Assignment 4

Preliminaries

1. You can reuse and build upon your previous code for this assignment.
2. You will need the functions *makeFvect.m* and *make_nlJacobian.m* you created in Assignment 2.
3. Like the previous assignment, the function *makeBt.m* provided. This function adds the time domain sinusoidal sources to the ***B*** vector for a given time.

4. In your submission, please provide all code in a zip file in a way that allows us to run the testbenches ourselves (include all code, not just the recent one).

5. Also submit a pdf file containing the answers to the questions, the output plots and the code for functions you have written for this assignment.

Question I (5marks)

Implement the function named nl_BE_method.m for computing the transient response of nonlinear circuits using the Backward Euler method. The stub for this function is provided below. Please feel free to modify the function in accordance with your preferences.

```
function [tpoints, X]= nl_BE_method(tEnd,h, outNode)
% This function uses BACKWARD EULER method to compute the transient reponse
% of a NONLINEAR circuit.
%Inputs: 1. tEnd:  The simulation starts at time = 0s  and ends at time =
%                  tEND s.
%        2. h: length  of step size.
%        3. outNode: is the node for which the transient is required.
%Output:  1. tpoints: list of time points.
%         2. X:  is the transient response at output node.
%
%Note: The function stub provided above is just an example. You can modify the
%      in function in any fashion.
%---------------------------------------------------------------
```

For this function you will need the functions *makeFvect.m* and *make_nlJacobian.m* you created in Assignment 2.

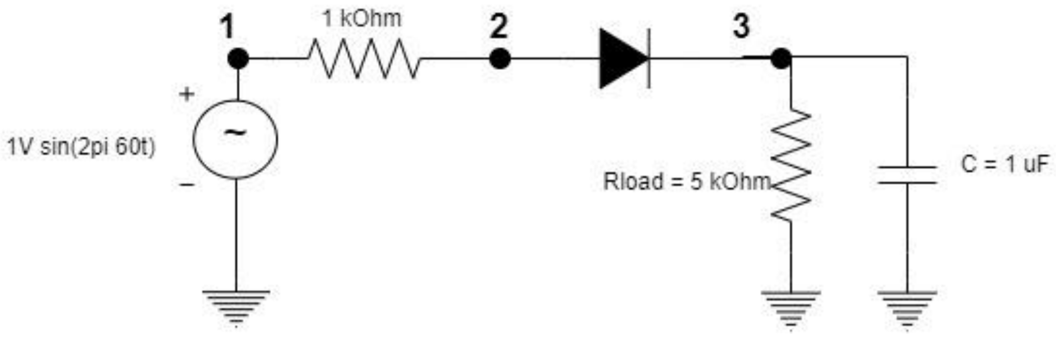

Figure 1: Half-wave rectifier.

Deliverables: After implementing the function, test your method using the circuit shown in Figure 1. Figure 1 shows a half-wave rectifier connected to a 60Hz sinusoidal source with 1Volt amplitude. The netlist of this file is provided in the script *Half_wave_rectifier_TR.m*. Run the script *TestBench_q1.m* to test your code. Make sure you make necessary changes in the testbench script to accommodate the changes (if any) you made in the BE function.

Please provide the plots of the voltages at all the three nodes of the above circuit.

Question II (15 Marks)

Implement the functions described in above to solve the Harmonic Balance equation described as

$$\bar{\boldsymbol{G}}\bar{\boldsymbol{X}} + s\,\bar{\boldsymbol{C}}\bar{\boldsymbol{X}} + \bar{\boldsymbol{F}}(\bar{\boldsymbol{X}}) = \bar{\boldsymbol{B}}(s)$$

where $s = 2\pi * frequency$.

To handle the Harmonic Balance elements, we are providing you with following functions:

a) *makeHB_Gmat.m* this function creates the matrix $\bar{\boldsymbol{G}}$ using the matrix $\boldsymbol{G}$. This function is already implemented for you. The stub for this function is provided below,

```
function Gbar = makeHB_Gmat(H)
% This function uses the MNA matrix, G, to compute the HB matrix Gbar.
% First, this function builds the matrix G and then it builds Gbar.
% So please make sure your method makeGmatrix.m works well or use the
% makeGmatrix method provided to you along with this assignment.
% input: H is the number of harmonics.
%        Number of Fourier coefficients can be computed as, Nh = 2*H+1;
% output: Gbar is the Harmonic balance matrix of size (n*Nh x n*Nh).
```

b) *makeHB_Cmat.m* this function creates the matrix $\bar{\boldsymbol{C}}$ using the matrix $\boldsymbol{C}$. This function is already implemented for you.

```
function Cbar = makeHB_Cmat(H)
% This function uses MNA matrix, C, to compute the HB matrix Cbar.
% First, this function builds the matrix C and then it builds Cbar.
% Therefore  please make sure your method makeCmatrix.m works well or use the
% makeCmatrix method provided to you along with this assignment.
% Inputs: H is the number of harmonics.
%         Number of Fourier coefficients can be computed as, Nh = 2*H+1;
% Output: Cbar is the Harmonic balance matrix of size (n*Nh x n*Nh).
```

c) The addition of Harmonic Balance voltage sources is already done for you in the functions, *makeGmatrix.m* and *makeBvect.m*.

You need to implement the following functions using the provided framework. The stubs for the functions below are provided to you,

a) *make_Gamma.m* this function creates the Direct Fourier Transform (DFT) matrix. It takes the number of harmonics, H, as input and provides the DFT matrix as a output.

b) *HB_fvect.m* is the function that adds the Harmonic Balance based nonlinear stamp to the vector $\bar{\boldsymbol{F}}(\bar{\boldsymbol{X}})$**.** This function takes the vector of Fourier coefficients of nodal voltages, $\bar{\boldsymbol{X}}$**,** as input and then returns the $\bar{\boldsymbol{F}}(\bar{\boldsymbol{X}})$**.**

~~c)~~ *HB_nljacobian.m* is the function that creates/updates the Harmonic Balance Jacobian, $\frac{\partial \bar{\boldsymbol{F}}(\bar{\boldsymbol{X}})}{\partial \bar{\boldsymbol{X}}}$, for the nonlinear elements. This function also takes $\bar{\boldsymbol{X}}$ (i.e., the vector of Fourier coefficients of nodal voltages) as input~~.~~

d) *HBsolve.m* this function solves the Harmonic Balance system of equations using Newton-Raphson Method. This method requires initial guess, and the number of Harmonics as input and returns a vector of nodal Fourier coefficients as output.

Deliverables: You will test your Harmonic Balance method using the circuit provided in Figure 1. The script *Half_wave_rectifier_HB.m* contains the Harmonic Balance netlist for the circuit in Figure 1. After implementing the required functions, run the file *TestBench_q2.m* to simulate the circuit.

1. Simulate the above circuit and then plot the time-domain steady state response for the circuit at nodes 1, 2, and 3.
2. To verify your simulation, compare it with the nonlinear transient response you implemented above.
3. Include the code above in your PDF file submission for the assignment.