Elec 4700 Assignment 4

Table of Contents

Steven Cramp 101030294

Introduction

This assignment looks at the implementation of a circuit simulator using a monti carlo simulation to solve for the value of a resistive nanostructure. This lab will look at the AC and DC simulation of the circuit. It will also look at the time domain and fast fourier of the circuit and the effects that sampling time and noise has on the circuit.

Part 1

The code sagment below was used to simulate the resistive nanostructure used in this assigninment. The nanostructure was constructed with a 60 nm gap. The voltage acrost the divice was swept from 0.1 to 10 volts and the resistance of the divice was found by finding the slope of the current vs applied voltage.

```
clc;clear;close all
%profile on
Numvolt=15;
for w= 1:Numvolt

C.q_0 = 1.60217653e-19;             % electron charge
C.hb = 1.054571596e-34;             % Dirac constant
C.h = C.hb * 2 * pi;                % Planck constant
C.m_0 = 9.10938215e-31;             % electron mass
C.kb = 1.3806504e-23;               % Boltzmann constant
C.eps_0 = 8.854187817e-12;          % vacuum permittivity
C.mu_0 = 1.2566370614e-6;           % vacuum permeability
C.c = 299792458;                    % speed of light
C.g = 9.80665; %metres (32.1740 ft) per sÂ²
a3p1 =0;
a3p2 = 1;

partical_colission = 1;% part 2
boxes =1;%part 3
specular = 0 ;% part 3 one is on zero is off
```

```
T= 300;%k temperature
Eme= 0.26* C.m_0;% kg effective mass of electron
Tmn = 0.2e-12;%s mean time between colissions
vth = sqrt(2*C.kb*T/Eme);% m/s thermal volocity
MFP = vth*Tmn; %m
m= 1000*2*w;% length of sim
dt = 1e-15/(w*2);% time step
N = 1000; %number of electons
dimx =200e-9;%m
dimy =100e-9;%m
c =zeros(1,N);
xpos=zeros(m,N);
ypos=zeros(m,N);
Fx=0;
Fy=0;

blocksize = 20e-9; %linspace(0,(dimy/2)-1e-9,5);

% create boxes
if (boxes)
    box1 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;dimy,dimy,
 dimy-blocksize,dimy-blocksize,dimy];
    box2 =
 [dimx-125e-9,dimx-75e-9,dimx-75e-9,dimx-125e-9,dimx-125e-9;blocksize,blocksize,
 0,0,blocksize];
else
    box1 =[0 0 0 0 0 ; 0 0 0 0 0];
    box2 =[0 0 0 0 0 ; 0 0 0 0 0];
end
% initiates points and ensures that they donot spawn ouside the
 boundries
if ~boxes
    xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
    ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
    xpos(xpos ==dimx)=xpos(xpos ==dimx)-1e-9;
    xpos(xpos ==0)=xpos(xpos ==0)+1e-9;
    ypos(ypos ==dimy)=ypos(ypos ==dimy)-1e-9;
    ypos(ypos ==0)=ypos(ypos ==0)+1e-9;
elseif(boxes)
    for l =1:N
        xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
        ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        while (ypos(1,l)<=0|ypos(1,l)>=dimy|
xpos(1,l)>=box1(1,1)-1e-9&xpos(1,l)<=box1(1,2)+1e-9&(ypos(1,l)<=box2(2,1)+1e-9|
ypos(1,l)>=box1(2,3)-1e-9))
            xpos(1,l) =(randi((dimx*1e9)+1,1,1)-1)/1e9 ;
            ypos(1,l) =(randi((dimy*1e9)+1,1,1)-1)/1e9;
        end
    end
end
% while(sum ((xpos>=box1(1,1)&xpos<= box1(1,1)& ypos>= box1(2,3))|
(xpos>=box2(1,1)&xpos<= box2(1,1)& ypos>= box2(2,3)))>=1)
%     xpos = (randi((dimx*1e9)+1,1,N)-1)/1e9;% m electron position x
```

```
%     ypos = (randi((dimy*1e9)+1,1,N)-1)/1e9;% m electron position y
% end

vx   = zeros(1,N);%m/s velocity in x
vy   = zeros(1,N);%m/s velocity in y
colision_count = zeros(1,N);
dtraveled=zeros(1,N);
colourL = ["R", "G", "B","LB", "P", "y", "BLk", "DB", "O","M","GG"];
colour = [[1 0 0];[0 1 0];[0 0 1];[0 1 1];[1 0 1];[1 1 0];[0 0 0];
[0 0.447 0.741];[0.85 0.325 0.098];[0.929 0.694 0.125];[0.466 0.674
 0.188]];
Temp = T;

 dx2 =1e-9;
    dy2=1e-9;
    nx2 =round( dimx/dx2);
    ny2 = round(dimy/dy2);
    Ex=zeros(ny2,nx2);
    Ey=zeros(ny2,nx2);
% initiated the partical velocities
if ~partical_colission
    angle = rand(1,N);
    vx = vth* cos(angle*2*pi);
    vy = vth* sin(angle*2*pi);
else
    vx =randn(1,N)*sqrt(C.kb*T/Eme);
    vy=randn(1,N)*sqrt(C.kb*T/Eme);
end
v = sqrt(vx.^2+vy.^2);



if boxes
%     figure (2)
%     subplot(2,1,1);
%     plot (box1(1,:),box1(2,:),'-k')
%     hold on
%     plot (box2(1,:),box2(2,:),'-k')
end
if partical_colission
%     figure
%     histogram(v)
    p =1-exp(-dt/Tmn);
else
    p=0;
end


if a3p1 ==1

    econcentration =1e15;
    J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;
    Vapplied =0.1;
    Ex = Vapplied/dimx
```

```
    Ey =0
    Fx = (C.q_0)*Ex
    Fy = (C.q_0)*Ey
    AccelerationX =(Fx./(Eme))
    AccelerationY =(Fy./(Eme))
end
if a3p2 == 1
     econcentration =1e15;
     Vapplied =linspace(0.1,10,15);
   boundy = [Vapplied(w) nan 0 nan];
   J_driftx = mean(vx)*econcentration*(-C.q_0)*dimy*100;

    [Ex, Ey]= calcEfeild (dimx,dimy,dx2,dy2,boundy,0, box1, box2);

end

for l=1:m

    if a3p2 ==1

    lincordx = floor(xpos(1,:)/dx2)+1;
    lincordy = floor(ypos(1,:)/dy2)+1;

    lincordx(lincordx==round(dimx/dx2))=1;
    lincordy(lincordy==round(dimy/dy2))=1;
%    for k = 1:N
    ExatPos=
 (Ex(sub2ind(size(Ex),lincordy,lincordx))+Ex(sub2ind(size(Ex),lincordy
+1,lincordx))+Ex(sub2ind(size(Ex),lincordy,lincordx
+1))+Ex(sub2ind(size(Ex),lincordy+1,lincordx+1)))/4;
    EyatPos=
 (Ey(sub2ind(size(Ey),lincordy,lincordx))+Ey(sub2ind(size(Ey),lincordy
+1,lincordx))+Ey(sub2ind(size(Ey),lincordy,lincordx
+1))+Ey(sub2ind(size(Ey),lincordy+1,lincordx+1)))/4;
 %   end

        Fx = (-C.q_0)*ExatPos;
        Fy = (-C.q_0)*EyatPos;

    end

    %updates position
    vx = vx+(Fx./(Eme))*(dt);
    vy =vy+(Fy./(Eme))*(dt);
    v = sqrt(vx.^2+vy.^2);
    J_driftx =[J_driftx, -C.q_0*econcentration *mean(vx)*dimy*100];

    xpos(l+1,:)=xpos(l,:)+(vx*dt);
    ypos(l+1,:)=ypos(l,:)+(vy*dt);

    % fineds the distance traveled by each partical
    dtraveled = dtraveled + sqrt ((xpos(l,:)-xpos(l
+1,:)).^2+(ypos(l,:)-ypos(l+1,:)).^2);
    slope =(vy./vx);
```

```
    % sets up colision detection by determining if the particals have
 the
    % distance to the edges

    dtt = sqrt(((dimy -ypos(l+1,:)).^2)+((((dimy-ypos(l+1,:))./
slope)).^2));
    dtb = sqrt(((0 -ypos(l+1,:)).^2)+((((-ypos(l+1,:))./slope)).^2));
    if(boxes)
        dttbf =((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*
 sqrt(((box1(2,3) -ypos(l+1,:)).^2)+((((box1(2,3)-ypos(l+1,:))./
slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l+1,:)<=box1(1,2))).*100;
        dtbbf = ((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*sqrt(((box2(2,1) -ypos(l+1,:)).^2)+((((box2(2,1)-
ypos(l+1,:))./slope)).^2))+~((xpos(l+1,:)>=box1(1,1)&xpos(l
+1,:)<=box1(1,2))).*100;
        dts1 =( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,1)-xpos(l+1,:))).^2+(box1(1,1)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
        dts2=( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*
 sqrt((slope.*(box1(1,2)-xpos(l+1,:))).^2+(box1(1,2)-xpos(l+1,:)).^2)
 +~( ypos(l+1,:)<=box2(2,1)|ypos(l+1,:)>=box1(2,3)).*100;
    else
        dttbf =ones(1,N).*100;
        dtbbf = ones(1,N).*100;
        dts1 =ones(1,N).*100;
        dts2=ones(1,N).*100;

    end
    %counts the number of colissions that have occured and
    pc =c;
    c=(((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))));
    colision_count = colision_count+(((dts1<1e-9|dts2<1e-9)|
((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9))));
    if specular
        % basic colission part one
        vy= -((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9).*2-1).*vy;
        vx= -((dts1<1e-9|dts2<1e-9).*2-1).*vx;
    else
        % re thermalized velocities for part 3
        %if rethermalized volocity is in the same direection as
 previouse
        %than flip signs
        signx=sign(vx);
        signy=sign(vy);



        vx= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vx;
        vy= ((((dts1<1e-9|dts2<1e-9)|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|
dtbbf<1e-9))))&~pc).*(randn(1,N)*sqrt(C.kb*T/Eme)) +(~((((dts1<1e-9|
dts2<1e-9))|((dtt<1e-9|dtb<1e-9|dttbf<1e-9|dtbbf<1e-9)))&~pc)).*vy;
```

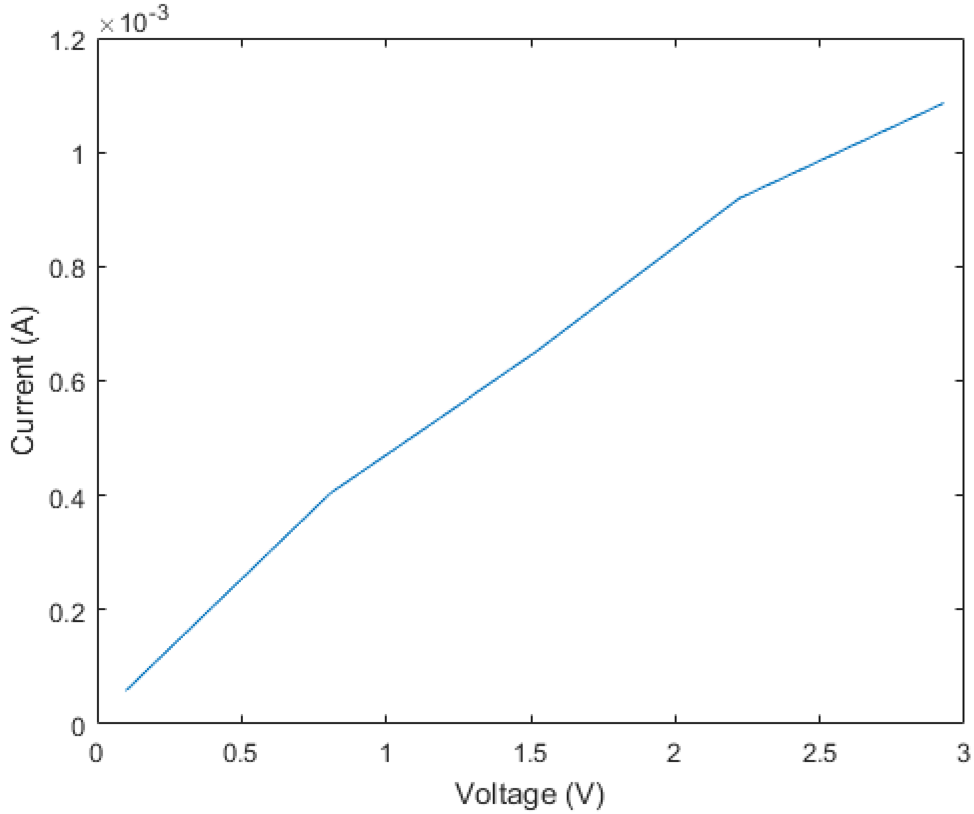
```
        vx = (((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc).*-1.*vx+(~(((dts1<1e-9&sign(vx)==1)|(dts2<1e-9
 &sign(vx)==-1))&~pc)).*vx;
        vy = (((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc)).*-1.*vy
+(~(((((dtt<1e-9&sign(vy)==1)|(dtb<1e-9&sign(vy)==-1)|
(dttbf<1e-9&sign(vy)==1)|(dtbbf<1e-9&sign(vy)==-1)))&~pc))).*vy;



    end
    % loop condition for end boundries
    xpos(l+1,:) = (xpos(l+1,:)>dimx).*0+(xpos(l+1,:)<0).*dimx+~(xpos(l
+1,:)>=dimx|xpos(l+1,:)<=0).*xpos(l+1,:);
    % colisions with other particals are only alouwed when partical is
 away
    % from the edges and has not colided with an edge Part 2 and 3
    colision = p>rand(1,N)&~(((dts1<30e-9|dts2<30e-9)|((dtt<30e-9|
dtb<10e-9|dttbf<30e-9|dtbbf<30e-9))))&~c;
    colision_count = colision_count+colision;
    vy=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vy;
    vx=colision.*(randn(1,N)*sqrt(C.kb*T/Eme))+(~colision).*vx;



    % skips the plot of the x boundry transition
    skip = (xpos(l+1,:)>=dimx|xpos(l+1,:)<=0);
    % progress = (l/m)*100
    v = sqrt(vx.^2+vy.^2);
    c=0;
    %finds the current temperature
    Temp =[Temp, mean((v.^2)*Eme/(2*C.kb))];


end

squarcount= zeros(round(dimy/1e-9),round(dimx/1e-9));
temps= zeros(round(dimy/1e-9),round(dimx/1e-9));


current(w) = mean(J_driftx);

end
```

The plot below shows the current vs voltage for the nanostructuire. From this plot it was found that the resistance of the device was:

```
f = fittype('a*x');
[fited, gof,
 fitinfo]=fit(current(1:Numvolt)',Vapplied(1:Numvolt)',f,'StartPoint',
[1]);
Res = fited.a;
fprintf("The resistance of the divice is: %d ohms", Res);
figure
```

```
plot(Vapplied(1:5),current(1:5))
xlabel('Voltage (V)')
ylabel('Current (A)')
```

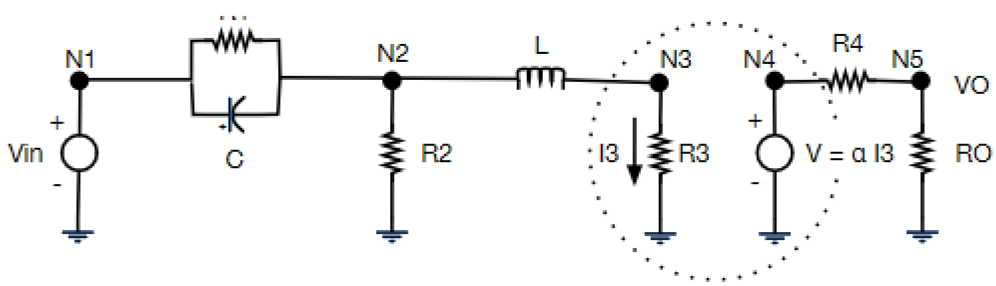
The resistance of the divice is: 3.737607e+03 ohms



Part 2

This section looks at the DC and AC simulation of the circuit in the figure below. From this figure the fallowing equations were generated. From these equations the falowing G and C matricies were created. The plots below shows the results that were found in PA9 and the results using the simulated nanostructure.

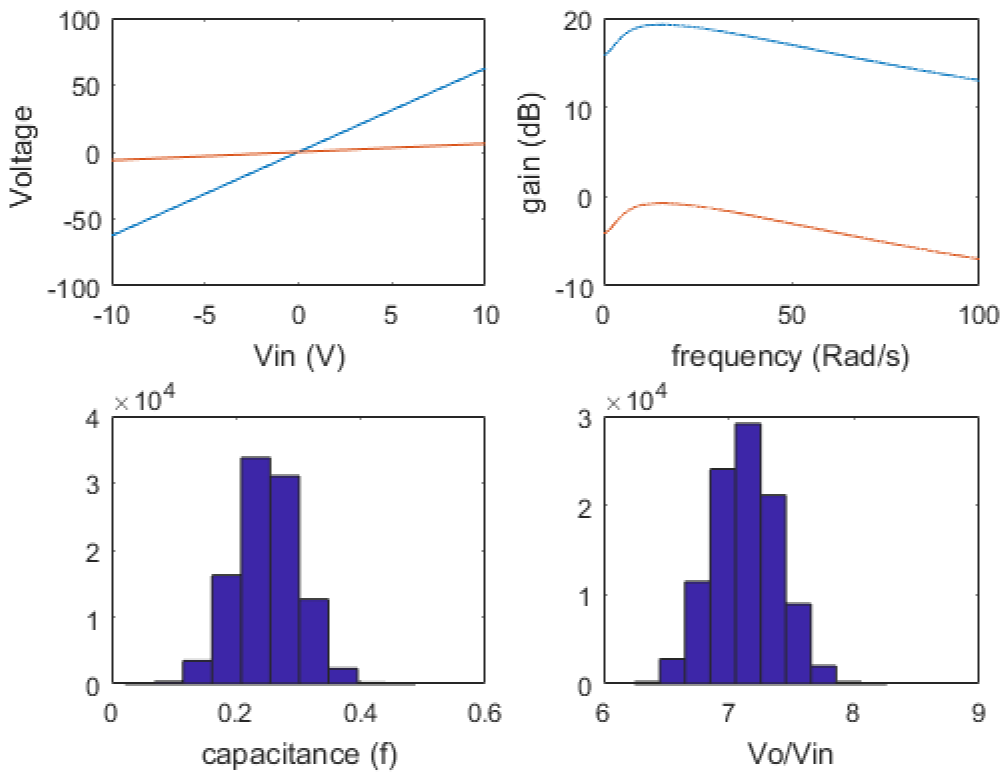
$0 = (Vn1 - Vn2)/R1 + Cd * (Vn1 - Vn2)/dt - Vn2/R2 - Il$ %0= L*d(IL)/dt + Vn1 - Vn2$ %0= IL - Vn3/R3$ %0= I4 - (Vn4 -Vn5)/R4$ %VIN = Vn1$ %0= Vn4 - a*Vn3/R3$ %0= I4 - Vn5/Ro$

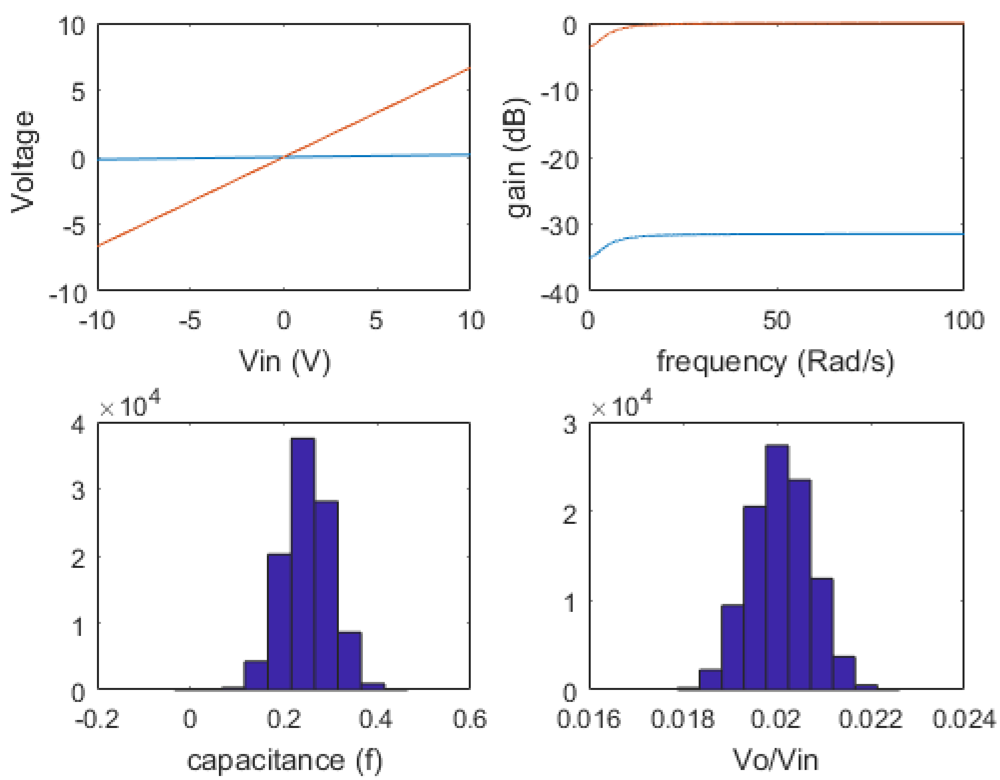


conduction matrix

```
R=Res;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/R 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/R 0 1 0 0;
    0 0 0 0 0 1 1/1000]
% capacitance matrix
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 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]

figure
V= solveCircuit(10);
figure
V2= solveCircuit(Res);
```

```
G =

    1.0000         0         0         0         0         0         0
         0   -1.0000    1.0000         0         0         0         0
    1.0000   -1.5000         0   -1.0000         0         0         0
         0         0   -0.0003    1.0000         0         0         0
         0         0         0         0   10.0000    1.0000  -10.0000
         0         0   -0.0268         0    1.0000         0         0
         0         0         0         0         0    1.0000    0.0010


C =

         0         0         0         0         0         0         0
         0         0         0    0.2000         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
```


Voltage
Vin (V)
gain (dB)
frequency (Rad/s)
×10^4
capacitance (f)
×10^4
Vo/Vin


Voltage
Vin (V)
gain (dB)
frequency (Rad/s)
×10^4
capacitance (f)
×10^4
Vo/Vin

From these plots it can be seen that the circuit has a bandpass or high pass filter responce given the resistance of R3.

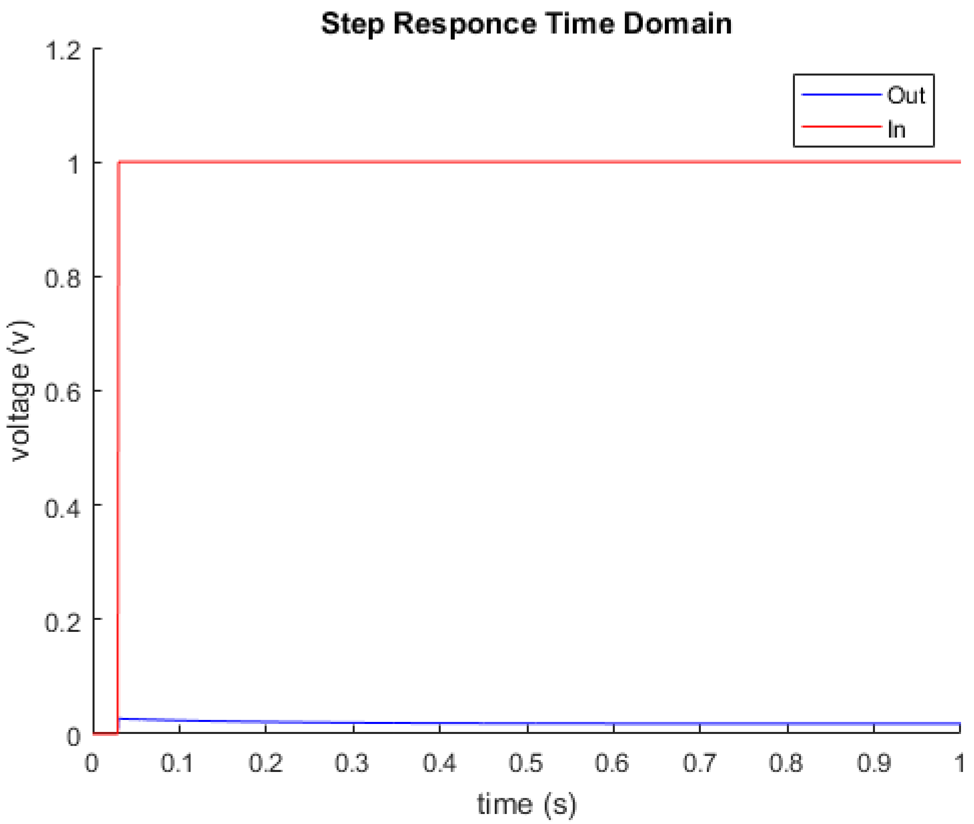
Part 3

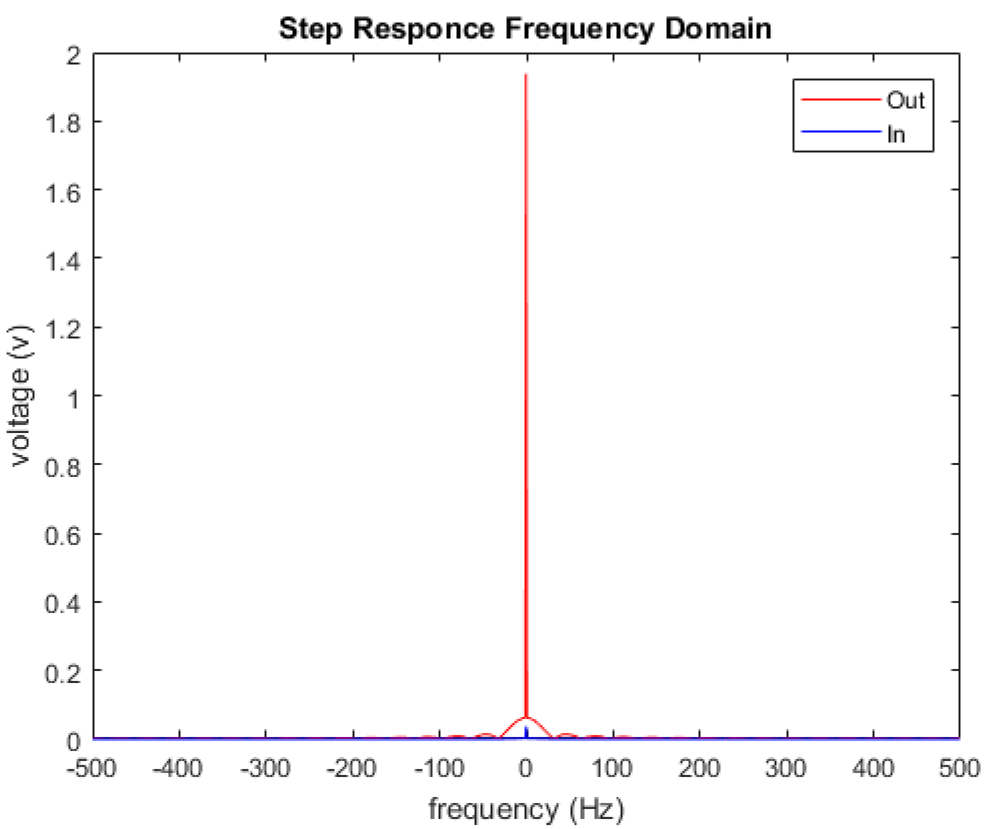
This section looks at the time domain simulation of the circuit above. By inspection and the results of the previouse section the circuit has a high pass filter responce with a low frequency pole. The time domain simulation was generated using the fallowing equation: $V+ = A^{-1} * [C * V/dt + F]$ where A is $(C/dt + G)$, C, G are the conductance and capacitance maticies and F is the input vector. This secton will apply three diffrent signals to the circuit, a heavy side step, sin wave and a guassian pulse. The plots below show the time domain and frequency domain results for each of the cases.

```
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000]
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 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]
dt = 1e-3;
figure
X = linspace(0, 1,1/dt);
temp= heaviside(X-0.03);
F = zeros(7,1001);
F(1,:)=[0,temp];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
    hold on
    if l>1
    plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

    end
%pause(0.01);
end
   title('Step Responce Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
```

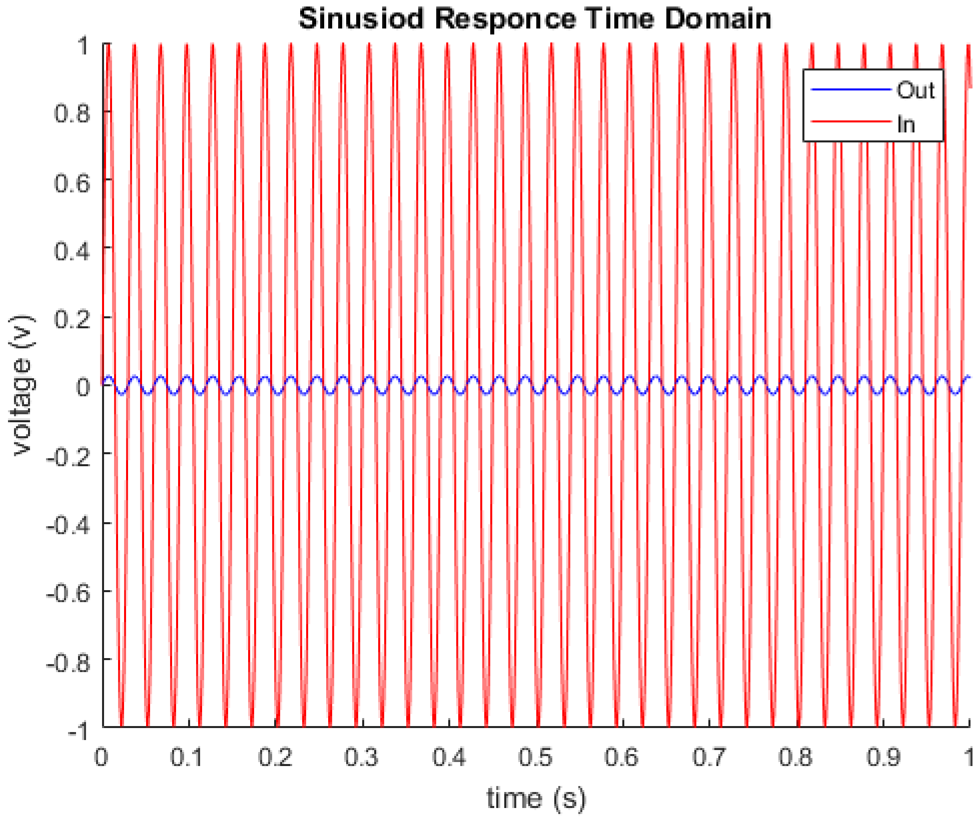
```
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Step Responce Frequency Domain')
legend('Out', 'In')
xlabel('frequency (Hz)')
ylabel('voltage (v)')
```

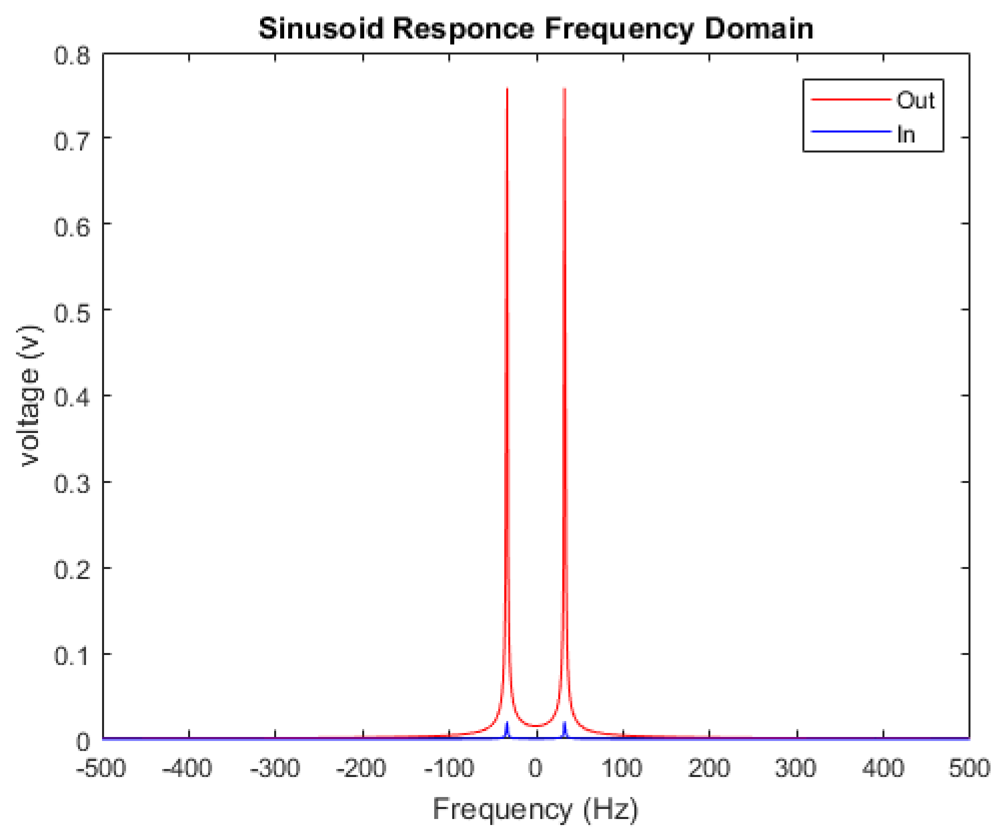
```
G =

    1.0000         0         0         0         0         0         0
         0   -1.0000    1.0000         0         0         0         0
    1.0000   -1.5000         0   -1.0000         0         0         0
         0         0   -0.0003    1.0000         0         0         0
         0         0         0         0   10.0000    1.0000  -10.0000
         0         0   -0.0268         0    1.0000         0         0
         0         0         0         0         0    1.0000    0.0010


C =

         0         0         0         0         0         0         0
         0         0         0    0.2000         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
```


Step Responce Time Domain
Out
In
voltage (v)
time (s)
1.2
1
0.8
0.6
0.4
0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9

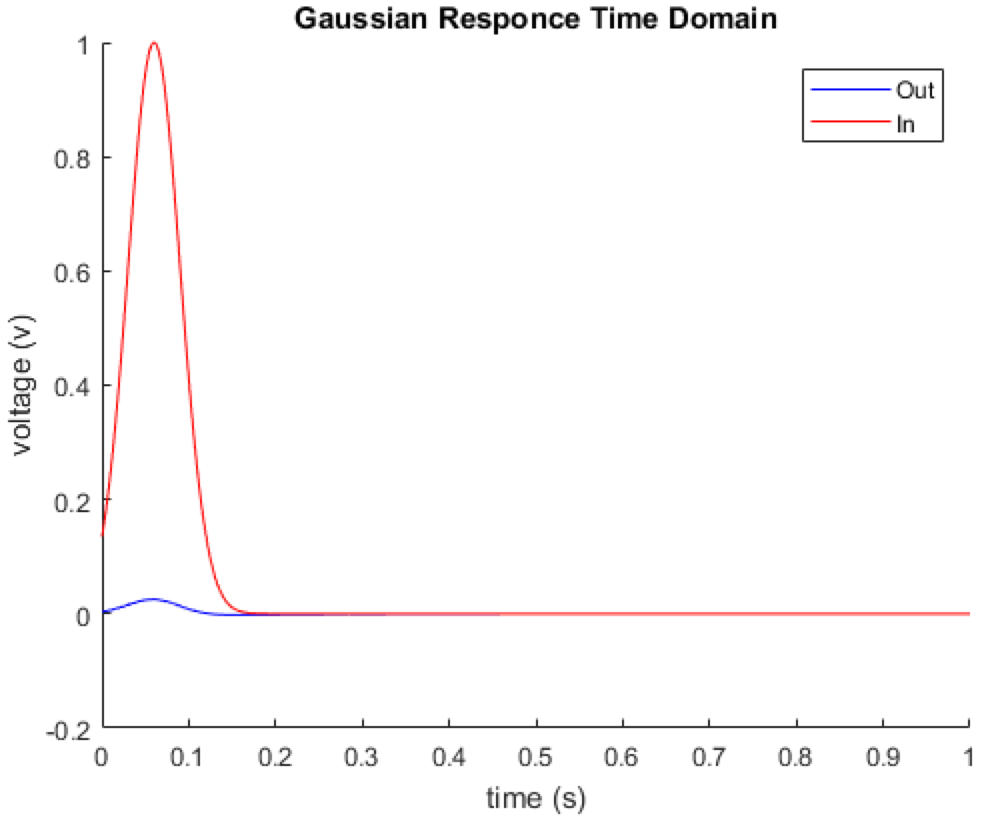

Step Responce Frequency Domain
Out
In
voltage (v)
frequency (Hz)
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-500
-400
-300
-200
-100
100
200
300
400
500

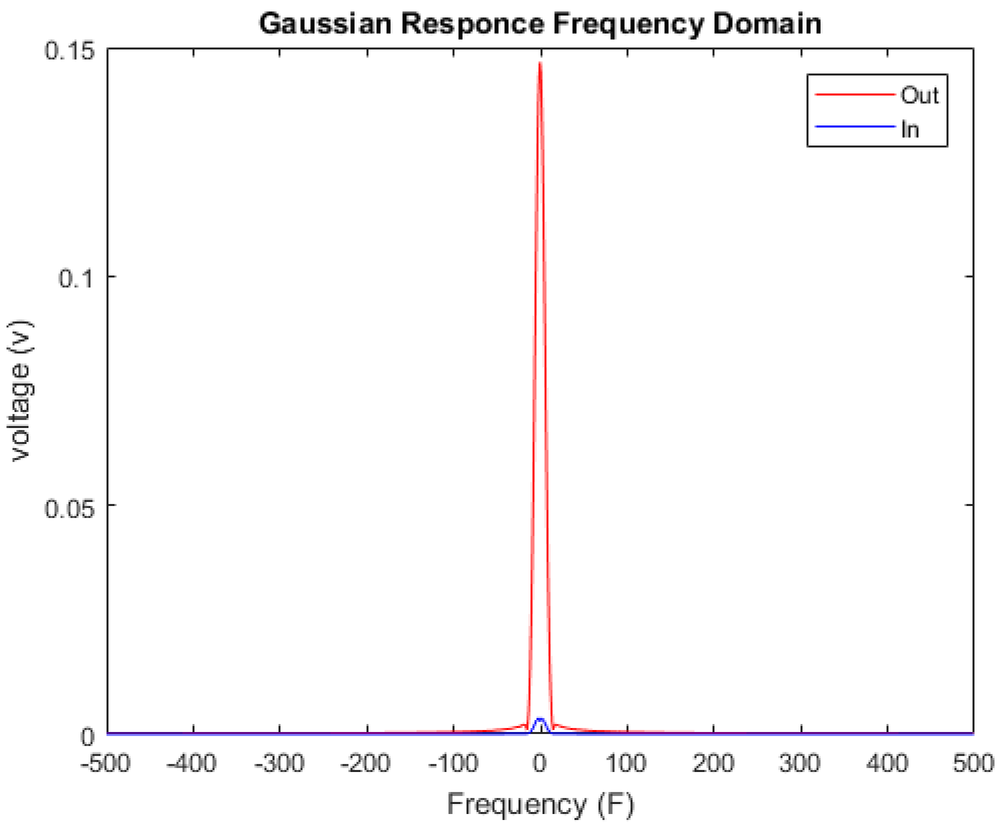
```
figure
dt = 1e-3;

X = linspace(0, 1,1/dt);
temp= sin(2*pi*X*(1/0.03));
F = zeros(7,1001);
F(1,:)=[0,temp];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
    hold on
   if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

   end
%pause(0.01);
end
title('Sinusiod Responce Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')
```


Sinusiod Responce Time Domain
voltage (v)
time (s)
Out
In
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Sinusoid Responce Frequency Domain
voltage (v)
Frequency (Hz)
Out
In
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

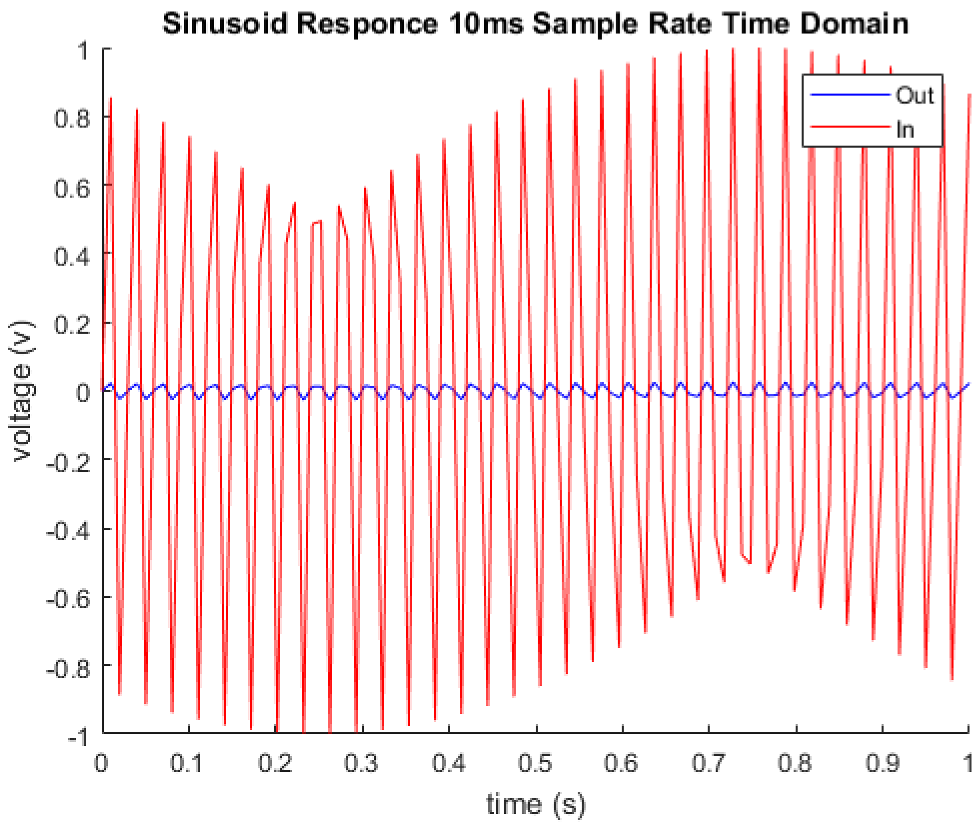
```
figure
dt = 1e-3;

X = linspace(0, 1,1/dt);
temp= exp(-0.5.*(((X-0.06).^2)./0.03.^2));
F = zeros(7,1001);
F(1,:)=[0,temp];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

  end
    pause(0.01);
end
 title('Gaussian Responce Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Gaussian Responce Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (F)')
ylabel('voltage (v)')
```

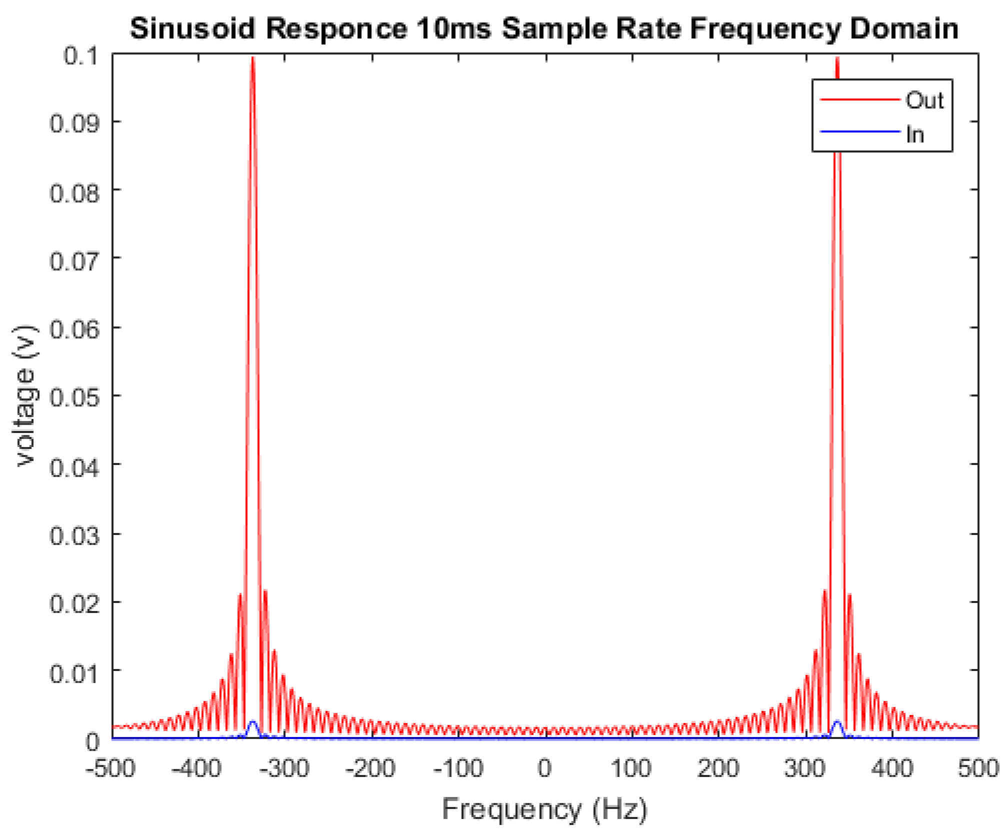

Gaussian Responce Time Domain
Out
In
voltage (v)
time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Gaussian Responce Frequency Domain
Out
In
voltage (v)
Frequency (F)
0.15
0.1
0.05
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

The plot below shows the results of the time domain simulation of the circuit given a larger timestep. From this plot it can be seen that the larger timestep causes the plot to become inproperly sampled to the point that it does not appear to be a sin wave. This can also be seen in the frequencey where the responce nolonger has a pure tone it instead has a sin x over x roll off.

```
figure
dt = 10e-3;

X = linspace(0, 1,1/dt);
temp= sin(2*pi*X*(1/0.03));
F = zeros(7,1/dt+1);
F(1,:)=[0,temp];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
    hold on
   if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

   end
%pause(0.01);
end
 title('Sinusoid Responce 10ms Sample Rate Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 10ms Sample Rate Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')
```


Sinusoid Responce 10ms Sample Rate Time Domain
Out
In
voltage (v)
time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Sinusoid Responce 10ms Sample Rate Frequency Domain
Out
In
voltage (v)
Frequency (Hz)
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

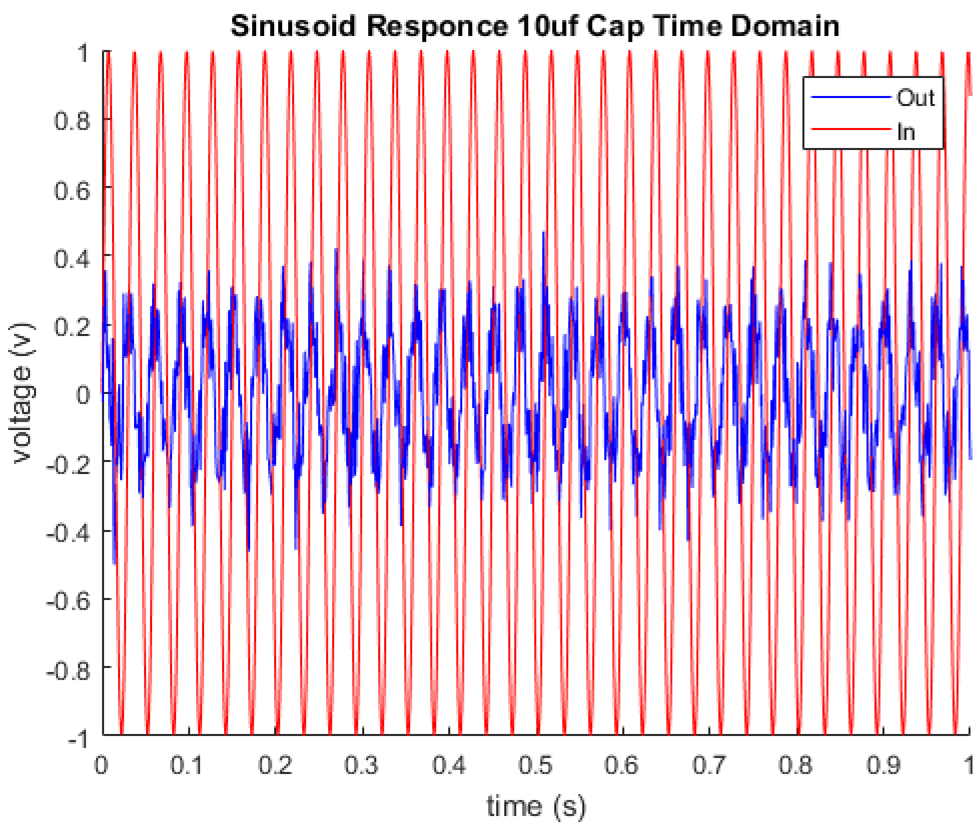
Part 4

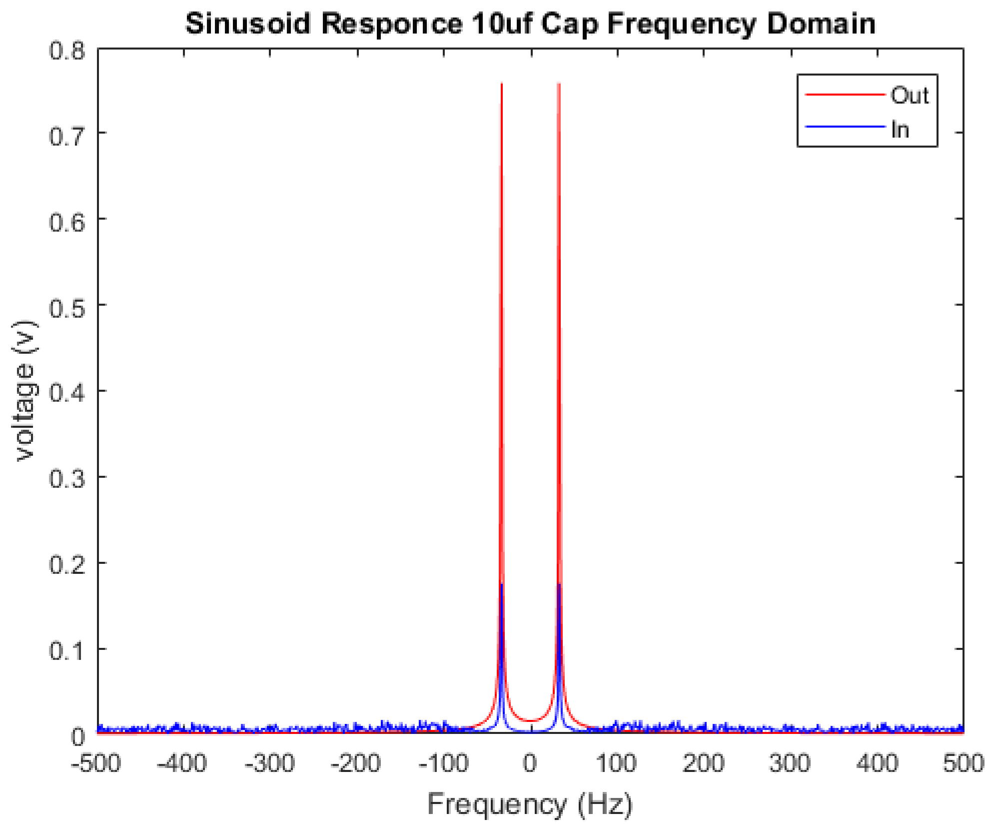
This section looks at the effect of a noise source and its bandwidth on the circuit. Feading a sin wave into the cicut the output voltage was ploted below. From the time domain simulation it can be seen clearly that the output is no longer a pure sin wave but instead has noise applied to it. This is also evedent in the frequency domain where the spikes repesienting the output are taller and have a wider base than the frequency responce found in the previous section. This indicates that there is more energy at these frequencies and that it is leaking over to the surounding frequncies.

```
cn = 0.00001;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000];
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 0 0 0 0 0;
    0 0 -cn 0 0 0 0;
    0 0 0 0 0 0 0;
    0 0 -100*cn 0 0 0 0;
    0 0 0 0 0 0 0]
figure
dt = 1e-3;

X = linspace(0, 1,1/dt);
temp=  sin(2*pi*X*(1/0.03));
current_noise= 0.001*randn(1,1000);
F = zeros(7,1001);
F(1,:)=[0,temp];
F(4,:)=[0,current_noise];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

  end
    pause(0.01);
end
  title('Sinusoid Responce 10uf Cap Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
```

```
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 10uf Cap Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')
```

```
C =

         0         0         0         0         0         0         0
         0         0         0    0.2000         0         0         0
    0.2500   -0.2500         0         0         0         0         0
         0         0   -0.0000         0         0         0         0
         0         0         0         0         0         0         0
         0         0   -0.0010         0         0         0         0
         0         0         0         0         0         0         0
```



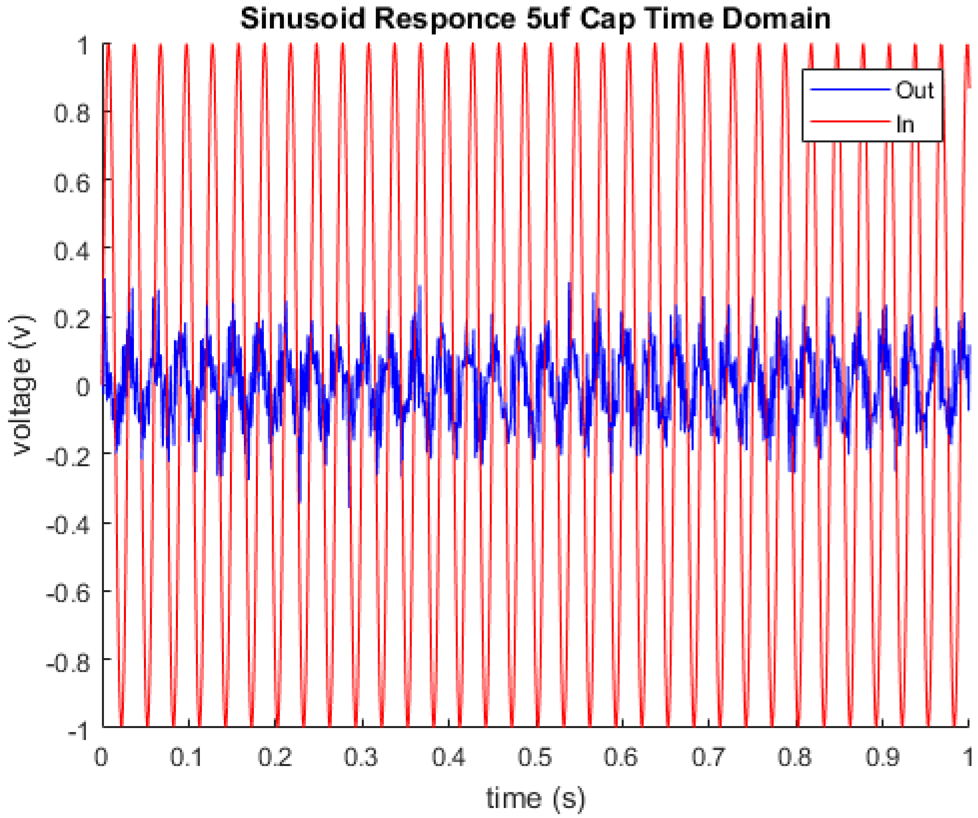


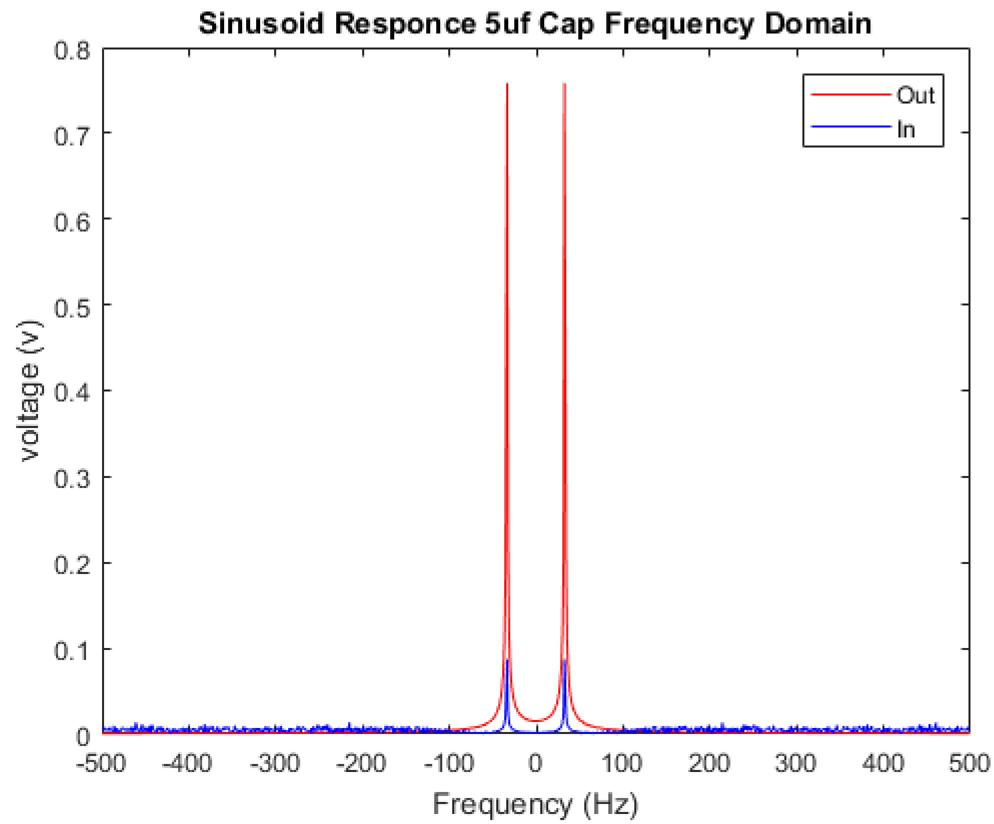
The plots below show the effect that varying the paralel capacitance of R3 has on the noise in the circuit. From the frequency domain plots it can be seen that the larger this capacitance is the effect of the noise on the cicuit is lessend and the smaller this capacitance is the more of an effect the noise has on the circuit. This can be seen by how the spikes representing the time domain of the output are larger when the capacitance is decreased and smaller when the capacitanc is increased. This comes from the equation for bandwidth: %BW = 1/(2*pi*C*R)% where R and C are the resistace of the nanostructure and C is the capacitance.

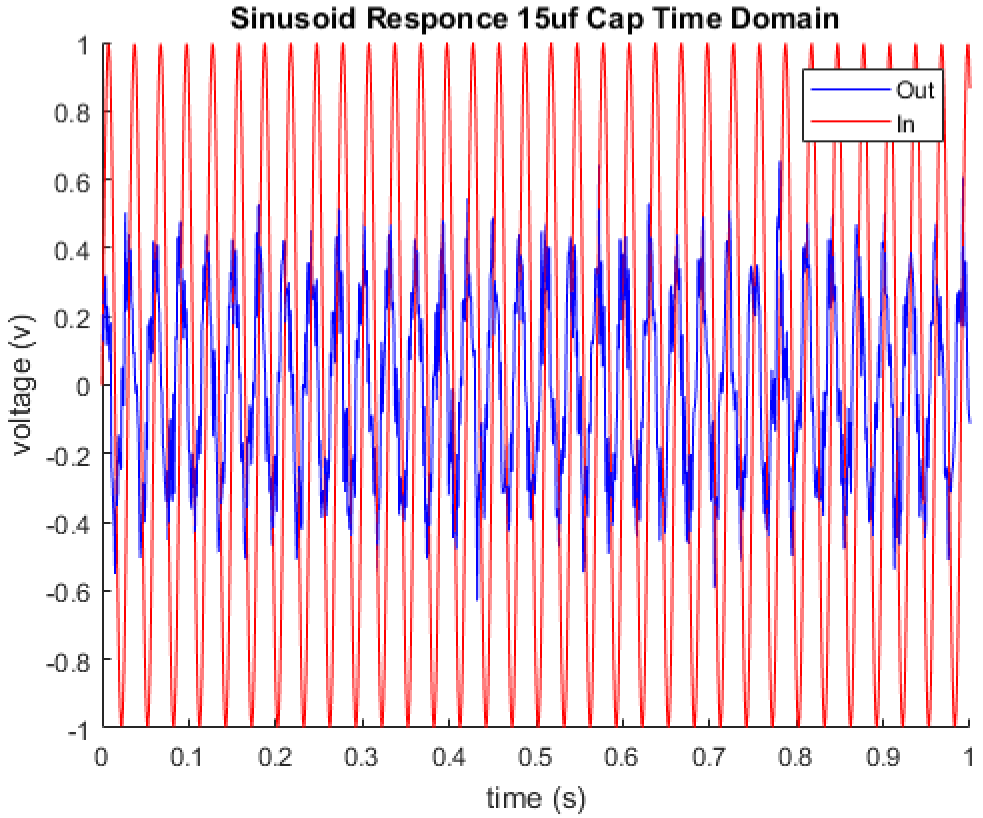
```
cn = 0.000005;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000];
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 0 0 0 0 0;
    0 0 -cn 0 0 0 0;
    0 0 0 0 0 0 0;
    0 0 -100*cn 0 0 0 0;
    0 0 0 0 0 0 0];
figure
dt = 1e-3;

X = linspace(0, 1,1/dt);
```

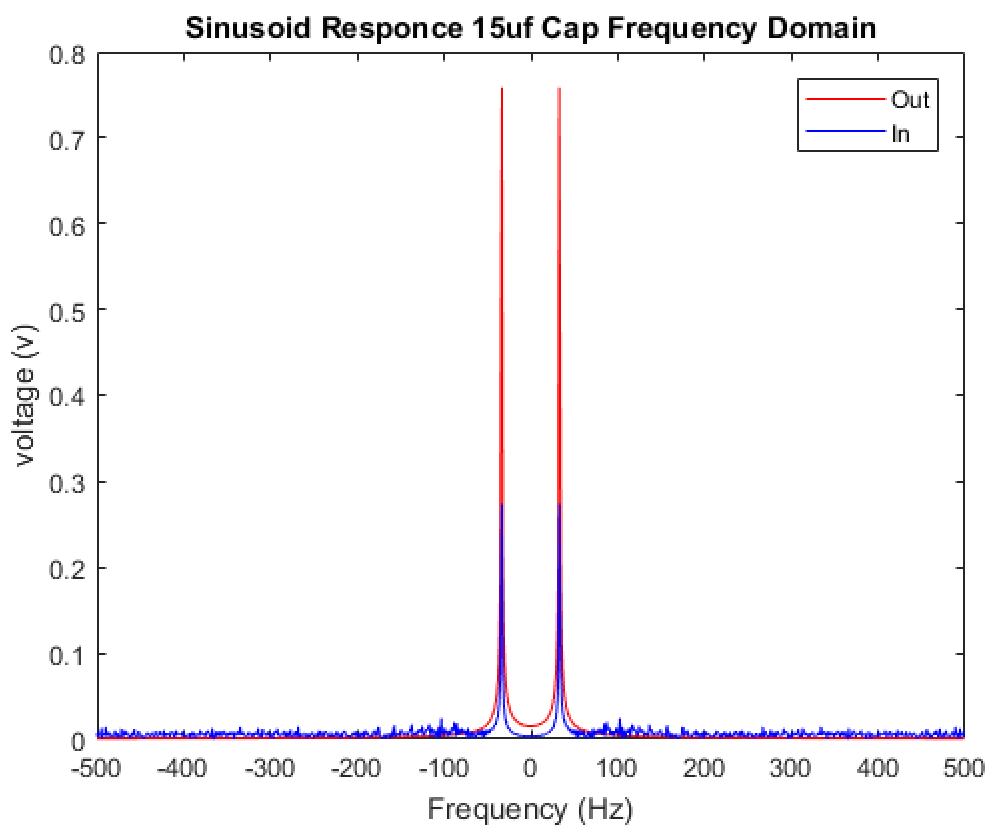
```
temp=  sin(2*pi*X*(1/0.03));
current_noise= 0.001*randn(1,1000);
F = zeros(7,1001);
F(1,:)=[0,temp];
F(4,:)=[0,current_noise];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

  end
    pause(0.01);
end
  title('Sinusoid Responce 5uf Cap Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 5uf Cap Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')

cn = 0.000015;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000];
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 0 0 0 0 0;
    0 0 -cn 0 0 0 0;
    0 0 0 0 0 0 0;
    0 0 -100*cn 0 0 0 0;
    0 0 0 0 0 0 0];
figure
dt = 1e-3;

X = linspace(0, 1,1/dt);
temp=  sin(2*pi*X*(1/0.03));
current_noise= 0.001*randn(1,1000);
```

```
F = zeros(7,1001);
F(1,:)=[0,temp];
F(4,:)=[0,current_noise];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

  end
    pause(0.01);
end
title('Sinusoid Responce 15uf Cap Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 15uf Cap Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')
cn = 0.00001;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000];
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 0 0 0 0 0;
    0 0 -cn 0 0 0 0;
    0 0 0 0 0 0 0;
    0 0 -100*cn 0 0 0 0;
    0 0 0 0 0 0 0];
figure
```


Sinusoid Responce 5uf Cap Time Domain
Out
In
voltage (v)
time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Sinusoid Responce 5uf Cap Frequency Domain
Out
In
voltage (v)
Frequency (Hz)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Sinusoid Responce 15uf Cap Time Domain
Out
In
voltage (v)
time (s)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

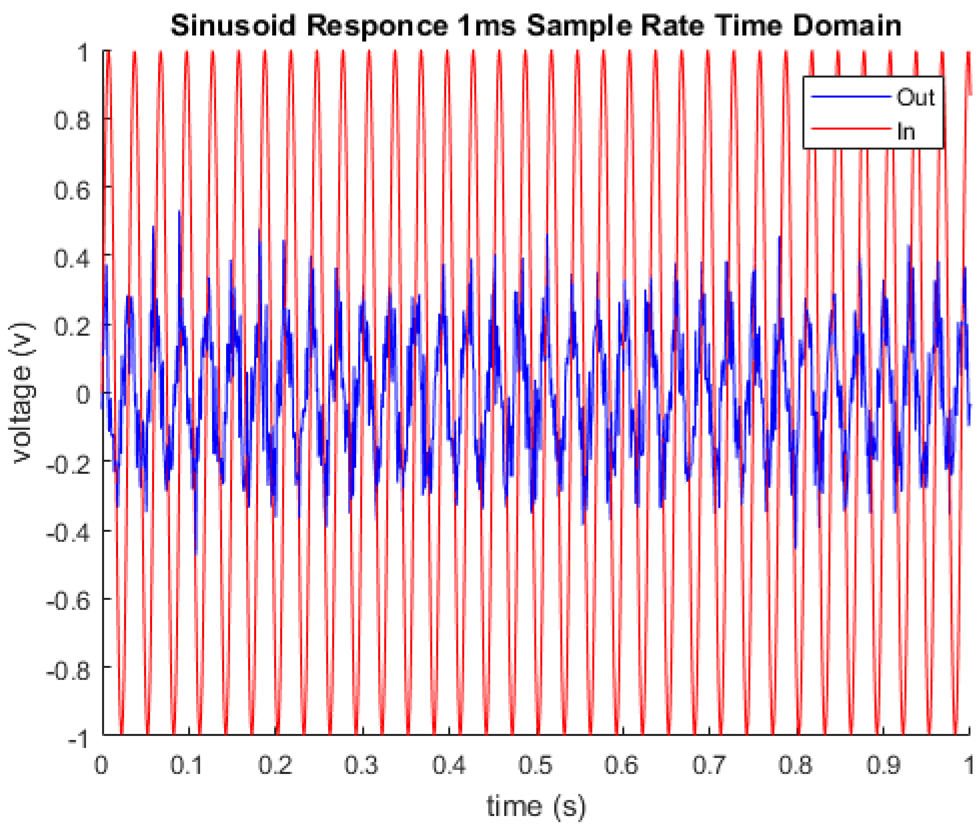

Sinusoid Responce 15uf Cap Frequency Domain
Out
In
voltage (v)
Frequency (Hz)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

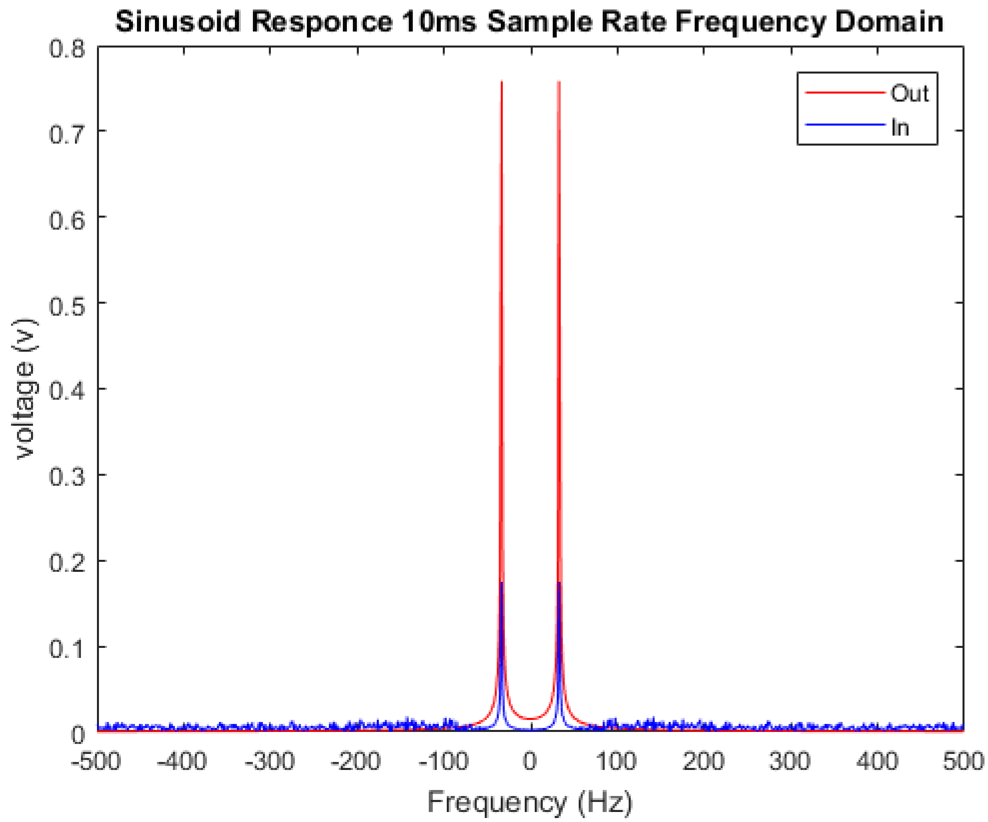
The plots generated with the code below are used to ilistrate the effects of the sampleing time on the noise produced. From the time domain it can be seen that the sampleing time reduces the visual effects of the noise but distorts the waveform by removing the high frequency content. In the frequency domain it can be seen that the sampleing time of the circuit creates aditional sinx over x noise that causes the spectrum to become unpure.

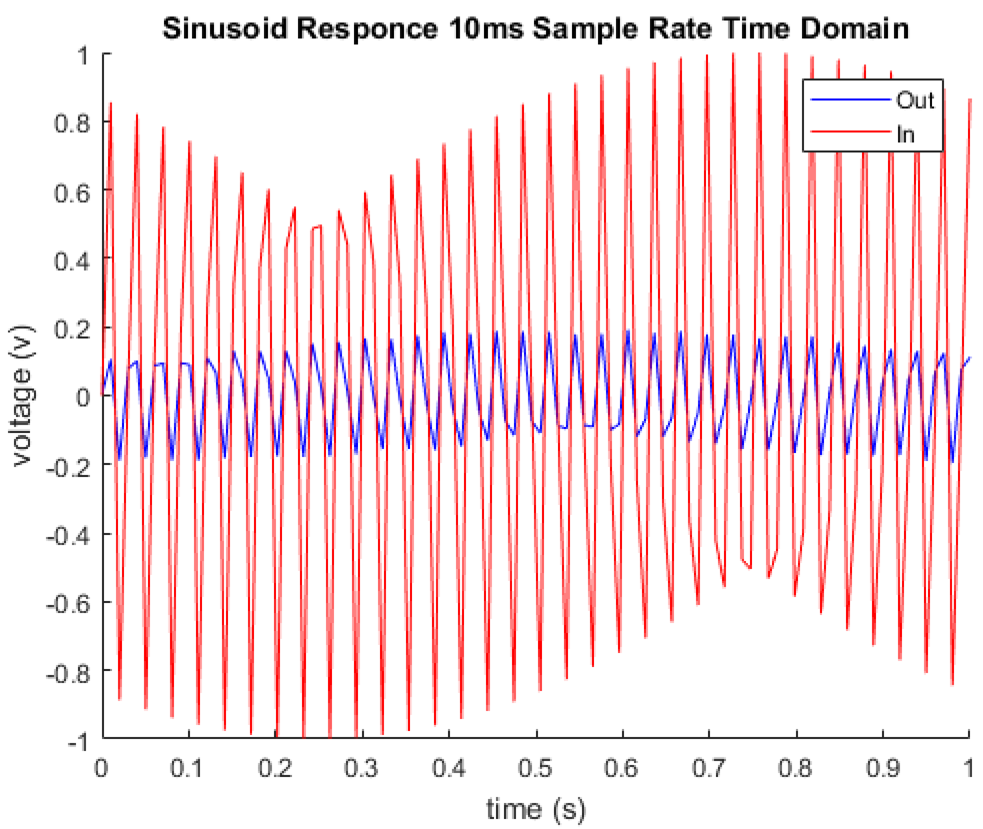
```
dt = 1e-3;

X = linspace(0, 1,1/dt);
temp=  sin(2*pi*X*(1/0.03));
current_noise= 0.001*randn(1,1000);
F = zeros(7,1001);
F(1,:)=[0,temp];
F(4,:)=[0,current_noise];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

   end
%pause(0.01);
end
```

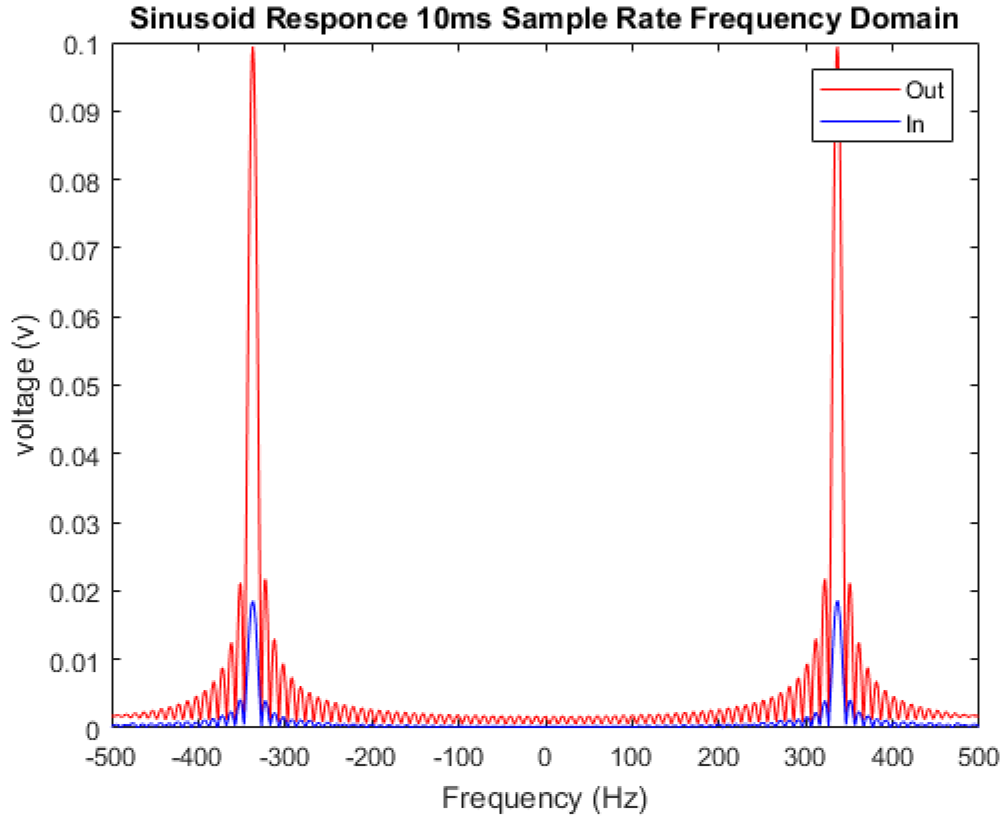
```
 title('Sinusoid Responce 1ms Sample Rate Time Domain')
legend('Out', 'In')
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 10ms Sample Rate Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')

cn = 0.00001;
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/Res 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/Res 0 1 0 0;
    0 0 0 0 0 1 1/1000];
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 0 0 0 0 0;
    0 0 -cn 0 0 0 0;
    0 0 0 0 0 0 0;
    0 0 -100*cn 0 0 0 0;
    0 0 0 0 0 0 0];
figure
dt = 10e-3;

X = linspace(0, 1,1/dt);
temp=  sin(2*pi*X*(1/0.03));
current_noise= 0.001*randn(1,100);
F = zeros(7,101);
F(1,:)=[0,temp];
F(4,:)=[0,current_noise];
A=(C/dt+G);
V = zeros(7,1001);
for l =1:1/dt

   V(:,l+1)=A\(C*(V(:,l)/dt)+F(:,l+1));
   hold on
  if l>1
     plot (X(l-1:l),V(7,l:l+1),'b',X(l-1:l),V(1,l:l+1),'r')

   end
%pause(0.01);
end
title('Sinusoid Responce 10ms Sample Rate Time Domain')
legend('Out', 'In')
```

```
xlabel('time (s)')
ylabel('voltage (v)')
figure
fs = 1/1e-3;
n= length(V);
FFTin = fftshift(fft(V(1,:)));
FFTout =fftshift(fft(V(7,:)));
f=(-n/2:n/2-1)*(fs/n);
plot(f,2*abs(FFTin)/n,'r',f,2*abs(FFTout)/n,'b')
title('Sinusoid Responce 10ms Sample Rate Frequency Domain')
legend('Out', 'In')
xlabel('Frequency (Hz)')
ylabel('voltage (v)')
```




Sinusoid Responce 10ms Sample Rate Frequency Domain
Out
In
voltage (v)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-500
-400
-300
-200
-100
0
100
200
300
400
500
Frequency (Hz)


Sinusoid Responce 10ms Sample Rate Time Domain
Out
In
voltage (v)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
time (s)



Part 6

If the nonlinear voltage source was added to the circuit a B matrex containing the non linear of the circuit would need to be added to the equations that difine the circuit for the transiant simulation the equation would be: $V+ = A^{-1} * [C * V/dt + F - B]$. This equation is set equal to zero and solved at each time step. The result is then divided from the left by $H = C/dt + G - J$ where J is the dirivitive with respect to voltage of the B matrix. The result of this is then added to the output vector and the process is repeated until the result is less than the persision of the simulator.

Conclution

In conclution this Assignment successfuly analised the DC, AC and Transiant protreties of a circuuit using the MNA method. This assignment also used the results of a monti carlo simulation to find the resistance of a nanostructure and used it within the solver to find the outputs of the circuit. It was also found that noise and sample rate have a domatic effect on the quality of the output signal.

Appendix 1: E-Feild solver

```
function [Ex, Ey]= calcEfeild (dimx,dimy,Dx,Dy,boundy,pr, box1, box2,
 box3)


sig = 1;
sigbox = 10e-2;
```

```
dx =Dx;
dy=Dy;

box1pos= [round(box1(1,1)/dx),round(box1(2,1)/dy)];
box2pos= [round(box2(1,1)/dx),round(box2(2,1)/dy)];
box1dim= [round((box1(1,2)-box1(1,1))/dx),round((box1(2,1)-box1(2,3))/
dy)];
box2dim= [round((box2(1,2)-box2(1,1))/dx),round((box2(2,1)-box2(2,3))/
dy)];


nx = round(dimx/dx);
ny = round(dimy/dy);
V3 = zeros(ny,nx);
G2= sparse(nx*ny,nx*ny);
V02=1;
BC2= boundy;
B2 = zeros(1,nx*ny);
cond = zeros (ny,nx);
% sets up conduction map

for p = 1:ny
    for m = 1:nx

        if ((m >=box1pos(1)&m <=box1pos(1)+box1dim(1))&p
 <=box1pos(2)&p >=box1pos(2)-box1dim(2))|((m >=box2pos(1)&m
 <=box2pos(1)+box2dim(1))&p <=box2pos(2)&p >=box2pos(2)-box2dim(2))
            cond(p,m) =sigbox;
        else
           cond(p,m) =sig;
        end
    end
end


%sets boundry conditions
for p = 1:size(B2,2)

    if (p== 1)
        if isnan(BC2(1))

        else
            B2(p)=BC2(1);
        end
    elseif (p == nx)
        if isnan(BC2(3))

        else
            B2(p)=BC2(3);
        end
    elseif (p == (1+(ny-1)*nx))
        if isnan(BC2(1))
```

```
        else
            B2(p)=BC2(1);
        end
    elseif(p == nx*ny)
        if isnan(BC2(3))

        else
            B2(p)=BC2(3);
        end
    elseif(mod(p,nx)==0)
        if isnan(BC2(3))

        else
            B2(p)=BC2(3);
        end
    elseif(mod(p-1,nx)==0)
        if isnan(BC2(1))

        else
            B2(p) = BC2(1);
        end
    elseif(1<p&p<nx)
        if isnan(BC2(4))

        else
            B2(p) =BC2(4);
        end
    elseif((1+(ny-1)*nx)<p&p<nx*ny)
        if isnan(BC2(2))

        else
            B2(p) =BC2(2);
        end
    else

    end

end
% creates conduction matrix
for p = 1:ny
    for m = 1:nx
        n = m+(p-1)*nx;
        nxm = (m-1)+(p-1)*nx;
        nxp = (m+1)+(p-1)*nx;
        nym = (m)+(p-2)*nx;
        nyp = m+(p)*nx;
        nxm2 = (m-2)+(p-1)*nx;
        nxp2=(m+2)+(p-1)*nx;
        nym2 = (m)+(p-3)*nx;
        nyp2 = m+(p+1)*nx;
        nxp3 = (m+3)+(p-1)*nx;
        nxm3 = (m-3)+(p-1)*nx;

        if m == 1
```

```
            G2(n,:)=0;
            G2(n,n)=1;

        elseif m==nx
            G2(n,:)=0;
            G2(n,n)=1;

        elseif p ==1

            ryp = (cond(p,m)+cond(p+1,m))/2;
            rxp = (cond(p,m)+cond(p,m+1))/2;
            rxm = (cond(p,m)+cond(p,m-1))/2;

            G2(n,n) = -(ryp+rxp+rxm);
            G2(n,nyp)= ryp;
            G2(n,nxm)= rxm;
            G2(n,nxp) = rxp;

        elseif p==ny
            rym = (cond(p,m)+cond(p-1,m))/2;
            rxp = (cond(p,m)+cond(p,m+1))/2;
            rxm = (cond(p,m)+cond(p,m-1))/2;

            G2(n,n) = -(rym+rxp+rxm);
            G2(n,nym)= rym;
            G2(n,nxm)= rxm;
            G2(n,nxp) = rxp;
        else
            ryp = (cond(p,m)+cond(p+1,m))/2;
            rym = (cond(p,m)+cond(p-1,m))/2;
            rxp = (cond(p,m)+cond(p,m+1))/2;
            rxm = (cond(p,m)+cond(p,m-1))/2;

            G2(n,n) = -(ryp+rym+rxp+rxm);
            G2(n,nyp)= ryp;
            G2(n,nym)= rym;
            G2(n,nxm)= rxm;
            G2(n,nxp) = rxp;
        end

    end
end
%create voltage map
V3 = G2\B2';
Vout2 = zeros(ny, nx);
for p = 1:ny
    for m = 1:nx
        n = m+(p-1)*nx;
        Vout2(p,m) =V3(n);
    end
end
if pr==1
```

```
figure
title('Voltage')
surf(Vout2)
xlabel('X')
ylabel('Y')
zlabel('V')
view(-45, -45)
end
Ex= size (Vout2);
Ey = size (Vout2);
for p = 1:ny
    for m = 1:nx

        if (m==1)
            Ex(p,m) = (Vout2(p,m+1)-Vout2(p,m))/dx;
        elseif m == nx
             Ex(p,m) = (Vout2(p,m)-Vout2(p,m-1))/dx;
        else
             Ex(p,m) = (Vout2(p,m+1)-Vout2(p,m-1))/(dx*2);

        end
         if (p==1)
            Ey(p,m) = (Vout2(p+1,m)-Vout2(p,m))/dy;
        elseif p == ny
             Ey(p,m) = (Vout2(p,m)-Vout2(p-1,m))/dy;
        else
             Ey(p,m) = (Vout2(p+1,m)-Vout2(p-1,m))/(dy*2);

        end
    end
end
Ex=-Ex;
Ey=-Ey;
end
```

Appendix 2: circuit solver

```
function VDC = solveCircuit(R)
%DC sweep
Vin = linspace(-10, 10,21);
% syms v(t) v1(t) v2(t) v3(t) v4(t) v5(t) v6(t)
% V=[v; v1 ;v2 ;v3 ; v4 ;v5; v6];
% conduction matrix
G = [1 0 0 0 0 0 0;
    0 -1 1 0 0 0 0;
    1 -1.5 0 -1 0 0 0;
    0 0 -1/R 1 0 0 0;
    0 0 0 0 10 1 -10;
    0 0 -100/R 0 1 0 0;
    0 0 0 0 0 1 1/1000];
% capacitance matrix
C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    0.25 -0.25 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];

for l = 1:21
```

```
    % input vector
F = [Vin(l); 0; 0; 0; 0; 0; 0];

% solve for voltage ignoring time dependent
VDC = G\F;

V0(l)=VDC(7);
V3(l)=VDC(3);
end
subplot(2,2,1)
plot (Vin, V0,Vin,V3)
xlabel('Vin (V)')
ylabel('Voltage ')
% frequency sweep
freq =linspace (0,16,10000)*2*pi;
% set input vector to one volt at the source to make future
 calculations
% easy
F = [1; 0; 0; 0; 0; 0; 0];
for l = 1:size(freq, 2)
    % solve for the V vector
   V =  (G+j*freq(l)*C)\F;

   V02(l)=20*log10(abs(V(7)));
   V32(l)=20*log10(abs(V(3)));
end
subplot(2,2,2)
plot (freq, V02,freq,V32)
xlim([0 100])
xlabel('frequency (Rad/s)')
ylabel('gain (dB)')
% create a normaly distrobuted array of capacitor valuse
cap = normrnd(0.25,0.05,[1,100000]);
subplot(2,2,3)
hist(cap)
xlabel('capacitance (f)')


for l = 1:size(cap, 2)
   % re crate the c matrix
   C = [0 0 0 0 0 0 0;
    0 0 0 0.2 0 0 0;
    cap(l) -(cap(l)) 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];
% solve for voltage
    V =  (G+j*pi*C)\F;

    Vo (l)=abs(V(7));

end
subplot(2,2,4)
```

```
hist(Vo)
xlabel ('Vo/Vin')


end
```

Published with MATLAB® R2017a