****

**Universidade Estadual de Santa Cruz – UESC**

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

**Relatório de implementações realizadas por Luiz Felipe do Rosário Alves Silva**

**Disciplina Análise Numérica.**

**Curso Ciência da Computação**

**Semestre 2025.1**

**Professor Gesil Sampaio Amarante II**

**Ilhéus – BA**

**2025**

ÍNDICE

Lista de Figuras

Linguagem(ns) Escolhida(s) e justificativas

Python foi escolhido devido à sua simplicidade, versatilidade e grande suporte de bibliotecas, como NumPy, que facilita o trabalho com dados numéricos e operações matemáticas. Sua sintaxe clara e direta contribui para um desenvolvimento rápido e eficiente.

[Link - Python](https://www.python.org/)

GeoGebra foi utilizado para criar gráficos e visualizar funções, permitindo uma melhor análise e compreensão dos dados. Esta ferramenta é especialmente útil para representar graficamente funções matemáticas e visualizar interações numéricas de forma intuitiva.

[Link - CodeSnap](https://www.geogebra.org/)

VS Code foi escolhido como o ambiente de desenvolvimento devido à sua leveza, alta personalização e excelente suporte a extensões. Ele oferece ferramentas poderosas para codificação, depuração e controle de versões, tornando o desenvolvimento mais produtivo e organizado.

[Link - VS Code](https://code.visualstudio.com/)

CodeSnap foi utilizado para capturar prints do código de maneira organizada e visualmente atraente, facilitando a documentação do projeto. Esta ferramenta é útil para criar imagens de trechos de código para incluir em relatórios e apresentações.

[Link - CodeSnap](https://marketplace.visualstudio.com/items?itemName=adpyke.codesnap)

Bissecção

# Estratégia de Implementação:

Para receber as expressões, utilizamos a função eval() com a variável x definida no escopo. Esse foi o método mais simples e facilitado para as funções pois o eval já está na base do Python.

Para o critério de parada, seguimos para o valor absoluto de f(x) menor ou igual à tolerância e o comprimento do intervalo menor que a tolerância. Pois, com ambos, podemos ter um melhor aproveitamento do código sem adaptações conforme outros testes.

Primeiro o programa verifica se tem mudança de sinal no intervalo inicial. Caso não, lança um erro informando que o intervalo é inválido.

Para evitar divisão por zero no cálculo de (b-a)/a, foi aplicada uma condição que zera o valor quando a = 0.

A saída foi registrada em arquivo .txt, com uma tabela organizada contendo os valores de cada iteração, e a raiz aproximada no final.

# Estrutura dos Arquivos de Entrada/Saída

Entrada: bisseccao-fun.txt

1. Função adaptada como String para ser reconhecida pelo eval() do próprio Python;
2. Intervalo de pontos de X onde possivelmente está uma das raízes. Separados por espaço somente. Podendo ser Int ou Float;
3. Tolerância (Sendo válido para valor absoluto e comprimento do intervalo, o que se aproximar primeiro).

Saída: bisseccao-res.txt:

1. Cabeçalho:

* a e b: Limites atuais do intervalo de busca;
* f(a) e f(b): Valores da função nos extremos;
* (b - a): Tamanho do intervalo;
* (b - a)/a: Taxa relativa de redução do intervalo;
* x: Ponto médio (candidato à raiz);
* f(x): Valor da função no ponto médio.

1. Raiz aproximada com 6 casas decimais por causa da flutuação referente à tolerância escolhida.

# Problema teste 3.3

O problema envolve calcular o tempo de subida de um amplificador RC, resolvendo numericamente equações que determinam quando a resposta atinge 10% e 90% do valor final.

Texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Função (10%):

1 - (1 + x + (x\*\*2 / 2)) \* math.e\*\*-x - 0.1, onde X = T.

Geogebra:

Gráfico

O conteúdo gerado por IA pode estar incorreto.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:**Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.**

Função (90%):

1 - (1 + x + (x\*\*2 / 2)) \* math.e\*\*-x - 0.9, onde X = T.

Geogebra:

Gráfico

O conteúdo gerado por IA pode estar incorreto.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída: Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.6

A questão busca determinar o ângulo de lançamento α\alphaα para que um míssil atinja um alvo, resolvendo uma equação trigonométrica com dados conhecidos.

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

Função:

(math.sin(x) \* math.cos(x)) / (0.8 - math.cos(x)\*\*2) - math.tan(math.radians(40))

Geogebra:

Gráfico, Gráfico de linhas

O conteúdo gerado por IA pode estar incorreto.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.8

O problema compara dois planos de financiamento. A taxa de juros de cada plano é obtida resolvendo uma equação não linear. O plano com menor raiz tem juros mais baixos.

Texto, Carta

O conteúdo gerado por IA pode estar incorreto.

Texto, Carta

O conteúdo gerado por IA pode estar incorreto.

Função (A):

5.283 \* x\*\*10 - 6.283 \* x\*\*9 + 1

Geogebra:

Gráfico, Gráfico de linhas

O conteúdo gerado por IA pode estar incorreto.

Entrada:

Uma imagem contendo Interface gráfica do usuário

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

Função (B):

6.512 \* x\*\*13 - 7.512 \* x\*\*12 + 1

Geogebra:

Gráfico, Gráfico de linhas

O conteúdo gerado por IA pode estar incorreto.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Posição Falsa

# Estratégia de Implementação:

As expressões matemáticas foram avaliadas usando eval(). Essa abordagem foi adotada por ser prática e já integrada à linguagem Python, sem necessidade de bibliotecas externas.

O critério de parada foi definido como o valor absoluto de f(c) menor ou igual à tolerância. Isso garante que o valor da função no ponto estimado da raiz esteja suficientemente próximo de zero. O intervalo não é usado como critério de parada nesse caso pois como não temos uma divisão simétrica. Neste caso, é um pouco mais interessante ver a evolução dos valores do intervalo no nosso resultado.

Todos os resultados foram registrados em um arquivo .txt, com uma tabela contendo os valores de cada iteração, e a raiz aproximada ao final com 6 casas decimais, respeitando a tolerância definida.

# Estrutura dos Arquivos de Entrada/Saída

Entrada: posicaofalsa-fun.txt

1. Função como string avaliada com eval();
2. Intervalo de busca: dois valores (a e b), separados por espaço;
3. Tolerância.

Saída: posicaofalsa-res.txt

1. Cabeçalho da Tabela:
   * a, b: Intervalo atual;
   * f(a), f(b): Valores da função nos extremos;
   * (b - a): Tamanho do intervalo (só para acompanhamento, não afeta parada);
   * c: Raiz aproximada da iteração (usando interpolação linear);
   * f(c): Valor da função no ponto c.
2. Resultado final:
   * Linha ao final com a raiz aproximada:

# Problema teste 3.3

Neste método, os valores de entrada são quase semelhantes. Para fins de análise, apenas a tolerância foi mudada. Para mostrar a diferença em quantidade de iterações.

A para o método anterior era de 0,000001 e a para o atual é de 0,0000001

Função 10%:

1 - (1 + x + (x\*\*2 / 2)) \* math.e\*\*-x - 0.1, onde X = T.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Em f(c) podemos ver que a diferença entre a 1ª, 2ª e 3ª iteração é

Função (90%):

1 - (1 + x + (x\*\*2 / 2)) \* math.e\*\*-x - 0.9, onde X = T.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.6

Da mesma forma,também na entrada a única diferença é a tolerância.

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.8

Nesse caso foi possível notar suficiência ao utilizar tolerância de 0,00001.

Função (A):

5.283 \* x\*\*10 - 6.283 \* x\*\*9 + 1

Entrada:

Uma imagem contendo Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto, Calendário

O conteúdo gerado por IA pode estar incorreto.

Aqui podemos notar a evolução de ‘**c**’nas iterações, e como a diferença vem diminuindo ao decorrer das mesmas.

Função (B):

6.512 \* x\*\*13 - 7.512 \* x\*\*12 + 1

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Newton-Raphson

# Estratégia de Implementação:

É um método iterativo baseado em derivadas. Ele começa com um chute inicial e usa a seguinte fórmula para refinar o valor de x:

x = x - (eval(expressao) / eval(derivada))

É necessário fornecer a derivada da função porque o método depende diretamente dela para saber em que direção e com que intensidade ajustar o valor de x.

Novamente o uso da função eval() para interpretar tanto a função quanto sua derivada diretamente como strings.

O critério de parada foi o f(x). Pois, como é um método rápido, essa condição já garante alta precisão em poucos passos. Já que, ela não precisa dos dois pontos e pode avançar rapidamente em direção à raiz.

# Estrutura dos Arquivos de Entrada/Saída

**Entrada: newtonraphson-fun.txt**

1. **Função** representada como string;
2. **Derivada da função**, também em string;
3. **Intervalo [a, b]** para chute inicial o programa começa pelo ponto médio.
4. **Tolerância**.

**Estrutura do Arquivo de Saída (newtonraphson-res.txt)**

1. Cabeçalho:
   * x\_n: Valor atual da iteração;
   * f(x\_n): Valor da função nesse ponto.
2. Raiz Aproximada.

# Problema teste 3.3

Entrada (10%):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Entrada (90%):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.6

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.8

Entrada (A):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Tabela

O conteúdo gerado por IA pode estar incorreto.

Entrada (B):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Tabela

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Secante

# Estratégia de Implementação:

Sem a necessidade de derivadas, o método começa com dois valores iniciais​, e a cada iteração calcula um novo valor x​\_n utilizando a fórmula:

**proximo\_x = (f\_atual \* x\_antigo - f\_antigo \* x\_atual) / (f\_atual - f\_antigo)**

Esse valor é calculado até que a função f(x\_n) seja suficientemente próxima de zero, dentro da tolerância estabelecida.

A função eval() é utilizada para interpretar a função como string e avaliá-la a cada iteração.

O critério de parada utilizado é apenas o valor absoluto de f(x\_n) porque o método da Secante, ao contrário de outros métodos, não precisa da atualização do intervalo, já que a função converge rapidamente em direção à raiz.

# Estrutura dos Arquivos de Entrada/Saída

Entrada (secante-fun.txt):

1. Função representada como string;
2. Dois valores iniciais de x (separados por espaço) como ponto de partida para o método;
3. Tolerância.

**Saída (secante-res.txt):**

1. Cabeçalho:
   * x\_n: Valor atual da iteração;
   * f(x\_n): Valor da função nesse ponto.
2. Resultado final:
   * Raiz aproximada encontrada, com precisão definida pela tolerância.

# Problema teste 3.3

Entrada (10%):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Entrada (90%):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Uma imagem contendo Interface gráfica do usuário

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.6

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Tabela

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 3.8

Entrada (A):

Uma imagem contendo Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Tabela

O conteúdo gerado por IA pode estar incorreto.

Entrada (B):

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Tabela

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Eliminação de Gauss

# Estratégia de Implementação:

A implementação começa com a leitura da matriz aumentada do sistema linear do arquivo. O código aplica a eliminação de Gauss para escalonar a matriz e torná-la triangular superior. Caso o pivô seja zero, a função trocar\_linhas é utilizada para trocar as linhas da matriz e evitar problemas de divisão por zero. Após o escalonamento, a substituição inversa é realizada para resolver o sistema, calculando as variáveis de baixo para cima. Cada solução de xxx é calculada a partir da última equação e retrocedendo, usando as soluções já calculadas. O resultado é então armazenado em um arquivo de saída, incluindo a matriz escalonada e as soluções das variáveis. Essa abordagem é eficiente e lida adequadamente com o caso de pivôs nulos.

# Estrutura dos Arquivos de Entrada/Saída

Entrada (eliminacaogauss-mat.txt):  
Contém a matriz aumentada do sistema linear, onde cada linha representa uma equação do sistema e a última coluna contém os termos independentes. A estrutura da entrada foi escolhida por ser direta e compatível com o método da eliminação de Gauss.

Saída (eliminacaogauss-res.txt):  
Contém a matriz escalonada após a eliminação de Gauss, seguida das soluções do sistema no formato xi=valor, uma por linha. Esse formato facilita a leitura e compreensão das soluções do sistema linear.

# Problema teste 4.1

Entrada:

Fundo preto com letras brancas

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 4.3

Entrada:

Foto preta e branca de um teclado

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 4.6

Entrada:

Uma imagem contendo Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Fatoração LU

# Estratégia de Implementação:

É realizada no código a partir da leitura da matriz aumentada do sistema, que é dividida em b (termos independentes) e U (coeficientes da matriz). O processo começa com o pivotamento parcial, que troca as linhas da matriz U para garantir que o maior valor absoluto na coluna seja o pivô, aumentando a estabilidade numérica.

Em seguida, o algoritmo de eliminação de Gauss é aplicado para transformar a matriz UU em uma matriz triangular superior, ao mesmo tempo em que são calculados os multiplicadores, que são armazenados em uma lista. Esses multiplicadores são usados para preencher a matriz triangular inferior L, onde a diagonal é preenchida com 1s e os valores abaixo da diagonal correspondem aos multiplicadores.

Após a decomposição, o sistema L ⋅ y = b é resolvido por substituição direta, e o sistema U ⋅ x = y é resolvido por substituição regressiva para encontrar a solução x. Essa abordagem garante a solução do sistema linear de forma eficiente e numericamente estável.

# Estrutura dos Arquivos de Entrada/Saída

Entrada (fatoracaoLU-mat.txt):

Contém uma matriz aumentada representando o sistema linear. O formato da entrada é uma matriz de nnn linhas e n+1n+1n+1 colunas, onde a última coluna contém os termos independentes do sistema. Essa escolha foi feita para simplificar a leitura e a manipulação dos dados, pois a matriz aumentada já contém tudo o que é necessário para resolver o sistema.

Saída (fatoracaoLU-res.txt):

Contém as soluções das variáveis do sistema linear. Cada solução é salva em uma linha do arquivo no formato xi = valor. O formato de saída foi escolhido para ser simples e direto, permitindo fácil leitura e compreensão dos resultados.

# Problema teste 4.1

Entrada:  
Fundo preto com letras brancas

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 4.3

Entrada:

Foto preta e branca de um teclado

O conteúdo gerado por IA pode estar incorreto.

Saída:

Uma imagem contendo Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 4.6

Entrada:  
Uma imagem contendo Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Jacobi

# Estratégia de Implementação:

Foi utilizada a biblioteca NumPy para facilitar as operações com matrizes e vetores. O código verifica automaticamente se a matriz é diagonalmente dominante, condição necessária para a convergência do método de Jacobi. Caso não seja, uma mensagem é salva no arquivo de saída e a execução é interrompida.

O critério de parada é baseado na norma do erro entre iterações consecutivas, com tolerância definida no arquivo de entrada. Isso garante precisão ajustável e controle sobre a convergência. Toda a saída é registrada em um arquivo, sem exibir nada no terminal, o que torna o uso mais limpo e adequado para integração ou testes.

# Estrutura dos Arquivos de Entrada/Saída

Entrada (jacobi-mat.txt):

* Primeira linha: tolerância (ex: 0.0001)
* Linhas seguintes: coeficientes das equações com o termo independente ao final

Saída (jacobi-res.txt):

* As soluções x₁, x₂, etc., uma por linha
* Número de iterações realizadas
* Ou uma mensagem de erro se a matriz não permitir convergência

# Problema teste 5.1

Entrada:

Calendário

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 5.2

Entrada:

Uma imagem contendo Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 5.5

Entrada:

Uma imagem contendo Calendário

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Gauss-Seidel

# Estratégia de Implementação:

Foi utilizada a biblioteca **NumPy** para facilitar o trabalho com matrizes e vetores. O código verifica automaticamente se a matriz é diagonalmente dominante, condição necessária para a convergência do método de Gauss-Seidel. Caso essa condição não seja satisfeita, uma mensagem é gravada no arquivo de saída e a execução é encerrada. O critério de parada adotado é baseado na norma do erro entre duas iterações consecutivas, comparado com uma tolerância fornecida pelo usuário no próprio arquivo de entrada. Toda a saída é registrada em arquivo, mantendo o terminal limpo e favorecendo o uso automatizado ou a verificação posterior dos resultados.

# Estrutura dos Arquivos de Entrada/Saída

Entrada (gaussseidel-mat.txt):

• Primeira linha: tolerância do erro   
• Linhas seguintes: coeficientes do sistema linear, com o termo independente ao final de cada linha

Saída (gaussseidel-res.txt):

• As soluções x₁, x₂, etc., uma por linha  
• Número de iterações realizadas  
• Ou uma mensagem de erro, caso a matriz não seja diagonalmente dominante e o método não possa ser aplicado

# Problema teste 5.1

Entrada:

Calendário

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 5.2

Entrada:

Uma imagem contendo Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 5.5

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Gauss-Jordan

# Estratégia de Implementação:

Descreva as escolhas que teve que fazer para poder superar eventuais dificuldades com limitações da linguagem, do ambiente de programação, etc.

De que forma resolveu tratar os erros, critérios de parada, formato de saída e quaisquer outros fatores importantes para a solução dos problemas.

# Estrutura dos Arquivos de Entrada/Saída

Descreva o formato dos arquivos de entrada e saída e justifique as escolhas.

# Problema teste 5.1

Entrada:

Calendário

O conteúdo gerado por IA pode estar incorreto.

Saída:

Texto

O conteúdo gerado por IA pode estar incorreto.

# Problema teste 5.2

Entrada:

Texto

O conteúdo gerado por IA pode estar incorreto.

Saída:

Imagem em preto e branco

O conteúdo gerado por IA pode estar incorreto.

# Dificuldades enfrentadas

Considerações Finais