ECSE 343: Numerical Methods in Engineering Final Project

Table of Contents

In this project, three different methods will be implemented to find the transient responses of for all the voltage nodes, namely, V1, V2, and V3 for circuit shown in Figure 1.

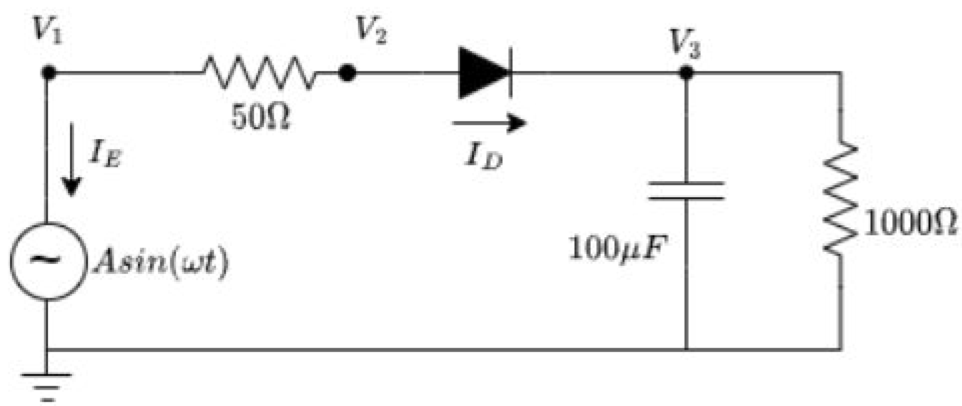


Figure 1: Half-Wave Rectifier.

Using equations 1 and 2, the compact Modified Nodal Analysis form as seen in equation 3 is implemented.

$$I_D = I_S\left(e^{\frac{V_2 - V_3}{V_T}} - 1\right) \tag{1}$$

$$\underbrace{\begin{bmatrix} \frac{1}{50} & -\frac{1}{50} & 0 & 1 \\ -\frac{1}{50} & \frac{1}{50} & 0 & 0 \\ 0 & 0 & \frac{1}{10^3} & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} V_1(t) \\ V_2(t) \\ V_3(t) \\ I_E(t) \end{bmatrix}}_{X(t)} + \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-4} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{C} \underbrace{\begin{bmatrix} \dot{V_1}(t) \\ \dot{V_2}(t) \\ \dot{V_3}(t) \\ \dot{I_E}(t) \end{bmatrix}}_{\dot{X}(t)} + \underbrace{\begin{bmatrix} 0 \\ I_S\left(e^{\frac{V_2(t) - V_3(t)}{V_T}} - 1\right) \\ -I_S\left(e^{\frac{V_2(t) - V_3(t)}{V_T}} - 1\right) \\ 0 \end{bmatrix}}_{D(X(t))} - \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ A\sin(\omega t) \end{bmatrix}}_{B(t)} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2}$$

$$\boldsymbol{F}(\boldsymbol{X}(t)) = \boldsymbol{G}\,\boldsymbol{X}(t) + \boldsymbol{C}\,\dot{\boldsymbol{X}}(t) + \boldsymbol{D}(\boldsymbol{X}(t)) - \boldsymbol{B}(t) = \boldsymbol{0} \qquad (3)$$

Backward Euler Approximation

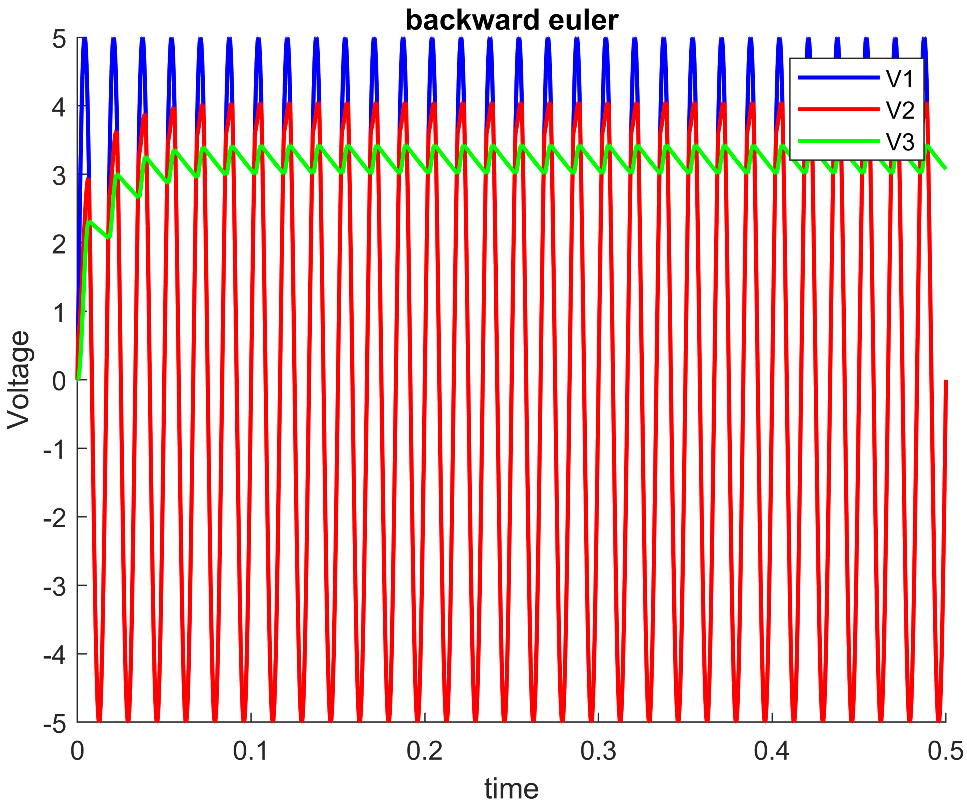
Using the backward euler approximation for the derivatives term $X'_{n+1} \approx \frac{X_{n+1} - X_n}{\Delta t}$ and substituting in equation 3 gives us the difference equation,

$$G * (X_{n+1}) + C * \left(\frac{X_{n+1} - X_n}{\Delta t}\right) + D(X_{n+1}) - B_{n+1} = 0$$

Which will be called $F(X_{n+1}) = G * (X_{n+1}) + C * \left(\frac{X_{n+1} - X_n}{\Delta t}\right) + D(X_{n+1}) - B_{n+1} = 0$

X_{n+1} will be solved for every time step dT, where the Newton-Raphson Method will be used using direct Jacobian calculation.

```
dT = 0.0001; % delta t
T = 0:dT:0.5;

%Initial conditions
V1 = 0;
V2 = 0;
V3 = 0;
Ie = 0;

% Xn
X = zeros(4,length(T));
X(:,1) = [V1; V2; V3; Ie];

for i=2:length(T)
    [Xout] = NR_Jacobian_bckwrd(X(:,i-1),X(:,i-1),1e-4,T(i),dT);
    X(:,i) = Xout;
end

figure()
title('backward euler')
xlabel('time')
ylabel('Voltage')
hold on
% plot solution here.
plot(T,X(1,:),'b-','LineWidth',1.5,'displayname','V1')
legend();
hold on
plot(T,X(2,:),'r-','LineWidth',1.5,'displayname','V2')
hold on
plot(T,X(3,:),'g-','LineWidth',1.5,'displayname','V3')
```



Forward Euler Approximation

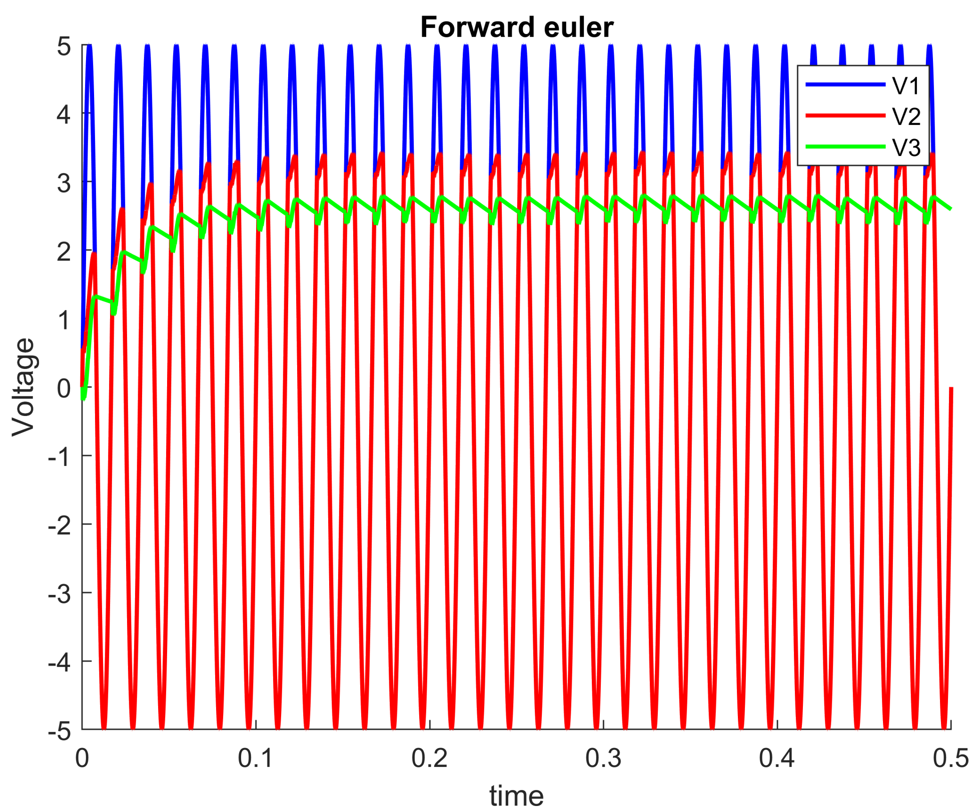
Using the forward euler approximation for the derivatives term $X'_n \approx \frac{X_{n+1} - X_n}{\Delta t}$ and substituting in equation 3 gives us the difference equation,

$$F(X_n, X_{n+1}) = GX_n + \frac{C}{\Delta t}(X_{n+1} - X_n) + D(X_n) - B_n = 0$$

X_{n+1} will be solved for every time step dT, where the Newton-Raphson Method will be used using direct Jacobian calculation.

```
clear all;
dT = 0.0001; % delta t
T = 0:dT:0.5;

%Initial conditions
V1 = 0;
V2 = 0;
V3 = 0;
Ie = 0;

% Xn
X = zeros(4,length(T));
X(:,1) = [V1; V2; V3; Ie];
```

```
%Xguess = X;



for i=2:length(T)
    [Xout] = NR_Jacobian_frwrd(X(:,i-1),X(:,i-1),1e-4,T(i),dT);
    X(:,i) = Xout;
end


figure()
title('Forward euler')
xlabel('time')
ylabel('Voltage')
legend();
hold on
% plot solution here.
plot(T,X(1,:),'b-','LineWidth',1.5,'displayname','V1')
hold on
plot(T,X(2,:),'r-','LineWidth',1.5,'displayname','V2')
hold on
plot(T,X(3,:),'g-','LineWidth',1.5,'displayname','V3')
```



Trapezoidal Rule

We can use the trapezoidal approximation, $X_{n+1} = X_n + \frac{\Delta t}{2}(X'_n + X'_{n+1}) \implies X'_n + X'_{n+1} = 2\left(\frac{(X_{n+1} - X_n)}{\Delta t}\right)$,

and substitute into the difference equation, which we define by adding $F(X_{n+1})$ and $F(X_n)$ to give

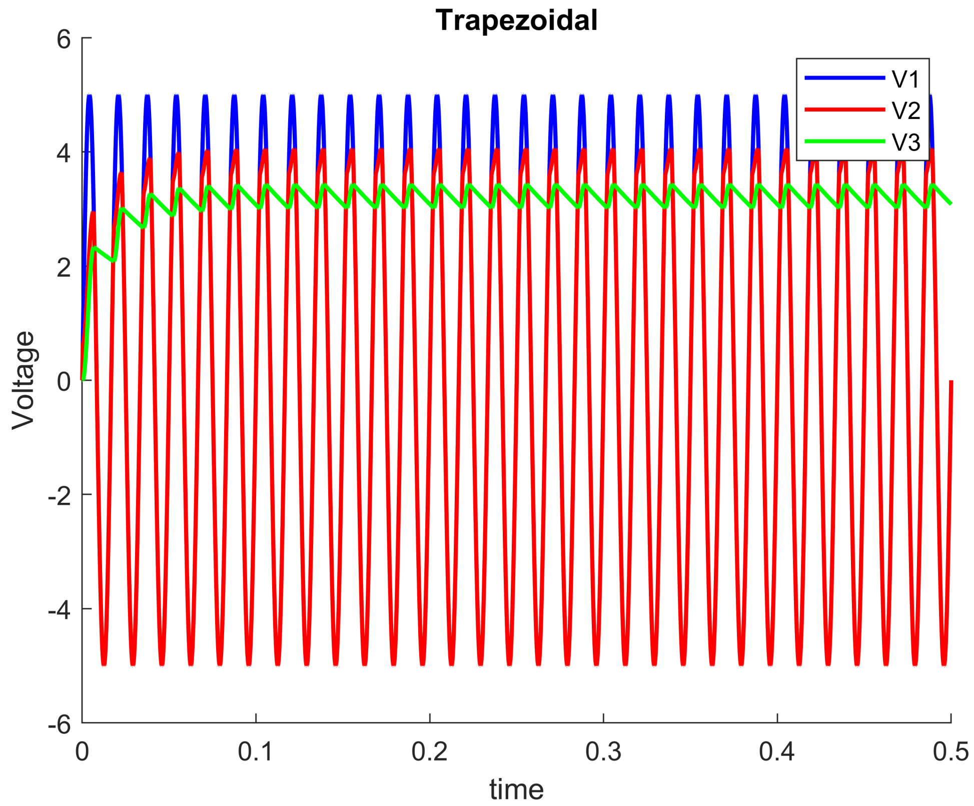
$F(X_{n+1}, X_n) = G * (X_{n+1} + X_n) + C * (X'_n + X'_{n+1}) + D(X_{n+1}) + D(X_n) - B_n - B_{n+1}$

After substituting we define the new function as,

$$F(X_{n+1}) = G * (X_{n+1} + X_n) + C * \left(2\left(\frac{(X_{n+1} - X_n)}{\Delta t}\right)\right) + D(X_{n+1}) + D(X_n) - B_n - B_{n+1}$$

X_{n+1} will be solved for every time step dT, where the Newton-Raphson Method will be used using direct Jacobian calculation.

```
dT = 0.0001; % delta t
T = 0:dT:0.5;

%Initial conditions
V1 = 0;
V2 = 0;
V3 = 0;
Ie = 0;

% Xn
X = zeros(4,length(T));
X(:,1) = [V1; V2; V3; Ie];
%Xguess = X;

for i=2:length(T)
    [Xout] = NR_Jacobian_trapezoidal(X(:,i-1),X(:,i-1),1e-4,T(i),dT);
    X(:,i) = Xout;
end


figure()
title('Trapezoidal')
xlabel('time')
ylabel('Voltage')
legend();
hold on

% plot solution here.
plot(T,X(1,:),'b-','LineWidth',1.5,'displayname','V1')
hold on
plot(T,X(2,:),'r-','LineWidth',1.5,'displayname','V2')
hold on
plot(T,X(3,:),'g-','LineWidth',1.5,'displayname','V3')
```



Appendix

```
function [Xout] = NR_Jacobian_bckwrd(x0, Xguess,tol,t,dt)

Is = 1e-13;
A = 5;
Vt = 0.025;
w = 2*pi*60; % f = 60Hz



% G Matrix
G = [1/50, -1/50, 0, 1;
    -1/50, 1/50, 0, 0;
    0, 0, 1/(10^3), 0;
    1, 0, 0, 0];

% C Matrix
C = [0, 0, 0, 0;
    0, 0, 0, 0;
    0, 0, 10^-4, 0;
    0, 0, 0, 0];

Bn = [0; 0; 0; A*sin(w*(t))];
%
```

```
Xout = Xguess;
counter = 1;
while 1
    % Calculate diode current
    Id = Is*(exp((Xout(2)-Xout(3))/Vt)-1);
    % Calculate D(x)
    Dx = [0; Id; -Id; 0];
    % Calculate Jacobian
    J = G+(C/dt)+[0, 0, 0, 0; 0, Is*(exp((Xout(2)-Xout(3))/Vt))/Vt, -Is*(exp((Xout(2)-Xout(3))/
    % Delta X = -(F(x)/J) where F(x) = G*(Xout)+(1/dt)*C*(Xout-x0)+Dx-Bn
    delX = -J\(G*(Xout)+(1/dt)*C*(Xout-x0)+Dx-Bn);
    % Update Xout
    Xout = Xout + delX;

    if norm(delX)<tol
        break
    end
    if counter>1000
        break
    end
    counter = counter + 1;
end

end
```

```
function [Xout] = NR_Jacobian_frwrd(x0, Xguess,tol,t,dt)

Is = 1e-13;
A = 5;
Vt = 0.025;
w = 2*pi*60; % f = 60Hz



% G Matrix
G = [1/50, -1/50, 0, 1;
    -1/50, 1/50, 0, 0;
    0, 0, 1/(10^3), 0;
    1, 0, 0, 0];

% C Matrix
C = [0, 0, 0, 0;
    0, 0, 0, 0;
    0, 0, 10^-4, 0;
    0, 0, 0, 0];

Bn = [0; 0; 0; A*sin(w*(t))];
%
```

```
Xout = Xguess;
counter = 1;
while 1
    % Calculate diode current
    %Id = Is*(exp((x0(2)-x0(3))/Vt)-1);
    Id = Is*(exp((Xout(2)-Xout(3))/Vt)-1);
    % Calculate D(x)
    Dx = [0; Id; -Id; 0];
    % Calculate Jacobian
    J = G+(C/dt)+[0, 0, 0, 0; 0, Is*(exp((Xout(2)-Xout(3))/Vt))/Vt, -Is*(exp((Xout(2)-Xout(3))/
    % Delta X = -(F(x)/J)
    delX = -J\(G*x0+(1/dt)*C*(Xout-x0)+Dx-Bn);
    % Upadate Xout
    %x0 = x0 + delX;
    Xout = x0 + delX;
    %Xout = Xout+ delX;



    if norm(delX)<tol
        break
    end
    if counter>1000
        break
    end
    counter = counter + 1;
end

end
```

```
function [Xout] = NR_Jacobian_trapezoidal(x0, Xguess,tol,t,dt)

Is = 1e-13;
A = 5;
Vt = 0.025;
w = 2*pi*60; % f = 60Hz



% G Matrix
G = [1/50, -1/50, 0, 1;
    -1/50, 1/50, 0, 0;
    0, 0, 1/(10^3), 0;
    1, 0, 0, 0];

% C Matrix
C = [0, 0, 0, 0;
    0, 0, 0, 0;
```

```
    0, 0, 10^-4, 0;
    0, 0, 0, 0];

% Calculate Bn
B = [0; 0; 0; A*sin(w*(t-dt))];
% Calculate Bn+1
Bn = [0; 0; 0; A*sin(w*(t))];

Xout = Xguess;
counter = 1;

Id_1 = Is*(exp((x0(2)-x0(3))/Vt)-1);
% calculate D(Xn)
Dx = [0; Id_1; -1*Id_1; 0];

while 1
    Id_2 = Is*(exp((Xout(2)-Xout(3))/Vt)-1);
    % Calculate D(Xn+1)
    Dxn = [0; Id_2; -1*Id_2; 0];
    J = G+((2*C)/dt)+[0, 0, 0, 0; 0, Is*(exp((Xout(2)-Xout(3))/Vt))*(1/Vt), -Is*(exp((Xout(2)-X
    % Delta X = -(F(x)/J) where F(x) = G*(Xout+x0)+(2/dt)*C*(Xout-x0)+Dx+Dxn-B-Bn
    delX = -J\(G*(Xout+x0)+(2/dt)*C*(Xout-x0)+Dx+Dxn-B-Bn); %-invJ*F(x+1,x)
    Xout = Xout + delX;

    if norm((G*(Xout+x0)+(2/dt)*C*(Xout-x0)+Dx+Dxn-B-Bn))<tol && norm(delX)<tol
        break
    end

    if norm(delX)<tol
        break
    end
    if counter>1000
        break
    end
    counter = counter + 1;
end

end
```