Universidade Federal do Rio de Janeiro

Escola Politécnica

Departamento de Engenharia Eletrônica

Circuitos Elétricos II - 2014.1

Relatório

Autores:

Bernardo Marques
(btm@poli.ufrj.br)

Bruno Campello de Andrade
(bruno.candrade@poli.ufrj.br)

Vinicius Allemand Mancebo Pinto
(vinicius.allemand@poli.ufrj.br)

Avaliador:

Antônio Carlos M. de Queiroz

Sumário

Capítulo 1

Introdução

1.1 Objetivo

O software foi desenvolvido em linguagem cpp e no ambiente de trabalho QT utilizando o compilador minGW. Nosso programa deve analisar circuitos lineares invariantes no tempo, encontrando a resposta em frequência e apresenta-las de três maneiras possíveis: décadas, oitavas e linear.

1.2 Organização do Documento

O Capítulo 2 descreve a estrutura do código.

O Capítulo 3 mostra os problemas que encontramos.

Capítulo 2

Código

Nosso código está divido em diferentes arquivos, que são: CE_MNARF.pro, gaussJordan.cpp, gerarArquivo.cpp, main.cpp, mainWindow.cpp, montarEstampas.cpp, resolver.cpp e seus respectivos headers(.h).

2.1 Estruturas

1. Sistema

Essa estrutura possui um subtipo que define qual o tipo de fonte em questão. E possui mais nove variáveis que servem para definir a fonte. É utilizada para armazenar as informações das fontes de entrada do circuito.

2. Acoplamentos de indutores

Essa estrutura possui dois vetores que definem os indutores acoplados. Possui quatro variáveis para definir os parâmetros do acoplamento. É utilizada para armazenar as informações de transformadores.

3. netlist

Essa é a principal estrutura do programa, ela possui todas as informações do circuito. Ela possui uma variável que guarda o seu nome, outra para guardar o seu tipo (que pode ser qualquer elemento da netlist). Além disso, possui uma variável do tipo fonte, outra do tipo acoplamento de indutores, e possui seis variáveis para definir seus valores na matriz da estampa.

4. Complexos

É uma estrutura com duas variáveis do tipo double, uma equivalente a parte real e outra equivalente a parte imaginária de um número complexo.

2.2 Principais funções

1. montarEstampas

Essa função é responsável pela montagem da estampa de cada elementos, por isso é necessária a entrada do nome da fonte e do omega, pois no caso de a fonte ser do tipo AC a estampa dos elementos irá depender da frequência natural da senóide.

2. resolver

Gauss Jordan é o algoritmo utilizado para resolver o sistema, essa função nada mais é do que tal algoritmo implementado. O algoritmo é simples: para obter o vetor da solução ele diagonaliza a matriz. A implementação é realizada da seguinte maneira: a coluna principal partindo da primeira linha até a última é varrida, tornando os pivots unitários, cada iteração faz operações entre colunas para tal e em seguida zera toda a coluna abaixo do

respectivo pivot. Depois de varrer a diagonal principal uma vez, a matriz vai estar triangularizada na parte de baixo. Em seguida o algoritmo percorre a diagonal principal de baixo para cima zerando as colunas acima de cada pivot, no fim de tal iteração a matriz estará completamente diagonalizada e o sistema resolvido.

3. gerarArquivo

Essa função utiliza o resultado gerado pelo algoritmo do Gauss Jordan, e gera um arquivo de texto que é utilizado para plotar a resposta em frequência. O cálculo desse arquivo depende de um parâmetro que pode gerar saídas para a resposta linear, em década ou em oitava.

4. mainwindow

Essa função é responsável pelo interface gráfica do programa.

Capítulo 3

3.1 13/05/2014

Problema:

1. Não foi apresentado o programa (.exe).

2. Apenas as análises linear e de oitava estavam funcionando.

3. O programa estava lento.

Causa:

1. Ainda não tinha sido gerado os (.dll).

2. A fórmula utilizada na função para análise de décadas estava incorreta.

3. O programa estava calculando muitos pontos tornando o processamento lento.

3.1 15/05/2014

Problema:

1. Dois dos três exemplos testaram não rodaram.

2. A análise do exemplo que rodou estava exibindo a fase errada em um dos nós.

Causa:

1. A estampa de indutores com acoplamentos estava implementada de maneira errada, causado erro quando chamada.

2. A dimensão da matriz onde o sistema de estampa era montada estava subdimensionada, causando erro com sistemas um pouco maiores.

3. A função que calculava o arco tangente para a estampa estava errada, o que gerava uma defasagem na fase.