Trabalho Final de Computação
Científica e Análise de Dados

Resolução do Problema Estacionário com Condições de Dirichlet via o Método dos Elementos Finitos Utilizando Métodos Iterativos

João Victor Lopez Pereira

3 de dezembro de 2024

Rio de Janeiro - RJ

Resumo:

Este documento apresenta como trabalho final da disciplina *Computação Científica e Análise de Dados* a resolução de um sistema genérico de equações diferenciais parciais com condições de Dirichlet utilizando o Método dos Elementos Finitos, conforme abordado nas aulas da disciplina *Introdução ao Método dos Elementos Finitos*, ministradas na Universidade Federal do Rio de Janeiro, durante o segundo semestre de 2024.

O método dos elementos finitos é amplamente utilizado para transformar equações diferenciais parciais e ordinárias em sistemas lineares da forma $Ax = b$, que podem ser resolvidos por métodos numéricos iterativos, como o método de Gauss-Jacobi. Esse método é particularmente interessante para esse contexto devido à estrutura da matriz A, que, nesse caso, é esparsa e tri-diagonal, o que reduz significativamente o custo computacional e o uso de memória, tornando-o adequado para sistemas de grande escala. No documento, também é realizada uma análise detalhada das vantagens do método de Gauss-Jacobi nesse cenário, considerando suas propriedades e desempenho em sistemas esparsos gerados pelo método dos elementos finitos.

Abstract:

This document presents the solution of a generic system of ordinary differential equations with Dirichlet conditions using the Finite Element Method, as covered in the course *Introduction to the Finite Element Method*, taught at the Federal University of Rio de Janeiro, during the second half of 2024, as the final project for the course *Scientific Computing and Data Analysis*.

The finite element method is widely used to transform partial differential equations into linear systems of the form $Ax = b$, which can be solved using iterative numerical methods such as the Gauss-Jacobi method. This method is particularly advantageous in this context due to the sparse and tri-diagonal structure of the matrix A, significantly reducing computational cost and memory usage, making it suitable for large-scale systems. The document also provides a detailed analysis of the advantages of the Gauss-Jacobi method in this scenario, focusing on its properties and performance in the sparse systems generated by the finite element method.

Sumário

Capítulo 1

Transformando uma EDO Genérica em um Sistema Matriz-vetor

1.1 Definição da Formulação Forte

Dada uma função $f : [0,1] \to \mathbb{R}$ e constantes $\alpha > 0$, $\beta \geq 0$ e $\gamma \geq 0$, queremos encontrar a função $u : [0,1] \to \mathbb{R}$ tal que:

$$(S) = \begin{cases} -\alpha u_{xx}(x) + \beta u(x) + \gamma u_x(x) = f(x) \\ \\ u(0) = u(1) = 0 \end{cases}$$

Sendo (S) conhecido como a formulação forte do problema.

1.2 Transição da Formulação Forte para a Formulação Fraca

Nesse momento iremos realizar uma série de manipulações algébricas para deixá-la em um formato mais próximo de uma formulação fraca, que é mais adequada para análise teórica e para implementação numérica.

Visto que $-\alpha u_{xx}(x) + \beta u(x) + \gamma u_x(x) = f(x)$, podemos multiplicar ambos os lados por uma função $v(x)$ tal que $v(1) = v(0) = 0$ que nos ajude a eliminar a segunda derivada u_{xx}:

$$-\alpha u_{xx}(x) + \beta u(x) + \gamma u_x(x) = f(x)$$

$$\left[-\alpha u_{xx}(x) + \beta u(x) + \gamma u_x(x)\right] v(x) = f(x)v(x)$$

$$-\alpha u_{xx}(x)v(x) + \beta u(x)v(x) + \gamma u_x(x)v(x) = f(x)v(x)$$

$$\int_0^1 \left[-\alpha u_{xx}(x)v(x) + \beta u(x)v(x) + \gamma u_x(x)v(x)\right] dx = \int_0^1 f(x)v(x)dx$$

$$\int_0^1 -\alpha u_{xx}(x)v(x)dx + \int_0^1 \beta u(x)v(x)dx + \int_0^1 \gamma u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx$$

Sabemos que dadas funções f e g:

$$\int f(x)g'(x)dx = f(x)g(x) - \int f'(x)g(x)dx$$

Logo, realizando a integração por partes no primeiro termo na equação:

$$-\alpha \left[u_x(x)v(x)\Big|_0^1 - \int_0^1 u_x(x)v_x(x)dx\right] + \int_0^1 \beta u(x)v(x)dx + \int_0^1 \gamma u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx$$

$$-\alpha \left[(u_x(1)v(1) - u_x(0)v(0)) - \int_0^1 u_x(x)v_x(x)dx\right] + \int_0^1 \beta u(x)v(x)dx + \int_0^1 \gamma u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx$$

$$-\alpha \left[\cancel{(u_x(1)v(1) - u_x(0)v(0))} - \int_0^1 u_x(x)v_x(x)dx\right] + \int_0^1 \beta u(x)v(x)dx + \int_0^1 \gamma u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx$$

$$\alpha \int_0^1 u_x(x)v_x(x)dx + \beta \int_0^1 u(x)v(x)dx + \gamma \int_0^1 u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx$$

1.3 Definição da Formulação Fraca

Definição: Pelo restante do documento, considere que uma função g é "suficientemente suave" se ela respeitar:

- g é contínua em todo o domínio;
- g possui derivadas contínuas até a ordem necessária;

Seja H um espaço de funções formado por funções u suficientemente suaves que satisfazem (W) e as condições de contorno $u(0) = u(1) = 0$. Seja V um espaço das funções de teste, composto por funções v suficientemente suaves e que satisfazem as condições de contorno $v(0) = v(1) = 0$.

Dados $\alpha > 0$, $\beta \geq 0$, $\gamma \geq 0$ e uma função $f : [0,1] \to \mathbb{R}$, precisamos determinar $u : [0,1] \to \mathbb{R}, u \in H$, tal que, $\forall v \in V$,

$$(W) = \begin{cases} \alpha \int_0^1 u_x(x)v_x(x)dx + \beta \int_0^1 u(x)v(x)dx + \gamma \int_0^1 u_x(x)v(x)dx = \int_0^1 f(x)v(x)dx. \\ \\ u(0) = u(1) = 0 \end{cases}$$

1.4 Definição do Problema Aproximado pelo Método de Galerkin

O método de Galerkin consiste em aproximar o espaço das soluções por um subespaço de dimensão finita para encontrarmos uma solução aproximada que satisfaça a formulação fraca do problema dentro de um subespaço apropriado, permitindo a construção de uma solução computacionalmente viável. Sendo assim:

Seja H^h um espaço de funções finito-dimensional composto por funções u^h suficientemente suaves que satisfazem (W) e as condições de contorno $u^h(0) = u^h(1) = 0$. Analogamente, seja V^h um espaço de funções finito-dimensional das funções de teste, formado por funções v^h suficientemente suaves que também atendem às condições de contorno $v^h(0) = v^h(1) = 0$.

Precisamos determinar $u^h \in H^h$ tal que, $\forall v^h \in V^h$,

$$\int_0^1 f(x)v^h(x)dx = \alpha \int_0^1 u_x^h(x)v_x^h(x)dx + \beta \int_0^1 u^h(x)v^h(x)dx + \gamma \int_0^1 u_x^h(x)v^h(x)dx.$$

1.5 Transição do Problema Aproximado para a Forma Matriz-vetor

Visto que $u^h \in H^h$, podemos tomar u^h como combinação linear das funções da base de H^h.

Seja

$$u^h(x) = \sum_{j=1}^{m} \varphi_j(x)c_j.$$

Logo, temos que:

$$u_x^h(x) = \sum_{j=1}^{m} \varphi_{xj}(x)c_j$$

Substituindo ambos em nossa equação:

$$\alpha \int_0^1 \sum_{j=1}^{m} \varphi_{xj}(x) c_j v_x^h(x) dx + \beta \int_0^1 \sum_{j=1}^{m} \varphi_j(x) c_j v^h(x) dx + \gamma \int_0^1 \sum_{j=1}^{m} \varphi_{xj}(x) c_j v^h(x) dx = \int_0^1 f(x) v^h(x) dx$$

$$\sum_{j=1}^{m} c_j \left[\alpha \int_0^1 \varphi_{xj}(x) v_x^h(x) dx + \beta \int_0^1 \varphi_j(x) v^h(x) dx + \gamma \int_0^1 \varphi_{xj}(x) v^h(x) dx \right] = \int_0^1 f(x) v^h(x) dx$$

Veja que podemos tomar $v^h(x)$ como sendo qualquer função da base V^h.

$v^h(x) = \varphi_i(x)$, $i \in \{1, \ldots, m\}$, e $v_x^h(x) = \varphi_{xi}(x)$

$$\sum_{j=1}^{m} c_j \left[\alpha \int_0^1 \varphi_{xj}(x) \varphi_{xi}(x) dx + \beta \int_0^1 \varphi_j(x) \varphi_i(x) dx + \gamma \int_0^1 \varphi_{xj}(x) \varphi_i(x) dx \right] = \int_0^1 f(x) \varphi_i(x) dx$$

Veja que essa equação vale para $i \in \{1, \ldots, m\}$. Dessa forma, podemos expressar nosso problema na forma matriz-vetor.

1.6 Definição do Problema na Forma Matriz-Vetor

Veja que podemos escrever a equação acima como um sistema de m equações ao variarmos o valor de i:

$$\begin{cases} \sum_{j=1}^{m} c_j \left[\alpha \int_0^1 \varphi_{xj}(x) \varphi_{x1}(x) dx + \beta \int_0^1 \varphi_j(x) \varphi_1(x) dx + \gamma \int_0^1 \varphi_{xj}(x) \varphi_1(x) dx \right] = \int_0^1 f(x) \varphi_1(x) dx \\ \vdots \\ \sum_{j=1}^{m} c_j \left[\alpha \int_0^1 \varphi_{xj}(x) \varphi_{xi}(x) dx + \beta \int_0^1 \varphi_j(x) \varphi_i(x) dx + \gamma \int_0^1 \varphi_{xj}(x) \varphi_i(x) dx \right] = \int_0^1 f(x) \varphi_i(x) dx \\ \vdots \\ \sum_{j=1}^{m} c_j \left[\alpha \int_0^1 \varphi_{xj}(x) \varphi_{xm}(x) dx + \beta \int_0^1 \varphi_j(x) \varphi_m(x) dx + \gamma \int_0^1 \varphi_{xj}(x) \varphi_m(x) dx \right] = \int_0^1 f(x) \varphi_m(x) dx \end{cases}$$

E veja que podemos escrever esse sistema na forma matriz vetor:

$$\mathcal{K}\mathcal{C} = \mathcal{F}$$

$$\begin{pmatrix} \mathcal{K}_{1,1} & \ldots & \mathcal{K}_{1,j} & \ldots & \mathcal{K}_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathcal{K}_{i,1} & \ldots & \mathcal{K}_{i,j} & \ldots & \mathcal{K}_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathcal{K}_{m,1} & \ldots & \mathcal{K}_{m,j} & \ldots & \mathcal{K}_{m,m} \end{pmatrix} \begin{pmatrix} \mathcal{C}_1 \\ \vdots \\ \mathcal{C}_i \\ \vdots \\ \mathcal{C}_m \end{pmatrix} = \begin{pmatrix} \mathcal{F}_1 \\ \vdots \\ \mathcal{F}_i \\ \vdots \\ \mathcal{F}_m \end{pmatrix}$$

Tal que:

$$\mathcal{K}_{i,j} = \alpha \int_0^1 \varphi_{xi}(x)\varphi_{xj}(x)dx + \beta \int_0^1 \varphi_i(x)\varphi_j(x)dx + \gamma \int_0^1 \varphi_i(x)\varphi_{xj}(x)dx$$

$$\mathcal{F}_i = \int_0^1 f(x)\varphi_i(x)dx$$

1.7 Definindo uma Função Base

Seja $x_1, \ldots, x_m$ uma discretização do intervalo $[0,1]$ tal que $\forall i \in \{1, \ldots, m\}$, $x_{i+1} - x_i = h$. Além disso, considere $x_0 = 0$ e $x_{m+1} = 1$ como sendo os extremos do intervalo tal que $x_{m+1} - x_m = h$ e $x_1 - x_0 = h$.

Definiremos $\varphi_i(x)$ como:

$$\varphi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{h}, & \forall x \in [x_{i-1}, x_i] \\ \dfrac{x_{i+1} - x}{h}, & \forall x \in [x_i, x_{i+1}] \\ 0, & \forall x \notin [x_{i-1}, x_{i+1}] \end{cases}$$

Observe que a função base $\varphi_i(x)$ é definida como uma função linear por partes intencionalmente de forma que a sua integração e derivação sejam simples. Além disso, $\varphi_i(x)$ é projetada para ser zero fora do intervalo $[x_{i-1}, x_{i+1}]$, o que significa que cada função base afeta apenas dois ou três pontos consecutivos. Essa escolha reduz significativamente a complexidade do sistema linear, já que a matriz resultante terá muitas entradas nulas. Isso nos é benéfico pois então trabalharemos com matrizes esparsas, que são computacionalmente mais eficientes, tanto em termos de armazenamento quanto de tempo de processamento.

Perceba também que:

$$\varphi_i(x_i) = \frac{x_i - x_{i-1}}{h} = \frac{h}{h} = 1.$$

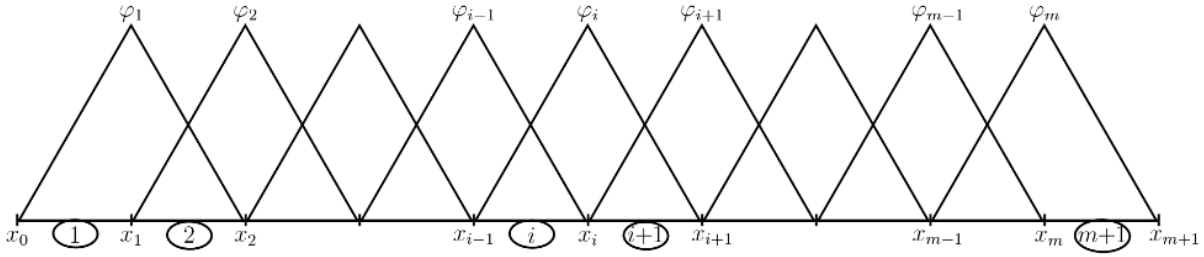

Figura 1.1: Definição das funções da base. Imagem de Bruno Alves do Carmo[1].

Capítulo 2

Resolvendo o Sistema Matriz-vetor

2.1 Tri-diagonalidade da Matriz

Vimos que, pela escolha que funções da base, grande parte dos termos da nossa matriz $\mathcal{K}$ zeram.

Veja que temos a definição de $\varphi_i(x)$ como:

$$\varphi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{h}, & \forall x \in [x_{i-1}, x_i] \\ \dfrac{x_{i+1} - x}{h}, & \forall x \in [x_i, x_{i+1}] \\ 0, & \forall x \notin [x_{i-1}, x_{i+1}] \end{cases}$$

Além disso, temos como definição da matriz $\mathcal{K}$:

$$\begin{pmatrix} K_{1,1} & \dots & K_{1,j} & \dots & K_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ K_{i,1} & \dots & K_{i,j} & \dots & K_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ K_{m,1} & \dots & K_{m,j} & \dots & K_{m,m} \end{pmatrix}, \quad i,j \in \{1, \dots, m\}$$

sendo:

$$K_{i,j} = \alpha \int_0^1 \varphi_{xi}(x)\varphi_{xj}(x)dx + \beta \int_0^1 \varphi_i(x)\varphi_j(x)dx + \gamma \int_0^1 \varphi_i(x)\varphi_{xj}(x)dx$$

Mas veja que $\varphi_i\varphi_j = 0$ quando $|i - j| > 1$, então temos que $\mathcal{K}$ é uma matriz esparsa tal que:

$$\mathcal{K} = \begin{pmatrix} K_{1,1} & K_{1,2} & 0 & 0 & 0 \\ K_{2,1} & K_{2,2} & K_{2,3} & 0 & 0 \\ 0 & K_{3,2} & K_{3,3} & \ddots & 0 \\ 0 & 0 & \ddots & \ddots & K_{m-1,m} \\ 0 & 0 & 0 & K_{m,m-1} & K_{m,m} \end{pmatrix}$$

Sendo assim, temos que resolver um sistema $\mathcal{K}\mathcal{C} = \mathcal{F}$ para $\mathcal{C}$ sendo $\mathcal{K}$ uma matriz tri-diagonal.

2.2 Métodos Iterativos

Podemos utilizar métodos iterativos semelhantes ao método do Ponto Fixo para resolver sistemas lineares se tivermos a garantia de que os autovalores da matriz são todos menores do que 1 visto que, dessa forma, o erro do método convergirá para 0 conforme provamos em [4]. Para a forma da matriz $\mathcal{K}$ específica desse problema, temos que os seus autovalores são menores do que 1 e, consequentemente, esses métodos podem ser utilizados.

Dessa forma, decidimos realizar um teste de performance ao computar o tempo gasto pelo `contra barra` da linguagem na qual implementamos o sistema dos Elementos Finitos (Julia) e outros métodos que implementamos.

Primeiramente, estamos realizando uma análise de convergência do erro para garantirmos que todos os métodos estão, de fato, dando o mesmo resultado numérico.

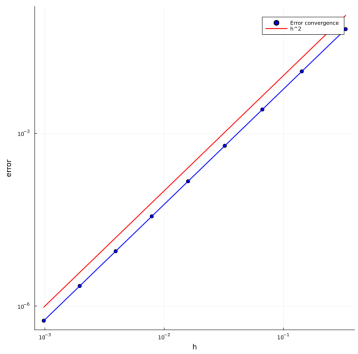

Figura 2.1: Convergência do Erro do `contra barra` do Julia.

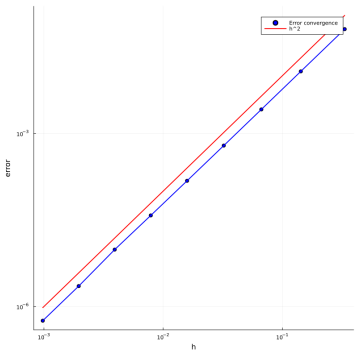

Figura 2.2: Convergência do Erro do Gauss Jacobi.

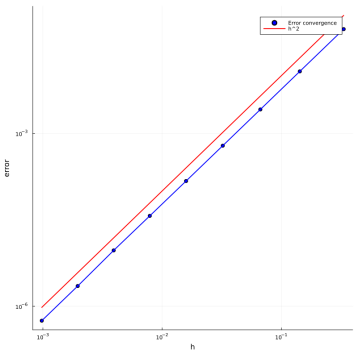

Figura 2.3: Convergência do Erro do Gauss Seidel.

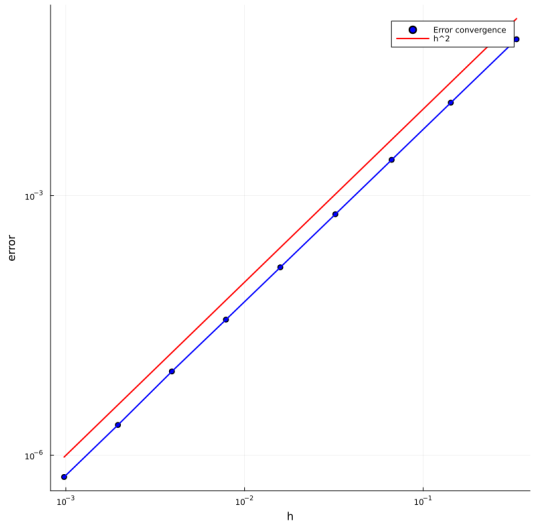

Figura 2.4: Convergência do Erro do Gauss Seidel Tri-diagonal.

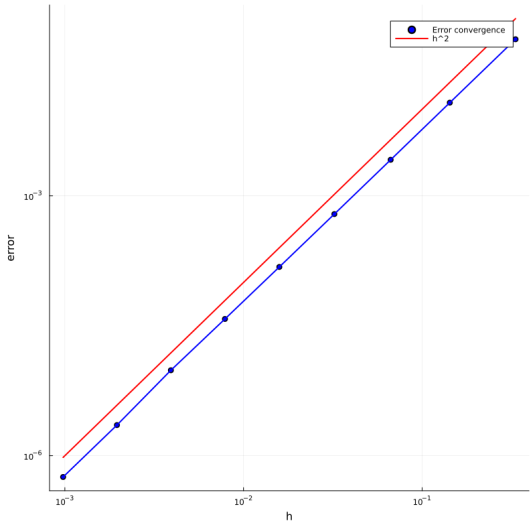

Figura 2.5: Convergência do Erro do Gauss Jacobi Tri-diagonal.

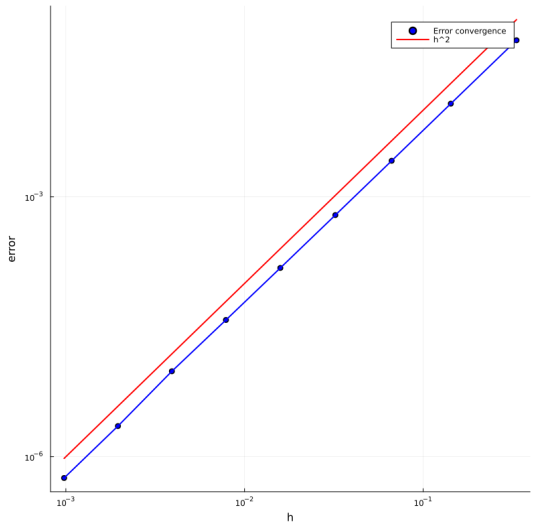

Figura 2.6: Convergência do Erro do Gauss Jacobi Paralelo.

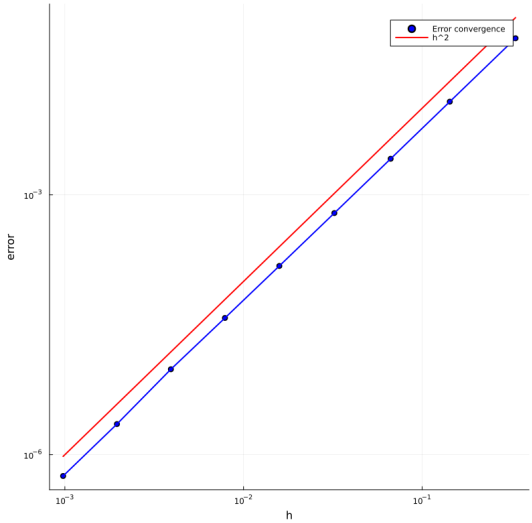

Figura 2.7: Convergência do Erro do Gauss Jacobi Tri-diagonal Paralelo.

Além disso, fizemos uma análise do tempo que alguns métodos iterativos e algumas de suas variações demoraram para montar as matrizes e resolver o sistema linear com o número de elementos variando em potências de 2 desde 2^2 a 2^{10}.

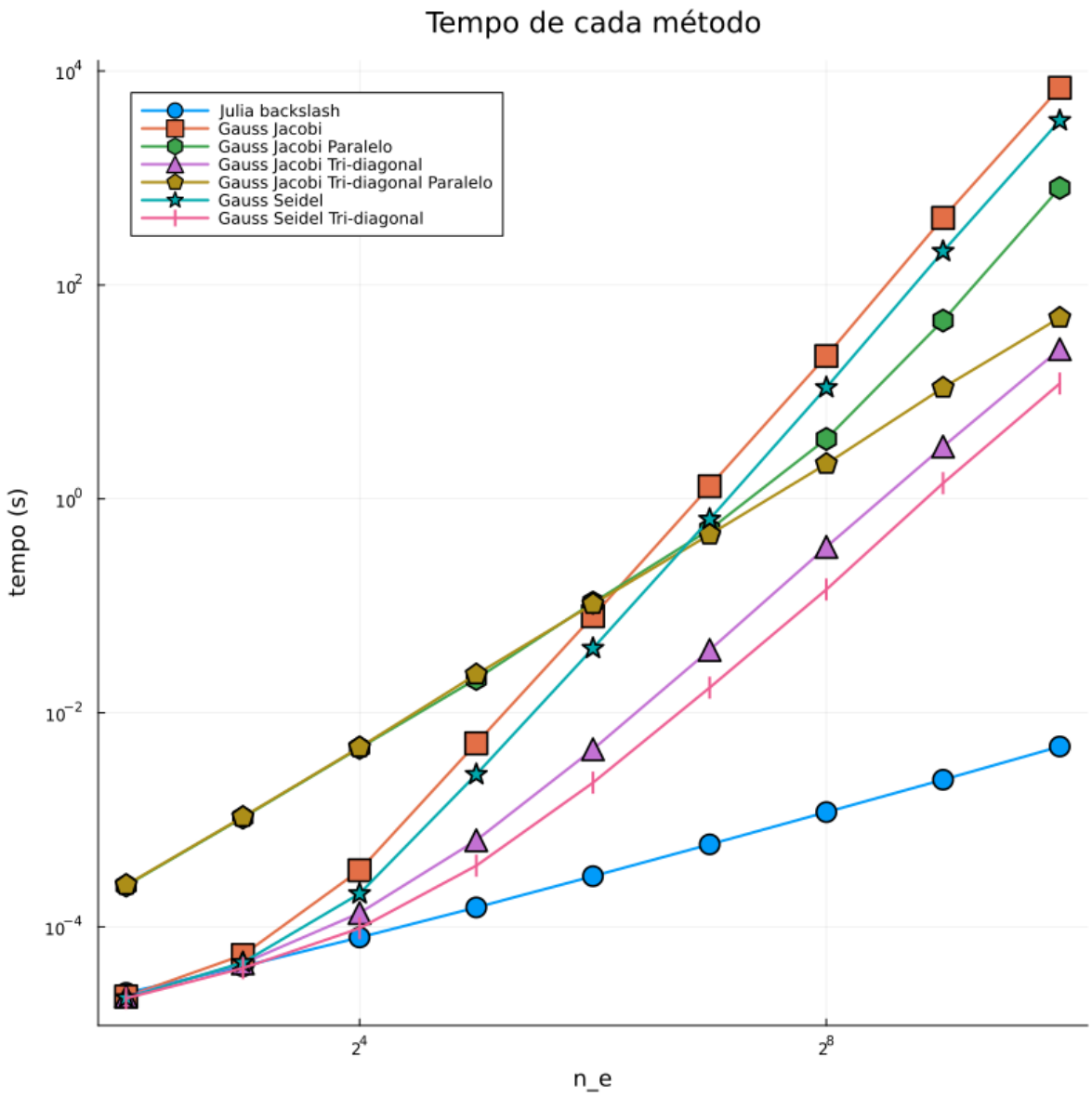

Figura 2.8: Comparação do tempo tomado por cada método.

Para números pequenos de elementos, os métodos paralelos não se mostraram eficientes devido ao alto custo inicial do paralelismo. Entretanto, ao aumentar o número de elementos para valores superiores a 64, o paralelismo começou a demonstrar maior viabilidade do que alguns outros métodos.

Uma análise interessante a ser feita é a inclinação de cada reta no gráfico, representada por $\frac{\log_{10}(tempo)}{\log_2(ne)}$. Por exemplo, o método Gauss-Jacobi Tri-diagonal Paralelo apresenta uma inclinação menor em relação aos demais métodos, o que sugere que, para valores de $ne > 2^{10}$, ele seria a melhor escolha (desconsiderando o uso do `contra barra` do Julia). No entanto, para números pequenos de elementos, este método foi o mais lento na resolução do sistema.

Outro ponto relevante é que o gráfico utiliza escala logarítmica ($\log_{10}$) no eixo yy. Essa escala evidencia a discrepância de desempenho entre os métodos aplicados a matrizes não esparsas e os métodos otimizados. Por exemplo, o Gauss-Jacobi demorou cerca de 10^4 segundos para resolver um sistema 1024×1024, enquanto o `contra barra` do Julia levou cerca de 10^{-3} segundos. Isso indica que o `contra barra` foi aproximadamente

10^7 vezes mais rápido que o método Gauss-Jacobi.

Essa diferença de desempenho reflete o fato do `contra barra` do Julia ser um método nativo e altamente otimizado, especialmente quando combinado com a biblioteca `spzeros` utilizada nos testes. Quando aplicada a uma matriz tri-diagonal, o `contra barra` ignora as regiões não inicializadas (fora das três diagonais principais), contribuindo para seu desempenho excepcional.

Por outro lado, os métodos analisados, com exceção do `contra barra`, foram implementados manualmente, sem as otimizações típicas de métodos nativos. Isso contribui significativamente para a ineficiência observada, principalmente em matrizes de maior dimensão.

Bibliografia

[1] Bruno Alves do Carmo. *Elementos Finitos*. Acessado em 7 de Setembro de 2024. URL: https://github.com/bacarmo/Elementos_Finitos.

[2] I-Shih Liu, Mauro A. Rincon e Waldecir Bianchini. *Introdução ao Método de Elementos Finitos*. Instituto de Matemática, UFRJ, p. 183. ISBN: 978-65-86502-00-8.

[3] João Victor Lopez Pereira. *Finite-Elements-Method*. Acessado em 20 de Outubro de 2024. URL: https://github.com/joaovictorlopezpereira/Finite-Elements-Method.

[4] João Victor Lopez Pereira. *Notas de Aula de Computação Científica e Análise de Dados*. Acessado em 19 de Novembro de 2024. URL: https://github.com/joaovictorlopezpereira/Notas-de-Aula-CoCADa.