Table of Contents

Swenson_MAE673_HW2

```
clear; close all; clc;
```

Figure 1: Different Inputs

```
% Show an example of the Lumped Delay and the Distributed Delay
t = 0:.01:10;
xLumped = heaviside(t-5.0001);
xDistributed = (.2 + .2*t.*heaviside(5.0001-t) + heaviside(t-5.0001))/1.2;

figure();
subplot(211)
plot(t,xLumped)
ylabel('Lumped Delay')
title('Lumped Delay and Distributed Delay Step Inputs')
ylim ([0 1.1]);

subplot(212)
plot(t,xDistributed)
ylabel('Distributed Delay')
xlabel('Time (s)'); ylim ([0 1.1]);
```

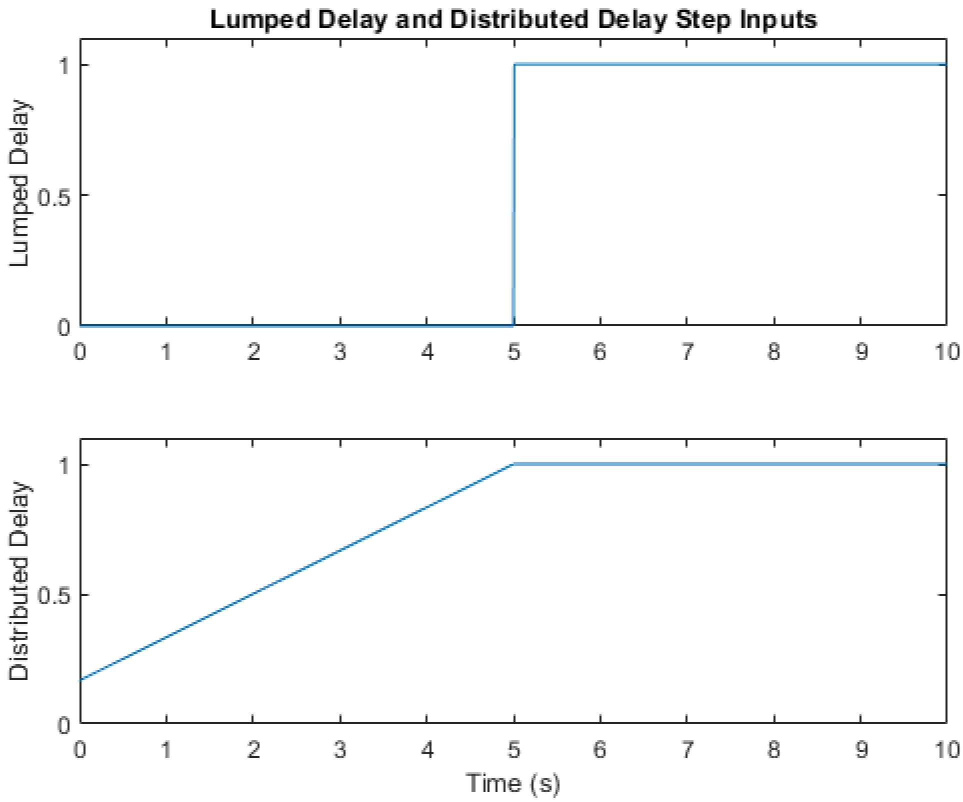


Figure 2: Gains and Normalized Delay of the ZV and DZV Systems

```
% Making use of standard form single DoF system.
% System Characteristics
w0 = 1;
Zeta = 0:.001:.999; % For zeta = 1, system is exponentially stable and A = B
 =1
ABTstore = zeros(length(Zeta),5);

for j = 1:length(Zeta);
    zeta = Zeta(j);
    Den = [1 2*zeta*w0 w0^2];
    poles = roots(Den);
    Beta = -real(poles); Beta = Beta(1);
    Omega = imag(poles); Omega = Omega(1);

    % Standard Lumped Delay Parameters
    Tau = pi/Omega;               % Time Delay
    A = exp(Beta*pi/Omega)/(1+exp(Beta*pi/Omega));      % Gain A
    Td = 2*pi/Omega;              % Damped Period of Oscillations

    % Function, range, etc for solving for theta root
    func = @(x)(Omega*exp(-Beta*x)+Beta*sin(Omega*x)-Omega*cos(Omega*x));
```

```
    %dfunc = @(x)(-Beta*Omega*exp(-Beta*x) + Beta*Omega*cos(Omega*x) +
 sin(Omega*x)*Omega^2);
    range = [pi/Omega 2*pi/Omega];
    x0 = range(1)*1.5; % Needed inital guess in middle of range to converge in
 range
    tol = 1e-6;
    %[Theta, store] = NewtonRaphson(x0,func,dfunc,tol);
    [Theta, count] = Bisection(range,func,tol);

    % DZV Filter Gain
    B = sin(Omega*Theta)/(sin(Omega*Theta) - Theta*Omega*exp(-Beta*Theta));


    % Normalized Delays
    Taubar = Tau/Td;
    Thetabar = Theta/Td;

    ABTstore(j,:) = [A, B, Taubar, Thetabar, Theta];

    % This figure checks to make sure the correct pole has been found
%     figure()
%     x = range(1):.01:range(2);
%     plot(x,func(x),Theta,func(Theta),'kx')
%     pause(.15)

end

figure();
subplot(211)
plot(Zeta,ABTstore(:,1),Zeta,ABTstore(:,2))
title('Gains and Normalized delays of ZV and DZV Filters')
legend('ZV','DZV','location','east'); ylabel('Gain')
subplot(212)
plot(Zeta,ABTstore(:,3),Zeta,ABTstore(:,4))
ylim([.3 1]);
xlabel('$\zeta$','interpreter','latex')
ylabel('Normalized Delay')
legend('ZV','DZV','location','northeast');
```

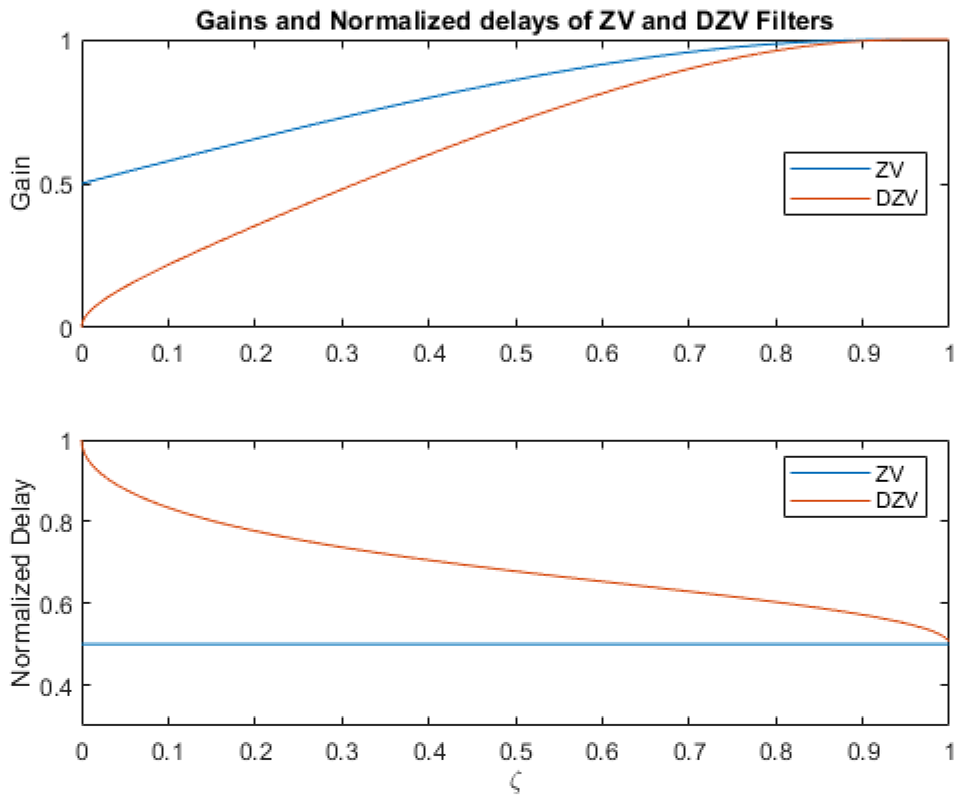


Figure 4: Spectra of Zeros

Need to calculate the range of zeros for each of the different filters

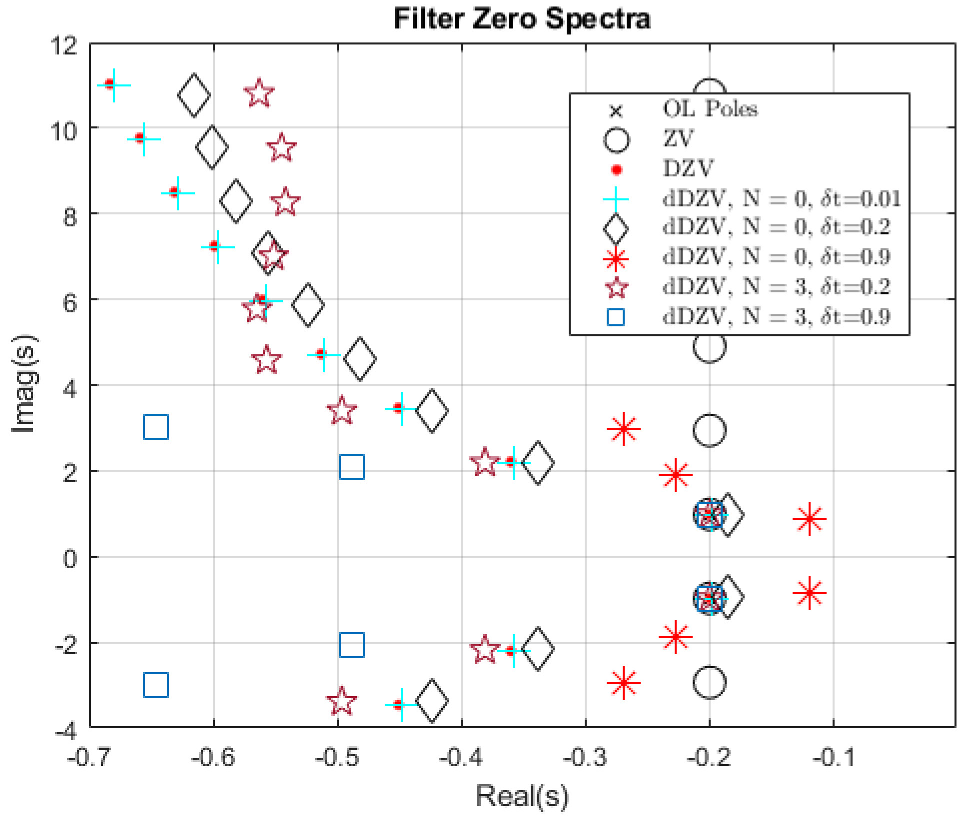
```
% OL System
w0 = 1;          zeta = 0.2;
Num = w0^2;      Den = [1 2*zeta*w0 w0^2];
OLpoles = roots(Den);
Omega = imag(OLpoles(1));
Beta = -real(OLpoles(1));

% ZV Filter
Tau = pi/Omega;
A = exp(Beta*pi/Omega)/(1+exp(Beta*pi/Omega));
kvec = [0:9]';
sZV = [-(1/Tau)*log(A/(1-A)) + 1i*pi*(2*kvec+1)/Tau; -(1/Tau)*log(A/(1-A)) -
 1i*pi*(2*kvec+1)/Tau];


% DZV Filter
func = @(x)(Omega*exp(-Beta*x)+Beta*sin(Omega*x)-Omega*cos(Omega*x));
range = [pi/Omega 2*pi/Omega];
tol = 1e-6;
[Theta, count] = Bisection(range,func,tol);
B = sin(Omega*Theta)/(sin(Omega*Theta) - Theta*Omega*exp(-Beta*Theta));
```

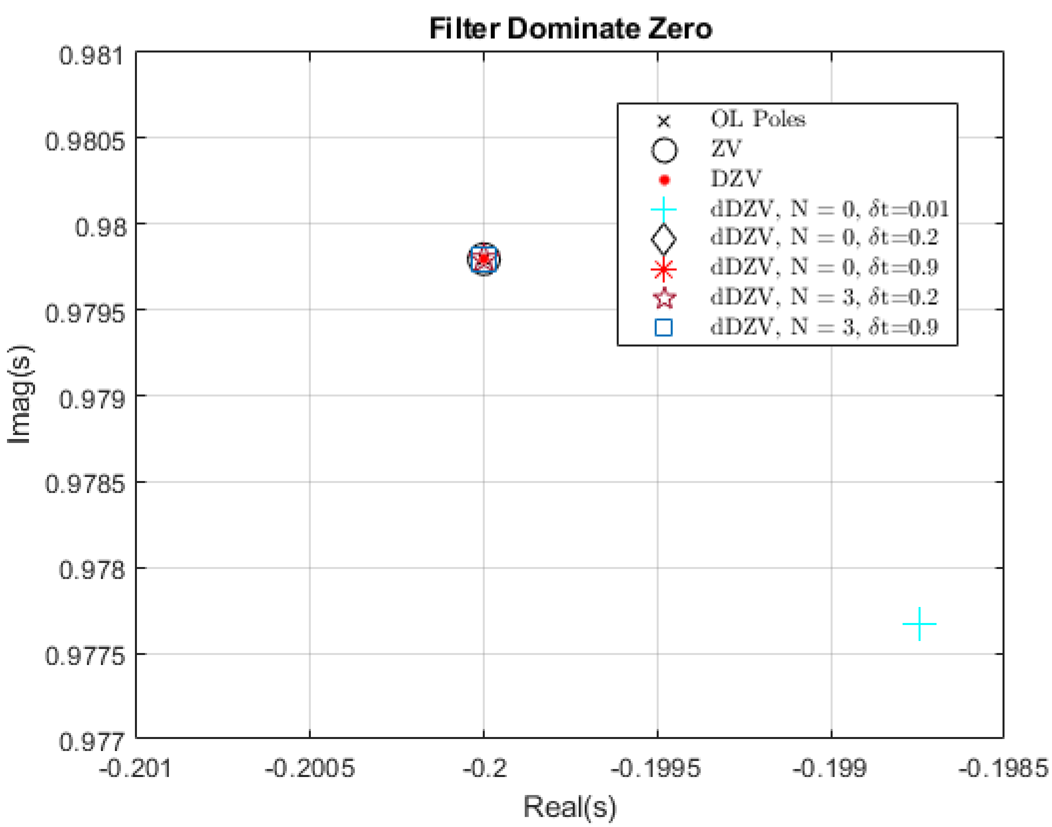
```matlab
% DZV Spectra
k = [-9:-1,1:9];
B1 = (1-B)/B;
s = (lambertw(k,B1*exp(B1))-B1)/Theta;


% dDZV Spectra
% Values from Paper but using my unrounded values from above
% Omega = sqrt(1-.04); Beta = .2;
% A = .655; tau = 3.21; B = .354; Theta = 4.98;

Nvec = [0; 3];
dtvec = [0.01; .2; .9];


dDZVstore = zeros(length(s)+2,length(Nvec)*length(dtvec));
avstore = zeros(4,3);
dpoles = zeros(length(s),1);
count = 0; countN = 0;

for j = 1:length(Nvec)
   N = Nvec(j);
   for jj = 1:length(dtvec)

        count = count + 1;
        dt = dtvec(jj);
        d = ceil(Theta/dt);
        r1 = -Beta + 1i*Omega;
        P1 = exp(r1*dt);

        if N > 1
            countN = countN+1;
            Mp1 = (P1^d)*( ((B*Theta*(P1-1))/((1-B)*dt)) + 1 );
            Pmat = [1   real(P1) real(P1^2) real(P1^3);
                    0   imag(P1) imag(P1^2) imag(P1^3);
                    1   1        1          1];
            Mvec = [real(Mp1) imag(Mp1) 1]';
            av = pinv(Pmat)*Mvec
            avstore(:,countN) = av;

%              if dt == .9
%                  av = [.274 .447 .285 -.0069]';
%              end
%              if dt == .2
%                  av = [.918 -0.076 -0.23 0.391]';
%              end

            CoeffVec = zeros(1,d+2);
            CoeffVec(1) = Theta*B;
            CoeffVec(2) = (dt*(1-B)-Theta*B);
            CoeffVec(end-3) = -dt*(1-B)*av(4);
            CoeffVec(end-2) = -dt*(1-B)*av(3);
            CoeffVec(end-1) = -dt*(1-B)*av(2);
            CoeffVec(end-0) = -dt*(1-B)*av(1);
```

```
            rootsdDZV = roots(CoeffVec);
        else
            CoeffVec = zeros(1,d+2);
            CoeffVec(1) = Theta*B;
            CoeffVec(2) = dt*(1-B)-Theta*B;
            CoeffVec(end) = -dt*(1-B);
            rootsdDZV = roots(CoeffVec);
        end

        dpoles = rootsdDZV;
        spoles = (log( abs(dpoles) ) + 1i*atan2(imag(dpoles),real(dpoles)) )/
dt;
        check = (real(spoles)<-1e-6) & (real(spoles)>-.7) & (imag(spoles)>-4)
 & (imag(spoles)<12);
        spoles = spoles(check);
        spolessorted = sort(spoles);

        if length(spoles) < 18
            spoles = [spoles; zeros(18-length(spoles),1)];
        end

        dDZVstore(:,count) = [N; dt; spoles(1:18)];
        stop = 1;

    end
end

% Plot the Spectra of Poles
figure()
plot(real(OLpoles),imag(OLpoles),'kx');
xlim([-.7 -0.001]); ylim([-4 12]);
grid on
hold on
plot(real(sZV),imag(sZV),'ko','MarkerSize',12)
plot(real(s),imag(s),'.r','MarkerSize',12)
plot(real(dDZVstore(3:end,1)),imag(dDZVstore(3:end,1)),'+c','MarkerSize',12)
plot(real(dDZVstore(3:end,2)),imag(dDZVstore(3:end,2)),'kd','MarkerSize',12)
plot(real(dDZVstore(3:end,3)),imag(dDZVstore(3:end,3)),'r*','MarkerSize',12)
plot(real(dDZVstore(3:end,5)),imag(dDZVstore(3:end,5)),'pentagram','MarkerSize',12)
plot(real(dDZVstore(3:end,6)),imag(dDZVstore(3:end,6)),'square','MarkerSize',12)
legend('OL Poles','ZV','DZV','dDZV, N = 0, $\delta$t=0.01','dDZV, N = 0, $
\delta$t=0.2',...
        'dDZV, N = 0, $\delta$t=0.9','dDZV, N = 3, $\delta$t=0.2',...
        'dDZV, N = 3, $\delta$t=0.9','location','best','interpreter','latex')
xlabel('Real(s)'); ylabel('Imag(s)'); title('Filter Zero Spectra')

figure()
plot(real(OLpoles),imag(OLpoles),'kx');
xlim([-.201 -.1985]); ylim([.977 .981]);
grid on
hold on
plot(real(sZV),imag(sZV),'ko','MarkerSize',12)
plot(real(s),imag(s),'.r','MarkerSize',12)
plot(real(dDZVstore(3:end,1)),imag(dDZVstore(3:end,1)),'+c','MarkerSize',12)
```

```
plot(real(dDZVstore(3:end,2)),imag(dDZVstore(3:end,2)),'kd','MarkerSize',12)
plot(real(dDZVstore(3:end,3)),imag(dDZVstore(3:end,3)),'r*','MarkerSize',12)
plot(real(dDZVstore(3:end,5)),imag(dDZVstore(3:end,5)),'pentagram','MarkerSize',12)
plot(real(dDZVstore(3:end,6)),imag(dDZVstore(3:end,6)),'square','MarkerSize',12)
legend('OL Poles','ZV','DZV','dDZV, N = 0, $\delta$t=0.01','dDZV, N = 0, $
\delta$t=0.2',...
        'dDZV, N = 0, $\delta$t=0.9','dDZV, N = 3, $\delta$t=0.2',...
        'dDZV, N = 3, $\delta$t=0.9','location','best','interpreter','latex')
xlabel('Real(s)'); ylabel('Imag(s)'); title('Filter Dominate Zero')
```

```
av =

   9.322377925304863
  -8.758781255954091
  -8.772548628955519
   9.208951959605656


av =

   0.917544745729268
  -0.076227078894545
  -0.232278906212169
   0.390961239377447


av =

   0.274244561368619
   0.447184967154886
   0.285481657925590
  -0.006911186449094
```


Filter Zero Spectra
OL Poles
ZV
DZV
dDZV, N = 0, δt=0.01
dDZV, N = 0, δt=0.2
dDZV, N = 0, δt=0.9
dDZV, N = 3, δt=0.2
dDZV, N = 3, δt=0.9
Imag(s)
Real(s)
12
10
8
6
4
2
0
-2
-4
-0.7
-0.6
-0.5
-0.4
-0.3
-0.2
-0.1

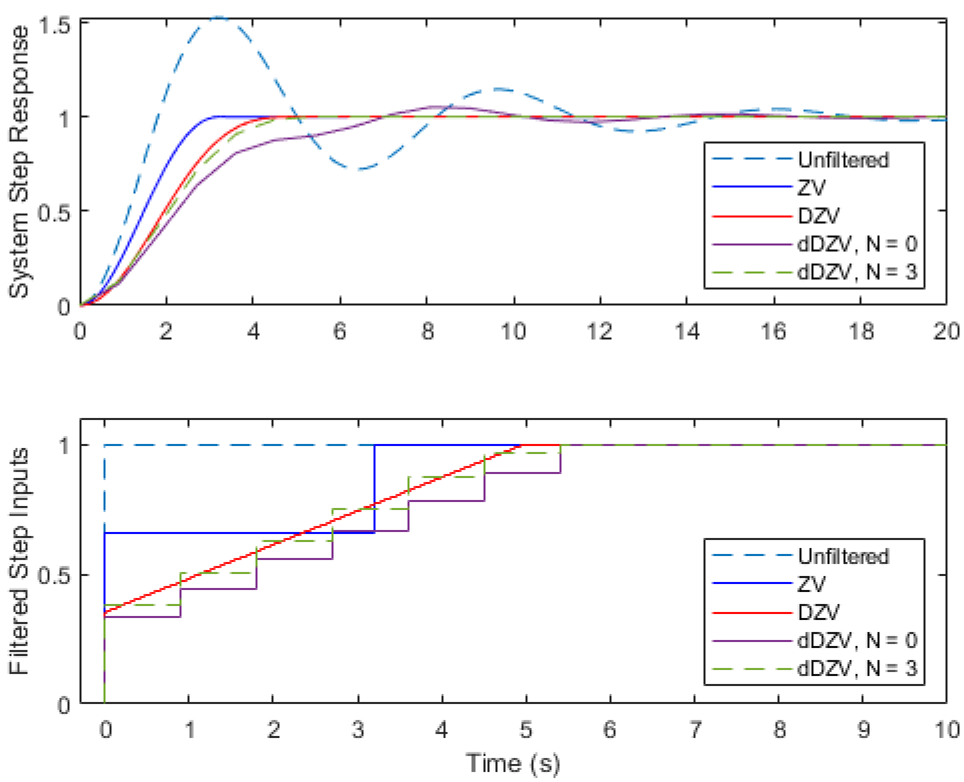

Filter Dominate Zero
OL Poles
ZV
DZV
dDZV, N = 0, δt=0.01
dDZV, N = 0, δt=0.2
dDZV, N = 0, δt=0.9
dDZV, N = 3, δt=0.2
dDZV, N = 3, δt=0.9
Imag(s)
Real(s)
0.981
0.9805
0.98
0.9795
0.979
0.9785
0.978
0.9775
0.977
-0.201
-0.2005
-0.2
-0.1995
-0.199
-0.1985

Step Response of System

```
dt = 0.9;
w0 = 1; zeta = 0.2;
sysOL = tf(w0^2,[1 2*zeta*w0 w0^2]);
sysdOL = c2d(sysOL,dt);


k = [0:50]';
wk = ones(length(k),1);
wk0 = heaviside(k+.0001 - d);
wk1 = heaviside(k+1.001-d);
wk2 = heaviside(k+2.001-d);
wk3 = heaviside(k+3.001-d);
test = [wk wk0 wk1 wk2 wk3];


Nvec = [0 3];
x = zeros(length(k)+1,length(Nvec));
ukdDZV = zeros(length(k),length(Nvec));

% Creating the shaper output
for jj = 1:2
    N = Nvec(jj);
    for j = 1:length(k)

        if N > 1
            av = avstore(:,3);
            C = Theta/( Theta*B + (1-B)*dt*(d - av(2) - 2*av(3) - 3*av(4)) );
            x(j+1,jj) = x(j,jj) + dt*( wk(j) - av(1)*wk0(j) - av(2)*wk1(j) -
 av(3)*wk2(j) - av(4)*wk3(j) );
        else

            C = Theta/( Theta*B + (1-B)*dt*(d) );
            x(j+1,jj) = x(j,jj) + dt*( wk(j) - wk0(j));
        end

        ukdDZV(j,jj) = C*( B*wk(j) + x(j,jj)*(1-B)/Theta );

    end
end


% Simulate the system response
t = [0:.01:.9*k(end)]';

for dumb =1;
r1 = ukdDZV(1,1)*ones(91,1);
r2 = ukdDZV(2,1)*ones(91,1);
r3 = ukdDZV(3,1)*ones(91,1);
r4 = ukdDZV(4,1)*ones(91,1);
r5 = ukdDZV(5,1)*ones(91,1);
r6 = ukdDZV(6,1)*ones(91,1);
```

```
r7 = ukdDZV(7,1)*ones(91,1);
r8 = ukdDZV(8,1)*ones(91,1);
r9 = ukdDZV(9,1)*ones(91,1);
r10 = ukdDZV(10,1)*ones(91,1);
r11 = ukdDZV(11,1)*ones(91,1);
udDZV0 =
 [r1;r2(2:end);r3(2:end);r4(2:end);r5(2:end);r6(2:end);r7(2:end);r8(2:end);r9(2:end);r10(2

r1 = ukdDZV(1,2)*ones(91,1);
r2 = ukdDZV(2,2)*ones(91,1);
r3 = ukdDZV(3,2)*ones(91,1);
r4 = ukdDZV(4,2)*ones(91,1);
r5 = ukdDZV(5,2)*ones(91,1);
r6 = ukdDZV(6,2)*ones(91,1);
r7 = ukdDZV(7,2)*ones(91,1);
r8 = ukdDZV(8,2)*ones(91,1);
r9 = ukdDZV(9,2)*ones(91,1);
r10 = ukdDZV(10,2)*ones(91,1);
r11 = ukdDZV(11,2)*ones(91,1);
udDZV3 =
 [r1;r2(2:end);r3(2:end);r4(2:end);r5(2:end);r6(2:end);r7(2:end);r8(2:end);r9(2:end);r10(2

%sysdOL = c2d(sysOL,.01);

end



uZV = A*ones(length(t),1) + (1-A)*heaviside((t+.0001)-Tau);
uDZV = B*ones(length(t),1) + ((1-B)/Theta)*(t - (t-Theta).*heaviside((t
+.0001)-Theta));
uStep = ones(length(t),1);

umat = [uStep, uZV, uDZV];
udDZV = [0 0; ukdDZV];
umat = [zeros(1,3); umat];
td = [0; t];
kd = [0; k];


[Ac,Bc,Cc,Dc] = tf2ss(cell2mat(sysOL.Numerator),cell2mat(sysOL.Denominator));
[Ad,Bd,Cd,Dd] =
 tf2ss(cell2mat(sysdOL.Numerator),cell2mat(sysdOL.Denominator));
sscOL = ss(Ac,Bc,Cc,Dc);
ssdOL = ss(Ad,Bd,Cd,Dd,dt);


[ystep, tostep, xstep] = lsim(sscOL,uStep,t);
[yZV, toZV, xZV] = lsim(sscOL,uZV,t);
[yDZV, toDZV, xDZV] = lsim(sscOL,uDZV,t);
[ydDZV0, todDZV0, xdDZV0] = lsim(ssdOL,ukdDZV(:,1),k*dt);
[ydDZV3, todDZV3, xdDZV3] = lsim(ssdOL,ukdDZV(:,2),k*dt);
```

```
figure();
subplot(211)
plot(t,ystep,'--'); hold on;
plot(t,yZV,'b');
plot(t,yDZV,'r');
plot(k*dt,ydDZV0);
plot(k*dt,ydDZV3,'--')
ylabel('System Step Response');
legend('Unfiltered','ZV ','DZV ','dDZV, N = 0','dDZV, N =
 3','location','southeast')

subplot(212)
stairs(td,umat(:,1),'--'); hold on;
stairs(td,umat(:,2),'b');
stairs(td,umat(:,3),'r');
stairs(kd*dt,udDZV(:,1));
stairs(kd*dt,udDZV(:,2),'--')
xlim([-.3 10]); ylim([0 1.1]);
legend('Unfiltered','ZV ','DZV ','dDZV, N = 0','dDZV, N =
 3','location','southeast')
ylabel('Filtered Step Inputs'); xlabel('Time (s)')
```



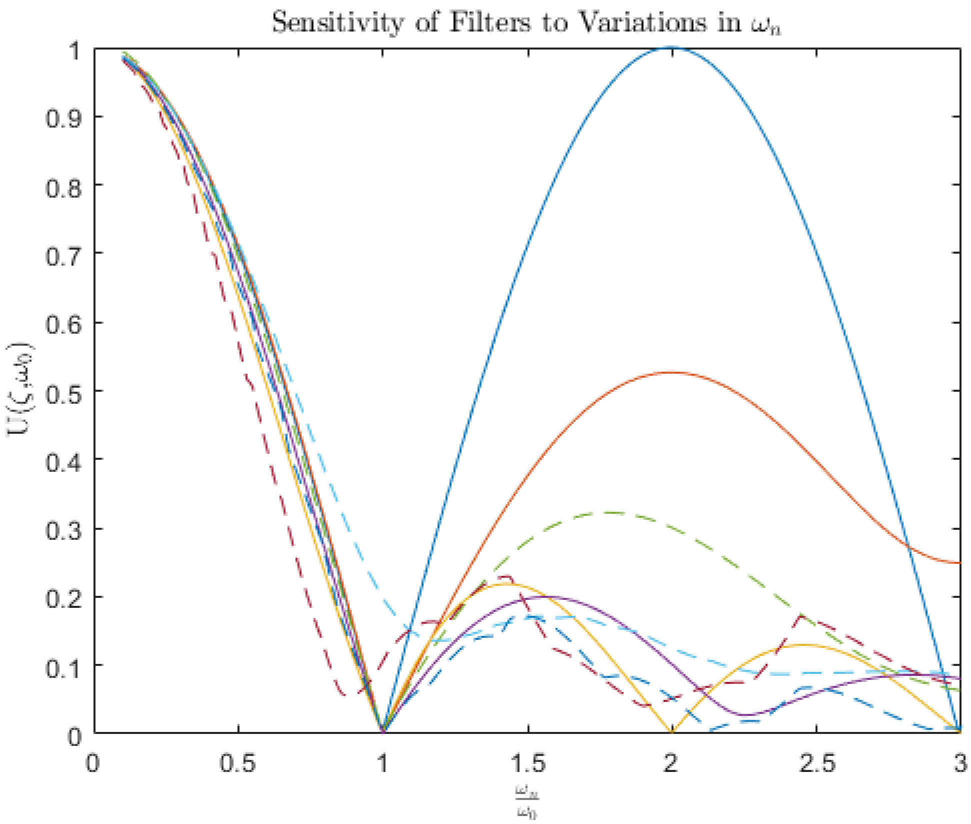
Sensitivity to Varying Natural Frequencies

Create the input for the convolved ZV shaper

```
psi_02 = Theta-Tau;     % psi term for zeta = .02
stzv_num_02 = 1; stzv_den_02 = [psi_02 0];
sysT_ZV_02 = tf(stzv_num_02,stzv_den_02);
ustzv_num_02 = A*ones(length(t),1) - A*heaviside(t-psi_02) + (1-
A)*heaviside(t-Tau) - (1-A)*heaviside(t-Theta);
ustzv_02 = lsim(sysT_ZV_02,ustzv_num_02,t);
%ustzv_02 = ustzv_02/max(ustzv_02);


% Create the ZV shaper for zeta = 0. Zeta = 0
Tau_0 = pi;
uZV_0 = .5*ones(length(t),1) + .5*heaviside(t - Tau_0 + .00000001);
Theta_0 = 2*pi;
psi_0 = Theta_0-Tau_0;
stzv_num_0 = 1;
stzv_den_0 = [psi_0 0];
sysT_ZV_0 = tf(stzv_num_0,stzv_den_0);
ustzv_num_0 = .5*ones(length(t),1) - .5*heaviside(t-psi_0) + .5*heaviside(t-
Tau_0) - .5*heaviside(t-Theta_0);
ustzv_0 = lsim(sysT_ZV_0,ustzv_num_0,t);
%ustzv_0 = ustzv_0/max(ustzv_0);


% Also need SS model for Plant with zeta = 0;
tfnum_0 = 1;
tfden_0 = [1 0 1];
[Ac0, Bc0, Cc0, Dc0] = tf2ss(tfnum_0,tfden_0);
sscOL_0 = ss(Ac0,Bc0,Cc0,Dc0) ;

% Need to generate new responses for the modified T*ZV inputs
[yTZV_0, toTZV_0, xTZV_0] = lsim(sscOL_0,ustzv_0,t);
[yTZV_02, toTZV_02, xTZV_02] = lsim(sscOL,ustzv_02,t);

% figure();plot(t,ustzv_02,t,ustzv_0)
% figure();plot(t,yTZV_0,t,yTZV_02)



% Creating the shaper output for N = 0 and dt = .2;
kfin = t(end)/.2;
k0 = [0:kfin]';
d = ceil(Theta/.2);

wk = ones(length(k0),1);
wk0 = heaviside(k0+.0001 - d);
wk1 = heaviside(k0+1.001-d);
wk2 = heaviside(k0+2.001-d);
wk3 = heaviside(k0+3.001-d);

ukdDZV_020 = zeros(length(k0),1);

for j = 1:length(k0)
    jj = 1;
    N = 0;
```

```
    if N > 1
        av = avstore(:,3);
        C = Theta/( Theta*B + (1-B)*dt*(d - av(2) - 2*av(3) - 3*av(4)) );
        x(j+1,jj) = x(j,jj) + dt*( wk(j) - av(1)*wk0(j) - av(2)*wk1(j) -
 av(3)*wk2(j) - av(4)*wk3(j) );
    else

        C = Theta/( Theta*B + (1-B)*dt*(d) );
        x(j+1,jj) = x(j,jj) + dt*( wk(j) - wk0(j));
    end

    ukdDZV_020(j,jj) = C*( B*wk(j) + x(j,jj)*(1-B)/Theta );

end





% Varying the natural frequency
wn = [.1:.01:3];
U = zeros(length(wn),8);

for j = 1:length(wn)

    num = 1;
    den_0 = [1 0 wn(j)^2];
    den_02 = [1 2*.2*wn(j) wn(j)^2];
    hginf = 1/wn(j)^2;

    [A0,B0,C0,D0] = tf2ss(num,den_0);
    [A02,B02,C02,D02] = tf2ss(num,den_02);


    % SS models for zeta = 0, 0.2 and discrete for dt = 0.2, 0.9
    ss_0 = ss(A0,B0,C0,D0);
    ss_02 = ss(A02,B02,C02,D02);
    ss_02_02d = c2d(ss_02,.2);
    ss_02_09d = c2d(ss_02,.9);

    ystep_0 = step(ss_0,t);
    ystep_02 = step(ss_02,t);
    ystep_02_02d = step(ss_02_02d,k0*.2);
    ystep_02_09d = step(ss_02_09d,k*dt);

%     figure();
%     plot(t,ystep_0,t,ystep_02); hold on;
%     plot(.2*k0,ystep_02_02d,dt*k,ystep_02_09d)
%     legend('ss_0','ss_02','ss_02_02d','ss_02_09d')

    yZV_0 = lsim(ss_0,uZV_0,t);                          % ZV with normal input,
 zeta = 0
    yZV_02 = lsim(ss_02,uZV,t);                        % ZV with normal input,
 zeta = 0.2
```

```
    yTDZV_0 = lsim(ss_0,ustzv_0,t);                % DZV with convolved
 input, zeta = 0
    yDZV_02 = lsim(ss_02,uDZV,t);                  % DZV systm, nomral DZV
 input, zeta = 0.2
    yTZV_02 = lsim(ss_02,ustzv_02,t);                 % TZV with zeta - .2
    ydDZV_020 = lsim(ss_02_02d,ukdDZV_020,.2*k0);    % dDZV system with dt
 = .2, zeta = .2, N = 0;
    ydDZV0 = lsim(ss_02_09d,ukdDZV(:,1),k*dt);         % dDZV system with dt
 = .9, zeta = .2, N = 0
    ydDZV3 = lsim(ss_02_09d,ukdDZV(:,2),k*dt);         % dDZV system with dt
 = .9, zeta = .2, N = 3;


%      figure();
%      plot(t,yZV_0,t,yZV_02,t,yTDZV_0,t,yDZV_02,t,yTZV_02);
%      hold on;
%      plot(.2*k0,ydDZV_020,dt*k,ydDZV0,dt*k,ydDZV3)
%      legend('ZV,z=0','ZV,z=.2','TDZV,z=0','DZV,z=.2','TZV,z=.2','dDZV 2
 0','dDZV 9 0','dDZV 9 3','location','southeast')


    U(j,1) = ( max(yZV_0) - hginf )/( max(ystep_0) - hginf );
    U(j,2) = ( max(yZV_02) - hginf )/( max(ystep_02) - hginf );
    U(j,3) = ( max(yTDZV_0) - hginf )/( max(ystep_0) - hginf );
    U(j,4) = ( max(yDZV_02) - hginf )/( max(ystep_02) - hginf );
    U(j,5) = ( max(yTZV_02) - hginf )/( max(ystep_02) - hginf );

    U(j,6) = ( max(ydDZV_020) - hginf )/( max(ystep_02_02d) - hginf );
    U(j,7) = ( max(ydDZV0) - hginf )/( max(ystep_02_09d) - hginf );
    U(j,8) = ( max(ydDZV3) - hginf )/( max(ystep_02_09d) - hginf );



end

figure();
plot(wn,U(:,1:4)); hold on
plot(wn,U(:,5:8),'--')
title('Sensitivity of Filters to Variations in $\omega_n
$','interpreter','latex')
xlabel('$\frac{\omega_n}{\omega_0}$','interpreter','latex')
ylabel('U($\zeta$,$\omega_0$)','interpreter','latex')
legend('ZV,$\zeta=0$','ZV,$\zeta=0.2$','TDZV,$\zeta=0$','DZV,
$\zeta=0.2$','TZV,$\zeta=0.2$','dDZV,$\Delta t = .2$, N =
 0','dDZV,$\Delta t = .9$, N = 0','dDZV $\Delta t = .9$, N =
 3','location','best','interpreter','latex')
```



Published with MATLAB® R2022b