Circuitos Elétricos II
Trabalho II - 2023.1

Universidade Federal do Rio de Janeiro
Departamento de Engenharia Eletrônica e de Computação
Professora: Fernanda Oliveira

Enunciado do segundo trabalho de simulação de circuitos elétricos. Para o trabalho, os alunos devem desenvolver um programa em Python (versão 3 ou maior). Os pacotes Numpy e Scipy são permitidos. O Matplotlib pode ser usado para testar o trabalho, mas a chamada da função do Matplotlib não deve ser incluída dentro da main. Para utilizar outros pacotes, converse com a professora. O trabalho é individual. Por favor não enviem ou recebam códigos de outros alunos.

Instruções:

1. O trabalho consiste no desenvolvimento de um programa em Python para o cálculo da reposta DC ou da resposta em frequência (AC) de um circuito utilizando a análise nodal modificada.

2. Para esse trabalho, modifique o código do trabalho 1 de forma que a função main calcule e retorne arrays que representam as tensões DC ou a resposta em frequência do circuito. Para a montagem do sistema ser utilizada a análise nodal modificada.

3. A estampa usada para indutores e transformadores deve ser a estampa da análise modificada.

4. O trabalho pode ser escrito em um ou mais arquivos ".py" (veja, abaixo, a padronização dos nomes dos arquivos) e pode usar funções ou classes, de acordo com a preferência do aluno, mas deve, necessariamente, conter uma função principal cujo nome é main. A função main recebe quatro valores por argumento, na seguinte ordem: uma string com o nome do arquivo; uma string igual a "DC" ou "AC"; uma lista de números inteiros que representam os nós desejados; uma lista com parâmetros de simulação. Para a simulação "DC" a lista de parâmetros é uma lista vazia. Para a simulação "AC" a lista de parâmetros contém, na seguinte ordem: um float representa a frequência inicial da simulação (em Hertz); um float representa a frequência final da simulação (em Hertz); um inteiro representa a quantidade de pontos por década. A função deve ser definida da seguinte forma: def main(arqNetlist, tipo, nos, parametros), de forma que, ao fazer a chamada da função dentro de um ".py" os valores devem ser definido na chamada, ex.: main('netlist1.txt', 'DC', [1,2], []) ou main('netlist1.txt', 'AC', [1,2], [0.1, 1e6, 100]). Reforçando, a função será chamada por um arquivo ".py" e não diretamente pelo terminal.

5. Para a simulação "DC", a função main deve retornar um array com as tensões nodais referente aos nós indicados por argumento. Na simulação "DC", capacitores devem ser considerados como circuito aberto e indutores devem ser considerados curto-circuito, o que deve acontecer naturalmente ao definir $\omega = 0\ rad/s$.

6. Para a simulação “AC” a função main retornar três arrays do numpy. O primeiro array retornado possui as frequências, em Hertz, utilizadas na simulação (eixo x do gráfico de resposta em frequência). As frequências devem ser escolhidas em escala log considerando a quantidade de pontos por década definida por argumento. O segundo array retornado possui os módulos (em dB) das tensões nodais desejadas para cada frequência simulada. A quantidade de colunas desse array é igual à quantidade de valores do array de frequências; e a quantidade de linhas é igual ao tamanho da lista do segundo argumento, onde a primeira linha é referente ao nó da posição 0 da lista, a segunda linha é referente ao nó da posição 1 da lista, e assim por diante. O terceiro array retornado possui as fases (em graus) das tensões nodais desejadas para cada frequência simulada. O terceiro array possui mesma dimensão e padrão do segundo array. Considere que os nós indicados na lista necessariamente existirão na netlist e que os valores da lista serão necessariamente inteiros maiores que zero.

7. O arquivo “.py” que contém a função main deve ter o nome seguindo o seguinte padrão: trab2nomesobrenome.py , onde nome deve ser substituído pelo seu primeiro nome e sobrenome pelo seu último sobrenome. Utilize somente minúsculas e não utilize caracteres especiais (nem o caractere de espaço). Por exemplo: trab2fernandaoliveira.py

8. Para garantir que não haverá arquivos de alunos diferentes com o mesmo nome, os nomes de todos demais arquivos “.py” utilizados (caso o aluno utilize mais de um arquivo “.py” além daquele que contém a função main) devem conter as iniciais do aluno.

Cada linha da netlist indica um novo componente e as informações necessárias sobre ele. A netlist pode conter comentários, que devem ser ignorados pelo programa, conforme indicado abaixo. Para resistores, fontes independentes ou controladas, AC ou DC, capacitores, indutores, transformadores, o seguinte padrão deve ser seguido:

- Comentários, linhas que não são usadas pelo programa: *<comentário>
- Resistor: R<identificação> <nó1> <nó2> <valor da resistência>
- Fonte de corrente controlada por tensão: G<identificação> <nóI(fonte drena a corrente desse nó)> <nóI(fonte injeta a corrente nesse nó)> <nóv(positivo)> <nóv(negativo)> <valor da transcondutância Gm>
- Indutor: L<identificação> <nó1> <nó2> <valor da indutância>
- Capacitor: C<identificação> <nó1> <nó2> <valor da capacitância>
- Transformador: K<identificação> <nó a> <nó b> <nó c> <nó d> <valor da indutância no primeiro enrolamento> <valor da indutância no segundo enrolamento> <valor da indutância mútua>
- Fonte de corrente controlada por corrente, considerando que a corrente de controle passa por um curto do nó C ao nó D: F<identificação> <nóI(fonte drena a corrente desse nó)> <nóI(fonte injeta a corrente nesse nó)> <nó de controle C> <nó de controle D> <valor do ganho de corrente B>

- Fonte de tensão controlada por tensão: E<identificação> <nó positivo> <nó negativo> <nó de controle positivo> < nó de controle negativo> <valor do ganho de tensão A>
- Fonte de tensão controlada por corrente, considerando que a corrente de controle passa por um curto do nó C ao nó D: H<identificação> <nó positivo> <nó negativo> <nó de controle C> <nó de controle D> <valor da transresistência Rm>
- Fonte de corrente AC (onde o pi que deve ser utilizado para transformar de graus para radianos é aquele definido no numpy; numpy.pi): I<identificação> <nó cuja corrente é drenada pela fonte> <nó cuja corrente é injetada pela fonte> AC <amplitude> <fase em graus>
- Fonte de corrente DC: I<identificação> <nó cuja corrente é drenada pela fonte> <nó cuja corrente é injetada pela fonte> DC <valor>
- Fonte de tensão DC: V<identificação> <nó positivo> <nó negativo> DC <valor>
- Fonte de tensão AC (onde o pi que deve ser utilizado para transformar de graus para radianos é aquele definido no numpy; numpy.pi): V<identificação> <nó positivo> <nó negativo> AC <amplitude> <fase em graus>

Tudo o que está entre os caracteres < e > deve ser substituído na netlist e os caracteres < e > em si também não aparecem na netlist.

Observação: um mesmo circuito pode ter fontes AC e DC. Em uma simulação DC as fontes AC devem ser consideradas iguais a zero; em uma simulação AC as fontes DC devem ser consideradas iguais a zero.