Escola Politécnica da Universidade de São Paulo

Exercício Programa – Cálculo Numérico e Introdução aos Sistemas de Potência

Lucas Sorensen Paes – 9836919

Renato de Oliveira Freitas – 9837708

São Paulo

2018

Introdução

O exercício programa tem como função apresentar aos alunos a aplicação de cálculos numéricos computacionais nas respectivas áreas de atuação profissional (no caso, Engenharia Elétrica – Sistemas de Potência).

Dessa forma, afim de determinar sobretensões e sobrecargas totais de uma rede elétrica e calcularmos as perdas totais nessa rede, utilizaremos o Método de Newton. Isso porque tal método parte de condições iniciais do sistema para calcular os valores desejados em cada trecho.

Objetivos

-Implementar o algoritmo do Método de Newton em C, testar com diferentes funções e posteriormente utilizá-lo no cálculo dos sistemas de potência.

-Fazer o cálculo da tensão complexa nas barras afim de verificar a existência de sobretensões na rede.

-Calcular potência ativa, reativa em diferentes trechos das redes.

-Fazer o cálculo das perdas totais na rede, permitindo identificar configurações de operação e intervenções a serem feitas na rede para garantir que ela opere de forma eficiente.

Linguagem de Programação utilizada: C

Software utilizado: Visual Studio Code

Análise do Exercício – Algoritmo

Temos que cada barra (ou nó) possui características intrínsecas. Essas são:

- Módulo da Tensão
- Ângulo da Tensão
- Potência Ativa
- Potência Reativa

Existem três tipos de barras nas redes calculadas. Cada barra vem com alguns dados pré-definidos e incógnitas. Segue tabela criada que facilitou o entendimento:

Tipos de Barras	Já dado	Incógnita	Equações	Quantidade
PQ	Pj;Qj	Θj;Vj	fPj = Pcalcj - Pespj	N1
			fQj = Qcalcj - Qespj	
PV	Pj;Vj	Θj;Qj	fPj = Pcalcj - Pespj	N2
			-	
Swing	Vj;Θj	Pj;Qj	-	N3
			-	

As incógnitas Vj e Θj são vetores. O objetivo é calculá-las e, a partir disso, calcular os desvios da potência. Dessa forma, obtemos o seguinte sistema linear de equações:

$$\begin{bmatrix} \dfrac{\partial \tilde{fp}}{\partial \tilde{\theta}} & \dfrac{\partial \tilde{fp}}{\partial \tilde{V}} \\ \dfrac{\partial \tilde{fq}}{\partial \tilde{\theta}} & \dfrac{\partial \tilde{fq}}{\partial \tilde{V}} \end{bmatrix} \cdot \begin{bmatrix} \Delta\tilde{\theta} \\ \Delta\tilde{V} \end{bmatrix} = \begin{bmatrix} -\tilde{fp}(\tilde{\theta},\tilde{V}) \\ -\tilde{fq}(\tilde{\theta},\tilde{V}) \end{bmatrix}$$

Essas são as informações básicas sobre o que deve ser feito. Em relação ao programa:

-Possui diversas funções, divididas em quatro grandes blocos:

1.Cálculo de Matrizes

2.Cálculo de Vetores

3.Cálculo das Redes

4.Cálculo dos Testes

Cada um desses blocos possui diversas funções internamente.

A main pode ser entendida sucintamente da seguinte forma:

1° Passo: Ler arquivo de dados

Para isso utilizamos a função fopen().

2° Passo: Organizar os vetores

Os dados lidos nem sempre vem em ordem.

Exemplo: é lida uma barra tipo PQ, depois uma PV e novamente uma PQ. Elas devem ser colocadas em conjunto para que possam ser equacionadas corretamente.

3° Passo: Criação de um laço for para alocar cada variável.

4° Passo: Cálculo utilizando Método de Newton

5° Passo: Apresentação dos Resultados

Resultados

<u>Testes Iniciais – Método de Newton</u>

O código feito para o método de Newton se encontra como uma função do exercício programa. A seguir apresentaremos os resultados obtidos para os testes iniciais:

• Use seu código para determinar o ponto de mínimo da função F(x, y) = (x − 2)^2 + (y − 3)^2 , calculando para tanto, o ponto onde seu gradiente se anula. (Quantas iterações do método de Newton são necessárias para convergência?)

Foram necessárias 2 iterações, sendo que o resultado final é (2,3).

```
2.000   0.000
0.000   3.000

-4.00e+00 -9.00e+00

p: 0.0 1.0

X: 2.00e+00 3.00e+00

2.000   0.000
0.000   3.000

0.00e+00 0.00e+00

p: 0.0 1.0

X: 2.00e+00 3.00e+00

Solucao: 2.00e+00 3.00e+00


Numero de iteracoes: 2
```

• Dada a função F(x1, x2, x3, x4) = (4*x1 −x2 +x3 −x1*x4, −x1 + 3*x2 −2*x3 − x2*x4, x1 − 2*x2 + 3*x3 − x3*x4, x2^1 + x2^2 + x2^3 − 1), determine a raiz que se obtém pelo método de Newton tomando x = (1, 1, 1, 1) como valor inicial.

O resultado é (0; 0,707; 0,707; 0). Necessárias 4 iterações.

```
3.000   -1.000  1.000   -1.000
-1.000  2.000   -2.000  -1.000
1.000   -2.000  2.000   -1.000
2.000   2.000   2.000   0.000

3.00e+00 -1.00e+00 1.00e+00 2.00e+00

p: 0.0 3.0 2.0 3.0

X: 0.00e+00 1.00e+00 1.00e+00 1.00e+00

3.000   -1.000  1.000   -0.000
-1.000  2.000   -2.000  -1.000
1.000   -2.000  2.000   -1.000
0.000   2.000   2.000   0.000

0.00e+00 0.00e+00 0.00e+00 1.00e+00

p: 0.0 3.0 2.0 3.0

X: 0.00e+00 7.50e-01 7.50e-01 1.00e+00

3.000   -1.000  1.000   -0.000
-1.000  2.000   -2.000  -0.750
1.000   -2.000  2.000   -0.750
0.000   1.500   1.500   0.000

0.00e+00 0.00e+00 0.00e+00 1.25e-01

p: 0.0 1.0 3.0 3.0

X: 0.00e+00 7.08e-01 7.08e-01 1.00e+00

3.000   -1.000  1.000   -0.000
-1.000  2.000   -2.000  -0.708
1.000   -2.000  2.000   -0.708
0.000   1.417   1.417   0.000

0.00e+00 -1.11e-16 -1.11e-16 3.47e-03

p: 0.0 1.0 3.0 3.0

X: 0.00e+00 7.07e-01 7.07e-01 1.00e+00

Solucao: 0.00e+00 7.07e-01 7.07e-01 1.00e+00

Numero de iteracoes: 4
```

• Utilize o método de Newton para determinar solução do sistema (n – 1) × (n – 1), cujas equações são $-x_{i-1} + 2x_i - x_{i+1} = e\ x_i\ n2$, $i = 1, ..., n - 1$, 2 com $x_0 = x_n = 0$, a partir da aproximação inicial nula. (Teste para n = 20, 40 e 80)

Solução:

(n = 20) -> (2.50e-02; 4.77e-02; 6.80e-02; 8.59e-02; 1.01e-01; 1.14e-01; 1.25e-01; 1.32e-01; 1.38e-01; 1.40e-01; 1.40e-01; 1.38e-01; 1.32e-01; 1.25e-01; 1.14e-01; 1.01e-01; 8.59e-02; 6.80e-02; 4.77e-02; 2.50e-02)

(n = 40) -> (1.31e-02; 2.56e-02; 3.75e-02; 4.88e-02; 5.94e-02; 6.94e-02; 7.88e-02; 8.75e-02; 9.56e-02; 1.03e-01; 1.10e-01; 1.16e-01; 1.21e-01; 1.26e-01; 1.30e-01; 1.34e-01; 1.36e-01; 1.38e-01; 1.40e-01; 1.40e-01; 1.40e-01; 1.40e-01; 1.38e-01; 1.36e-01; 1.34e-01; 1.30e-01; 1.26e-01; 1.21e-01; 1.16e-01; 1.10e-01; 1.03e-01; 9.56e-02; 8.75e-02; 7.88e-02; 6.94e-02; 5.94e-02; 4.88e-02; 3.75e-02; 2.56e-02; 1.31e-02)

(n = 80) -> (6.71e-03; 1.33e-02; 1.97e-02; 2.59e-02; 3.20e-02; 3.79e-02; 4.37e-02; 4.93e-02; 5.47e-02; 6.00e-02; 6.52e-02; 7.01e-02; 7.49e-02; 7.96e-02; 8.40e-02; 8.83e-02; 9.25e-02; 9.65e-02; 1.00e-01; 1.04e-01; 1.07e-01; 1.11e-01; 1.14e-01; 1.17e-01; 1.20e-01; 1.22e-01; 1.25e-01; 1.27e-01; 1.29e-01; 1.31e-01; 1.33e-01; 1.34e-01; 1.36e-01; 1.37e-01; 1.38e-01; 1.39e-01; 1.39e-01; 1.40e-01; 1.40e-01; 1.41e-01; 1.41e-01; 1.40e-01; 1.40e-01; 1.39e-01; 1.39e-01; 1.38e-01; 1.37e-01; 1.36e-01; 1.34e-01; 1.33e-01; 1.31e-01; 1.29e-01; 1.27e-01; 1.25e-01; 1.22e-01; 1.20e-01; 1.17e-01; 1.14e-01; 1.11e-01; 1.07e-01; 1.04e-01; 1.00e-01; 9.65e-02; 9.25e-02; 8.83e-02; 8.40e-02; 7.96e-02; 7.49e-02; 7.01e-02; 6.52e-02; 6.00e-02; 5.47e-02; 4.93e-02; 4.37e-02; 3.79e-02; 3.20e-02; 2.59e-02; 1.97e-02; 1.33e-02; 6.71e-03)