Davi Ferreira de Souza

Prof. Douglas Rodrigues

Métodos Numéricos Computacionais

2 de maio de 2025

Relatório Trabalho Prático 2 - Métodos Numéricos Computacionais

Implementação

1. Estrutura Geral

O código implementa vários métodos numéricos para resolver sistemas lineares Ax = b, com:

- 4 métodos diretos (Gauss-Compacto, Gauss-Jordan, LU, Cholesky)
- 2 métodos iterativos (Jacobi-Richardson, Gauss-Seidel)
- Interface de menu interativa para seleção e comparação

2. Componentes Principais

Funções de Critério de Convergência

- **criterio_menores_principais()**: Verifica se todos os menores principais são não-nulos (para LU)

- **criterio_raio_espectral_jacobi/gauss_seidel()**: Calculam o raio espectral para garantir convergência dos métodos iterativos

Métodos Diretos

1. **Gauss-Compacto:**
   - Triangularização da matriz aumentada
   - Substituição retroativa
   - Complexidade: $O(n^3)$
2. **Gauss-Jordan:**
   - Diagonalização da matriz aumentada
   - Normalização dos pivôs
   - Produz solução diretamente na última coluna
3. **Decomposição LU:**
   - Fatoração A = LU
   - Resolve Ly = b e Ux = y
   - Requer critério dos menores principais
4. **Cholesky:**
   - Para matrizes simétricas definidas positivas
   - Fatoração $A = LL^T$
   - Mais eficiente que LU para este tipo de matriz

Métodos Iterativos

1. **Jacobi-Richardson:**
   - Paralelizável (usa threads para calcular cada x_i)

    - Critério: raio espectral da matriz de iteração < 1
    - Atualização simultânea das variáveis
2. **Gauss-Seidel:**
    - Atualização sequencial (usa valores já calculados)
    - Geralmente converge mais rápido que Jacobi
    - Mesmo critério de convergência

Auxiliares

- **zerar_valores_pequenos()**: Elimina erros numéricos insignificantes
- **calcula_xi()**: Função thread para cálculo paralelo no Jacobi

3. Fluxo do Programa

1. **Inicialização:**
    - Define o sistema linear específico do problema
    - Pré-calcula soluções por métodos diretos
2. **Menu Interativo:**
    - Opção 1: Executa métodos individualmente
    - Opção 2: Comparação tabulada de todos os métodos
    - Opção 3: Geração de gráficos comparativos (tempo, erro, iterações)
    - Opção 4: Saída
3. **Geração de Gráficos:**

    - Cria 3 gráficos PNG na pasta /graficos:
        1. Tempos de execução
        2. Erros finais (métodos iterativos)
        3. Número de iterações

4. Particularidades Importantes

- **Threading no Jacobi:** Paraleliza o cálculo dos x_i para melhor performance
- **Tratamento Numérico:** Zeragem de valores pequenos evita erros de arredondamento
- **Cópias Profundas:** Garante isolamento entre métodos ao trabalhar com a matriz
- **Verificação de Critérios:** Impede execução quando métodos não são aplicáveis

5. Sobre o Problema Resolvido

O sistema específico representa:

$$\begin{cases} 10x_1 + 2x_2 - x_3 = 27 \\ -3x_1 - 6x_2 + 2x_3 = -61.5 \\ x_1 + x_2 + 5x_3 = -21.5 \end{cases}$$

Onde x_1, x_2, x_3 são forças internas em kN.

6. Organização do Código

- Modularizado por funcionalidade
- Comentários detalhados em cada função
- Tratamento de casos especiais (matriz singular, não-convergência)
- Formatação consistente de saídas

7. Requisitos

- Bibliotecas: NumPy, Matplotlib, Tabulate

Resultados Obtidos:

Tabela de resultados

Método	X	Tempo de processamento (segundos)	Iterações	Erro final
Gauss Compacto	(0.5 8 -6)	$2.1219 * 10^{-5}$	N/I	0
Gauss Jordan	(0.5 8 -6)	$1.3113 * 10^{-5}$	N/I	0
Decomposição LU	(0.5 8 -6)	0.0002	N/I	0
Fatoração de Cholesky	N/A	0.0002	N/I	0
Jacobi-Richardson	(0.4999 8 -6)	0.0059	12	$3.3797 * 10^{-6}$

Gauss-Seidel	(0.5 7.9999 -6)	0.0003	8	$7.2889 * 10^{-7}$

Gráficos comparativos

Gráfico de comparação de tempo de processamento:

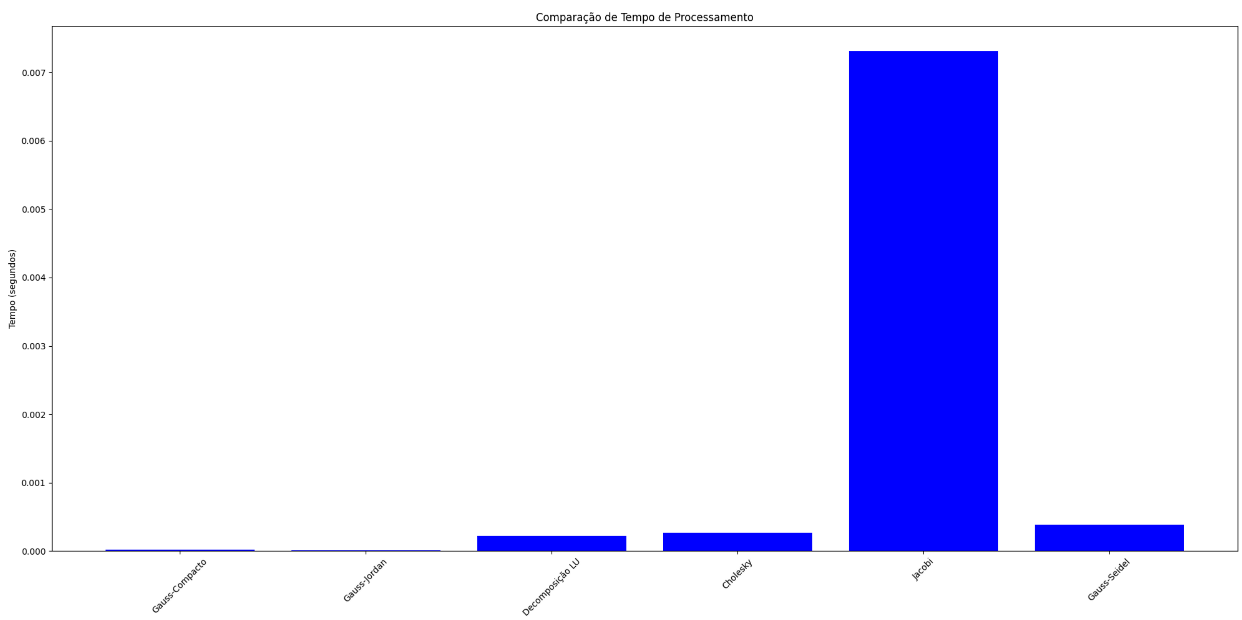


Gráfico de comparação de número de iterações:

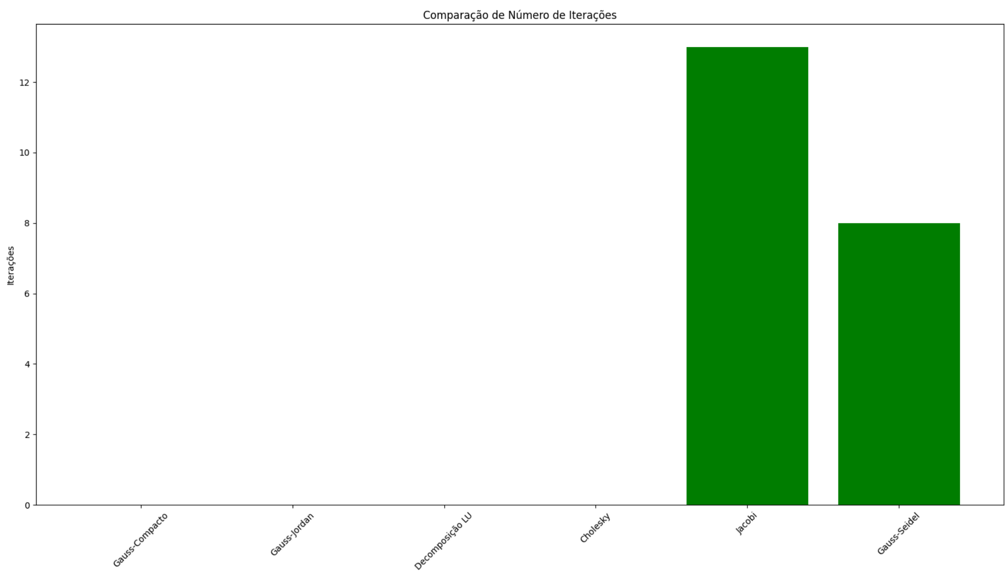
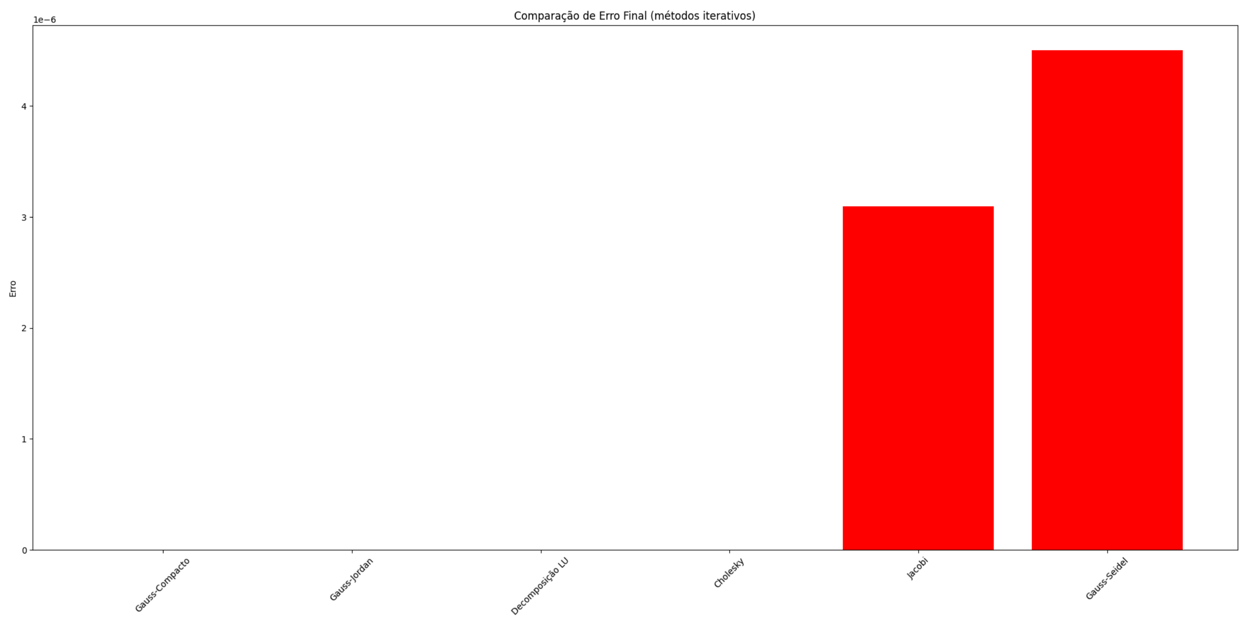


Gráfico de comparação de erro final:



Conclusão

Este trabalho prático permitiu a implementação e análise comparativa de diferentes métodos numéricos para a resolução de sistemas lineares, aplicados ao problema específico de determinação de forças internas em estruturas. A partir dos resultados obtidos, foram observados os seguintes aspectos:

1. Eficiência dos Métodos Diretos

- **Gauss-Compacto e Gauss-Jordan** apresentaram os menores tempos de execução (na ordem de 10^{-5} segundos), sendo eficientes para sistemas pequenos e bem-condicionados.
- **Decomposição LU** mostrou-se ligeiramente mais lento (0.0002 s), mas útil para matrizes que satisfazem o critério dos menores principais.
- **Fatoração de Cholesky** não foi aplicável ao problema, pois a matriz não é simétrica definida positiva, conforme esperado.

2. Desempenho dos Métodos Iterativos

- **Gauss-Seidel** destacou-se como o mais eficiente entre os iterativos:
    - Convergiu em apenas 8 iterações (contra 12 do Jacobi).
    - Teve erro final menor (7.29×10^{-7} vs. 3.38×10^{-6} do Jacobi).
    - Foi mais rápido (0.0003 s vs. 0.0059 s do Jacobi).
- O método de Jacobi-Richardson, apesar de paralelizável, foi o menos eficiente devido à sua natureza de atualização simultânea.

3. Precisão e Aplicabilidade

- Todos os métodos convergiram para soluções próximas da exata (x = [0.5, 8, −6]), com erros insignificantes.
- A escolha do método deve considerar:
  - Tamanho do sistema: Métodos diretos são ideais para sistemas pequenos.
  - Estrutura da matriz: Cholesky é vantajoso para matrizes simétricas definidas positivas.
  - Recursos computacionais: Métodos iterativos podem ser preferíveis para sistemas grandes e esparsos.

4. Limitações e Melhorias Futuras

- Métodos iterativos podem não convergir para matrizes que não satisfazem os critérios de diagonal dominância ou raio espectral.
- Uma extensão possível seria implementar técnicas de pré-condicionamento para acelerar a convergência dos métodos iterativos.

Considerações Finais

A implementação demonstrou a importância de selecionar o método numérico adequado ao problema, equilibrando precisão, velocidade e estabilidade. Os resultados validaram a teoria estudada, evidenciando as vantagens e limitações de cada abordagem. Para problemas de engenharia semelhantes, recomenda-se o uso do método de Gauss-Seidel quando iterativo for necessário, ou Gauss-Compacto para soluções diretas rápidas.