ELEC 4700 Assignment 4

Table of Contents

Circuit Modeling - Jinseng Vanderkloot 101031534 - Due: April 10, 2022

Question 1 and 2

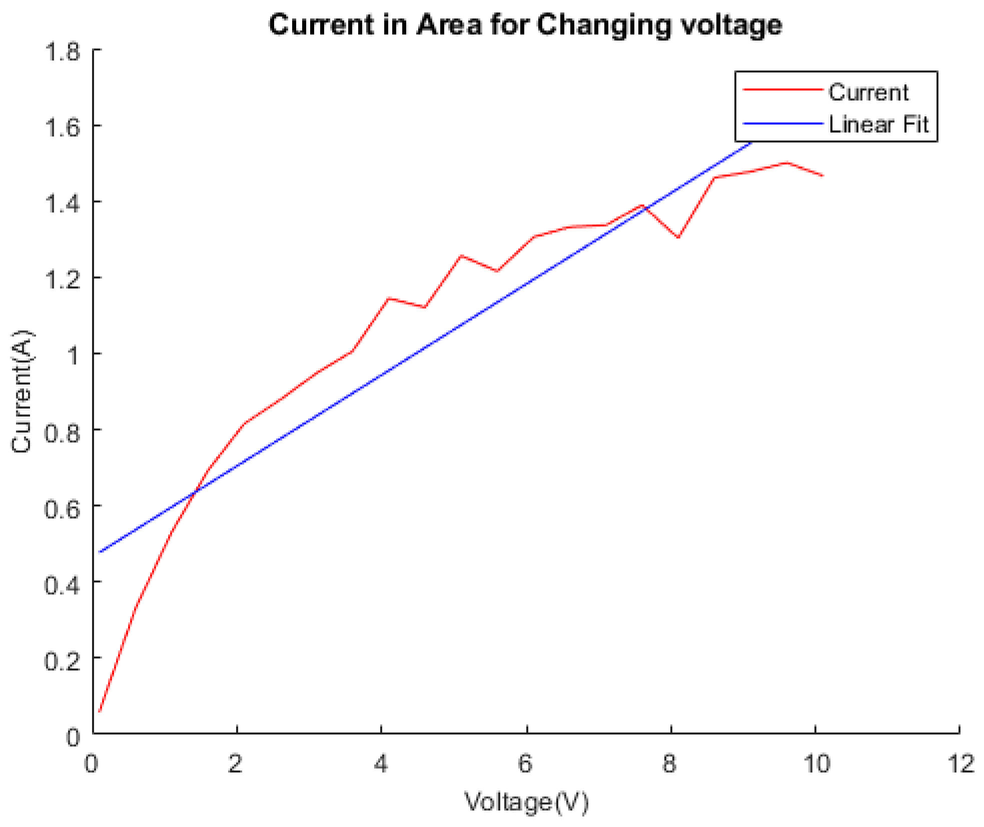
```
%Modify Assignment 3 Questions 2 to calculate the resistance for variable
%This will sweep the bottleneck area from 0.1V to 10V to calculate the
%resistance in the area and apply a linear fit to the IV curve


%Initial
m0 = 9.10938215e-31;            % electron mass
mn = 0.26*m0;                   % Effective mass
Temp = 300;                     % Inital Temp (K)
kb = 1.3806504e-23;             % Boltzmann constant
tmn = 0.2e-12;                  % Mean time between collision
q = 1.602e-19;                  % Charge of an Electron (Assignment 3 for
 force)

% Region Area
wArea = 200e-9;
lArea = 100e-9;

%Thermal Velocity (Question 1.A)
vt=sqrt((2*kb*Temp)/mn);        % Sim in 2D so (2*kb*Temp), 3D is (3*kb*Temp)

%Electron motion
numElec = 1000;                  %Number of simulated Electrons.
numEPlot = 30;                  %Number of plotted Electrons
dt = (lArea*wArea)/2;           %Typically 1/100 of region size
stepsTot = 200;                 %Total amount of steps (1000 was a long
 simulation)
tTot= stepsTot*dt;              %Total Simulation time
x = zeros(1,numElec);           %Inital X matrix
y = zeros(1,numElec);           %Inital y matrix
vx = zeros(1,numElec);          %Inital velocity x matrix
```

```
vy = zeros(1,numElec);           %Inital velocity y matrix
vtot = zeros(1,numElec);         %Inital velocity matrix
xfield = zeros(1,numElec);       %Inital Position in E field x
yfield = zeros(1,numElec);       %Inital Position in E field y
vxForce = zeros(1,numElec);       %Inital Force due to position in E field x
vyForce = zeros(1,numElec);       %Inital Force due to position in E field y

%Probability of Scatter
scatOn = 1;                      %Turn Scatter on (1) or off(0)
Pscatter = 1-exp(-dt/tmn);       %Scatter Equation
tScatter = zeros(1,numElec);

%Bottle Neck Boundary reduced to prevent bleeding
% X limit (no particle between X1 and X2 - Cosider Y limits)
boxX1=80e-9;
boxX2=120e-9;
% Y Limit (no particles between 0 and Y1 and Y2 to Y limit)
boxY1 = 40e-9;
boxY2 = 60e-9;

%Electric Field
x0 = 0.5; %voltage at right side of area increased to 0.5V to see huge effect
x1 = 0; %Voltage at left side of area
nx = wArea*1e9; % # of colums (changes to have same ratio as A1 area 2/1)
ny = lArea*1e9; % # of rows
xBox = 40; %Width of box (changes to same box ratio as A1 area l= 40 and w=40)
yBox = 40; %Hight of box
boxCond = 0.01;

%Electron Graph
for cnt = 1:numElec
    x(cnt)=rand()*wArea;
    y(cnt)=rand()*lArea;
    %If the electrons are place in the box, re-roll position
    while (x(cnt)>=boxX1 && x(cnt)<=boxX2 && (y(cnt)<=boxY1 ||
 y(cnt)>=boxY2))) %Relocate them if in boundary
        x(cnt)=rand()*wArea;
        y(cnt)=rand()*lArea;
    end
    vx(cnt)=(vt/sqrt(2))*randn();  % velocity * Gaussian dist
    vy(cnt)=(vt/sqrt(2))*randn();  % velocity * Gaussian dist
    vtot(cnt)= sqrt (vx(cnt)^2)+(vy(cnt)^2);
    colors= rand(numElec,3);
end

%Boundary Energy/Velocity loss coefficient (reduction in velocity =
%reduction in temprature)
vloss = 0.95;

vol = 0.1 : 0.5 : 10.1; %Sweep the device from 0.1V to 10V
vStep = 1;

% Main Loop
for voltage = vol
```

```
    V=A2_Function(nx, ny, xBox, yBox, boxCond, voltage, x1);
    Vmap = reshape(V, [ny, nx]);
    [Ex,Ey] = gradient(-Vmap);
    t=0;
    intCNT = 2;
    while t < tTot
        t = t + dt;

        %Store old position
        oldx = x;
        oldy = y;

        %Update to new position
        x(1:numElec) = x(1:numElec) + (vx(1:numElec).*dt);
        y(1:numElec) = y(1:numElec) + (vy(1:numElec).*dt);

        vtot(1:numElec)= sqrt ((vx(1:numElec).^2)+(vy(1:numElec).^2));

        for check = 1:numElec
            %Scatter
            if scatOn==1
                if Pscatter > rand()
                    vx(check)=(vt/sqrt(2))*randn();
                    vy(check)=(vt/sqrt(2))*randn();
                    tScatter(check)= 0; %If collision, time goes to 0
                else
                    tScatter(check)= tScatter(check) + dt; %track time
 increaing while no collision
                end
            end

            % Bring X and Y of particle location to values able to be tracked
            % in electric field mesh to apply field forces
            xfield(check) = round(x(check)*1e9);
            yfield(check) = round(y(check)*1e9);

            %Make sure field effects still apply to particles which go across
            %area
            if xfield(check)>200
                xfield(check) = 200;
            end
            if xfield(check)<1
                xfield(check) = 1;
            end
            if yfield(check)>100
                yfield(check) = 100;
            end
            if yfield(check)<1
                yfield(check) = 1;
            end
            %Apply field force to particles
            vxForce(check)= ((q *Ex(yfield(check),xfield(check))/(wArea/nx))/
mn)*dt; % V=(q*E)/mass F=q*E
```

```
            vyForce(check)= ((q *Ey(yfield(check),xfield(check))/(lArea/ny))/
mn)*dt; % V=(q*E)/mass F=q*E
            vx(check) = vx(check)+ vxForce(check);
            vy(check) = vy(check)+ vyForce(check);

            %Apply Boundary Conditions
            %If bottom contact, bounce off in opposite direction
            if (y(check)<=0)
                y(check) = 0;
                vy(check) = -vy(check);
            end
            %If top contact, bounce off in opposite directio
            if (y(check)>=lArea)
                y(check) = lArea;
                vy(check) = -vy(check);
            end
            %If left side of box, come out right side
            if(x(check)<=0)
                x(check) = x(check) + wArea;
            end
            %If right side of box, come out left side
            if(x(check)>=wArea)
                x(check) = x(check) - wArea;
            end

            %Apply bottle neck conditions
            %If contact on left walls of boundary (not in Gap)
            if (oldx(check)<boxX1 && x(check)>=boxX1 && (y(check)<= boxY1 ||
 y(check)>= boxY2) && oldx(check)>((1/5)*wArea) && oldx(check)<((4/5)*wArea))
                x(check)=boxX1;
                vx(check) = -(vx(check)*vloss);
            end
            %If contact on right walls of boundary (not in Gap)
            if (oldx(check)>boxX2 && x(check)<=boxX2 && (y(check)<= boxY1 ||
 y(check)>= boxY2) && oldx(check)>((1/5)*wArea)&& oldx(check)<((4/5)*wArea))
                x(check)=boxX2;
                vx(check) = -(vx(check)*vloss);
            end
            %If contact with bottom boundary in gap
            if (x(check)>boxX1 && x(check)< boxX2 && oldy(check)>boxY1 &&
 y(check)<= boxY1)
                y(check)= boxY1;
                vy(check) = -(vy(check)*vloss);
            end
            %If contact with top boundary in gap
            if (x(check)>boxX1 && x(check)< boxX2 && oldy(check)<boxY2 &&
 y(check)>=boxY2)
                y(check)=boxY2;
                vy(check) = -(vy(check)*vloss);
            end
        end
        %Current over time with Field - Mistake multiplying current by L and W
        %- Only multiply with W cause density is A/m * m
        Ix(:,intCNT)= q*10e19*mean(vx)* wArea; %Current = e * n * vd * Area
 (Particle density)

        intCNT = intCNT +1;
    end
    AvgIx (:,vStep) = mean (Ix);
    vStep = vStep +1;
end
figure(1); hold on;
p = polyfit(vol,AvgIx,1);
line = polyval(p,vol);
plot(vol,AvgIx,"r");
plot(vol,line,'b');
title('Current in Area for Changing voltage'),xlabel('Voltage(V)', 'FontSize',
 10), ylabel('Current(A)', 'FontSize', 10);
legend ('Current', 'Linear Fit');
fprintf('Linear equation is I = %e * V + %e \r', p(1), p(2));
R3 = 1/(p(1));
fprintf('The resistance for R3 = %e Ohms \r',R3);
hold on;
```

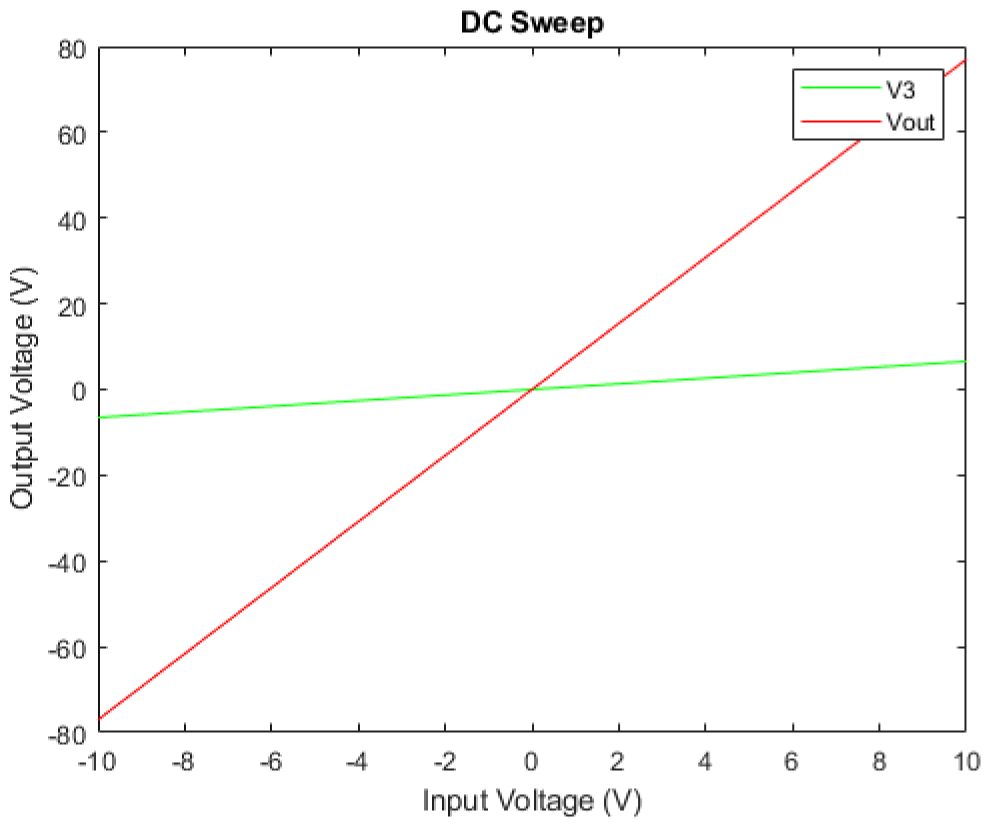
```
Linear equation is I = 1.194209e-01 * V + 4.654612e-01  The resistance for R3
 = 8.373740e+00 Ohms
```

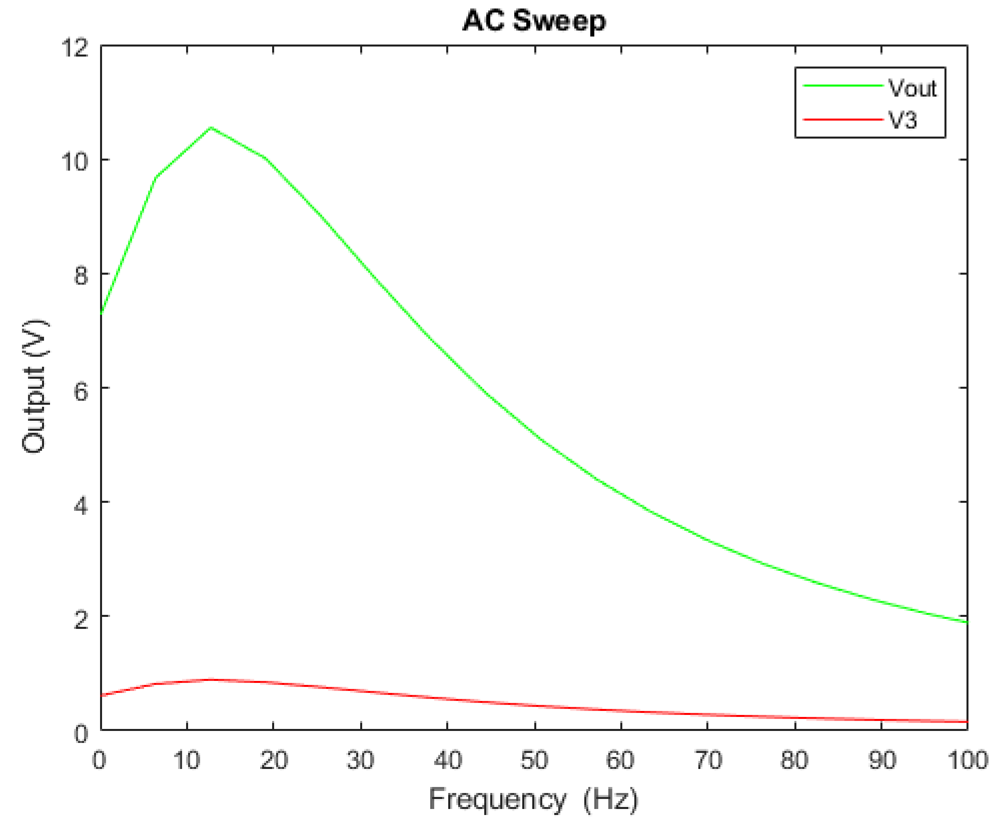


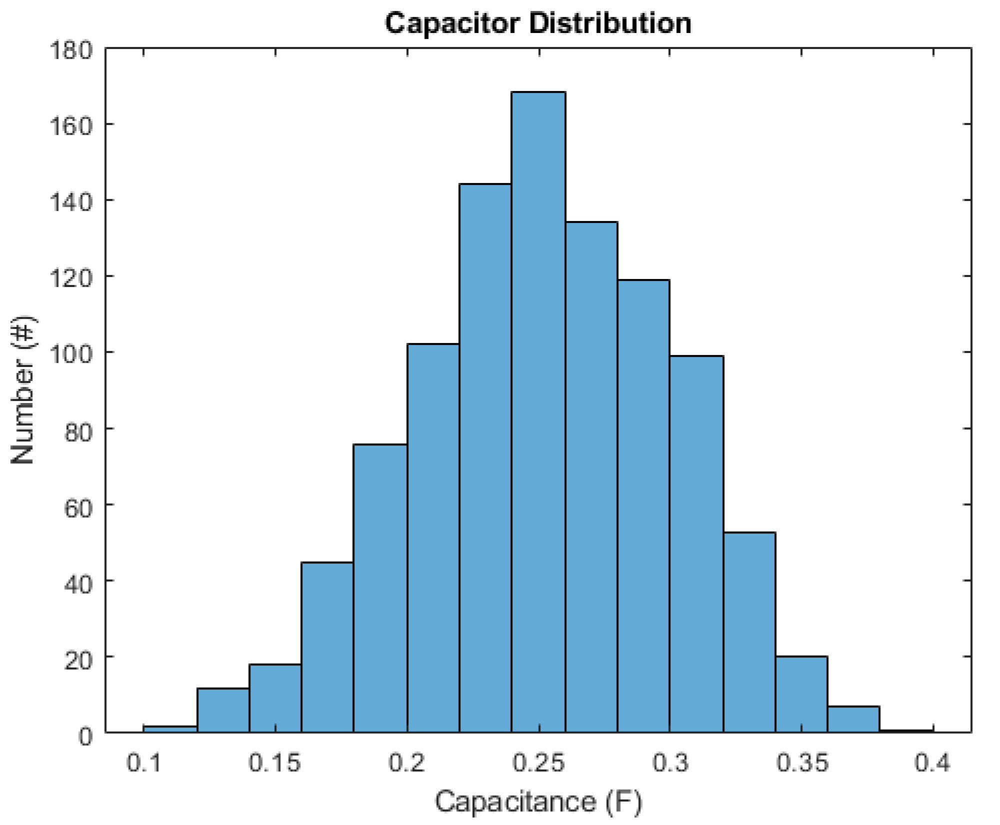
Question 3

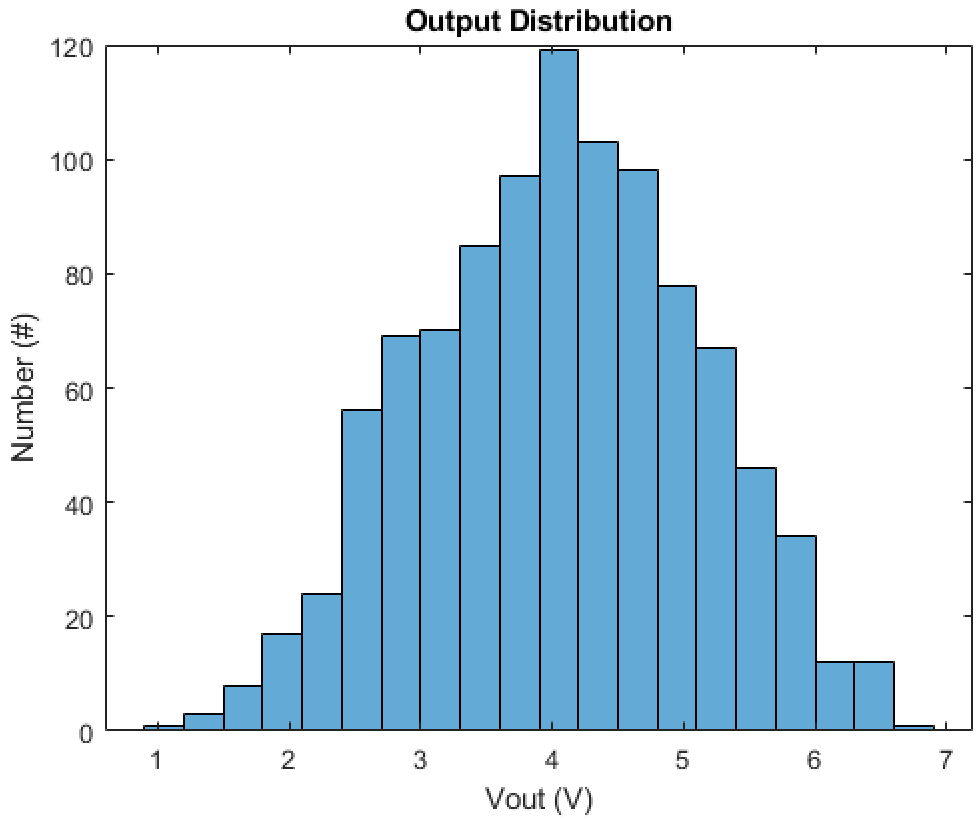
```
%This is PA7 with DC and AC sweep of circuit for A4

% The Differential Equations for the circuit are the following:

% 0 = Iin - G1(V1-V2) - C*d(V1-V2)/dt
% 0 = G1(V1-V2) + C*d(V1-V2)/dt - V2*G2 - IL
% 0 = IL - G3*V3
% 0 = I4 - G4*(V4-V5)
% 0 = G4*(V4-V5) - Go*Vo
% Vin = V1
% V4 = I3*Alpha = -Alpha*G3*V3
% dIL = dt(V2-V3)/L


% 1B Frequency Domain
        %Values to put into matrix

% 0 = Iin - G1(V2-V1) - jwC*(V2-V1)
        % V1*(-G1)   Iin   V2*(G1)
% 0 = G1(V2-V1) + jwC*(V2-V1) - V2*G2 - IL
        % V1(G1)   V2(-G1 + -G2)   -IL
% 0 = IL - G3*V3
        % IL    V3(-G3)
% 0 = I4 - G4*(V4-V5)
        % I4   V4(-G4)  V5 (G4)
% 0 = G4*(V4-V5) - Go*V5
        %V4(G4)    V5 (-G4-G0)
% Vin = V1
        %Vin(t)     V1
% V4 = I3*Alpha = -Alpha*G3*V3
        %V4    V3 (-Alpha*G3)
% IL = (1/jwL)*(V2-V3)
        %1/L   V2    -V3

C=0.25;
G1 = 1;
G2 = 1/2;
G3 = 1/8.5;
G4 = 1/0.1;
G0 = 1/1000;
L = 0.2;
alpha = 100;

%F = V1, V2, V3, V4, V5, Vin, I4, IL

CVec = [-C, C, 0, 0, 0, 0, 0, 0;
      C, -C, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, -L;]; %Not 1/L

GVec = [-G1, G1, 0, 0, 0, -1, 0, 0;
        G1, -G1-G2, 0, 0, 0, 0, 0, -1;
        0, 0, -G3, 0, 0, 0, 0, 1;
        0, 0, 0, -G4, G4, 0, 1, 0;
        0, 0, 0, G4, -G4-G0, 0, 0, 0;
        1, 0, 0, 0, 0, 0, 0, 0;
        0, 0, -alpha*G3, 1, 0, 0, 0, 0;
        0, 1, -1, 0, 0, 0, 0, 0;];

FVec = [0;0;0;0;0;1;0;0];

%DC Sweep
vin = linspace(-10,10);
threeV = zeros(size(vin));
outputV = zeros(size(vin));
for a = 1:size(vin,2)
    V = (GVec+CVec)\(FVec.*vin(a));
    threeV(a) = V(3);
    outputV(a) = V(5);
end

figure('name',"DC Sweep")
plot(vin,threeV,'g');
hold on
plot(vin,outputV,'r');
hold off
title('DC Sweep');
xlabel('Input Voltage (V)');
ylabel('Output Voltage (V)');
legend('V3','Vout');

% AC Sweep
freq = linspace(0.01, 100);
threeV = zeros(size(vin));
outputV = zeros(size(freq));
for a = 1:size(freq,2)
    w =(2*pi*freq(a));
    jw = 1i*w; % 1i makes value complex 1 -> 0+1j
    V = (GVec+jw*CVec)\FVec;
    outputV(a) = V(5);
    threeV(a) = V(3);
end

figure("Name","AC Sweep")
plot(2*pi*freq,real(outputV),'g'); %Real() gets rid of imag value warning
xlim([0 100])
hold on
plot(2*pi*freq,real(threeV),'r');
hold off
title('AC Sweep')
xlabel('Frequency  (Hz)')
```

```
ylabel('Output (V)')
legend('Vout','V3');


%Capacitor normal Distribution
C = 0.25 + 0.05 * randn(1000,1); %Random C values
outputV = zeros(size(C));
for a = 1:size(C,1)
    CVec(1:2,1:2)=C(a); %CVec does not auto update for changing C so need this
 statement
    V = (GVec+(1i*pi)*CVec)\FVec;
    outputV(a) = V(5);
end

figure("Name","Capacitor Distribution")
histogram(C);
title('Capacitor Distribution');
xlabel('Capacitance (F)');
ylabel('Number (#)');

figure("Name","Output Distribution");
histogram(real(outputV));
title('Output Distribution');
xlabel('Vout (V)');
ylabel('Number (#)');
```




AC Sweep
Output (V)
Frequency (Hz)
Vout
V3
0
2
4
6
8
10
12
0
10
20
30
40
50
60
70
80
90
100

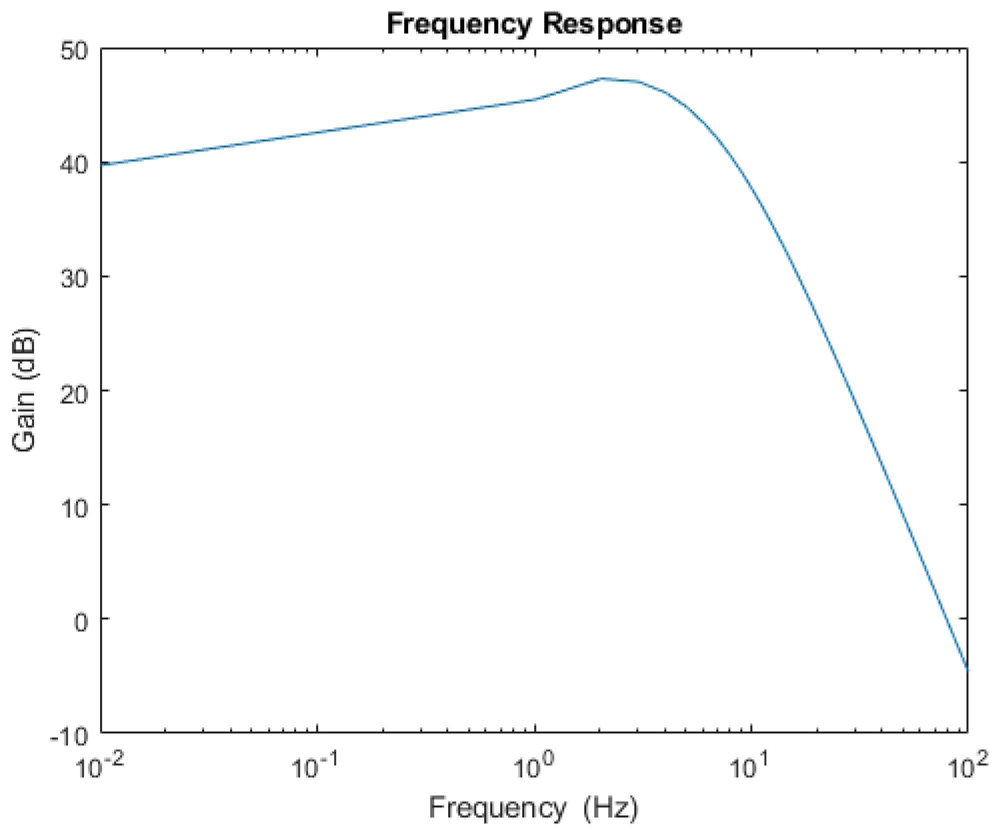

Capacitor Distribution
Number (#)
Capacitance (F)
0
20
40
60
80
100
120
140
160
180
0.1
0.15
0.2
0.25
0.3
0.35
0.4



Question 4 - Transient Circuit Analysis

```
%Simulate the circuit response for 1s in 1000 steps for a step function, a
%sine function and a gaussian pulse.

C=0.25;
G1 = 1;
G2 = 1/2;
G3 = 1/8.5;
G4 = 1/0.1;
G0 = 1/1000;
L = 0.2;
alpha = 100;

% C and G Matrix
%F = V1, V2, V3, V4, V5, Vin, I4, IL

CVec = [-C, C, 0, 0, 0, 0, 0, 0;
      C, -C, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, -L;];
```

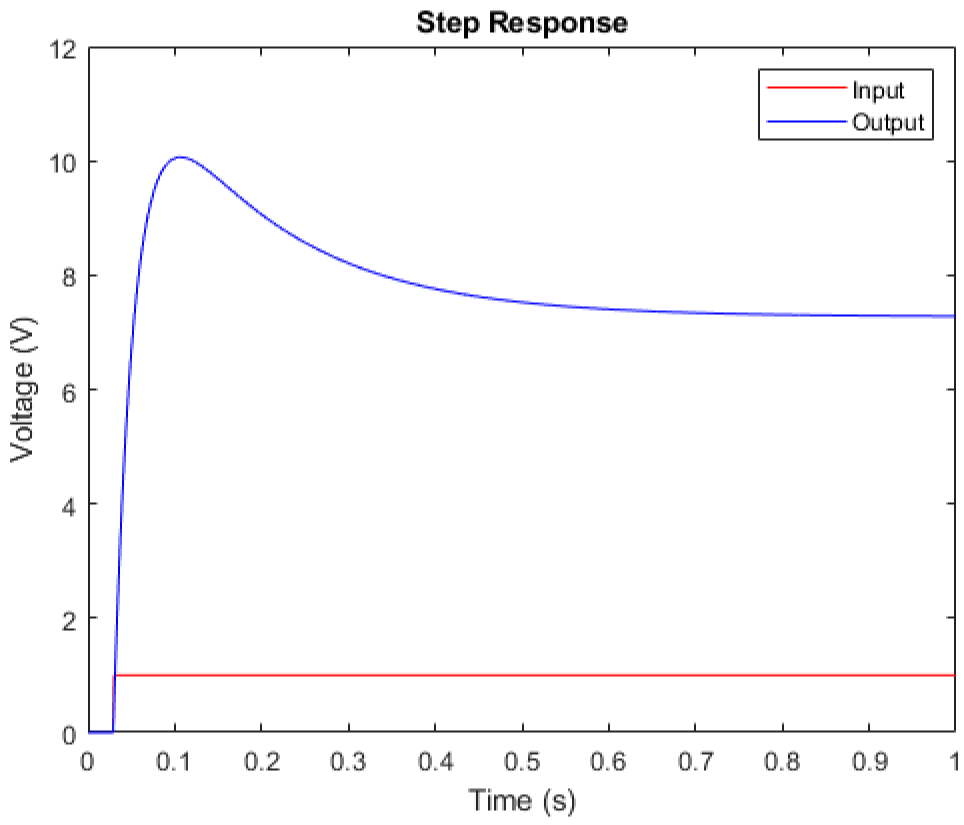
```
GVec = [-G1, G1, 0, 0, 0, -1, 0, 0;
        G1, -G1-G2, 0, 0, 0, 0, 0, -1;
        0, 0, -G3, 0, 0, 0, 0, 1;
        0, 0, 0, -G4, G4, 0, 1, 0;
        0, 0, 0, G4, -G4-G0, 0, 0, 0;
        1, 0, 0, 0, 0, 0, 0, 0;
        0, 0, -alpha*G3, 1, 0, 0, 0, 0;
        0, 1, -1, 0, 0, 0, 0, 0;];

FVec = [0;0;0;0;0;1;0;0];

vin = linspace(-10,10);
outputV = zeros(size(vin));
freq = linspace(0.01, 100);

for a = 1:size(freq,2)
    w =(2*pi*freq(a));
    jw = 1i*w; % 1i makes value complex 1 -> 0+1j
    V = (GVec+jw*CVec)\FVec;
    outputV(a) = V(5);
end

%Plot Gain
figure('Name','Circuit Frequency Response');
semilogx(freq, 20*log(abs(outputV)));
title('Frequency Response');
xlabel('Frequency  (Hz)');
ylabel('Gain (dB)');

% Question 4a) By inpection of the frequency response of the circuit, the
% result is a low pass filter with the cut off frequency being around 2Hz.

%Question 4b) The circuit has 40-46dB of gain for near DC values (any
%freqeuncy less than 2Hz). There is a slight slope for the gain so it is
%not ideal.

%Question 4c)
%   C(dV/dt) + GV = F
%   C[(Vn - Vn-1)/ delta_t] + G Vn = F(tn)
%   C (Vn/delta_t) - C (Vn-1 / delta_t) + G Vn = F(tn)
%   Vn = [C(Vn-1 / delta_t) + F(tn)] / (Vn/delta_t)
```

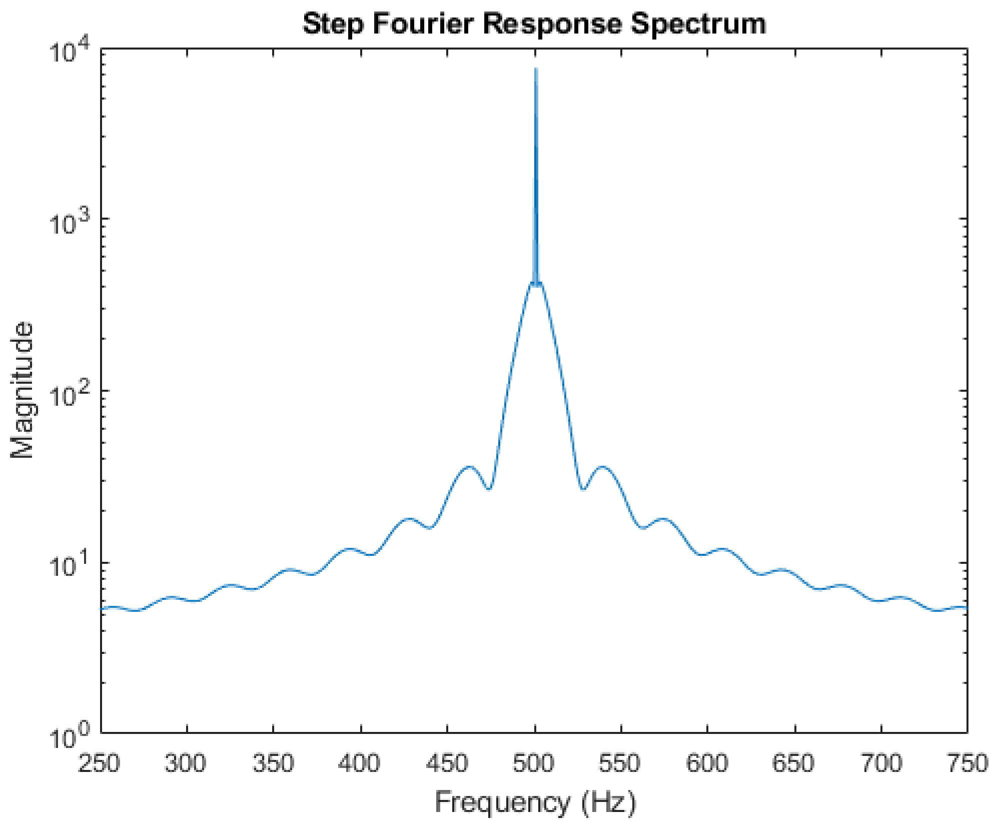


Q4d - Step Response

```
t = 0:0.001:1;                                 %Time 1000 steps 0 to 1
F1 = [0;0;0;0;0;0;0;0];                        % initial values
vout = zeros(size(GVec,1),size(t,2));  % output vector
vin= zeros(1,size(t,2));                      % input vector
vin(1,1) = 0;                                  %initial input value

%Loop through time
for n=1:size(t,2)-1
    vout(:,n) = F1;
    if n<30                         %Produce pulse
        vin(n+1)=0;
    else
        vin(n+1)=1;
    end
    F2 = FVec * vin(n+1);             % Capturing next input
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
vout(:,n+1) = F1;

figure("Name","Step Response")
plot(t, vin,'r');
hold on
plot(t, vout(5,:),'b');
```

```
title('Step Response');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Step Fourier Response Spectrum")
semilogy(abs(fftshift(fft(vout(5,:)))))
title('Step Fourier Response Spectrum');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750]);
```

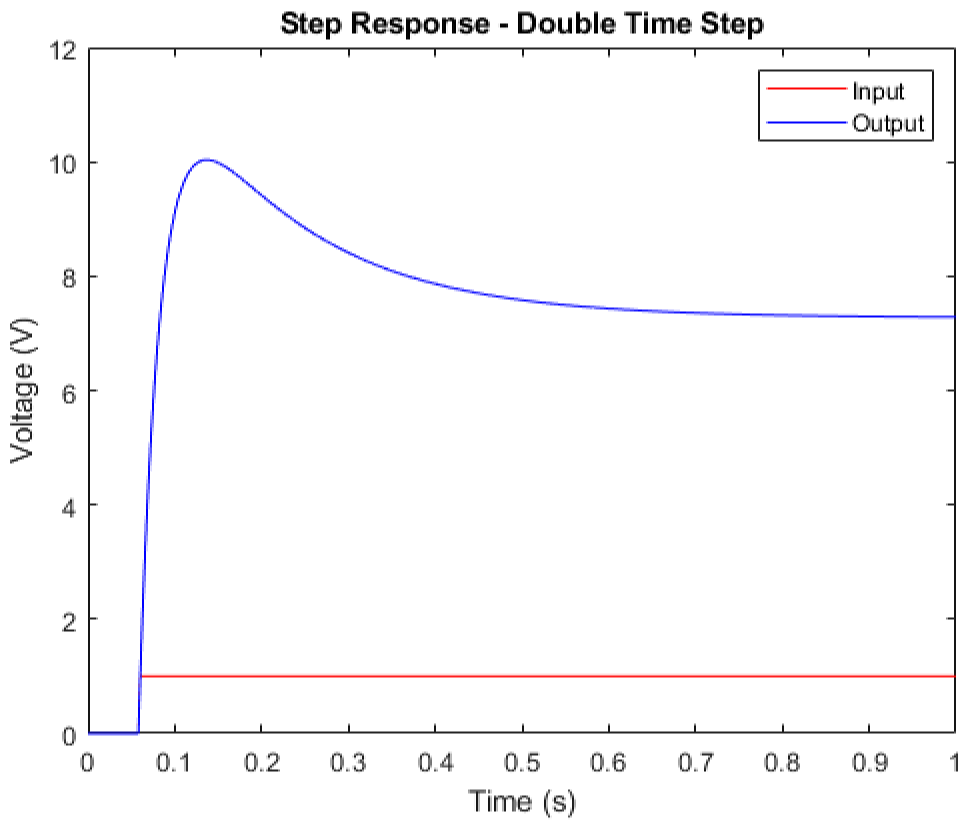


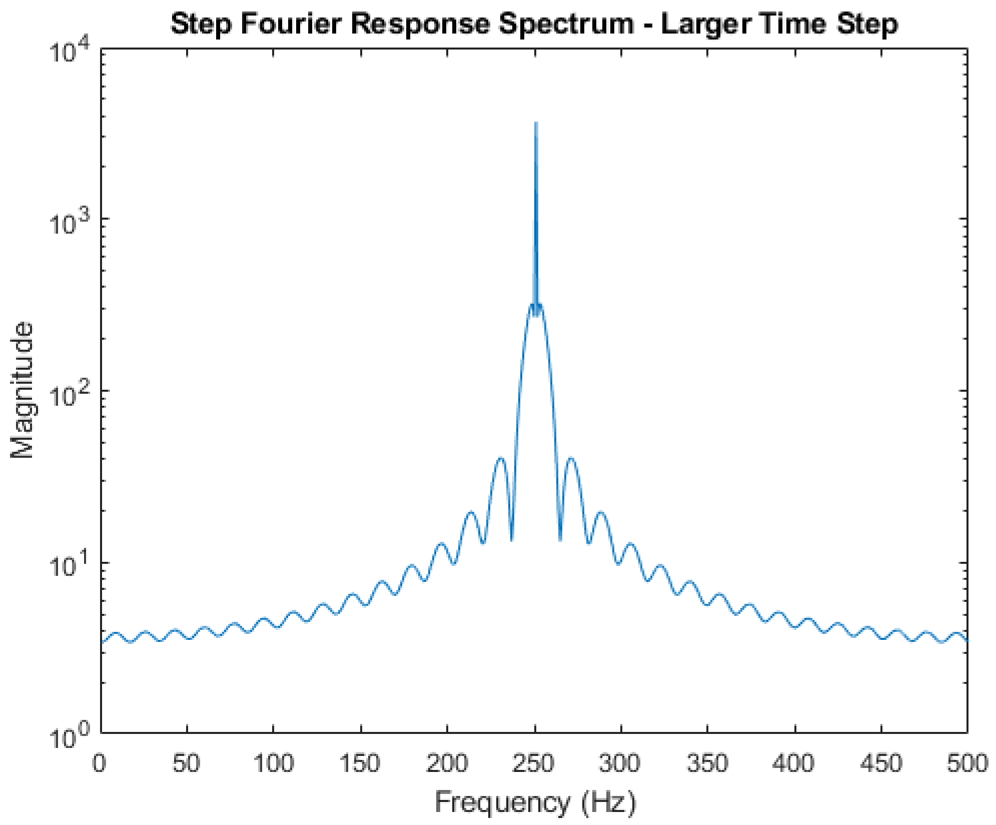


Q4d - Step Responce increasing time step

```
t = 0:0.002:1;                                %Time 1000 steps 0 to 1
F1 = [0;0;0;0;0;0;0;0];                       % initial values
vout = zeros(size(GVec,1),size(t,2));        % output vector
vin= zeros(1,size(t,2));                     % input vector
vin(1,1) = 0;                                 %initial input value

%Loop through time
for n=1:size(t,2)-1
    vout(:,n) = F1;
    if n<30                              %Produce pulse
        vin(n+1)=0;
    else
        vin(n+1)=1;
    end
    F2 = FVec * vin(n+1);                  % Capturing next input
    F1 = ((CVec./0.002) + GVec)\((CVec./0.002)*F1 + F2);
end
vout(:,n+1) = F1;

figure("Name","Step Response - Double Time Step")
plot(t, vin,'r');
hold on
plot(t, vout(5,:),'b');
```

```
title('Step Response - Double Time Step');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Step Fourier Response Spectrum - Larger Time Step")
semilogy(abs(fftshift(fft(vout(5,:)))))
title('Step Fourier Response Spectrum - Larger Time Step');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([0 500])

% When the time step doubles, the reaction time of the circuit is half and
% the base freqeuncy is half as well for the step response (250Hz).
```



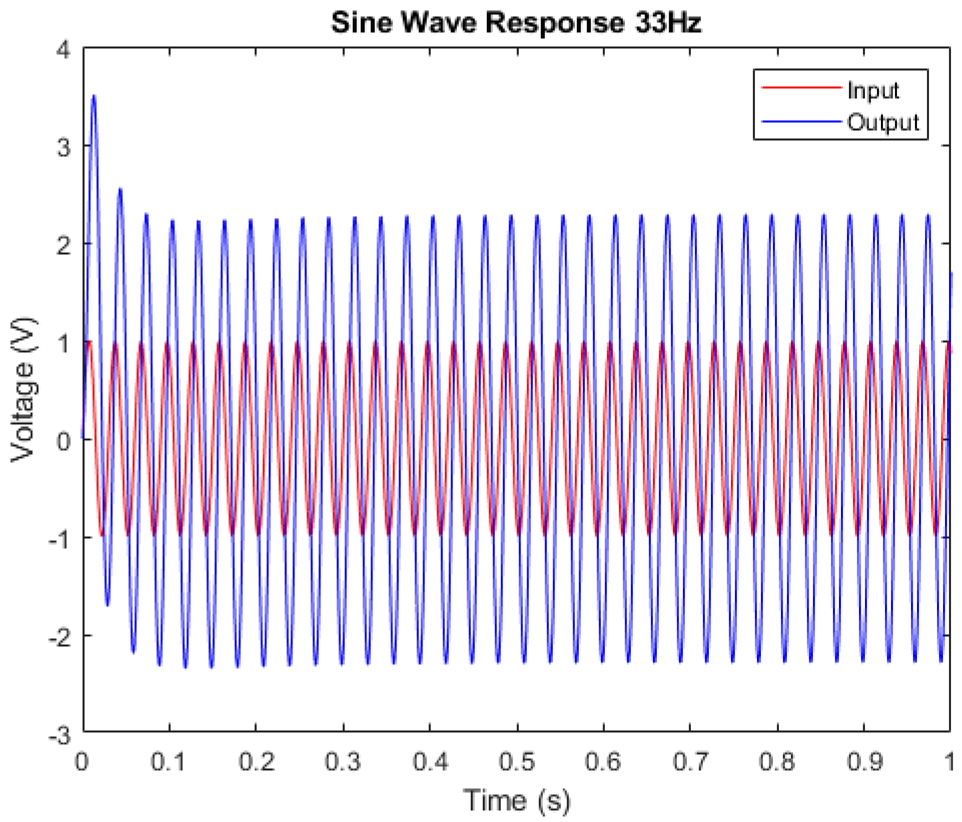


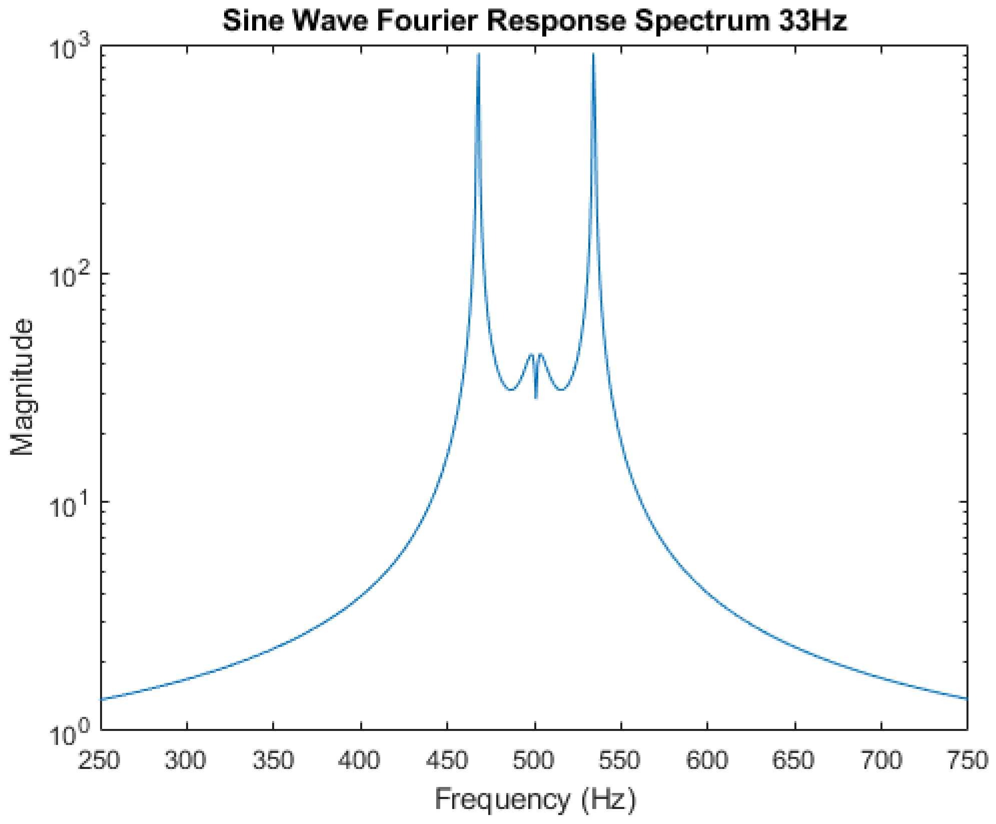
Q4d - Sine wave @ Differnt frequency

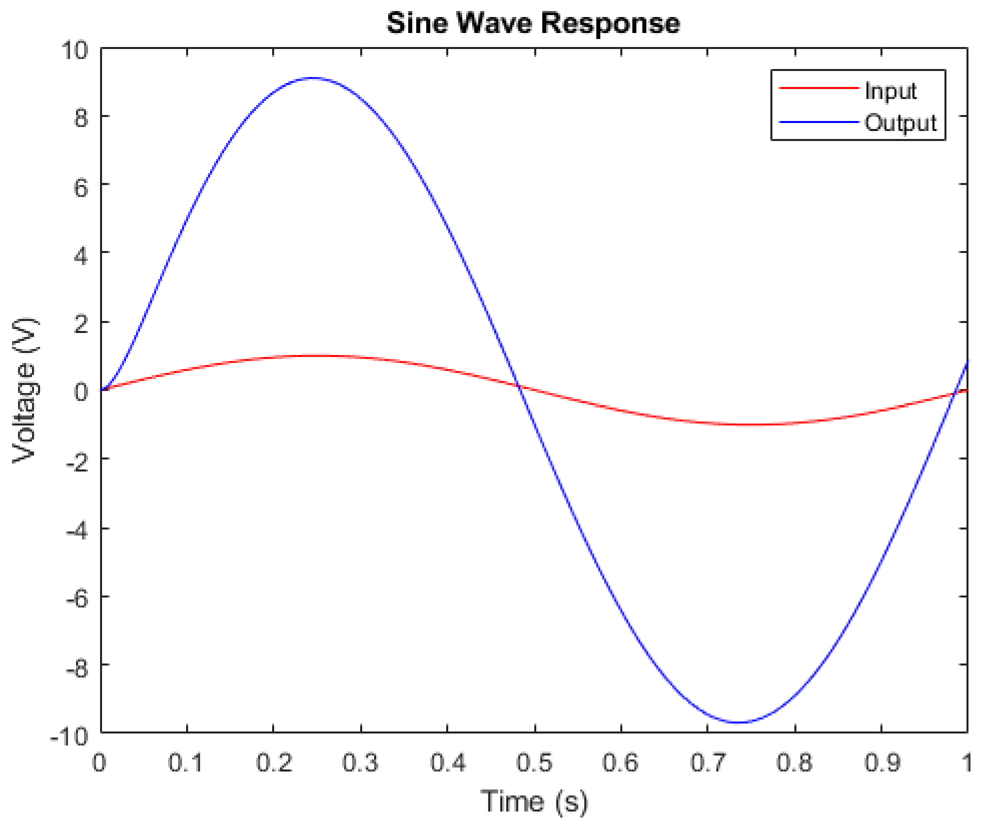
```
t = 0:0.001:1;
freq2 = 1/0.03; % Frequency 33Hz
F1 = [0;0;0;0;0;0;0;0];                % initial values
vout2 = zeros(size(GVec,1),size(t,2)); % Vector to hold outputs
vin2 = zeros(1,size(t,2));           % Vector to hold inputs
vin2(1,1) = sin(2*pi*freq2*t(1));

%Loop through time
for n=1:size(t,2)-1
    vout2(:,n) = F1;
    vin2(1,n+1) = sin(2*pi*freq2*t(n+1));
    F2 = FVec * vin2(1,n+1);
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
vout2(:,n+1) = F1;

figure("Name","Sine Wave Response 33Hz")
plot(t, vin2,'r');
hold on
plot(t, vout2(5,:),'b');
title('Sine Wave Response 33Hz');
xlabel('Time (s)');
ylabel('Voltage (V)');
```

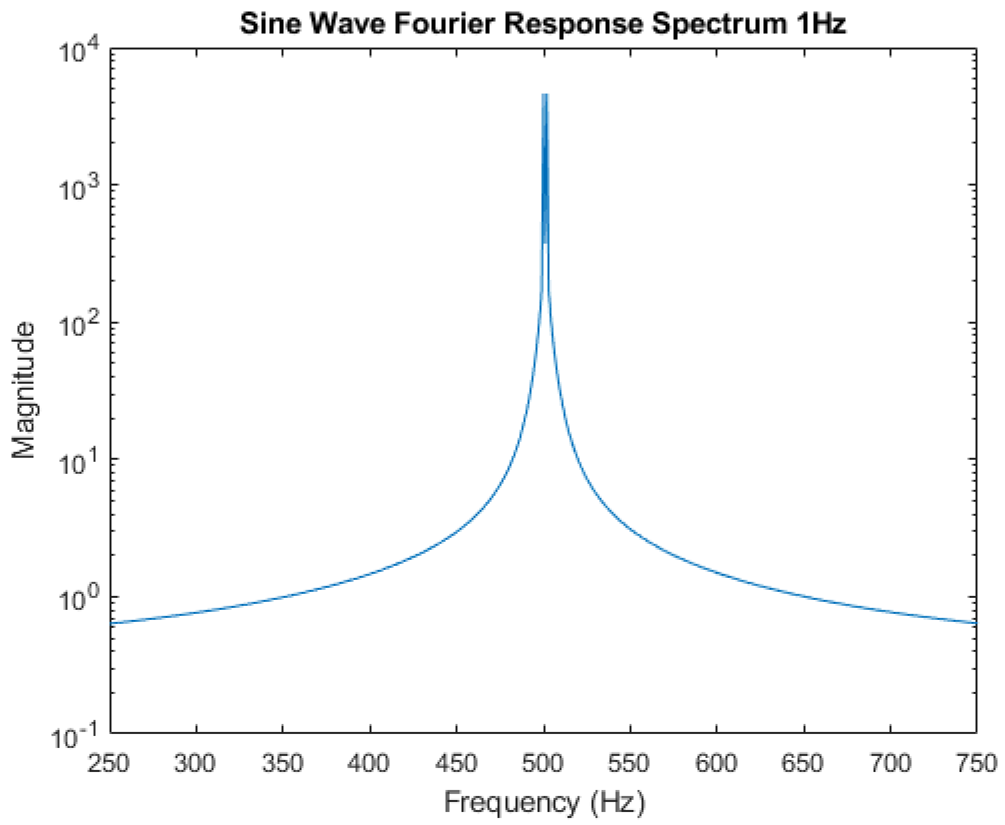
```
legend({'Input','Output'});

figure("Name","Sine Wave Fourier Response Spectrum 33Hz")
semilogy(abs(fftshift(fft(vout2(5,:)))))
title('Sine Wave Fourier Response Spectrum 33Hz');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750])

t = 0:0.001:1;
freq2 = 1; % Frequency in 1Hz
F1 = [0;0;0;0;0;0;0;0];                  % initial values
vout2 = zeros(size(GVec,1),size(t,2)); % Vector to hold outputs
vin2 = zeros(1,size(t,2));          % Vector to hold inputs
vin2(1,1) = sin(2*pi*freq2*t(1));

%Loop through time
for n=1:size(t,2)-1
    vout2(:,n) = F1;
    vin2(1,n+1) = sin(2*pi*freq2*t(n+1));
    F2 = FVec * vin2(1,n+1);
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
vout2(:,n+1) = F1;

figure("Name","Sine Wave Response 1Hz")
plot(t, vin2,'r');
hold on
plot(t, vout2(5,:),'b');
title('Sine Wave Response');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Sine Wave Fourier Response Spectrum 1Hz")
semilogy(abs(fftshift(fft(vout2(5,:)))))
title('Sine Wave Fourier Response Spectrum 1Hz');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750])

freq2 = 1/0.01; % Frequency in 100Hz
F1 = [0;0;0;0;0;0;0;0];                  % initial values
vout2 = zeros(size(GVec,1),size(t,2)); % Vector to hold outputs
vin2 = zeros(1,size(t,2));          % Vector to hold inputs
vin2(1,1) = sin(2*pi*freq2*t(1));

%Loop through time
for n=1:size(t,2)-1
    vout2(:,n) = F1;
    vin2(1,n+1) = sin(2*pi*freq2*t(n+1));
    F2 = FVec * vin2(1,n+1);
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
```

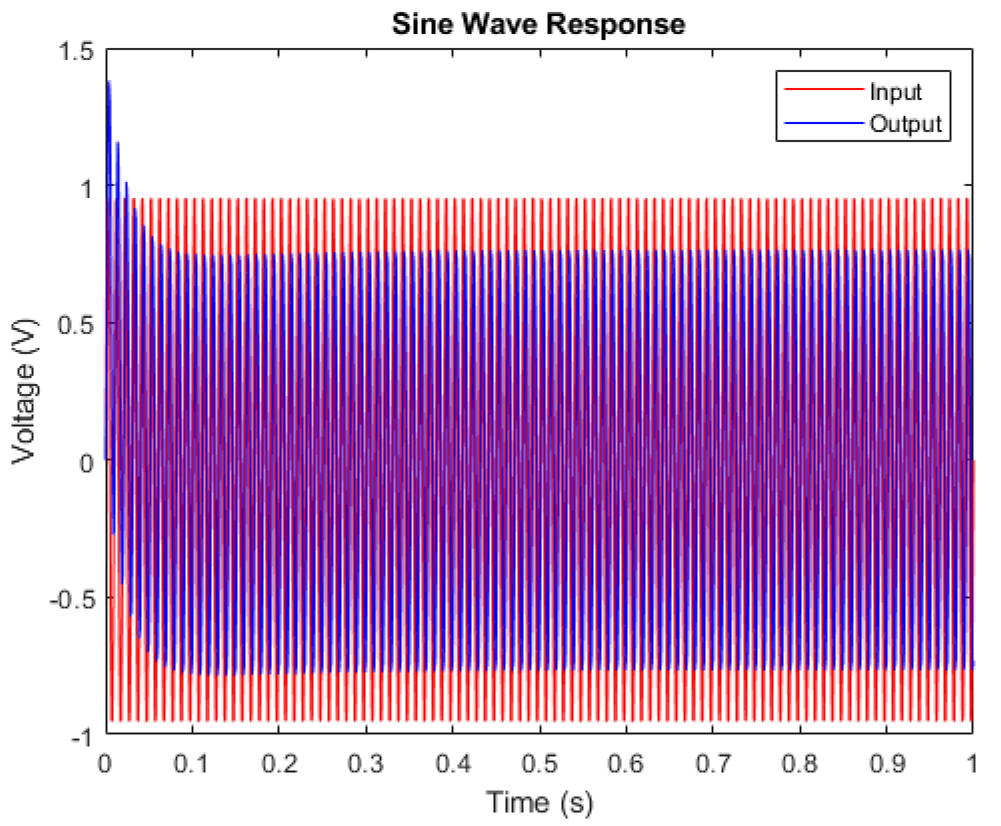
```
vout2(:,n+1) = F1;

figure("Name","Sine Wave Response 100Hz")
plot(t, vin2,'r');
hold on
plot(t, vout2(5,:),'b');
title('Sine Wave Response');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Sine Wave Fourier Response Spectrum 100Hz")
semilogy(abs(fftshift(fft(vout2(5,:)))))
title('Sine Wave Fourier Response Spectrum 100Hz');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750])

% Since the circuit is a low pass amplifier filter, the Gain of the circuit
% is optimal at the cutoff frquency. The 33Hz case had a gain of about
% double. increaseing the frequnecy to 100Hz reduces the output gain to a
% decimal value so the ouput is smaller than the input. Any higher and
% therer would be no ouput as the signal would be completely blocked. Any
% case below the cutoff freqeuncy would achive the maximum gain of the
% circuit.
```

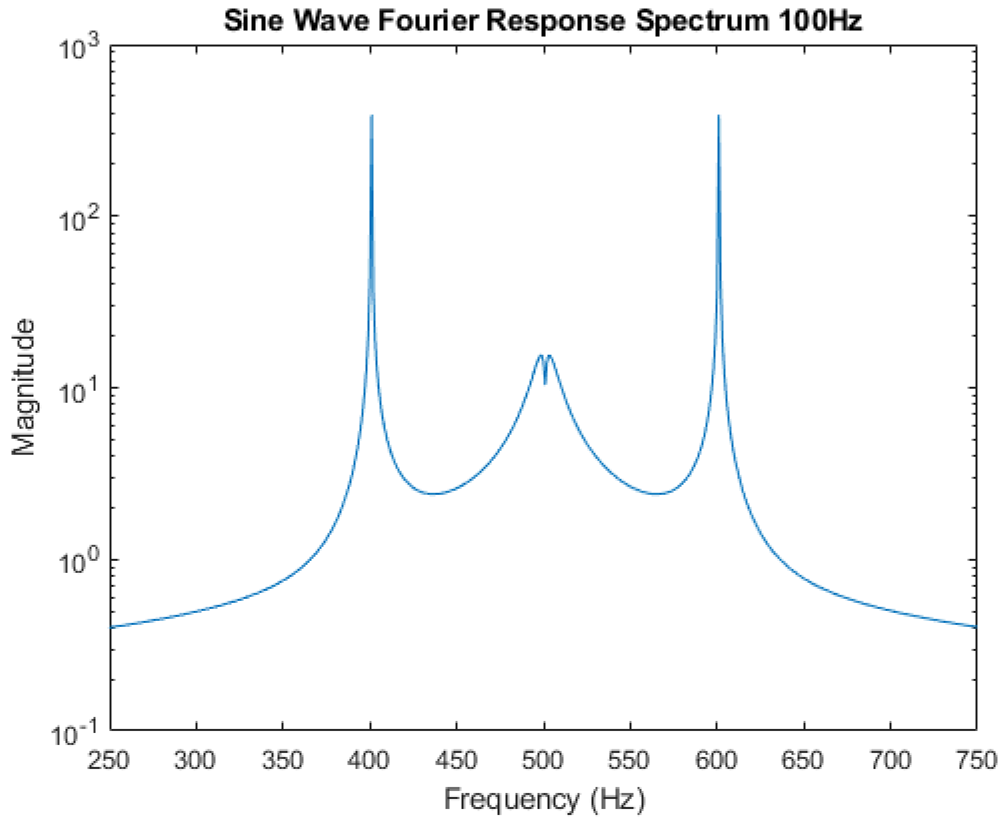



Sine Wave Fourier Response Spectrum 33Hz
Magnitude
Frequency (Hz)
10^3
10^2
10^1
10^0
250
300
350
400
450
500
550
600
650
700
750


Sine Wave Response
Voltage (V)
Time (s)
Input
Output
10
8
6
4
2
0
-2
-4
-6
-8
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Sine Wave Fourier Response Spectrum 1Hz
Magnitude
Frequency (Hz)
10^4
10^3
10^2
10^1
10^0
10^-1
250
300
350
400
450
500
550
600
650
700
750

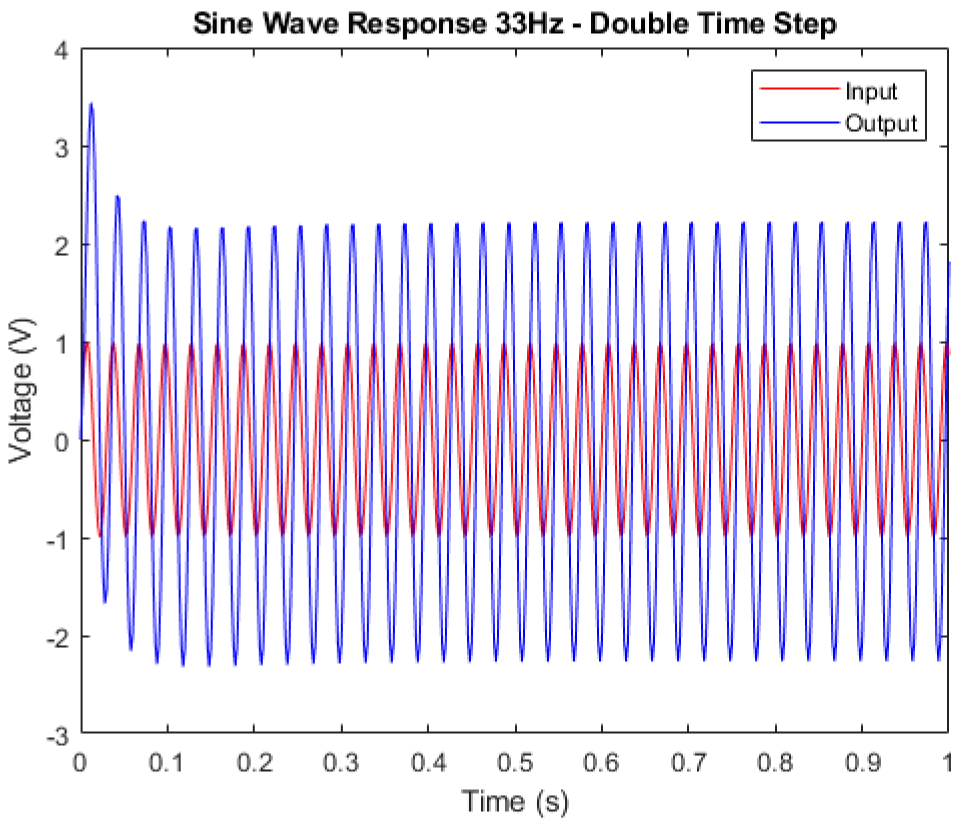

Sine Wave Response
Input
Output
Voltage (V)
Time (s)
1.5
1
0.5
0
-0.5
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

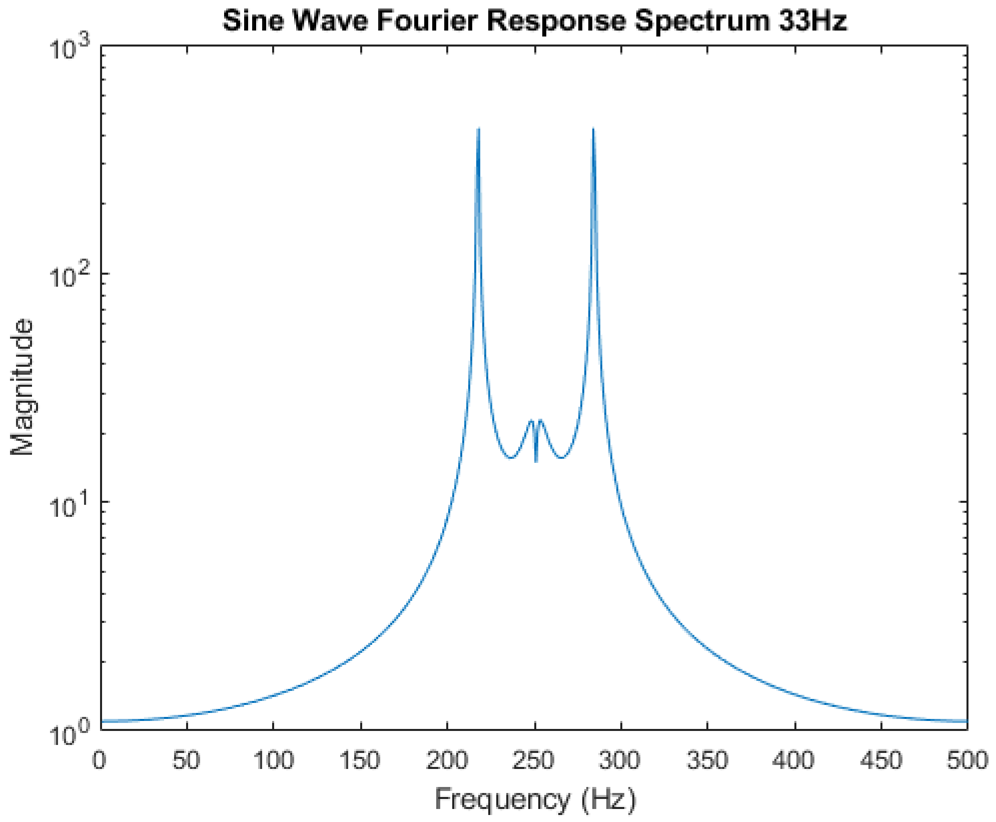


Q4d - Sine wave @ Different frequency increasing time step

```
t = 0:0.002:1;
freq2 =1/0.03; % Frequency 33Hz
F1 = [0;0;0;0;0;0;0;0];                         % initial values
vout2 = zeros(size(GVec,1),size(t,2)); % Vector to hold outputs
vin2 = zeros(1,size(t,2));           % Vector to hold inputs
vin2(1,1) = sin(2*pi*freq2*t(1));

%Loop through time
for n=1:size(t,2)-1
    vout2(:,n) = F1;
    vin2(1,n+1) = sin(2*pi*freq2*t(n+1));
    F2 = FVec * vin2(1,n+1);
    F1 = ((CVec./0.002) + GVec)\((CVec./0.002)*F1 + F2);
end
vout2(:,n+1) = F1;

figure("Name","Sine Wave Response 33Hz - Double Time Step")
plot(t, vin2,'r');
hold on
plot(t, vout2(5,:),'b');
title('Sine Wave Response 33Hz - Double Time Step');
```

```
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Sine Wave Fourier Response Spectrum 33Hz")
semilogy(abs(fftshift(fft(vout2(5,:)))))
xlabel('Frequency (Hz)');
title('Sine Wave Fourier Response Spectrum 33Hz');
ylabel('Magnitude');
xlim([0 500])

% When the time step doubles, the reaction time of the circuit is half and
% the base freqeuncy is half as well for the AC response (250Hz base).
```

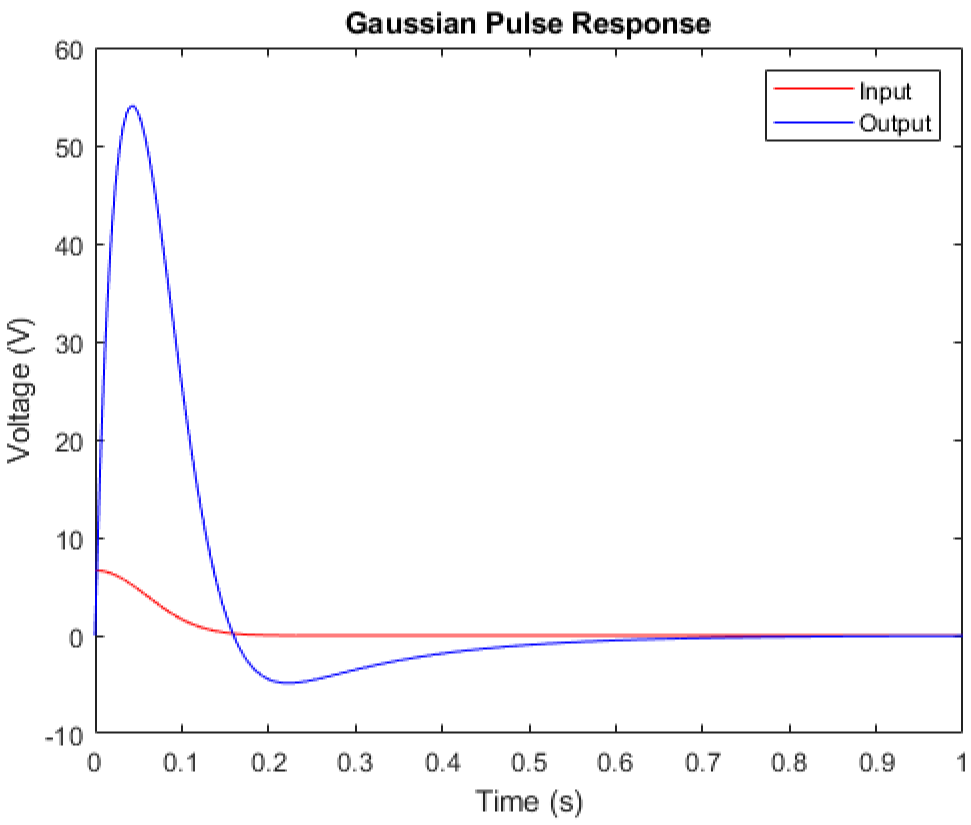


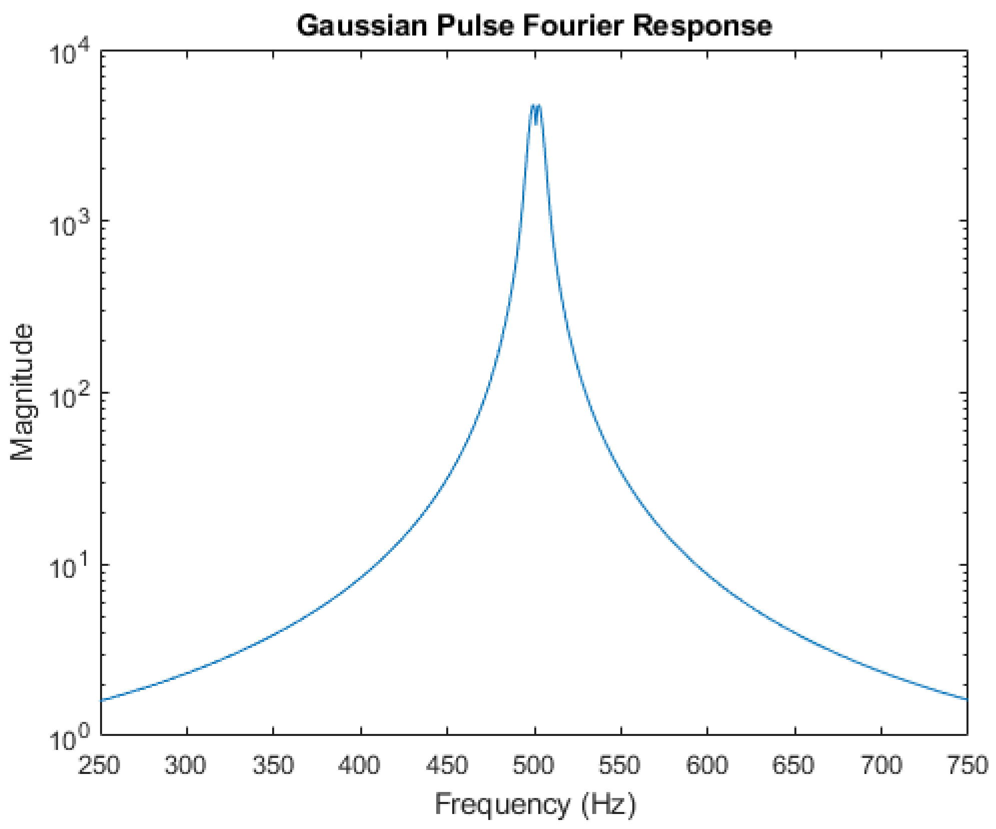


Q4d - Gaussian Pulse Response

```
F1 = [0;0;0;0;0;0;0;0];                         % initial values
vout3 = zeros(size(GVec,1),size(t,2)); % Vector to hold outputs
vin3 = zeros(1,size(t,2));          % Vector to hold inputs

t = 0:0.001:1;

%https://www.gaussianwaves.com/2014/07/generating-basic-signals-gaussian-
pulse-and-power-spectral-density-using-fft/
std = 0.03;
delay = 0.06^2;
vin3(1,1) = 1/(sqrt(2*pi*delay))*(exp(-t(1).^2/(2*delay)));

for n=1:size(t,2)-1
    vout3(:,n) = F1;
    vin3(1,n+1) = 1/(sqrt(2*pi*delay))*(exp(-t(n+1).^2/(2*delay)));
    F2 = FVec * vin3(1,n+1);
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
vout3(:,n+1) = F1;

figure("Name","Gaussian Pulse Response")
plot(t, vin3,'r');
hold on
```

```
plot(t, vout3(5,:),'b');
title('Gaussian Pulse Response');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Gaussian Pulse Fourier Response Spectrum")
semilogy(abs(fftshift(fft(vout3(5,:)))))
title('Gaussian Pulse Fourier Response');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750])
```



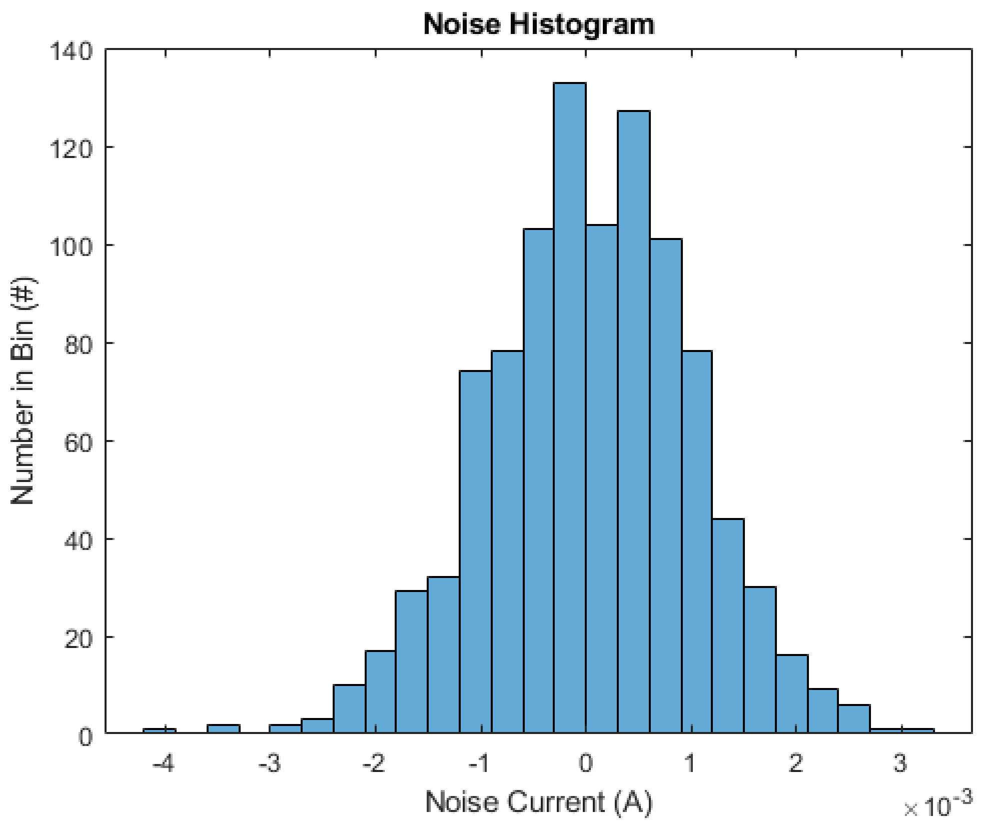


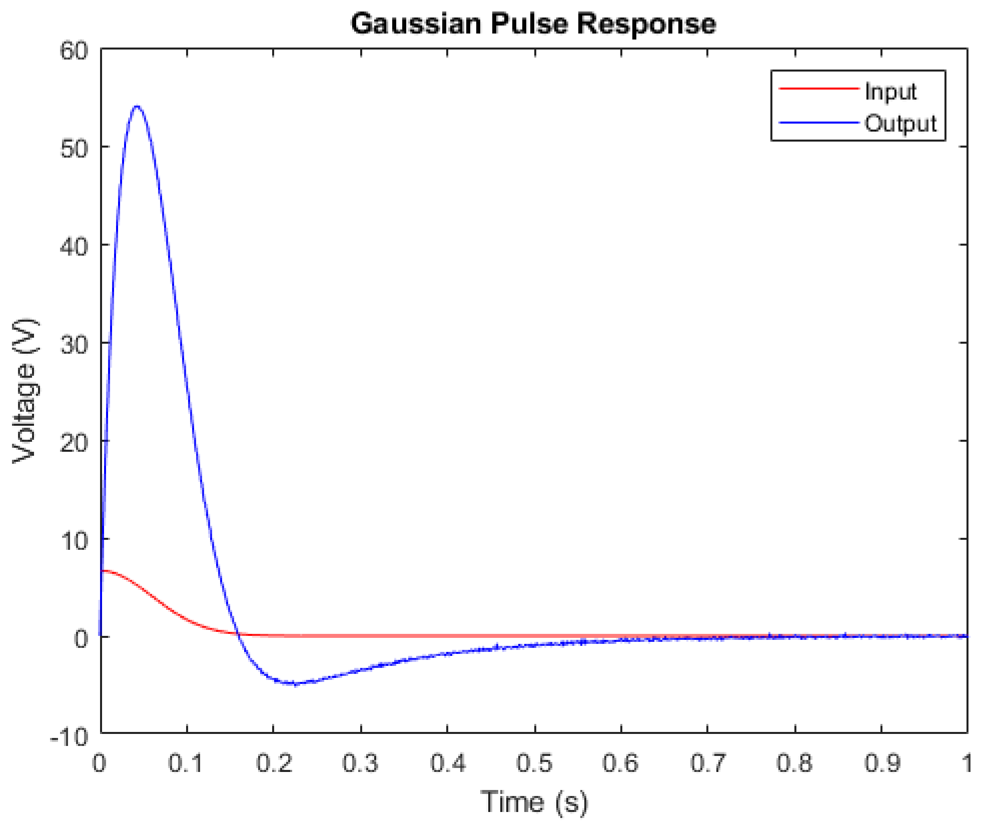
Question 5 - Introduction of circuit noise

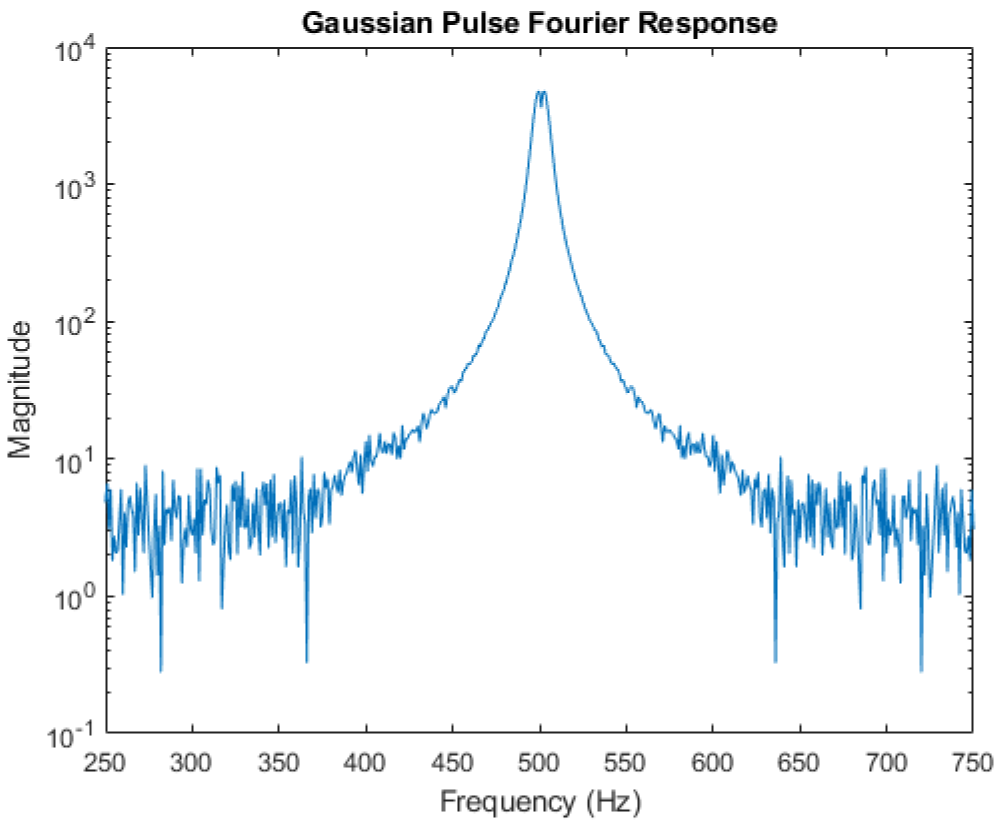
```
C=0.25;
Cn = 0.00001;
G1 = 1;
G2 = 1/2;
G3 = 1/8.5;
G4 = 1/0.1;
G0 = 1/1000;
L = 0.2;
alpha = 100;

%F = V1, V2, V3, V4, V5, Vin, I4, IL

% 0 = Iin - G1(V1-V2) - C*d(V1-V2)/dt
% 0 = G1(V1-V2) + C*d(V1-V2)/dt - V2*G2 - IL
% 0 = IL - G3*V3 - V3*Cn + In
% 0 = I4 - G4*(V4-V5)
% 0 = G4*(V4-V5) - Go*Vo
% Vin = V1
% V4 = I3*Alpha = -Alpha*G3*V3
% dIL = dt(V2-V3)/L

% Updated C Matrix
```

```
CVec = [-C, C, 0, 0, 0, 0, 0, 0;
      C, -C, 0, 0, 0, 0, 0, 0;
      0, 0, Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, -alpha*Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, -L;];

GVec = [-G1, G1, 0, 0, 0, -1, 0, 0;
        G1, -G1-G2, 0, 0, 0, 0, 0, -1;
        0, 0, -G3, 0, 0, 0, 0, 1;
        0, 0, 0, -G4, G4, 0, 1, 0;
        0, 0, 0, G4, -G4-G0, 0, 0, 0;
        1, 0, 0, 0, 0, 0, 0, 0;
        0, 0, -alpha*G3, 1, 0, 0, 0, 0;
        0, 1, -1, 0, 0, 0, 0, 0;];

FVec = [0;0;1;0;0;1;0;0]; %Include new current source for R3

t = 0:0.001:1;
F1 = [0;0;0;0;0;0;0;0];
In = 0.001*randn(1,size(t,2));
vout = zeros(size(GVec,1),size(t,2));
vin= zeros(1,size(t,2));
vin(1,1) = In(1);

std = 0.03;
delay = 0.06^2;
vin(1,1) = 1/(sqrt(2*pi*delay))*(exp(-t(1).^2/(2*delay)));


for n=1:size(t,2)-1
    vout(:,n) = F1;
    vin(1,n+1) = 1/(sqrt(2*pi*delay))*(exp(-t(n+1).^2/(2*delay)));
    F2 = FVec * In(n+1); %Add noise to node 3
    F2(6) = vin(n+1); %Add input
    F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
end
vout(:,n+1) = F1;

figure("Name","Current Noise Histogram")
histogram(In);
xlabel('Noise Current (A)');
ylabel('Number in Bin (#)');
title('Noise Histogram');


figure("Name","Gaussian Pulse Response")
plot(t, vin,'r');
hold on
plot(t, vout(5,:),'b');
title('Gaussian Pulse Response');
xlabel('Time (s)');
```

```
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Gaussian Pulse Fourier Response Spectrum")
semilogy(abs(fftshift(fft(vout(5,:)))))
title('Gaussian Pulse Fourier Response');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([250 750])
```



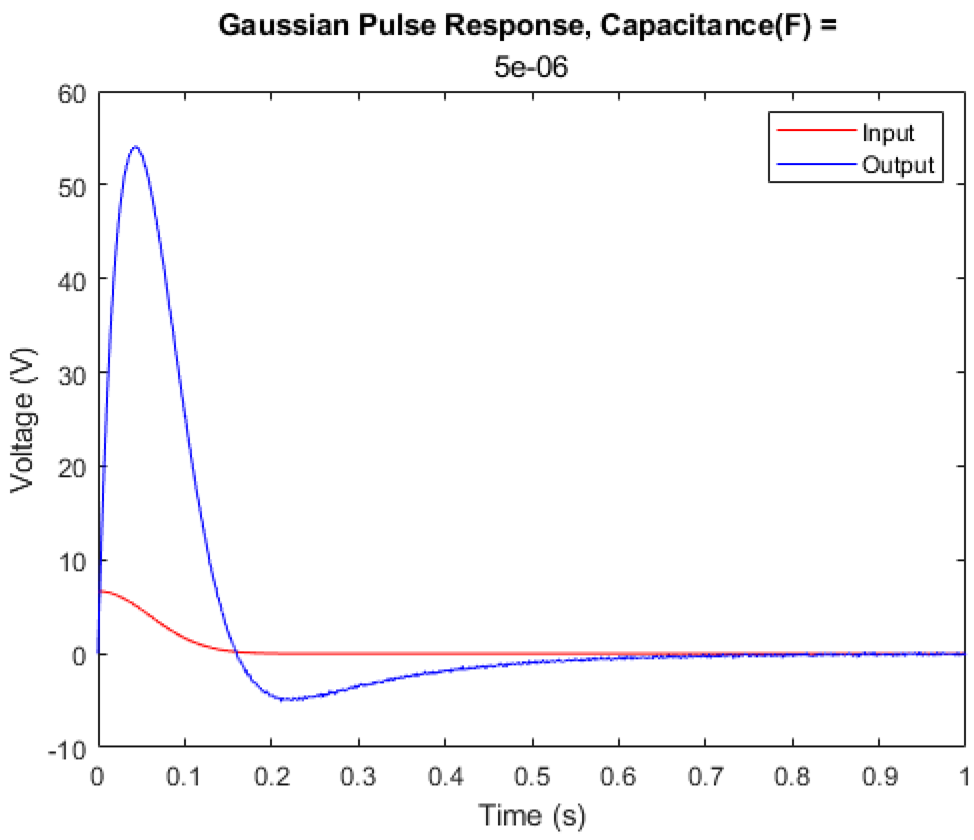

Gaussian Pulse Response
Input
Output
Voltage (V)
Time (s)
60
50
40
30
20
10
0
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

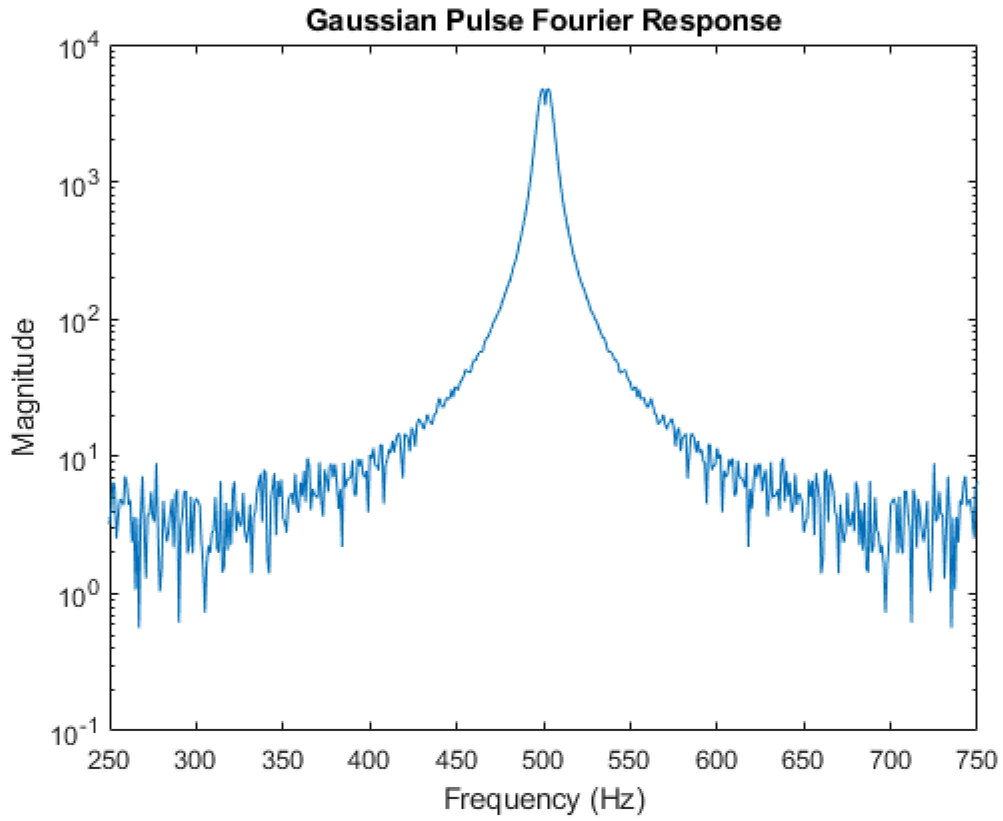

Gaussian Pulse Fourier Response
Magnitude
Frequency (Hz)
10^4
10^3
10^2
10^1
10^0
10^-1
250
300
350
400
450
500
550
600
650
700
750

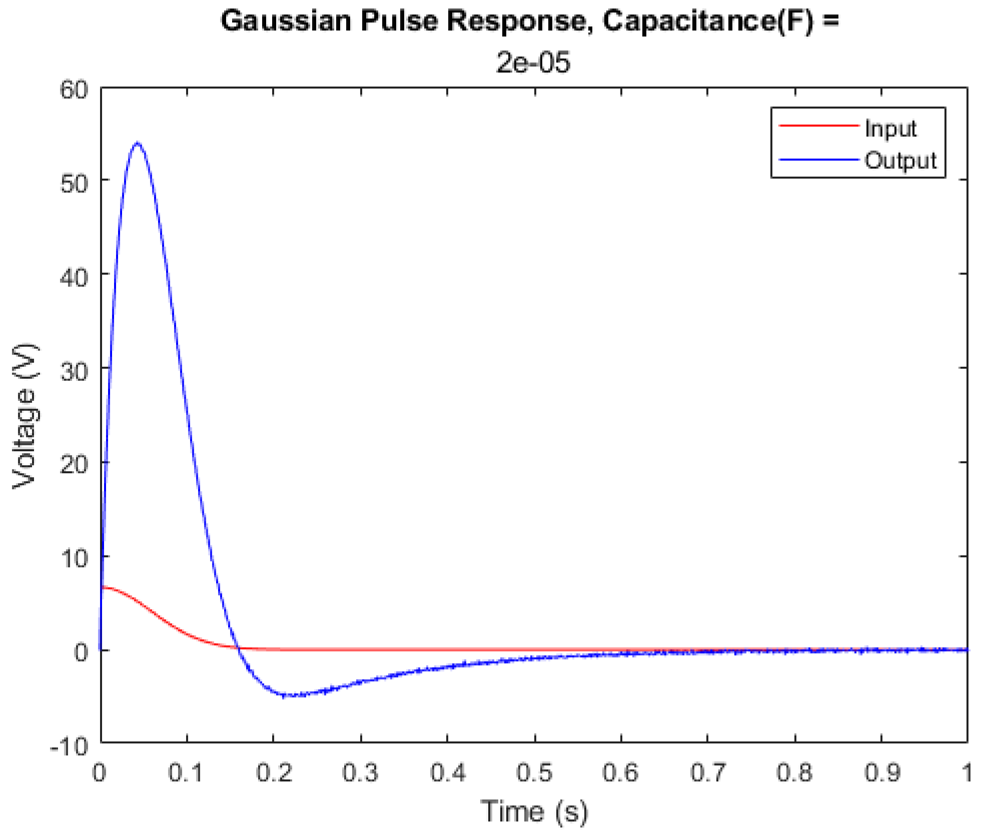
Q5 Changing Cn

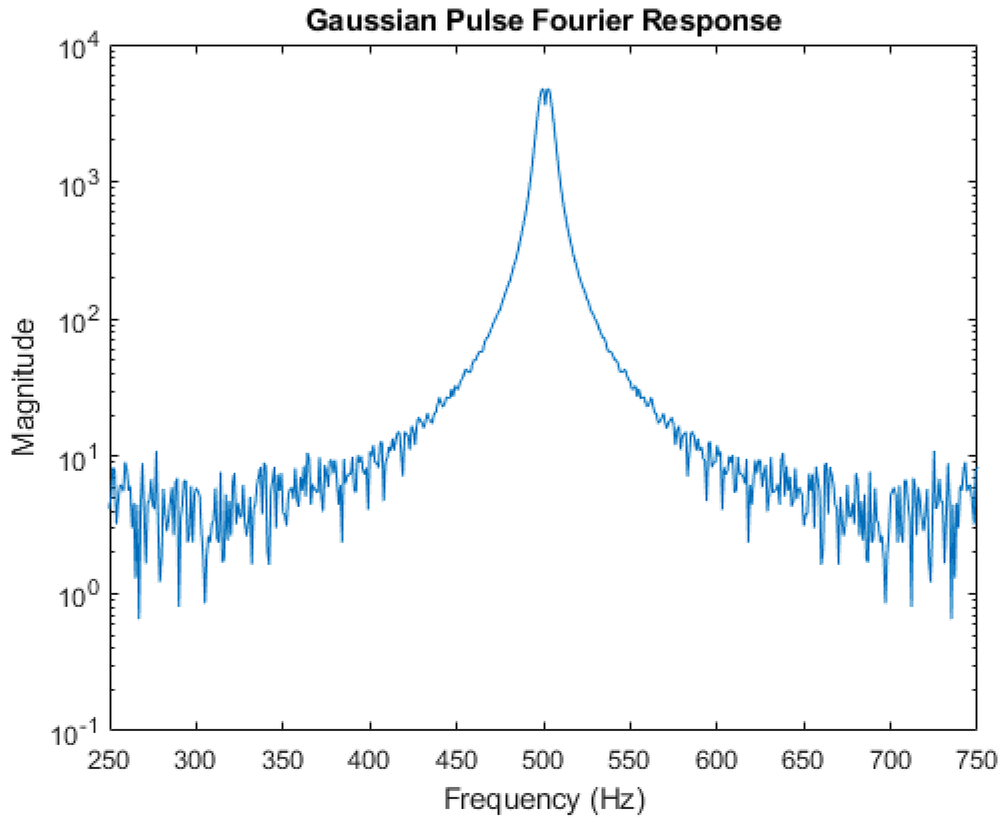
```
t = 0:0.001:1;
F1 = [0;0;0;0;0;0;0;0];
In = 0.001*randn(1,size(t,2));
vout = zeros(size(GVec,1),size(t,2));
vin= zeros(1,size(t,2));
vin(1,1) = In(1);

std = 0.03;
delay = 0.06^2;
vin(1,1) = 1/(sqrt(2*pi*delay))*(exp(-t(1).^2/(2*delay)));

a = linspace(0.000005, 0.00005,4);

for x = 1:size(a,2)
    Cn = a(x);

    CVec = [-C, C, 0, 0, 0, 0, 0, 0;
      C, -C, 0, 0, 0, 0, 0, 0;
      0, 0, Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, -alpha*Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, -L;];

    for n=1:size(t,2)-1
        vout(:,n) = F1;
        vin(1,n+1) = 1/(sqrt(2*pi*delay))*(exp(-t(n+1).^2/(2*delay)));
        F2 = FVec * In(n+1); %Add noise to node 3
        F2(6) = vin(n+1); %Add input
        F1 = ((CVec./0.001) + GVec)\((CVec./0.001)*F1 + F2);
    end
    vout(:,n+1) = F1;

    figure("Name","Gaussian Pulse Response ")
    plot(t, vin,'r');
    hold on
    plot(t, vout(5,:),'b');
    title('Gaussian Pulse Response, Capacitance(F) = ',Cn);
    xlabel('Time (s)');
    ylabel('Voltage (V)');
    legend({'Input','Output'});

    figure("Name","Gaussian Pulse Fourier Response Spectrum")
    semilogy(abs(fftshift(fft(vout(5,:)))))
    title('Gaussian Pulse Fourier Response');
    xlabel('Frequency (Hz)');
    ylabel('Magnitude');
    xlim([250 750]);
end
```

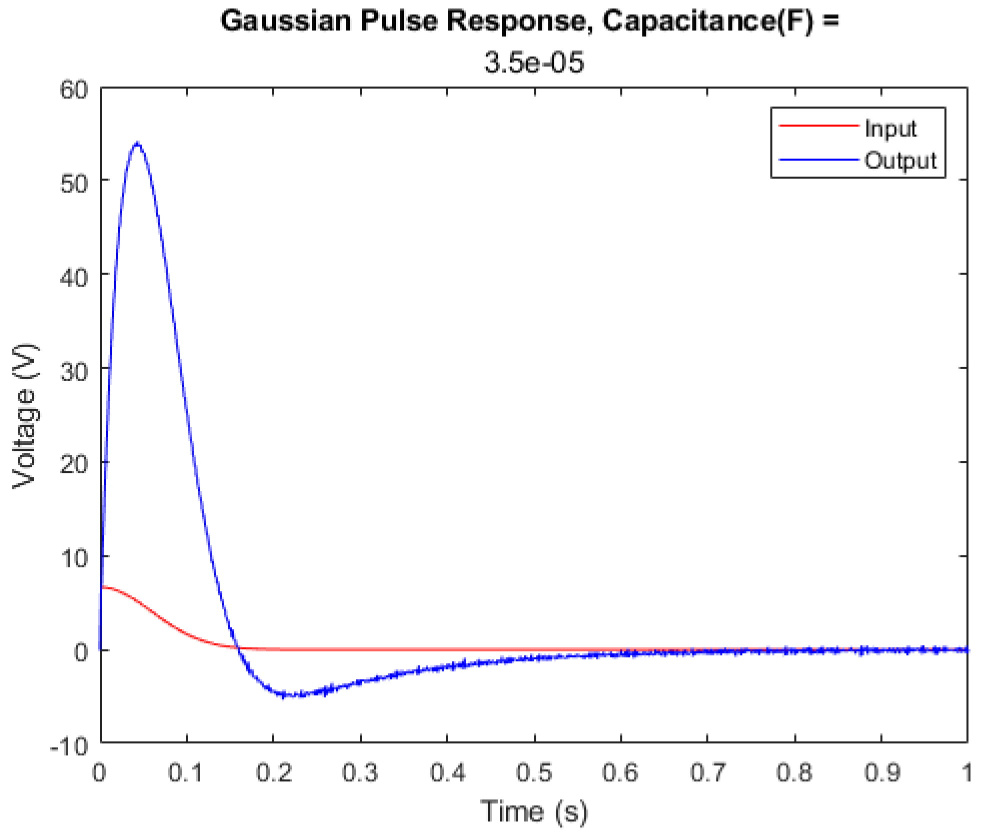
```
% Increasing the value of the capacitor chnages the effect of the noise on
% the circuit, the value cannot be increase to much or elese the circuit
% fails.
```

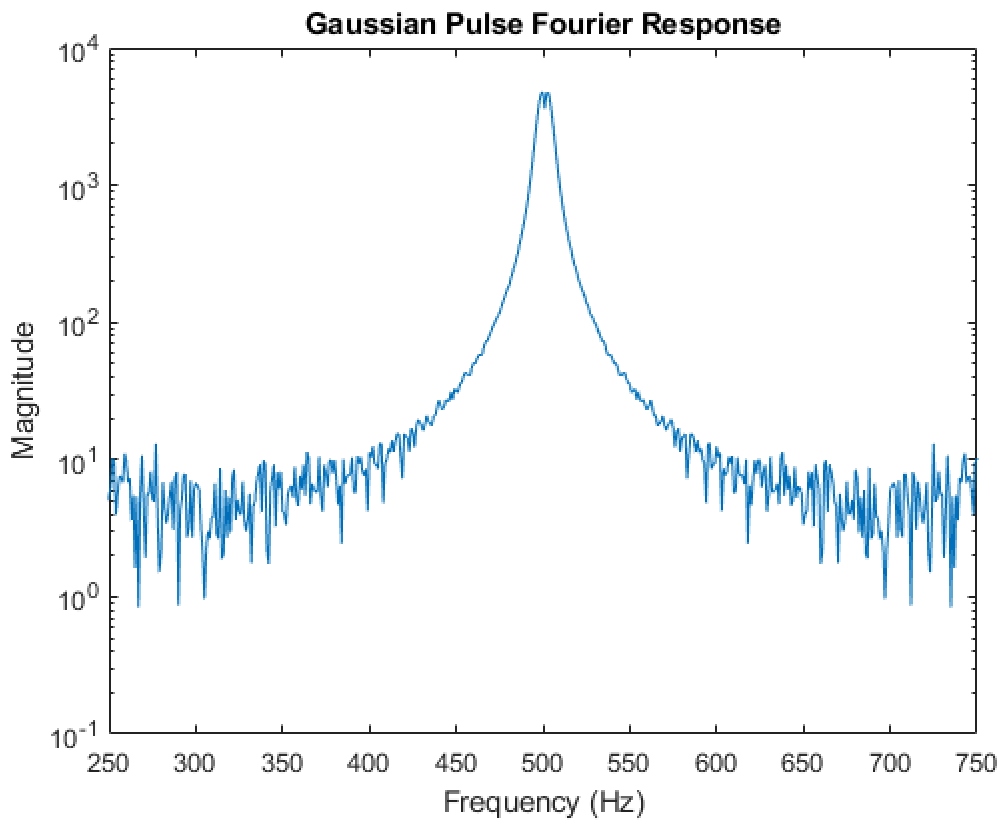


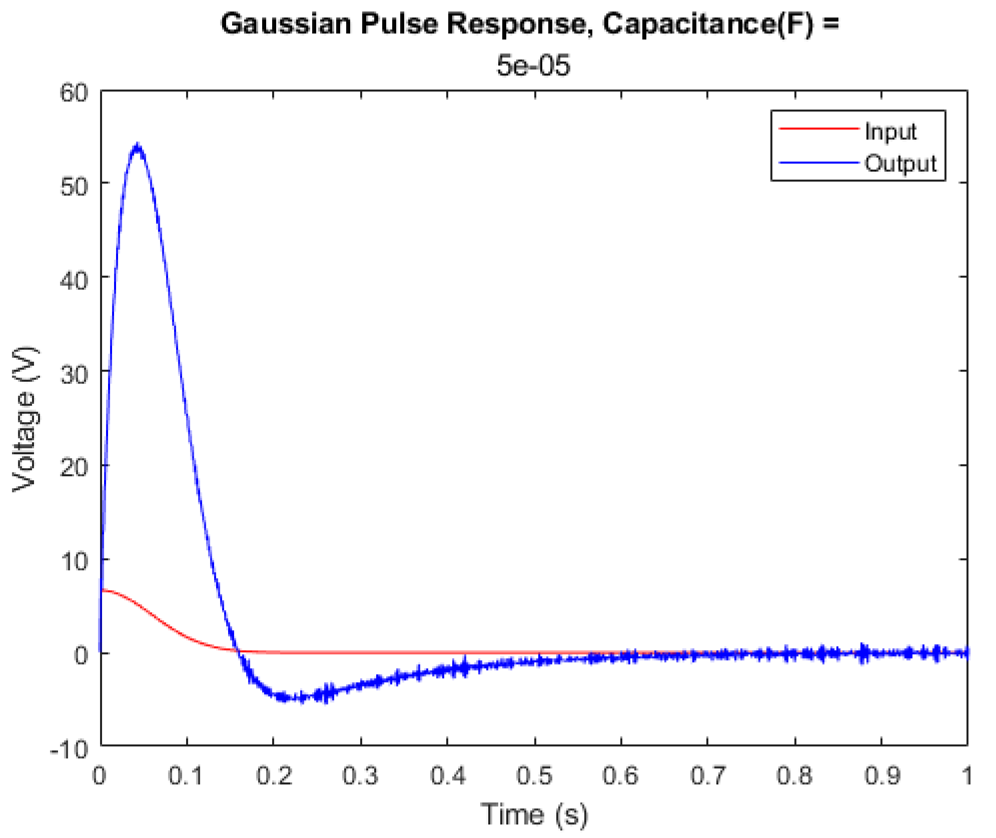

Gaussian Pulse Fourier Response
Magnitude
Frequency (Hz)
10^4
10^3
10^2
10^1
10^0
10^-1
250
300
350
400
450
500
550
600
650
700
750

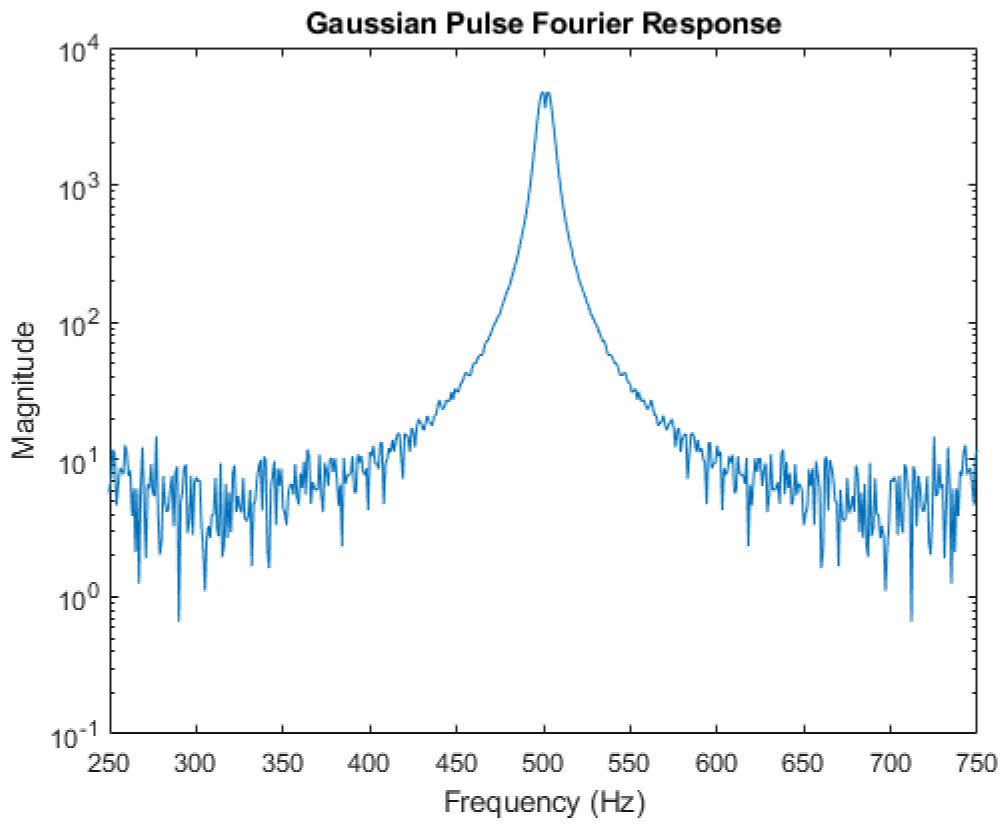

Gaussian Pulse Response, Capacitance(F) =
2e-05
Voltage (V)
Time (s)
Input
Output
60
50
40
30
20
10
0
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Gaussian Pulse Fourier Response
Magnitude
Frequency (Hz)
10^{4}
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
250
300
350
400
450
500
550
600
650
700
750


Gaussian Pulse Response, Capacitance(F) =
3.5e-05
Input
Output
Voltage (V)
Time (s)
60
50
40
30
20
10
0
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Gaussian Pulse Fourier Response
Magnitude
Frequency (Hz)
10^{4}
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
250
300
350
400
450
500
550
600
650
700
750

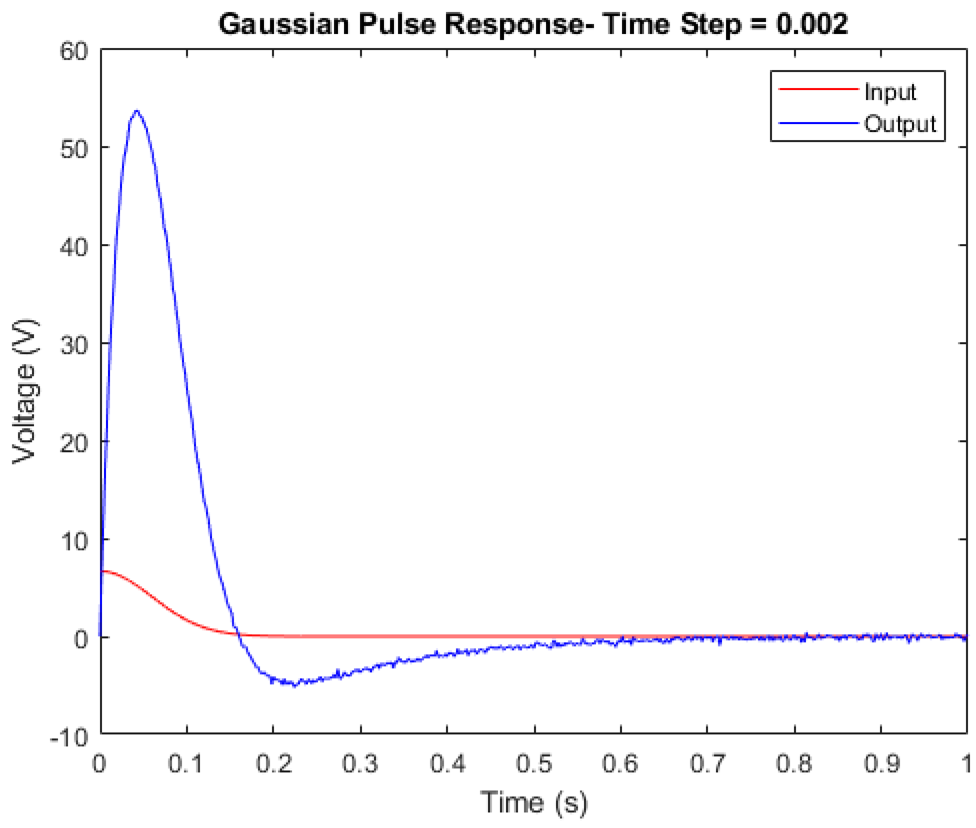

Gaussian Pulse Response, Capacitance(F) =
5e-05
Input
Output
Voltage (V)
Time (s)
60
50
40
30
20
10
0
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

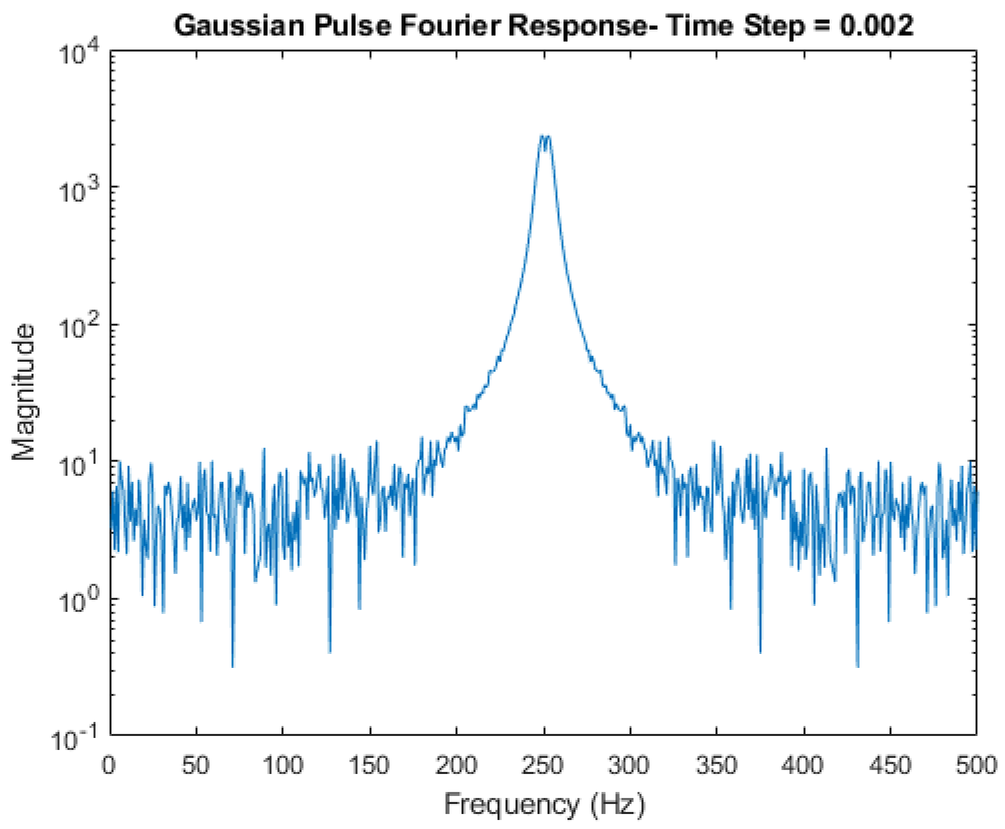


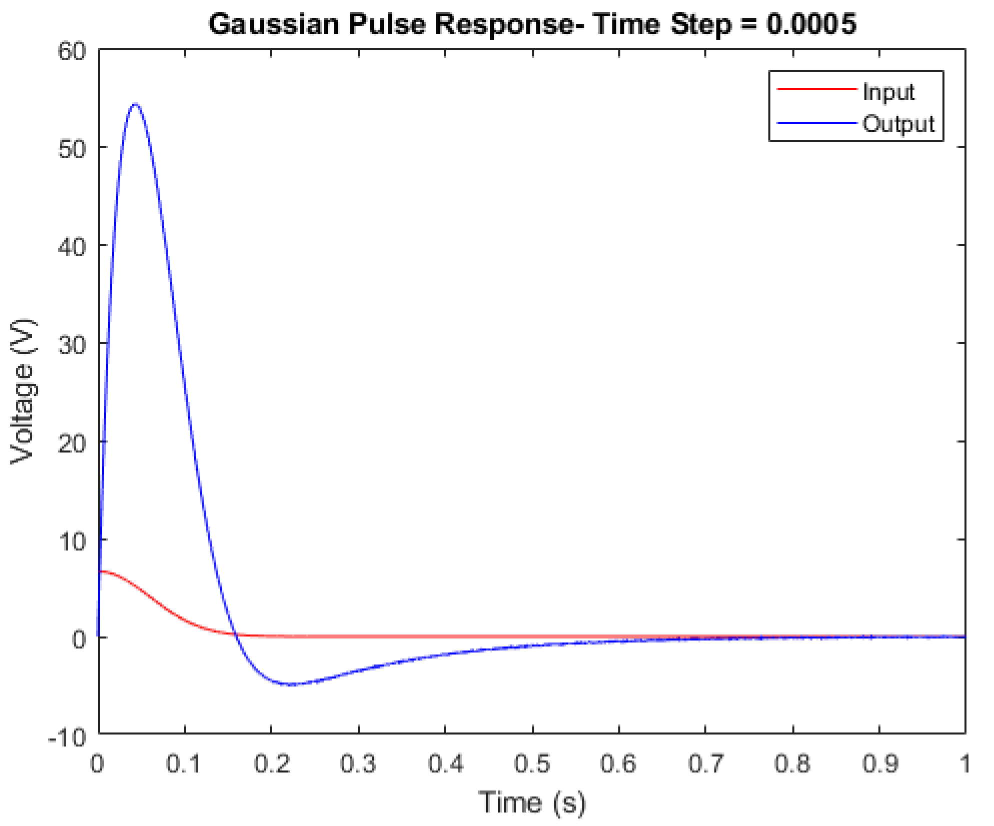
Q5 Changing Time Step

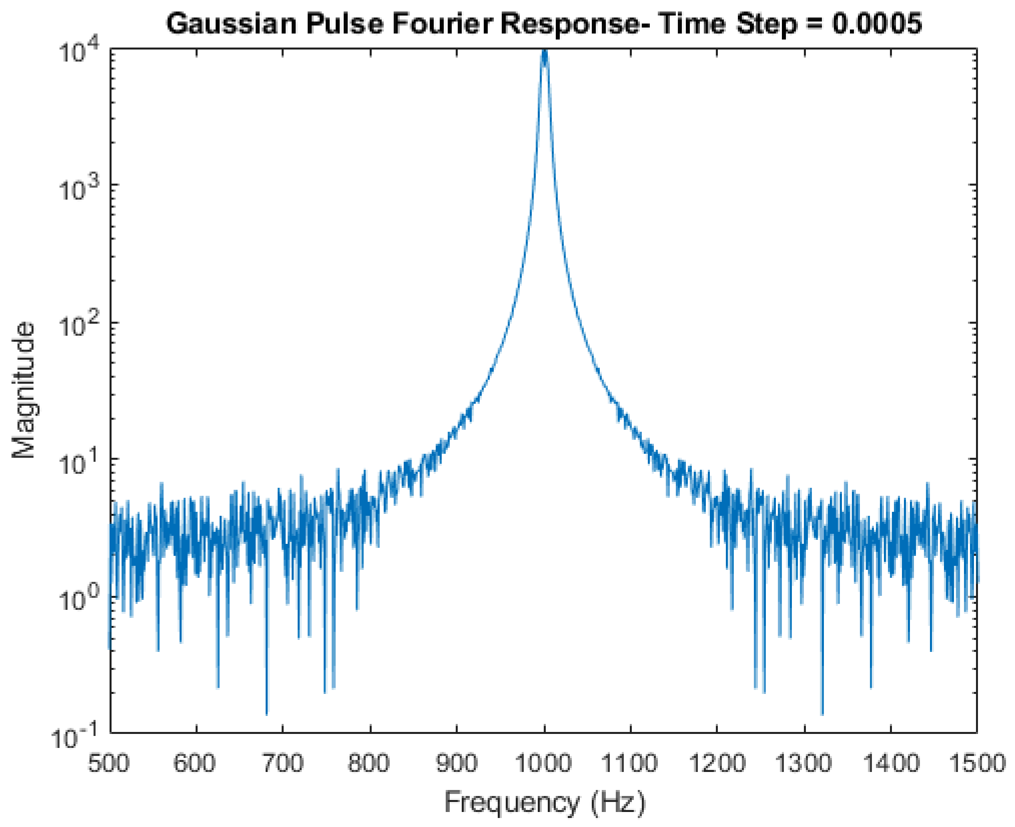
```
C=0.25;
Cn = 0.00001;
G1 = 1;
G2 = 1/2;
G3 = 1/8.5;
G4 = 1/0.1;
G0 = 1/1000;
L = 0.2;
alpha = 100;

CVec = [-C, C, 0, 0, 0, 0, 0, 0;
      C, -C, 0, 0, 0, 0, 0, 0;
      0, 0, Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, 0;
      0, 0, -alpha*Cn, 0, 0, 0, 0, 0;
      0, 0, 0, 0, 0, 0, 0, -L;];

GVec = [-G1, G1, 0, 0, 0, -1, 0, 0;
        G1, -G1-G2, 0, 0, 0, 0, 0, -1;
        0, 0, -G3, 0, 0, 0, 0, 1;
        0, 0, 0, -G4, G4, 0, 1, 0;
        0, 0, 0, G4, -G4-G0, 0, 0, 0;
        1, 0, 0, 0, 0, 0, 0, 0;
        0, 0, -alpha*G3, 1, 0, 0, 0, 0;
        0, 1, -1, 0, 0, 0, 0, 0;];

FVec = [0;0;1;0;0;1;0;0]; %Include new current source for R3

t = 0:0.002:1;
F1 = [0;0;0;0;0;0;0;0];
In = 0.002*randn(1,size(t,2));
vout = zeros(size(GVec,1),size(t,2));
vin= zeros(1,size(t,2));
vin(1,1) = In(1);

std = 0.03;
delay = 0.06^2;
vin(1,1) = 1/(sqrt(2*pi*delay))*(exp(-t(1).^2/(2*delay)));


for n=1:size(t,2)-1
    vout(:,n) = F1;
    vin(1,n+1) = 1/(sqrt(2*pi*delay))*(exp(-t(n+1).^2/(2*delay)));
    F2 = FVec * In(n+1); %Add noise to node 3
    F2(6) = vin(n+1); %Add input
    F1 = ((CVec./0.002) + GVec)\((CVec./0.002)*F1 + F2);
end
vout(:,n+1) = F1;

figure("Name","Gaussian Pulse Response - Time Step = 0.002")
plot(t, vin,'r');
hold on
plot(t, vout(5,:),'b');
title('Gaussian Pulse Response- Time Step = 0.002');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Gaussian Pulse Fourier Response Spectrum")
semilogy(abs(fftshift(fft(vout(5,:)))))
title('Gaussian Pulse Fourier Response- Time Step = 0.002');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([0 500]);

t = 0:0.0005:1;
F1 = [0;0;0;0;0;0;0;0];
In = 0.0005*randn(1,size(t,2));
vout = zeros(size(GVec,1),size(t,2));
vin= zeros(1,size(t,2));
vin(1,1) = In(1);

std = 0.03;
delay = 0.06^2;
vin(1,1) = 1/(sqrt(2*pi*delay))*(exp(-t(1).^2/(2*delay)));
```

```
for n=1:size(t,2)-1
    vout(:,n) = F1;
    vin(1,n+1) = 1/(sqrt(2*pi*delay))*(exp(-t(n+1).^2/(2*delay)));
    F2 = FVec * In(n+1); %Add noise to node 3
    F2(6) = vin(n+1); %Add input
    F1 = ((CVec./0.0005) + GVec)\((CVec./0.0005)*F1 + F2);
end
vout(:,n+1) = F1;

figure("Name","Gaussian Pulse Response - Time Step = 0.0005")
plot(t, vin,'r');
hold on
plot(t, vout(5,:),'b');
title('Gaussian Pulse Response- Time Step = 0.0005');
xlabel('Time (s)');
ylabel('Voltage (V)');
legend({'Input','Output'});

figure("Name","Gaussian Pulse Fourier Response Spectrum")
semilogy(abs(fftshift(fft(vout(5,:)))))
title('Gaussian Pulse Fourier Response- Time Step = 0.0005');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
xlim([500 1500]);
```




Gaussian Pulse Fourier Response- Time Step = 0.002
Magnitude
Frequency (Hz)
10^{4}
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
0
50
100
150
200
250
300
350
400
450
500


Gaussian Pulse Response- Time Step = 0.0005
Input
Output
Voltage (V)
Time (s)
60
50
40
30
20
10
0
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



Published with MATLAB® R2021b