Austin Shaw

Project 1 Report

3/5/2022

Question 1:

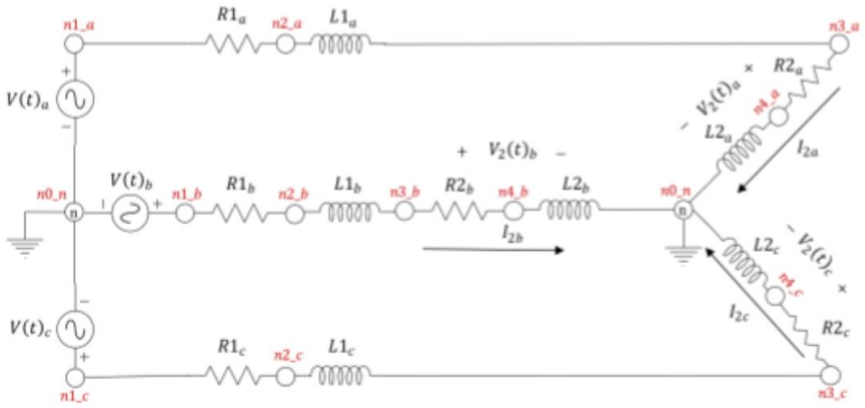

Figure 1: The circuit representation for problem 1.

a. The time domain simulation for 0.2 seconds can be performed by changing the *"Simulation Time"* variable in the *"settings.py"* file. Other settings for the program can be found in this file as well, such as the time step to use (10^{-5} seconds is the default value). Then, using the *"run_solver.py"* program, enter the name of the .json file as *"RL_circuit"* (I.e. without the .json extension). Finally, the plot should be revealed.

b.

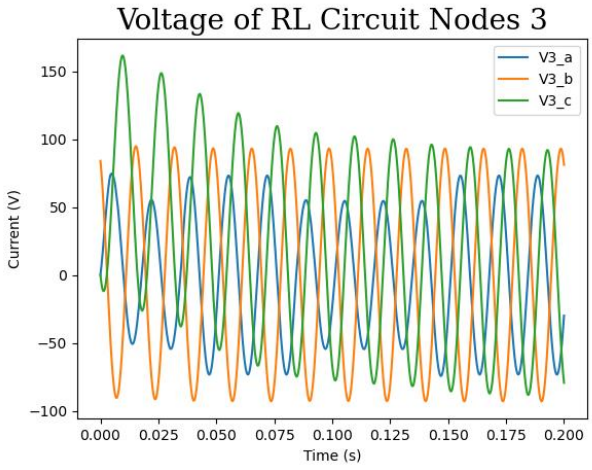

Figure 2: A plot of the voltage simulation across nodes *n3_a* (V3_a), *n3_b* (V3_b), *and n3_c* (V3_c) all respective to ground in Figure 1 and using MNA stamping methods. The time step was 10^{-5} seconds and the elapsed time was 0.2 seconds. Figure 4 shows the companion model for the inductor used in this circuit.

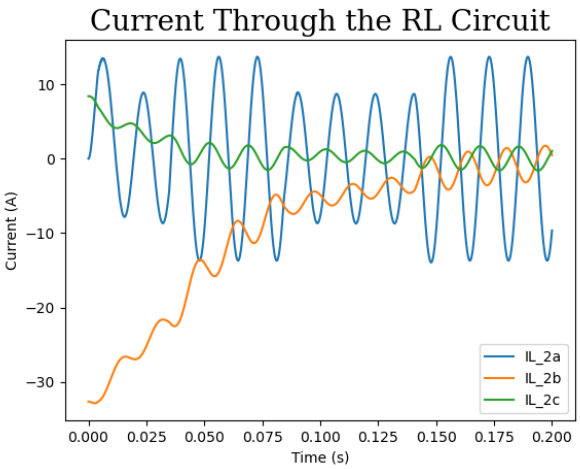

Figure 3: A plot of the current simulation through inductors $L2_a$ (IL_2a), $L2_a$ (IL_2b), and $L2_a$ (IL_2c) in Figure 1 using MNA stamping methods. A time step of 10^{-5} seconds and the total simulation time was 0.2 seconds were used. Note that the simulation utilizes Figure 4 as a companion model for an inductor.

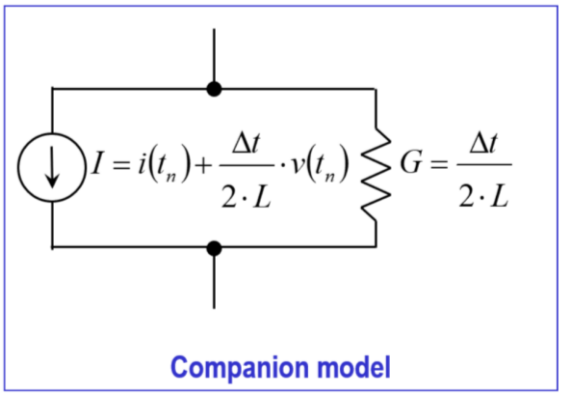

Figure 4: The equivalent companion model used for inductors in the simulator program. This companion model is derived from the trapezoidal method for discretization of an inductor. Figure adapted from "Lecture 6 - Time-domain analyses in Power Grids - Part – I" in Circuit Simulation and Optimization presented February 7th, 2022 by Amritanshu Pandey.

c. To begin, we notice that both Figure 3 and Figure 4, the circuit is in an unstable state due to the approximation methods employed and the poor initial conditioning of the model. The approximation methods we used included the trapezoidal method to discretize and inductor and create the companion model shown in Figure 4. Thus, we begin by approximating initial conditions as DC analysis where we assume that the operating point is the circuit's DC equivalent at time t = 0 seconds. Because of our poor initial conditions, and the need to assume that the conductance of the inductor in Figure 4 is very large initially, the circuit will not start at steady-state. However, at around 0.2 seconds, it appears the RL circuit approaches its steady-state. However, a longer simulation time will need to be run in order to confirm these results.

d. Running the simulator with the dense stamping method for the circuit given in Figure 1 took an average of 9.791 seconds to run over the course of three trials. I believe there is improvement to be made with the dense stamping method that I used, mainly because my coding could be optimized. As for the sparse

stamping methods, I would expect it to take approximately the same amount of time to run the simulation. This is due to two reasons. First, the simulation is not run for a very long time in general, thus any difference would be difficult to see. Next, the matrices used in the program are not very sparse, mainly due to the size of the matrix and the dependency between variables. Whenever a new variable is added, the sparsity tends to increase. Hence, a much larger circuit comprised of the same elements found in Figure 1 would be more appropriate to study the effects of sparse solvers.

e. Note that for the *Simulink* models, we can see that after the RL circuit in my simulation reaches steady-state, both simulations agree. Thus, if my model had better conditioning, such as AC analysis model, our results would probably more quickly. In other words, the simulation of my RL circuit would reach its steady-state sooner.

Question 2

- Note that I used the Simple Induction Motor model because I could not complete the full model within the timeframe.

a. To perform the time-domain simulation of the three-phase induction motor, the user must first open the *"settings.py"* file and confirm the settings they would like to use. Under default conditions, the max number of iterations for Newton Raphson method is 5 iterations, the tolerance of Newton Raphson convergence is 10^{-5} of the function's value, the time step is 10^{-5} seconds, and the simulation time is 0.2 seconds (note that for the induction motor, steady-state is not reached until approximately 1.6 seconds). Then, to initiate the simulation, run the program *"run_solver.py"*. Finally, enter the .json file for the simple induction motor named *"Simple_IM.json"* without the extension.

b. The steady-state of the three-phase induction motor was reached at approximately 1.6 seconds in the simulation time. Once, the induction motor reached steady state, the values of the induction motor variables were recorded. At steady-state, the amplitudes of the stator currents were 14.3 A for both I_{ds} and I_{qs} and the amplitudes of the rotor currents were approximately 1.1 A for both I_{dr} and I_{qr}. Finally, the angular frequency of the rotor finally settled at a magnitude of approximately 377 radians/second and the electrical torque was 3.3 (units).

c.

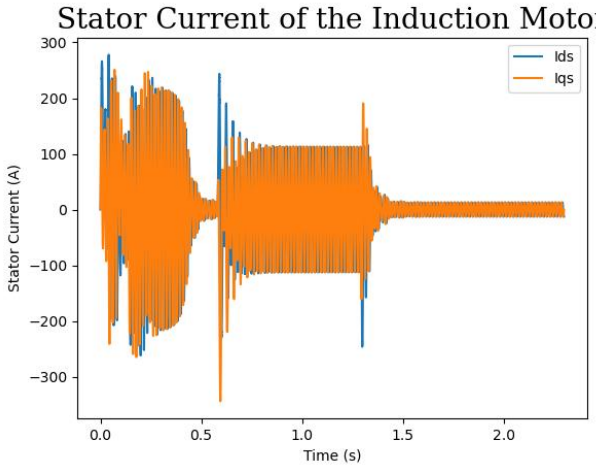

Figure 5: Simulation of the simple induction motor's stator current for a stimulation of 2.3 seconds, a time step of 10^{-5} seconds, max iterations of 5, and a tolerance level of 10^{-5}. Note that the steady is reached approximately around 1.6 seconds and steady state values are reported in part *b.*

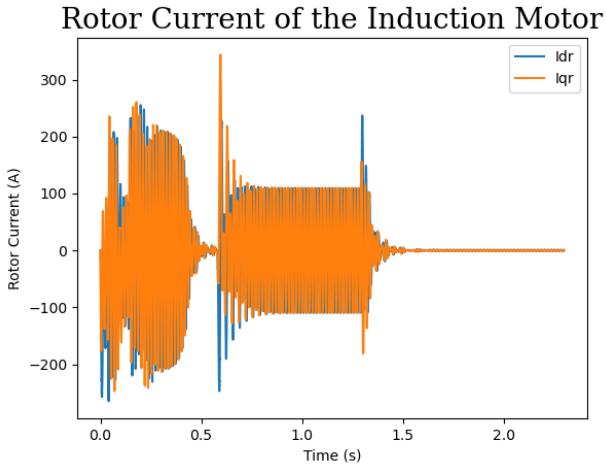

Figure 6: Simulation of the simple induction motor's rotor current for a stimulation of 2.3 seconds, a time step of 10^{-5} seconds, max iterations of 5, and a tolerance level of 10^{-5}. Note that the steady is reached approximately around 1.6 seconds and steady state values are reported in part *b.*

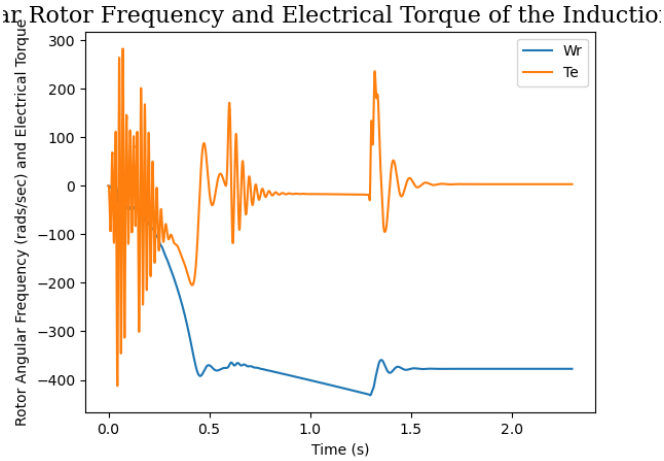

Figure 7: Simulation of the induction motor's electrical torque and angular frequency for a stimulation of 2.3 seconds, a time step of 10^{-5} seconds, max iterations of 5, and a tolerance level of 10^{-5}. Note that the steady is reached approximately around 1.6 seconds and steady state values are reported in part *b.*

d. From Figure 5, Figure 6, and Figure 7, we can see the effect that the induction motor has on the circuit during start-up. In our case, the induction motor draws almost 20 times more current than its steady state equivalent. Thus, it is very important to have these models in order to determine the effects that large loads would produce on a circuit or grid. Not only that, we can see the emphasized change in electrical torque produced by the induction motor on start-up. Now, we notice at a time of approximately 0.5 seconds that the induction motor almost approached its steady-state values. However, the system was still unbalanced and quickly fell out of equilibrium until it eventually restabilized 1.1 seconds later. Therefore, it took approximately 1.6 seconds for the induction motor to reach its steady state in the simulation.

e. I ran my program under two different parameters. First, the program was run with a time step equal to 10^{-5} seconds for a total simulation time of 1.6 seconds. Under these parameters, the program took 113.44 seconds to finish. The other simulation consisted of a time step of 10^{-5} seconds and a simulation time of 2.3 seconds. This simulation took a total time of 163.83 seconds to complete. Now, both of these are rather long run times and I believe the biggest contributor is the time step I chose and the use of Newton Raphson method. Under Newton Raphson method, I required a convergence of 10^{-5} for the physical dependencies of all variables. Thus, the change in the functions that describes the physics of the model had to agree within a value of 10^{-5} in order for Newton Raphson method to succeed. Using sparse matrices, I would expect the run time of the simulation to be very similar again. This is because we used KVL equations in our matrices for $Yv = J$, so Y is a rather dense matrix.

f. From the simulations produced by *Simulink*, we can see that the simulations produced in part *c* agree once the induction motor has reached steady state. Moreover, we can see the same start-up