Assignment 4

Emad Gad,

ELG7132D

Topics in Electronics I

Simulation of Radio Frequency Circuits

November 2, 2017

The goal of this assignment is to train the student on the implementation steps involved in the Harmonic Balance approach. The target circuit of this assignment is the common-emitter amplifier circuit used in assignment 2. For the purpose of this assignment, this circuit shall be stimulated with a sinusoidal signal having 1mv amplitude at the input voltage, v_{in}, and a frequency of 5MHz, i.e.,

$$v_{\text{in}}(t) = 1 \times 10^{-3} \cos(2 \times \pi \times 5 \times 10^{6} \times t) \tag{1}$$

The objective of the assignment is to compute the steady-state response at the output node, i.e. $v_{\text{out}}(t)$, via computing the Fourier coefficients, through the Harmonic Balance technique. Harmonic

In order to gain confidence in implementation and the general understanding of the entire process and algorithm, the student will be asked to compare the results of her/his implementation with the results obtained from a commercial simulator, the HSPICE RF. The circuit netlist that will be given to HSPICE RF is identical to the circuit netlist given to HiSPICE, but with few modifications intended for the commercial simulators. The two circuit netlists will be released on the Blackboard Learn on Virtual Campus.

Part (a)

Computing the DC solution vector

The circuit shown in Fig. 1 is excited using the sinusoidal source expressed by Equation (1).

The goal of this part is to find the DC operating point, and use this point to construct the initial guess for the Harmonic Balance technique $\bar{\boldsymbol{X}}^{(i)}$.

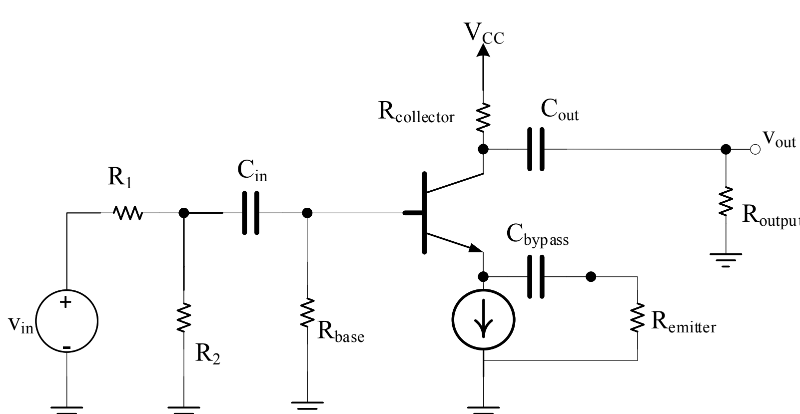

Figure 1: The circuit shcematic

As mentioned in the lecture and the notes, computing the DC operating point in the circuit $\boldsymbol{x}_{\mathrm{DC}}$ is done by solving the circuit DC equations,

$$\boldsymbol{G}\boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{b}_{\mathrm{DC}} \tag{2}$$

Your task in this part is to write a program in Matlab to compute the DC operating point, i.e. the vector $\boldsymbol{x}_{\mathrm{DC}}$. To do this, you will have to do the following procedure

(a) Start with an initial guess for the DC operating vector, call it $\boldsymbol{x}^{(i)}$, where $i = 0$.

(b) Compute the error due to the guess $\boldsymbol{x}^{(i)}$ call it $\boldsymbol{\phi}(\boldsymbol{x}^{(i)})$, where

$$\boldsymbol{\phi}(\boldsymbol{x}^{(i)}) = \boldsymbol{G}\boldsymbol{x}^{(i)} + \boldsymbol{f}\left(\boldsymbol{x}^{(i)}\right) - \boldsymbol{b}_{\mathrm{DC}} \tag{3}$$

(c) If the norm of this error vector is smaller than a small threshold ϵ, e.g. $\epsilon = 10^{-14}$, then stop: the solution has been reached. Assign $\boldsymbol{x}^{i}$ to $\boldsymbol{x}_{DC}$, then STOP.

(d) Compute the Jacobian matrix at $\boldsymbol{x} = \boldsymbol{x}^{(i)}$, and call it $\boldsymbol{J}(\boldsymbol{x}^{(i)})$, where

$$\boldsymbol{J}(\boldsymbol{x}^{(i)}) = \boldsymbol{G} + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \tag{4}$$

and $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ is the matrix of partial derivatives.

(e) Update the initial guess using the following update

$$\boldsymbol{x}^{(i+1)} = \boldsymbol{x}^{(i)} - \boldsymbol{J}^{-1}\boldsymbol{\phi}(\boldsymbol{x}^{(i)}) \tag{5}$$

(f) Go to step (b) replacing i by $i + 1$.

The following details should help you accomplish your task with the help of the HiSPICE-Matlab interface.

(a) **Obtaining your initial Guess $\boldsymbol{x}^{(0)}$.**

In order to ensure that the above Newton method converges, you will need to choose the initial condition given on the netlist file, which is specified using the `.ic` statement. The initial condition is accessed when the circuit netlist file is first parsed using the function `mna_parse_circuit`, e.g.

`my_circuit_struct = mna_parse_circuit(circuit_file_name).`

You can find the vector of initial conditions specified by the `.ic` statement on the circuit netlist file from the field called `xic` in the structure `my_circuit_struct` returned by the above function. You can access this vector using something like the following

```
X_0_init_guess = my_circuit_struct.xic;
```

(b) **Computing the DC source vector $\boldsymbol{b}_{\mathrm{DC}}$**

This vector can be computed using the Matlab-HiSPICE interface by calling the function $mna\_compute\_source\_vector$ at time $t = 0$, i.e.

```
bDC = mna_compute_source_vector(my_circuit_struct,0)
```

(c) **Computing the nonlinear function vector $\boldsymbol{f}(\boldsymbol{x}(t))$ and the Jacobian matrix of the nonlinear part $\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}$.**

You can compute both of $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{J}(\boldsymbol{x})$ using the Matlab-HiSPICE interface function called`mna_compute_circuit` as shown next

```
[fx b Jf] = mna_compute_circuit(my_circuit_struct,t,x)
```

The input arguments that need to be passed to this function are as follows

- `my_circuit_struct` is the structure returned by the call to `mna_parse_circuit`
- `t` is the time instant t at which those quantities need to be computed, and
- `x` is the vector of MNA variables at this time instant, i.e. $\boldsymbol{x}(t)$

The output arguments returned by this function are as follows,

- `fx` is the vector of nonlinear functions $\boldsymbol{f}(\boldsymbol{x}(t))$ computed at $\boldsymbol{x}(t) =$`x` (the input argument),
- `b` is the value of source vector computed at time $t =$`t` (the input argument)
- `Jf` is the Jacobian matrix in sparse format when $\boldsymbol{x} =$`x` (the input argument). This is the matrix of partial derivatives of the nonlinear part, i.e.,

$$\mathtt{Jf} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_N} \\ \vdots & & & \vdots \\ \frac{\partial f_N}{\partial x_1} & \frac{\partial f_N}{\partial x_2} & \cdots & \frac{\partial f_N}{\partial x_N} \end{bmatrix} \tag{6}$$

Part (b)

Computing the Steady-State Response using the HB

In this part, you are required to develop a Matlab program to compute the Steady-State response of the circuit shown in Figure 1 due to the input voltage shown in Equation (1).

Your program must adhere to the following guidelines.

(a) The first line in the program should have the following format,

```
function [freq Xo Xc Xs] = compute_hb_response(n_time_points,node_name)
```

The input arguments to the program should be

- `n_time_points`: is the number of time samples used in computing the steady-state response.
- `node_name`: is a string between single quotes that describes the name of the node, whose voltage steady-state response is to be computed. For example, `node_name='nvin'`, where `'nvin'` is the label given on the netlist to the input node on the circuit.

The output arguments of the program are described as follows,

- `freq`: is a vector of the frequency values used in the HB analysis problem. For this circuit, this vector will be

$$\begin{bmatrix} 0 \\ 5 \times 10^6 \\ 10 \times 10^6 \\ 15 \times 10^6 \\ \cdots \end{bmatrix}$$

- `Xo` is a scalar value for the DC coefficient of the response.
- `Xc` is a vector for the $\cos$ Fourier coefficients of the steady-state response,
- `Xs` is a vector for the $\sin$ Fourier coefficients of the steady-state response,

(b) Your program must adjust automatically to the case of even or odd time samples, i.e. the integer value passed in the input argument `n_time_samples`. For example, if `n_time_samples` = 7, then both `Xc` and `Xs` should be vectors of length 3 entries. However, if `n_time_samples` = 8, then `Xc` must be a vector of length 4, while `Xs` should be a vector of length 3, to balance the even number of time samples.

(c) Your program must be contained in only one `.m` file, called `compute_hb_response.m`. Notice that Matlab allows you to define internal functions, which are functions that are called from the main function `compute_hb_response`, within the same file `compute_hb_response.m`. An example for this arrangement may look like the Program 1

Program 1 An example of how to include internal functions within the same `.m` file.

```
function [freq Xo Xc Xs]=compute_hb_response(n_time_points,node_name)
%******************************
% ++++++++++++++ ELG7132D: Assignment +++++++++++++
% ++++++++++++++ Student Name:  John Doe
% ++++++++++++++ Student #:
% ++++++++++++++ Main function goes first +++++++++++++
⋮
% Call compute_function_1
⋮
% Call compute_function_2
⋮
% End of compute_hb_response
% *************************************************
% *************************************************
function [x,y,z] = compute_function_1(l,q)
⋮
% Do some computations here
⋮
% End of compute_function_1
% *************************************************
% *************************************************
function [x,y,z] = compute_function_2(r,s)
⋮
% Do some computations here
⋮
% End of compute_function_2
% *************************************************
```

(d) The first few lines in your program must provide a comment to show your ID. An example would look like the following

```
%********************************
% ELG7132D: Assignment 3
% Student Name:  Emad Gad
% Student ID #:  12345678
```

Part (c)

Comparing the results with HSPICE

In this part you will run the commercial HSPICE RF simulator on the same circuit of Figure 1 and compare your results with the results returned by HSPICE RF. Further instructions on how to do that will be released on the Blackboard Learn on Virtual Campus.

Part (d)

The Submitted Report

The goal of your report is to show a comparison between the results returned from your program and the HSPICE program. The file of your report should be a PDF file, whose name must be derived from your student id number and your last name, e.g. `12345678_Gad.pdf`.

Your report should include the following four items.

(a) Your report must include a table with a format similar to the format of Table 1, which shows the *magnitude* of each frequency (including DC) in the spectrum of the response from both programs.

Table 1: An example illustrating the format of the table included in your report.

frequency value	Results from My Program	Results from HSPICE
0	X_0	X_0
5×10^6	$\sqrt{X_1^{C2} + X_1^{S2}}$	$\sqrt{X_1^{C2} + X_1^{S2}}$
10×10^6	$\sqrt{X_2^{C2} + X_2^{S2}}$	$\sqrt{X_2^{C2} + X_2^{S2}}$
$\vdots$	$\vdots$	$\vdots$

(b) In this item you will be required to show a comparison for the behaviour of voltage gain (at 5MHz) versus the amplitude of the input voltage v_{in} as computed through your program and the HSPICE commercial simulator.

In other words, assume that your input voltage is given by

$$v_{\text{out}} = A\cos\left(2 \times \pi \times 5 \times 10^6 \times t\right) \tag{7}$$

then your task is to repeat the simulation of the Harmonic-Balance and HSPICE programs using different values for A, given by

$$1\mu\text{ V}, 10\mu\text{ V}, 100\mu\text{ V}, 1\text{ mV}, 10\text{ mV}, 100\text{ mV}, 1\text{ V}, 10\text{ V}.$$

Your results should be tabulated as shown in the table below.

Table 2: An example illustrating the format of the table included in your report.

Amplitude A	Results from My Program	Results from HSPICE
1μV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
10μV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
100μV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
1 mV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
10 mV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
100 mV	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
1 V	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
10 V	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$
15 V	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$	$\frac{\sqrt{X_1^{C^2}+X_1^{S^2}}}{A}$

Hints:

- Notice here that you may need to increase the number of sampling points to enable convergence, especially at high values of the input amplitude A.

- Notice also that the HB will not converge for values of A higher than 1V. You will have to use the notion of source stepping explianed in the lectures and the notes.

(c) Using the Matlab program you developed above compute the total harmonic distortion, THD, defined by

$$THD = \log_{10} \frac{\sum_{i=2}^{M} \sqrt{X_i^{C^2} + X_i^{C^2}}}{\sqrt{X_1^{C^2} + X_1^{C^2}}}, \tag{8}$$

where $X_i^{(C,S)}$ are the Fourier coefficients of the output voltage v_{out}, for all for the different values of the amplitude A listed in the above table. Repeat the same calculations, but using the results obtained from HSPICE. Plot, using log-log scale, THD versus A from both programs. Include the resulting graph in your report.

(d) In your report, answer the following question.

- Does the total Harmonic distortion increase or decrease with the increase in the amplitude of the input voltage A?
- Explain your answer.

Part (e)

IMPORTANT INSTRUCTIONS. What you need to submit

You are required to submit only two files (DO NOT compress them in one zipped file):

(a) A PDF file for your report named after your student ID # and your last name, e.g `1234567_Gad`, and including all four items listed above.

(b) A `compute_hb_response.m` file which includes all of your code. This file must be self-contained, in the sense that it should not need other files or programs to run, of course, with the exception of the files already supplied with the Matlab-HiSPICE interface (e.g. `mna_parse_circuit.m`): you do not need to include any of these files with your submission. The person marking the assignment must be able to run all of the program from the Matlab command prompt by simply typing

```
[freq Xo Xc Xs] = compute_hb_response(n_time_points,node_name)
```

If running this program failed, you will not get the mark related to this part.

(c) DO NOT compress your files or submit a .zip or .rar archive file.