Circuitos Elétricos II
Trabalho II - 2022.2

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

Enunciado do primeiro trabalho de análise 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 (cmath também, mas não considero que seja necessário). 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 que recebe uma string com o nome de um arquivo de texto que contém a descrição de circuito por uma "netlist", e que calcula e retorna um array do numpy com os fasores que representam as tensões nodais do circuito calculadas através da análise nodal simples no regime permanente senoidal.

2. Para esse trabalho, modifique o código do trabalho 1 (o código do trabalho 1 também será utilizado para o trabalho 3, então não modifique o arquivo sem antes criar um backup) de forma que a função main calcule e retorne um array do numpy, onde cada posição do array contém os fasores referentes às tensões nodais do circuito obtidos através da análise nodal simples no regime permanente senoidal.

3. Considere que o circuito contém somente fontes de corrente independentes senoidais, fontes de corrente controladas por tensão, resistores, capacitores, indutores e transformadores. Considere também que, se o circuito tiver mais de uma fonte senoidal, elas terão a mesma frequência.

4. Para esse trabalho, mesmo que a netlist indique alguma condição inicial, ela deve ser ignorada (conforme a análise no regime permanente senoidal). O valor DC da fonte também deve ser ignorado nessa função. Lembrando que fasores são números complexos, e, no Python, podem ser representados no formato a+bj.

5. 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. Essa função recebe por argumento uma string com o nome de um arquivo de texto que contém a "netlist" (isto é, a função deve ser definida da seguinte forma: def main(arqNetlist), de forma que, ao fazer a chamada da função dentro de um ".py" o nome do arquivo deve ser colocado como uma string, ex.: main('netlist1.txt') ; reforçando, a função será chamada por um arquivo ".py" e não diretamente pelo terminal). A função main deve calcular e retornar um array do numpy, onde cada posição do array contém os fasores referentes às tensões nodais do circuito obtidos através da análise nodal simples no regime permanente senoidal.

6. O array retornado pela função main deve conter uma quantidade de elementos igual ao número de nós exceto pelo terra. O elemento da posição 0 do array deve ser igual ao fasor da tensão calculada para o nó 1, o elemento da posição 1 deve ser igual ao fasor da tensão calculada para o nó 2, o elemento da posição 2 deve ser igual ao fasor da tensão calculada para o nó 3... A numeração dos nós deve respeitar a numeração da netlist (considere que a numeração da netlist começa de zero, que é o nó terra, e que nenhum número é pulado).

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. 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 de corrente controladas por tensão, e fontes de corrente independentes senoidais, capacitores, indutores e 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>
- Fonte de corrente senoidal, onde sua função deve considerar a expressão i(t) = <amplitude> * cos (2*π*<frequência>*t + <fase>*π/180),e o pi utilizado deve ser aquele definido no numpy (numpy.pi), quando a netlist possuir a seguinte linha: I<identificação> <nó cuja corrente é drenada pela fonte> <nó cuja corrente é injetada pela fonte> SIN <valor DC> <amplitude> <frequência em Hz> <fase em graus>
- Indutor: L<identificação> <nó1> <nó2> <valor da indutância> <condição inicial, considerando o sentido do nó 1 para o nó 2>
- Capacitor: C<identificação> <nó1> <nó2> <valor da capacitância> <condição inicial, considerando o nó 1 como positivo>
- 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>

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