****

**Universidade Estadual de Santa Cruz – UESC**

**Relatórios de Implementações de Métodos da Disciplina Análise Numérica**

**Relatório de implementações realizadas por Iago Gomes Santana**

**Disciplina Análise Numérica.**

**Curso Ciência da Computação**

**Semestre 2021.1**

**Professor Gesil Sampaio Amarante II**

**Ilhéus – BA**

**2021**

ÍNDICE

Lista de Figuras

Linguagem Escolhida e justificativas

A linguagem escolhida foi o Python (versão 3.8.5) por sua facilidade em lidar com tipos diferentes de dados e praticidade ao programar algumas funções matemáticas. Outro ponto importante para a escolha foi com relação a leitura do .txt de entrada e a conversão para f(x), algo que eu tive muita dificuldade em implementar, mas, com ajuda de amigos acabei chegando a algum lugar.  
Uma desvantagem que percebi no python foi que, em alguns casos, quando envolvem operações com números decimais com muitos dígitos após a vírgula, os resultados podem ser imprecisos. Para contornar isso eu descobri uma função chamada Decimal() que resolve esse problema de precisão.

Bissecção

# Estratégia de Implementação:

Esse método me lembrou bastante uma busca binária, então eu acabei usando a mesma estratégia para implementar separando os valores de a, b, f(a), f(b) e calculando c e f(c) com base neles. A partir daí, foi só fazer as verificações para saber se f(c) seria minha resposta, se ficaria no lugar de f(a) ou f(b). No caso das duas últimas, foi só multiplicar os resultados de f(a) e f(c), caso o resultado seja negativo, c assume o lugar de b, caso contrário, c assume o lugar de a.

# Estrutura dos Arquivos de Entrada/Saída Entrada:

valor ao lado do nome da variável

function = '1-(1+x+x^2/2)\*E^(-x)-0.9'

a = '0'

b = '10'

error = '0.009'

# Saída:

resposta final

# Problema teste 1, 2, 3...

Eu não consegui interpretar a questão 3 para entender o que era pra fazer exatamente. Nas outras, o algoritmo funcionou

# Dificuldades enfrentadas

An… por algum motivo, meu python não estava lendo arquivos, eu tentei de vários jeitos diferentes, copiei/colei tutorial da internet, e ainda assim, não funcionou, não faço ideia do motivo, até porque já trabalhei com arquivos em python antes e nunca tivesse esse problema

Secante

# Estratégia de Implementação:

Aqui eu apenas calculei o resultado das funções f(x0) e f(x1) e usei os seus resultados na fórmula do método da secante, conseguindo assim o valor de xk+2. E eu repeti esse processo em um looping “infinito” até que o valor de f(xk+2) seja menor que meu erro

# Estrutura dos Arquivos de Entrada/Saída

Igual a estrutura da Bissecção

# Problema teste 1, 2, 3...

Mesma coisa da Bissecção, consegui resultados para o problema teste 1 e 2, mas o terceiro eu empaquei

# Dificuldades enfrentadas

An… por algum motivo, meu python não estava lendo arquivos, eu tentei de vários jeitos diferentes, copiei/colei tutorial da internet, e ainda assim, não funcionou, não faço ideia do motivo, até porque já trabalhei com arquivos em python antes e nunca tivesse esse problema

Eliminação de Gauss

# Estratégia de Implementação:

Novamente, implementei 2 funções auxiliares, uma para calcular m e outra para calcular l, dessa vez não foi algo tão necessário, mas eu achei mais organizado fazer assim. Usei um for e uma variável de controle (mx) para calcular m com base no estado atual da matriz e usei esse m na função l(x, m, y) para ter l.

A partir daí, foi só fazer a substituição voltando da linha final até a inicial da matriz, substituindo os valores de x por seus respectivos valores descobertos na iteração anterior

# Estrutura dos Arquivos de Entrada/Saída

preencher a matriz como está sugerido no arquivo python, a estrutura ficou limitada ao caso feito em sala de aula, em que temos algo do tipo:  
[a1x1 a2x2 a3x3] = y

[b1x1 b2x2 b3x3] = y

[c1x1 c2x2 c3x2] = y

Isso foi uma falha na implementação que só percebi quando era tarde demais, mas mesmo assim, segue um exemplo de entrada:

MATRIZ = [

[3.0, -0.1, -0.2, 7.85],

[0.1, 7.0, -0.3, -19.3],

[0.3, -0.2, 10.0, 71.4]

]

# Problema teste 1, 2, 3...

Por conta dessa limitação citada acima, não foi possível responder às questões propostas, o algoritmo não estava preparado para os Xs fazerem operações entre eles mesmos

# Dificuldades enfrentadas

Matrizes com incógnitas que fazem operações enquanto estão sendo multiplicadas por uma mesma constante

Fatoração LU

# Estratégia de Implementação:

Para a implementação desse método, eu desenvolvi 2 funções auxiliares para o cálculo do somatório para a matriz L e U, como elas tem uma pequena diferença, achei melhor deixá-las separadamente.

Eu criei uma matriz de tamanho igual a da questão e inicializei ela com zeros. Usei as fórmulas para preencher a primeira linha da matriz U e a primeira coluna da matriz L. Após isso, usei a outra fórmula para preencher as outras posições de ambas matrizes, de modo a preencher apenas a diagonal principal e o que estiver acima para U, e a parte abaixo da diagonal principal de L, deixando assim zeros nas posições que fogem dessa regra.

A partir daí, foi só fazer a substituições e encontrar o resultado para o sistema

# Estrutura dos Arquivos de Entrada/Saída

preencher a matriz como está sugerido no arquivo python, a estrutura ficou limitada ao caso feito em sala de aula, em que temos algo do tipo:  
[a1x1 a2x2 a3x3] = y

[b1x1 b2x2 b3x3] = y

[c1x1 c2x2 c3x2] = y

Isso foi uma falha na implementação que só percebi quando era tarde demais, mas mesmo assim, segue um exemplo de entrada:

MATRIZ = [

[3.0, -0.1, -0.2, 7.85],

[0.1, 7.0, -0.3, -19.3],

[0.3, -0.2, 10.0, 71.4]

]

# Problema teste 1, 2, 3...

Por conta dessa limitação citada acima, não foi possível responder às questões propostas, o algoritmo não estava preparado para os Xs fazerem operações entre eles mesmos.

Mas COnsegui fazer os exercícios 5.1 e 5.5, funcionou direitinho nesses casos

# Dificuldades enfrentadas

Matrizes com incógnitas que fazem operações enquanto estão sendo multiplicadas por uma mesma constante

Considerações Finais