Assignment 2

Emad Gad,

ELG7132D

Topics in Electronics I: Simulation of Radio Frequency Circuits

October 3, 2017

Part (a)

Description of the Assignment

The objective in this assignment is to give the students a better and deeper understanding of the process of computing the steady-state response of a nonlinear circuit using transient time-domain simulation. You may need to read the submission instructions at the end of the assignment before you start working to get an idea of the required deliverables and plan your work accordingly.

Part (b)

Introduction

Consider the circuit shown in Figure 1. It is known that this circuit functions as a common-emitter voltage amplifier, with a certain voltage gain that can be computed using small-signal analysis. This means, for example, that if the input to the circuit is sinusoidal of frequency 5MHz, with 10 mv amplitiude, i.e.

$$v_{\text{in}}(t) = 0.01\cos(2 \times \pi \times 5 \times 10^6 \times t) \tag{1}$$

then the voltage at the output node, after the circuit has reached its steady-state response and all transient has died out, is expected to look close to a sinusoidal (except for a slight distortion) and to have an amplitude value of approximately $0.01 \times G$, where G is the voltage gain calculated using small-signa analysis.

Your tasks in this assignment

(a) Compute the voltage gain of the circuit using small-signal analysis. You may consider that $\beta = 100$ for this transistor

(b) Write a Matlab programs to simulate the steady-state response of the circuit via transient time-domain simulation and using the Trapezoidal Rule method.

You can validate the correctness of your program by seeing that, at steady-state, the amplitude of the output node voltage (v_{out}) obtained from your program matches the result of the voltage gain (as computed in the first task above) when multiplied by the amplitude of the input voltage anplitude of 0.01 Volts.

In order to achieve your task, it is recommended that you follow the suggested steps below. Detailed submission instructions will be provided at the end of the assignment.

Part (c)

Computing the initial point $\boldsymbol{x}(t = 0)$ via finding the DC operating point

The goal of this part is to find the DC operating point, and use this point to construct the starting point for the TR method, i.e., $\boldsymbol{x}(t = 0)$.

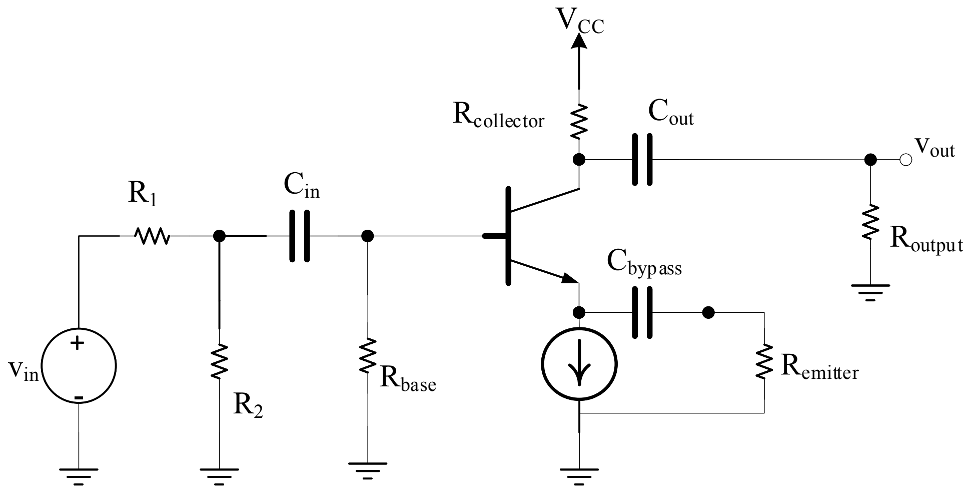

Figure 1: The circuit schematic

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

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

Your task in this part is to write a program in Matlab to compute the DC operating point, i.e. the vector $\boldsymbol{x}_{\text{DC}}$. 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

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

(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 $\boldsymbol{b}_{\text{DC}}$**

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.,

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

Part (d)

Implementation of the Trapezoidal Rule (TR)

After having computed the DC operating point in the previous part, you are now ready to proceed with computing the transient response of the circuit response $\boldsymbol{x}(t)$ at discrete time points. It is recommended that you set the discrete time points at equi-distant time steps as follows,

$$t_i = 0, h, 2h, \cdots ih, \cdots \tag{7}$$

where h is the size of the time step and is given by

$$h = \frac{T}{50} \tag{8}$$

with T being the period of the source, which for this circuit is given by

$$T = \frac{1}{5 \times 10^6} \text{ Seconds} \tag{9}$$

The TR requires computing the source vector ($\boldsymbol{b}(t)$) at specific time points. For this situation, the TR requires computing $\boldsymbol{b}(0), \boldsymbol{b}(h), \boldsymbol{b}(2h), \boldsymbol{b}(3h), \cdots$. This can be accomplished in the HiSPICE MATLAB interface program via calling the function.
`mna_compute_source_vector(my_circuit_struct,t)`

Part (e)

Assignment Submission Instructions

In submitting your assignment you will need to upload a number of MATLAB files in addition to a PDF file. All the MATLAB files should clearly state your student ID. This is typically done by giving a number of comments at the beginning of the file, and following the function definition line. For example, those lines should read something like the following.

```
function output = evaluate_something_somehow(input)
% ********************* ***************
% ELG 7132D. Assignment 2
% Student Name:  Emad Gad
% Student ID Number:  987654321
% Date:  October, 9, 2016
% Additional comments go here
% ********************* **************
```

IMPORTANT The Blackboard allows you to upload more than one file during submission. Upload the files one at a time. **Do not compress or zip** the files, or else your assignment will not be marked.

The list of required files for this submission are given next.

(a) A MATLAB file (`.m`) that implements the process of computing the DC operating point. This file should be given the following name

`compute_DC_point.m`

This file needs to be implemented as a function file, which means that first line in the file should be as follows

```
function xDC = compute_DC_point (x_initial_guess)
```

As shown above, the input argument to this function should be a vector that is used as an initial guess to start the Newton iteration, i.e. $\boldsymbol{x}^{(0)}$.

The output argument is the solution of the DC equations, or the DC operating point, which is obtained at the convergence of the Newton interactions.

(b) A MATLAB (`.m`) file that implements the TR method. This file MUST be named as follows.

`compute_TR_time_response.m`

That file should provide your own implementation of the TR method. It must be written in the form of a function (not a script). In other words, its first line should be as follows

```
function x_transient = compute_TR_time_response (x_0, ...
time_points,...
output_node_name)
```

The input arguments to this file are specified as follows.

(i) `x_0` This input specifies a vector of values that are given to the initial time point, i.e. $\boldsymbol{x}(t = 0) \in \mathbb{R}^N$ or $\boldsymbol{x}_0$. Typically this vector is the DC operating point obtained from a call to the previous function `compute_DC_point`.

(ii) `time_points` This input specifies the set of discrete time points at which the time-domain response of the circuit is to be computed. As has been recommended above, this will be vector which will contain multiples of the step size size, i.e., $h, 2h, 3h, \cdots$. However, your implementation should be general enough to treat any set of of discrete time points.

(iii) `output_node_name` This input defines a character string that identifies the node name whose voltage waveform is to be computed. Most likely, this will be the node voltage (v_{out}) at the output of amplifier of the amplifier (between the collector of the transistor and resistance R_3).

The output argument of this function is given as follows

(i) `x_transient` This is a vector of the values of the transient response for the node whose name has been given in the input argument `output_node_name`. The size of this vector must be the same size of the input vector `time_points`, and its entries should give the values of the node voltage at the specified time points, that is, $x_m(t = h), x_m(t = 2h), \cdots, x_m(t = ih), \cdots$, where m corresponds to the integer index assigned to the node `output_node_name`.

(c) A PDF report file summarizing your results. This file should be named using your last name and student ID, e.g., `Gad_987654321.pdf`. This report is intended to allow you to describe your experience, and reflect upon the insights that you gained in carrying out the assignment. A rich and detailed report will be assigned higher mark than reports with little details and poor discussions. As a basic framework that may help guide you in writing the report, your report should *at least* include, but does not need to be limited to, the following items.

(i) The small-signal analysis used to calculate the small-signal voltage gain of the circuit.

(ii) A graph showing the waveform of the input voltage and the output voltage at steady-state. The waveform should be plotted for only one period of the input source, at steady-state, i.e., for a period of time t between $nT \leq t \leq (n + 1)T$, where $T = 0.2$ ms and n is some large integer. The graph must show the relation between the input and output waveform, where a relation between the input and output matching the expected result (i.e., voltage gain of 5 V/V) indicates the success of your implementation.

(iii) A graph showing the frequency components of the output waveform. Those frequency components are the Fourier coefficients of the output waveform, and can be obtained using the Fast Fourier Transform (FFT) function in MATLAB, or using the IDFT matrix $\boldsymbol{\Gamma}^{-1}$

(iv) Comment on the number of cycles T that the circuit took before reaching the steady-state response, and the number of time steps that had to be taken in marching in time to reach the steady-state.

(v) Discuss the reasons for the difficulties encountered in obtaining the steady-state using transient time-domain simulation.

(vi) Give comments on what hapens if you choose larger or samller step size than has been suggested.

(vii) Create a graph that shows the frequency content of the output voltage at steady-state. To do that, use the fft (Fast Fourier Transform) available in Matlab.

(viii) Comment on the effect of increasing the amplitude of the input voltage on the time-domain and frequency-domain of the output voltage.