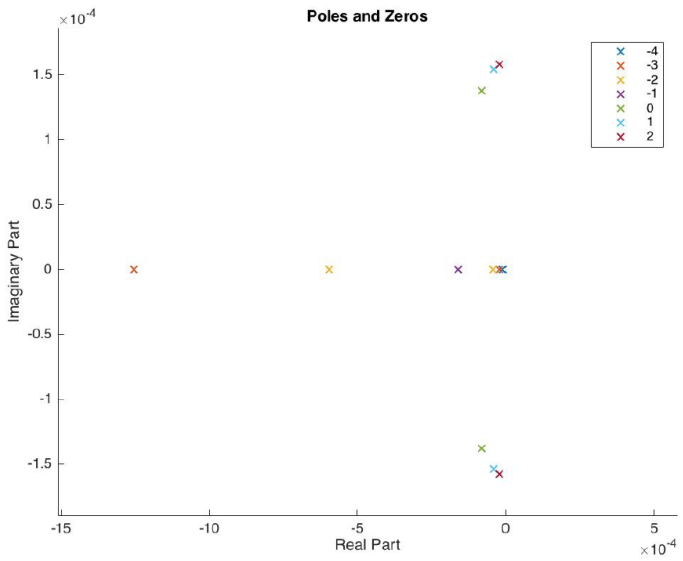

Figure 3: varying Q

The following figure shows the varying wc:

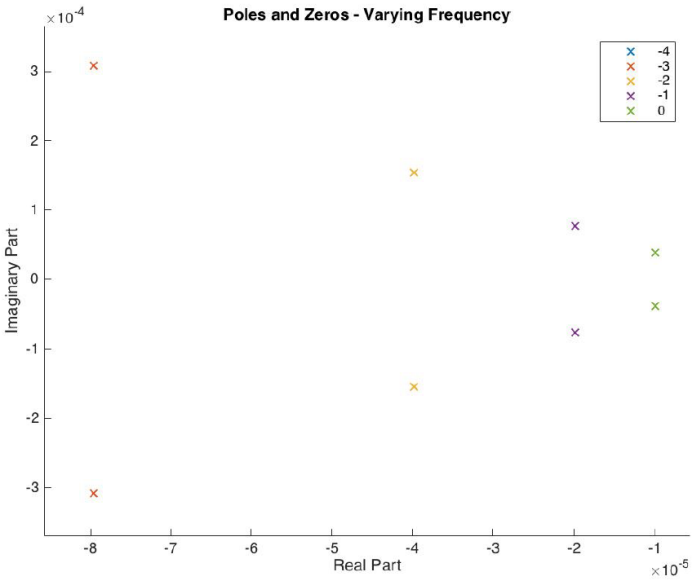

Figure 4: varying cut-off frequency

(b) By using bilinear transform, our analog filter into a digital representation can be obtained:

$$H(s) = \frac{1}{(\frac{s}{w})^2 + \frac{1}{Q}\frac{s}{w} + 1}$$

Homework 3

We let $s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$

$$H(s) = \frac{1}{(\frac{s}{w})^2 + \frac{1}{Q}\frac{s}{w} + 1}$$

Then the C++ code can be written as:

```
///////////////START/////////////////
b0 = 1.0; b1 = 0.0; b2 = 0.0;
a0 = 1.0; a1 = 1.0/(center*qval*2*pi); a2 = 1.0/(qval*2*pi)*(qval*2*pi);
///////////////END///////////////////
```

Figure 5: Low pass filter in s domain

```
// TODO: apply bilinear transform
///////////////START/////////////////
double T = 1/fs;
bz0 = b0; bz1 = 2*b0; bz2 = 1.0;
az0 = 4*a2/(T*T)-2*a1/T+a0; az1 = 2*a0-8*a2/(T*T); az2 =
    4*a2/(T*T)+2*a1/T+a0;
///////////////END///////////////////
```

Figure 6: Low pass filter in z domain by applying Bilinear Transform

Problem 2

(a) So as to make the filter smoothly varying over time, the user controls will be tracked using leaky integrators.

```
void setTau(float tau, float fs) {
    ///////////////START/////////////////
    a1 = exp(-1.0/(tau*fs));
    b0 = 1-a1;

    ///////////////END/////////////////
}
void reset() {
    // reset filter state
    z1=0;
}
void process (float input, float& output) {
    ///////////////START/////////////////
    z1 += b0 * (input -z1);
    output = z1;
    ///////////////END/////////////////
}
```

Figure 7: Code for SlewedParameter class

(b) Implement a low-frequency oscillator (LFO) with sine of a phase counter:

```
//------------------------------------------------
float WahWah::LFO(float f0)
{
    ///////////////START//////////////////
    //Initialize
    float phase = 0;
    //increment by delta
    float change = 2*pi*f0/fs;
    //Mode 2pi for phase:
    phase = fmodf(phase+change, 2*pi);
    //look up next sample
    float sample = sin(phase);
    return sample;
    ////////////////END////////////////////

}
```

Figure 8: Code for LFO class