Table of Contents

```
% Matthew Lazarus 100962142
```

Assignment 4

Part 2

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

Unlike the previous question, use stamps to generate the desired MNA matrices. These stamps were developed in ELEC 4506.

```
global G C b; %define global variables
G = zeros(5,5); % Define G, 5 node circuit (do not include additional
 variables)
C = zeros(5,5); % Define C, 5 node circuit (do not include additional
 variables)
b = zeros(5,1); % Define b, 5 node circuit (do not include additional
 variables)

vol(1,0,10);
vcvs(4,0,3,0,10);

res(1,2,1);
res(2,0,2);
res(3,0,10);
res(4,5,0.1);
res(5,0,1000);

cap(1,2,0.25);
ind(2,3,0.2);
```

The C and G matrices are:

```
C
```

```
G

%Solve DC Case to get values at t=0.
Xprev=zeros(8,1);


C =

  Columns 1 through 7

    0.2500   -0.2500         0         0         0         0         0
   -0.2500    0.2500         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

  Column 8

         0
         0
         0
         0
         0
         0
         0
   -0.2000


G =

  Columns 1 through 7

    1.0000   -1.0000         0         0         0    1.0000         0
   -1.0000    1.5000         0         0         0         0         0
         0         0    0.1000         0         0         0         0
         0         0         0   10.0000  -10.0000         0    1.0000
         0         0         0  -10.0000   10.0010         0         0
    1.0000         0         0         0         0         0         0
         0         0  -10.0000    1.0000         0         0         0
         0    1.0000   -1.0000         0         0         0         0

  Column 8

         0
    1.0000
   -1.0000
         0
         0
         0
         0
         0
```

(d) A

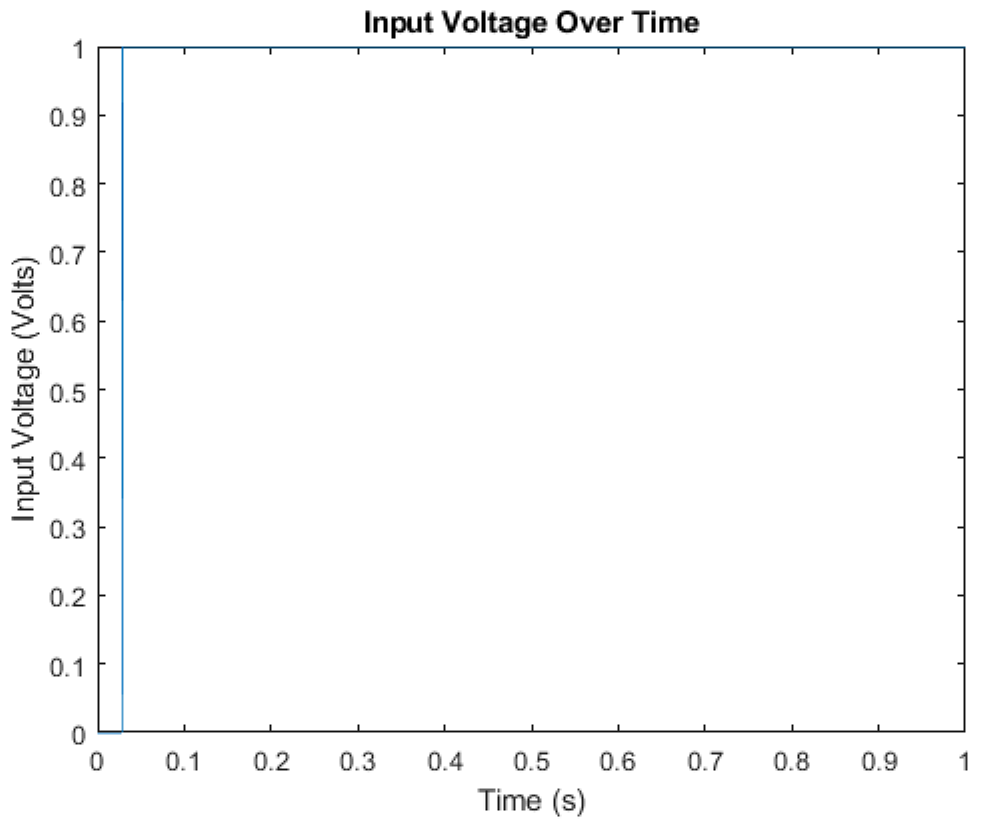
Now use Trapezoidal Rule to simulate over time.

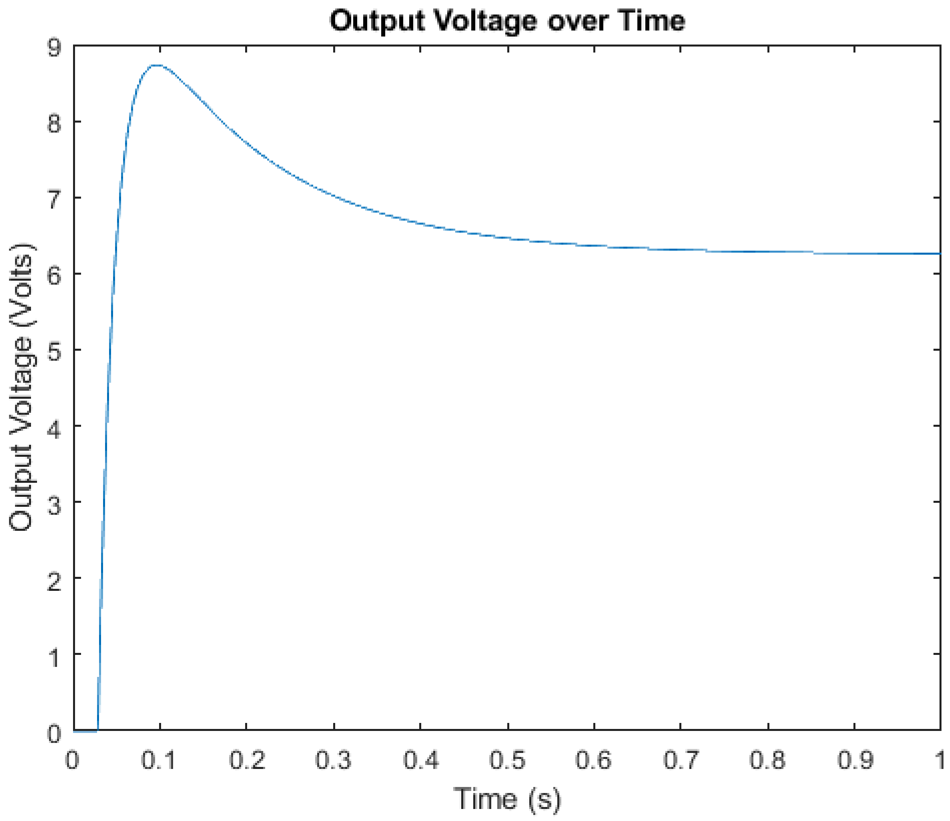
```
h=1/1000;
vInput = zeros(1000,1);
vOut = zeros(1000,1);
% Form matrix of input voltage over time.
for count = 1:1000
    t=count*h;
    if(t>=0.03)
        vInput(count)=1;
    end
end
b(6) = vInput(1);
for count = 1:1000
    bNext = b;
    bNext(6) = vInput(count);
    Xnext = (G+(2*C/h))\((2*C/h - G)*Xprev+b+bNext);
    vOut(count) = Xnext(5);

    b = bNext;
    Xprev = Xnext;
end
```

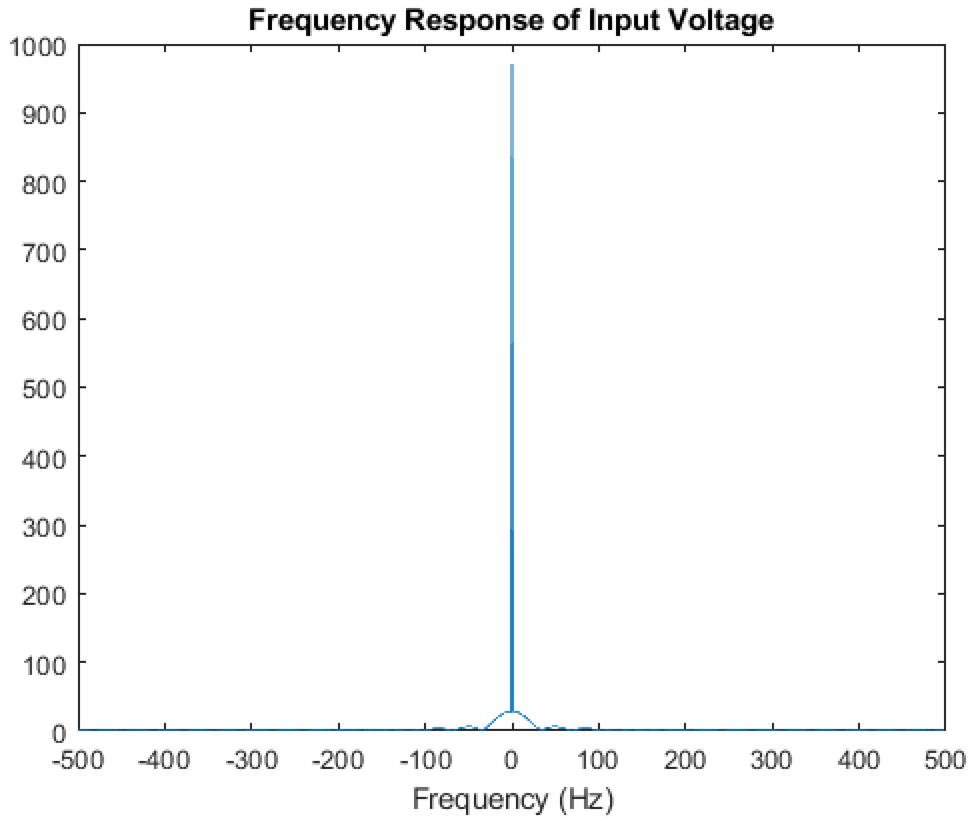
The Time domain and frequency domain plots of the voltage step can be seen below.

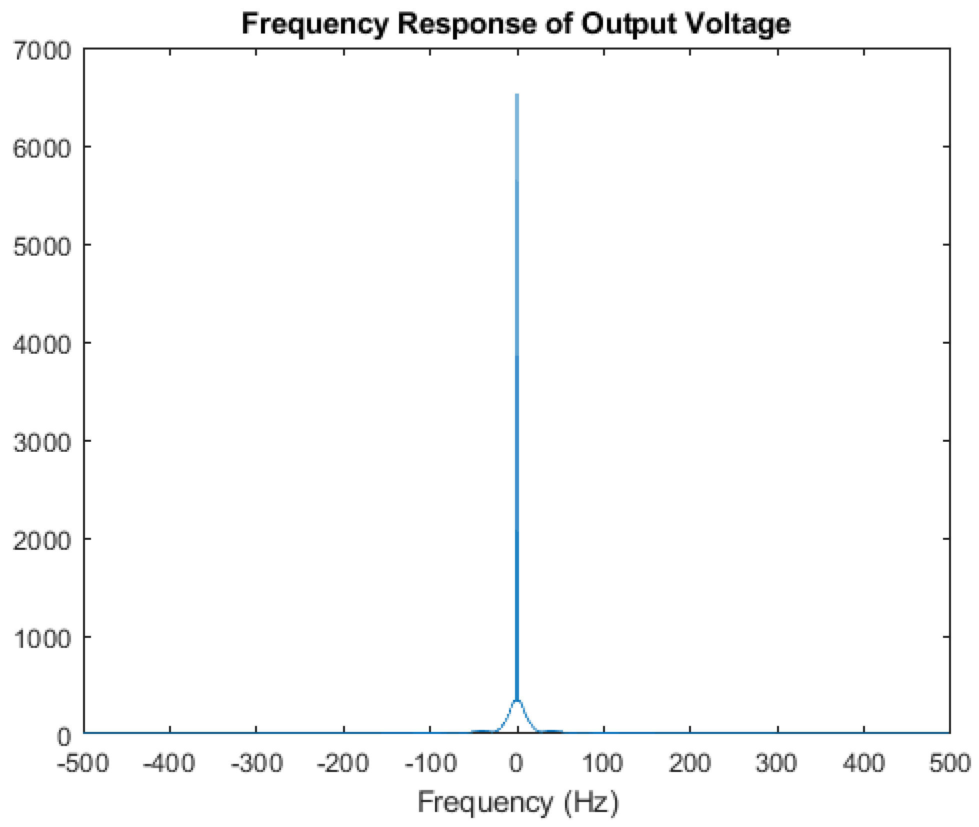
```
fftVin = abs(fftshift(fft(vInput)));
fftVout = abs(fftshift(fft(vOut)));
n=length(fftVin);
fs=1/h;
fshift=(-n/2:n/2-1)*(fs/n);

figure;
plot(linspace(0,1,1000),vInput)
xlabel('Time (s)')
ylabel('Input Voltage (Volts)')
title('Input Voltage Over Time')

figure;
plot(linspace(0,1,1000),vOut)
xlabel('Time (s)')
ylabel('Output Voltage (Volts)')
title('Output Voltage over Time')

figure;
plot(fshift, fftVin);
xlabel('Frequency (Hz)')
title('Frequency Response of Input Voltage')

figure;
```

```
plot(fshift, fftVout);
xlabel('Frequency (Hz)')
title('Frequency Response of Output Voltage')
```




Output Voltage over Time
Output Voltage (Volts)
Time (s)
0
1
2
3
4
5
6
7
8
9
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Frequency Response of Input Voltage
Frequency (Hz)
0
100
200
300
400
500
600
700
800
900
1000
-500
-400
-300
-200
-100
0
100
200
300
400
500



d) B

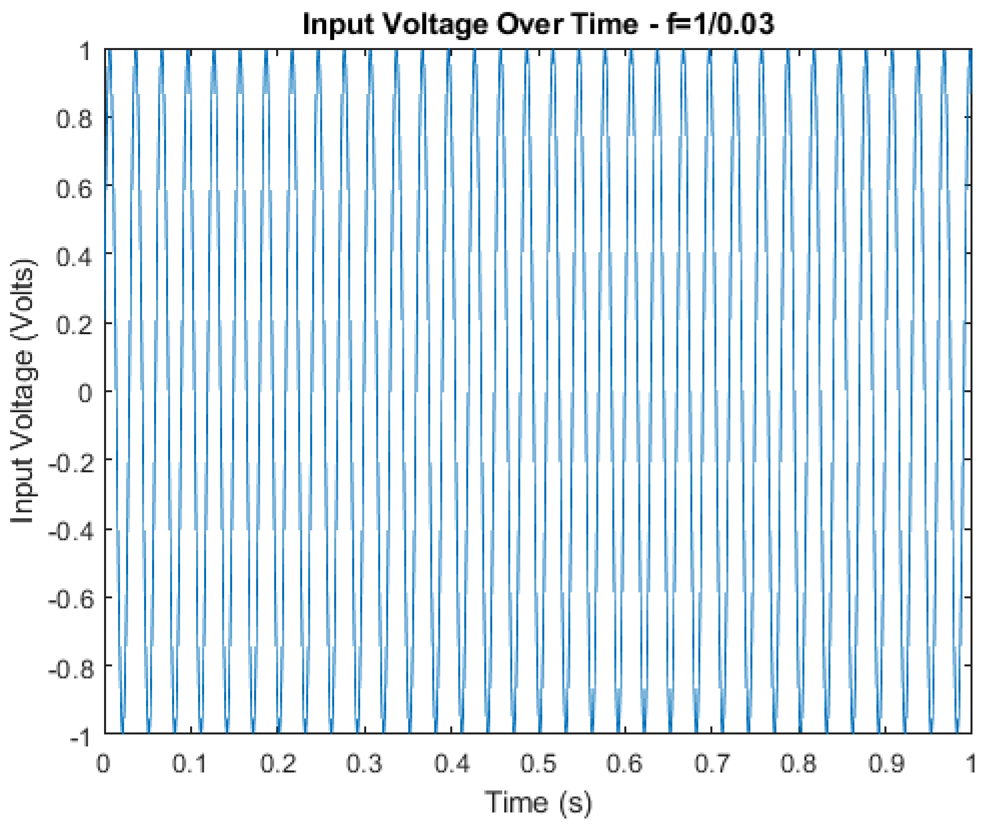
Now repeat the process for a sinusoidal signal.

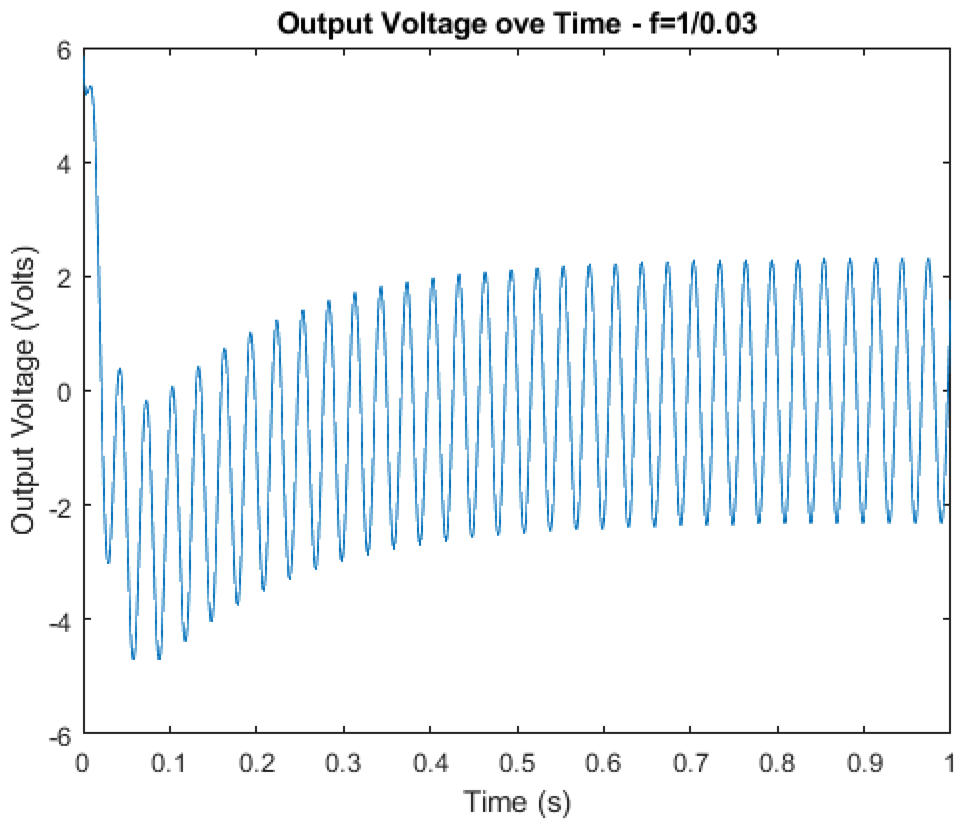
```
h=1/1000;
vInput = zeros(1000,1);
% vInput2 = zeros(1000,1);
% vInput3 = zeros(1000,1);
vOut = zeros(1000,1);
% vOut2 = zeros(1000,1);
% vOut3 = zeros(1000,1);
f1=1/0.03;
% f2=1/0.3;
% f3=1/0.003;
for count = 1:1000
    t=count*h;
    vInput(count) = sin(2*pi*f1*t);
%     vInput2(count) = sin(2*pi*f2*t);
%     vInput3(count) = sin(2*pi*f3*t);
end
b(6) = vInput(1);
%f=1/0.03
for count = 1:1000
    bNext = b;
    bNext(6) = vInput(count);
    Xnext = (G+(2*C/h))\((2*C/h - G)*Xprev+b+bNext);
    vOut(count) = Xnext(5);

    b = bNext;
    Xprev = Xnext;
end

fftVin = abs(fftshift(fft(vInput)));
fftVout = abs(fftshift(fft(vOut)));
n=length(fftVin);
fs=1/h;
fshift=(-n/2:n/2-1)*(fs/n);


% %f=1/0.3
% for count = 1:1000
%     bNext = b;
%     bNext(6) = vInput2(count);
%     Xnext = (G+(2*C/h))\((2*C/h - G)*Xprev+b+bNext);
%     vOut2(count) = Xnext(5);
%
%     b = bNext;
%     Xprev = Xnext;
% end
%
% %f=1/0.003
% for count = 1:1000
%     bNext = b;
%     bNext(6) = vInput3(count);
%     Xnext = (G+(2*C/h))\((2*C/h - G)*Xprev+b+bNext);
%     vOut3(count) = Xnext(5);
%
%     b = bNext;
%     Xprev = Xnext;
% end
```

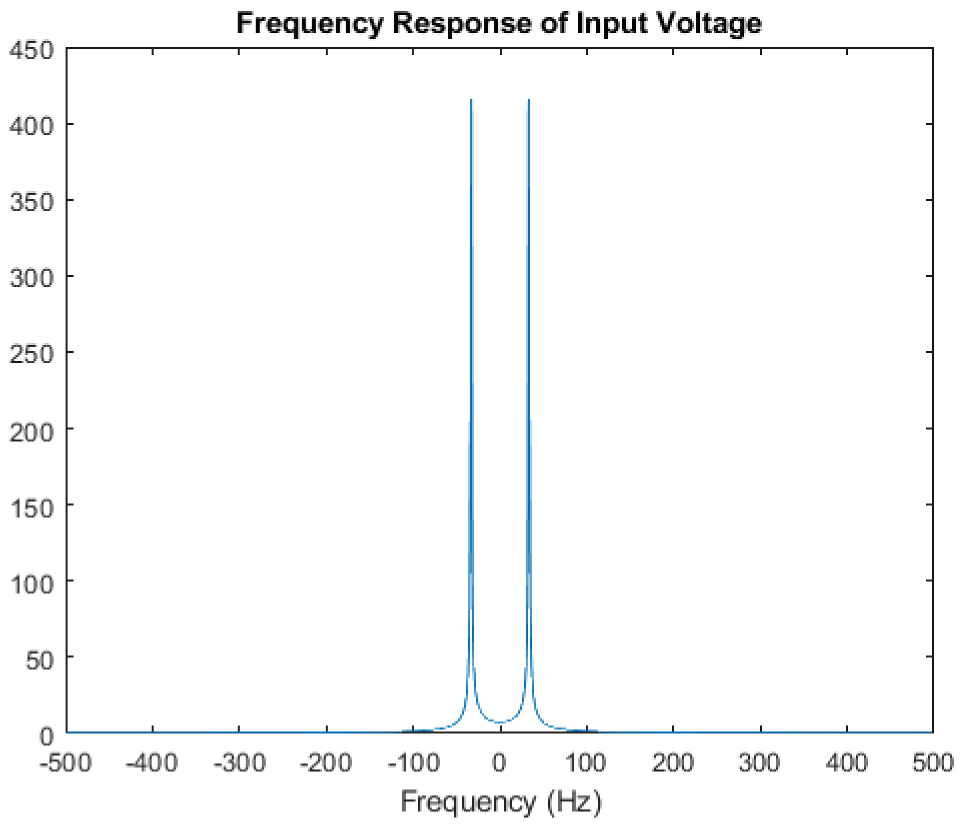
Seen in the figures below are the time domain and frequency domain signals corresponding to a sinusoidal input voltage.

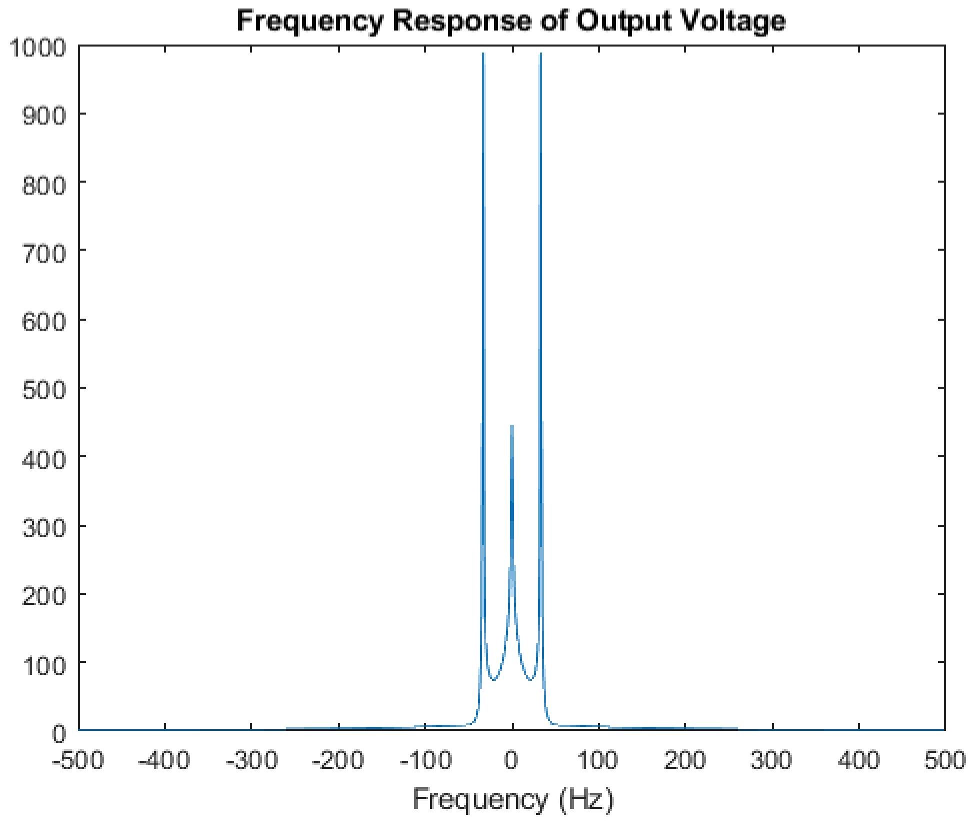
```
figure;
plot(linspace(0,1,1000),vInput)
xlabel('Time (s)')
ylabel('Input Voltage (Volts)')
title('Input Voltage Over Time - f=1/0.03')

figure;
plot(linspace(0,1,1000),vOut)
xlabel('Time (s)')
ylabel('Output Voltage (Volts)')
title('Output Voltage ove Time - f=1/0.03')

figure;
plot(fshift, fftVin);
xlabel('Frequency (Hz)')
title('Frequency Response of Input Voltage')
```

```
figure;
plot(fshift, fftVout);
xlabel('Frequency (Hz)')
title('Frequency Response of Output Voltage')
```




Output Voltage ove Time - f=1/0.03
Output Voltage (Volts)
Time (s)
6
4
2
0
-2
-4
-6
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Frequency Response of Input Voltage
Frequency (Hz)
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500



After simulating with varying frequencies, it was seen that higher frequencies lead to poorer results from the finite difference method. This is becuase the voltage was changing greatly over each timestep. Therefore, if a higher frequency were to be used, a higher sampling frequency should be used as well.

```
% figure;
% plot(linspace(0,1,1000),vInput2)
% xlabel('Time (s)')
% ylabel('Input Voltage (Volts)')
% title('Input Voltage Over Time - f=1/0.3')
%
% figure;
% plot(linspace(0,1,1000),vOut2)
% xlabel('Time (s)')
% ylabel('Output Voltage (Volts)')
% title('Output Voltage ove Time - f=1/0.3')
%
% figure;
% plot(linspace(0,1,1000),vInput3)
% xlabel('Time (s)')
% ylabel('Input Voltage (Volts)')
% title('Input Voltage Over Time - f=1/0.003')
%
% figure;
% plot(linspace(0,1,1000),vOut3)
% xlabel('Time (s)')
% ylabel('Output Voltage (Volts)')
% title('Output Voltage ove Time - f=1/0.003')
```

d) C

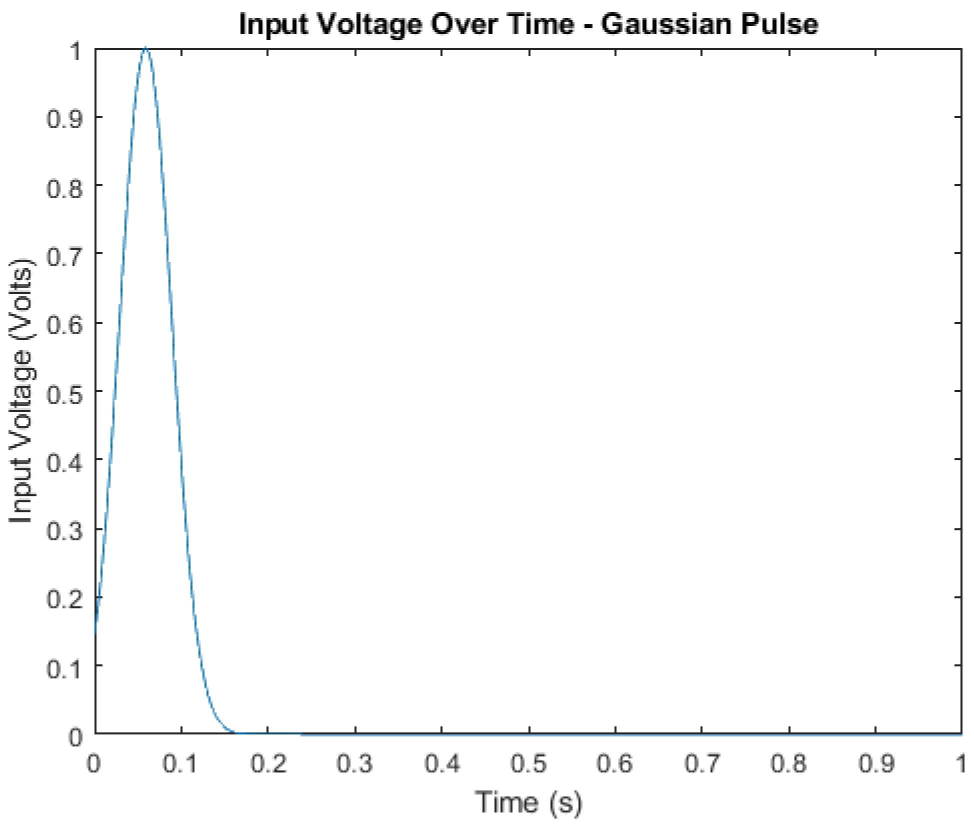
Lastly, repeat the process for a gaussian pulse.

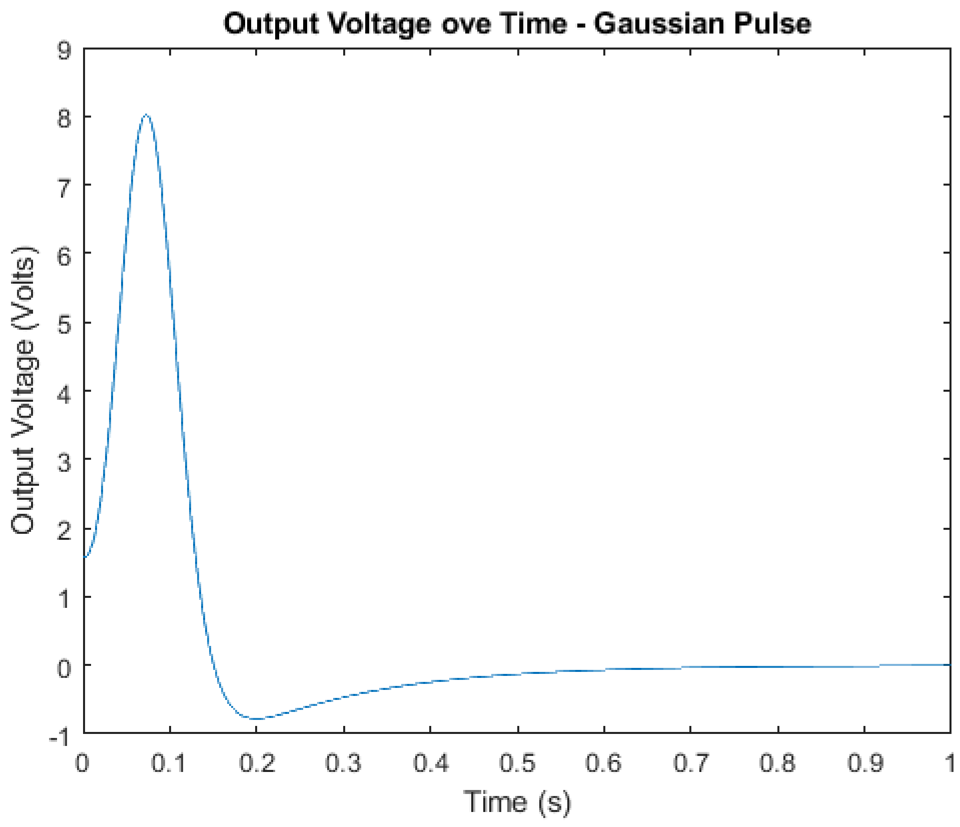
```
vInput = zeros(1000,1);
vOut = zeros(1000,1);
for count = 1:1000
    t=count*h;
    % Gaussian pulse, shifted by 0.06s and compressed to have std
 deviation
    % of 0.03.
    vInput(count) = exp(-0.5*((t-0.06)/0.03)^2);
end

b(6) = vInput(1);
for count = 1:1000
    bNext = b;
    bNext(6) = vInput(count);
    Xnext = (G+(2*C/h))\((2*C/h - G)*Xprev+b+bNext);
    vOut(count) = Xnext(5);

    b = bNext;
    Xprev = Xnext;
end


fftVin = abs(fftshift(fft(vInput)));
fftVout = abs(fftshift(fft(vOut)));
n=length(fftVin);
fs=1/h;
fshift=(-n/2:n/2-1)*(fs/n);
```

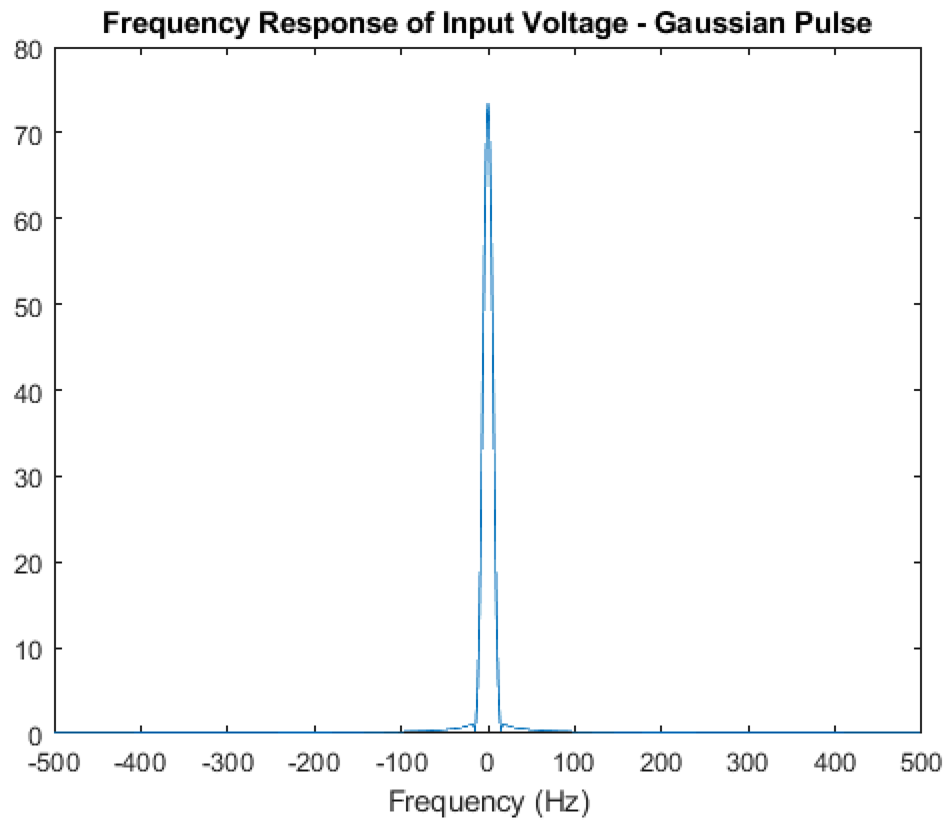
The figures below contain the time domain and frequency domain response o the circuit to a Gaussian input voltage.

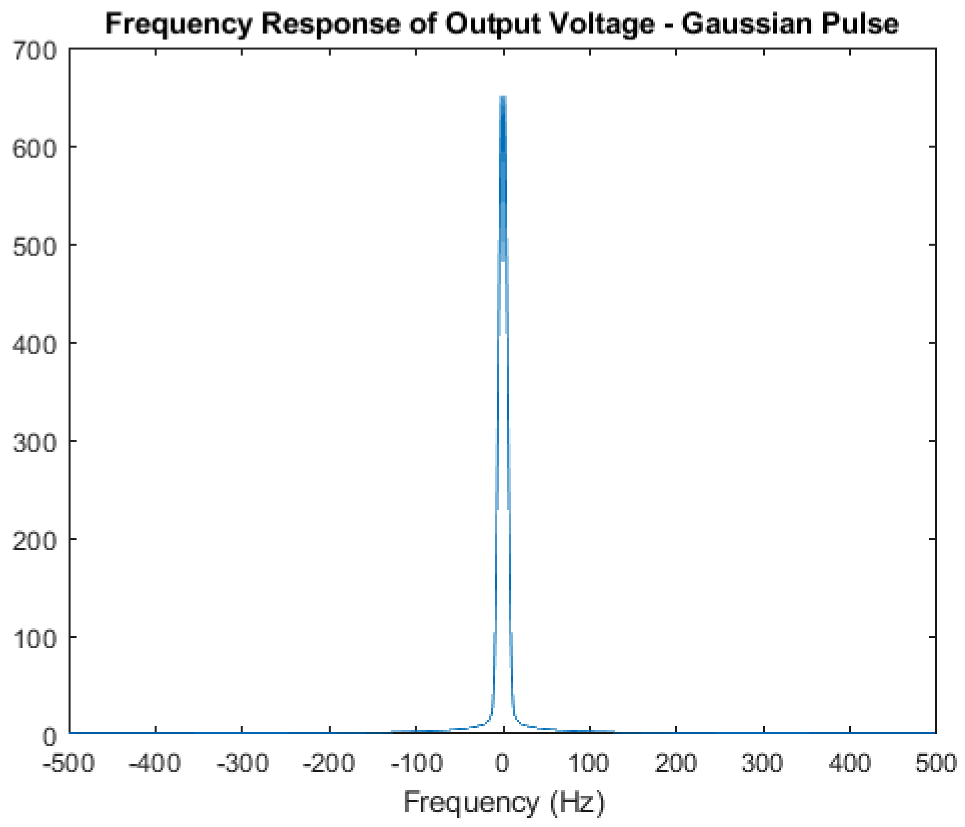
```
figure;
plot(linspace(0,1,1000),vInput)
xlabel('Time (s)')
ylabel('Input Voltage (Volts)')
title('Input Voltage Over Time - Gaussian Pulse')

figure;
plot(linspace(0,1,1000),vOut)
xlabel('Time (s)')
ylabel('Output Voltage (Volts)')
title('Output Voltage ove Time - Gaussian Pulse')

figure;
plot(fshift, fftVin);
xlabel('Frequency (Hz)')
title('Frequency Response of Input Voltage - Gaussian Pulse')
```

```
figure;
plot(fshift, fftVout);
xlabel('Frequency (Hz)')
title('Frequency Response of Output Voltage - Gaussian Pulse')
```




Output Voltage ove Time - Gaussian Pulse
Output Voltage (Volts)
Time (s)
9
8
7
6
5
4
3
2
1
0
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Frequency Response of Input Voltage - Gaussian Pulse
Frequency (Hz)
80
70
60
50
40
30
20
10
0
-500
-400
-300
-200
-100
0
100
200
300
400
500



d) v

After testing, increasing the time scale was seen to produce worse results. This is because a bigger time step resulted in poorer guesses from the finite difference method (Trapezoidal Rule), producing a more fragmented signal.

Published with MATLAB® R2018a