



Universidade Federal de Uberlândia
Faculdade de Matemática

Licenciatura em Matemática

ESTUDO E APLICAÇÃO DO MÉTODO MULTIMALHA NA RESOLUÇÃO DE SISTEMAS LINEARES

Maria Eduarda Martins Leandro

Uberlândia-MG
2023

Maria Eduarda Martins Leandro

**ESTUDO E APLICAÇÃO DO MÉTODO
MULTIMALHA NA RESOLUÇÃO DE
SISTEMAS LINEARES**

Trabalho de Conclusão de Curso apresentado à Faculdade de Matemática da Universidade Federal de Uberlândia como requisito parcial para obtenção do título de Licenciatura em Matemática.

Orientador: Prof. Dr. Josuel Kruppa Rogenski

**Uberlândia-MG
2023**



**Universidade Federal de Uberlândia
Faculdade de Matemática**

Coordenação do Curso de Licenciatura em Matemática

A banca examinadora, conforme abaixo assinado, certifica a adequação deste trabalho de conclusão de curso para obtenção do grau de Licenciatura em Matemática.

Uberlândia, ____ de _____ de 20____

BANCA EXAMINADORA

Prof. Dr. Josuel Kruppa Rogenski

Prof. Dr. Rafael Alves Figueiredo

Prof. Dr. Santos Alberto Enriquez Remigio

**Uberlândia-MG
2023**

A minha amada mãe, aos meus padrinhos e ao meu querido namorado.

AGRADECIMENTOS

A minha querida mãe, pai, irmãos e familiares, pelo amor, apoio e incentivo constantes ao longo desta jornada, que me impulsionaram a superar desafios e acreditar em meu potencial.

Ao professor Dr. Josuel Kruppa Rogenski, pela orientação valiosa, paciência e conhecimento compartilhado. Sua dedicação à minha formação acadêmica foi fundamental para o sucesso deste trabalho.

Aos meus amigos e colegas de curso, pelas discussões construtivas, troca de ideias e pelo apoio mútuo durante todo o processo.

Ao professor Rafael Alves Figueiredo que sempre esteve disposto a me auxiliar, sanando as eventuais dúvidas.

Ao corpo docente da FAMAT, pelos ensinamentos durante a graduação, em especial a Ana Cláudia Molina Zaqueu Xavier.

E, por último, mas não menos importante, dedico este trabalho ao meu namorado que durante todo o processo esteve ao meu lado me apoiando com palavras de incentivo.

Que este trabalho possa contribuir de alguma forma para o avanço do conhecimento na área da Matemática Aplicada e que sirva como um testemunho do meu comprometimento com a busca pelo saber.

RESUMO

A busca por soluções de sistemas lineares aparece em diversos contextos, como no ramo da otimização, teoria de controle, tomografia computacional e no estudo de problemas da dinâmica dos fluidos. Fenômenos físicos ligados à dinâmica dos fluidos são geralmente modelados através de equações diferenciais, que muitas vezes não possuem solução analítica conhecida. Em particular, é frequente a necessidade de se obter soluções para equações diferenciais parciais elípticas, como a equação de Poisson ou a equação de Laplace. A busca por uma solução numérica para esse tipo de equação demanda a solução de sistemas lineares esparsos, de grande dimensão e nem sempre com estrutura bem definida, que usualmente também necessitam ser resolvidos numericamente. Nesse contexto, o presente trabalho tem como principal fonte de estudo o desenvolvimento e a implementação de métodos multimalha objetivando verificar suas características e avaliar sua eficiência na resolução de sistemas. Essa técnica é muito utilizada para acelerar a convergência dos sistemas de equações ao explorar a vantagem que alguns métodos numéricos possuem na efetiva suavização de altas frequências do erro em diferentes níveis de refinamento. Métodos multimalha em ciclo V composto por dois, três ou quatro níveis são utilizados para resolver sistemas lineares esparsos decorrentes da discretização da equação de Laplace e Poisson via diferenças finitas de segunda ordem de precisão. Análises comparativas de tempo de execução e número de ciclos necessários foram realizadas e mostram o potencial do uso da técnica como acelerador de convergência para problemas de grande dimensão.

Palavras-chave: métodos multimalha, sistemas de equações esparsos, equação de Poisson, solução de sistemas lineares.

ABSTRACT

The search for solutions to linear systems appears in different contexts, such as in the field of optimization, control theory, computational tomography, and the study of fluid dynamics problems. Physical phenomena related to fluid dynamics are often modeled using differential equations, which often lack a known analytical solution. In particular, there is a frequent need to obtain solutions for elliptic partial differential equations, such as the Poisson equation or the Laplace equation. The search for a numerical solution to this type of equation requires solving sparse linear systems of large dimension and not always with well-defined structure, which usually also need to be solved numerically. In this context, the present work focuses on the development and implementation of multigrid methods to investigate their characteristics and evaluate their efficiency in solving systems. This technique is widely used to accelerate the convergence of equation systems by exploiting the advantage that some numerical methods have in effectively smoothing high-frequency errors at different refinement levels. Multigrid methods in a V-cycle composed of 2, 3, or four levels are used to solve sparse linear systems resulting from the discretization of the Laplace and Poisson equations via second-order accurate finite differences. Comparative analyses of runtime and the number of cycles needed were conducted and demonstrate the potential use of the technique as a convergence accelerator for large-scale problems.

Keywords: multigrid methods, sparse equation systems, Poisson equation, solution of linear systems.

SUMÁRIO

Lista de Figuras	I
Lista de Tabelas	II
1 INTRODUÇÃO	1
2 EQUAÇÃO DIFERENCIAL PARCIAL	4
2.1 Classificação das equações diferenciais parciais	4
2.1.1 Classificação das equações diferenciais parciais de 2 ^a ordem	5
2.2 Exemplos de equações diferenciais parciais bidimensionais	5
2.2.1 Equação do calor	5
2.2.2 Equação da onda	6
2.2.3 Equação de Poisson	6
2.2.4 Equação de Laplace	7
3 SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS PARCIAIS ELÍPTICA	8
3.1 Discretização	9
3.1.1 Métodos numéricos	10
3.1.2 Métodos analíticos	14
3.2 Condições auxiliares	15
3.2.1 Condições iniciais	15
3.2.2 Condições de contorno	15
3.3 Solução de sistemas lineares	16
3.3.1 Métodos diretos	16
3.3.2 Métodos iterativos	19
3.3.3 Critérios de convergência para métodos iterativos	22
3.3.4 Norma	23
4 MÉTODO MULTIMALHA	24
4.1 Resíduo	25
4.2 Elementos de um método multigrid: suavizadores	26
4.3 Método de correção e ciclos	27
4.3.1 Método de Correção	27
4.3.2 Ciclos	28
4.4 Processo de restrição	29
4.5 Processo de prolongamento	31
5 MATRIZ DO PROBLEMA	33
5.1 Soluções de sistemas de equações	34

6	RESULTADOS E DISCUSSÕES	37
6.1	Preliminares	37
6.2	Resultado do problema	38
6.3	Análise do processo de suavização do erro	39
6.4	Aplicação do método multimalha	40
6.5	Speedup	42
6.6	Solução da equação de Poisson	44
7	CONCLUSÕES	48
	Referências Bibliográficas	50
	Apêndice A Apêndice	52
A.1	Operadores	52
A.1.1	Resíduo	52
A.1.2	Operador de Restrição	52
A.1.3	Operador de Prolongação	53
A.2	Gauss-Jacobi	54
A.2.1	Multigrid com Gauss - Jacobi	56
A.3	Gauss-Seidel	63
A.3.1	Multigrid com Gauss - Seidel	65

LISTA DE FIGURAS

3.1	Etapas para obtenção da solução numérica. Fonte: A Autora baseado em Fortuna (2000, p.31)[8].	9
3.2	Discretização do domínio de uma região contínua. Fonte: A Autora baseado em Fortuna (2000)[8].	10
3.3	Representação unidimensional da malha uniforme. Fonte: A Autora baseado em Almeida (2017) [7].	11
3.4	Representação bidimensional da malha uniforme. Fonte: A Autora baseado em Almeida (2017)[7].	11
3.5	Representação unidimensional da malha não uniforme. Fonte: A Autora.	12
3.6	Representação bidimensional da malha não uniforme. Fonte: A Autora.	12
4.1	Número de ondas sobre uma malha fina com $N = 12$ pontos e sobre uma malha grossa com $N = 6$ pontos (BRIGGS et al., 2000, p. 32).	25
4.2	Representação gráfica da aproximação inicial da solução. Fonte: (Briggs et al., 1999)[3].	27
4.3	Esquema de representação do ciclo V. Fonte: Franco[9].	28
4.4	Esquema de representação do ciclo F. Fonte: Franco[9].	28
4.5	Esquema de representação do ciclo W. Fonte: Franco[9].	29
4.6	Operação de restrição para o caso unidimensional. Fonte: Franco [9]	29
4.7	Operação de restrição para o caso bidimensional. Fonte: Franco [9]	30
4.8	Operação de restrição para o caso meia ponderação (HW). Fonte: Franco [9].	30
4.9	Operação de restrição para o caso ponderação completa (FW), unidimensional. Fonte: Franco[9]	31
4.10	Operação de restrição para o caso ponderação completa (FW), bidimensional. Fonte: Franco[9]	31
4.11	Operação de prolongação para o caso linear. Fonte: Franco[9]	32
4.12	Operação de prolongação para o caso bilinear. Fonte: Franco[9]	32
5.1	Discretização de uma placa quadrada 2×2 com condição de contorno de $10^\circ C$	34
5.2	Discretização da região e ênfase nos pontos internos. Fonte: A Autora.	35
6.1	Ciclo V no Método Multimalha.	38
6.2	Suavização do erro pelo método Gauss-Jacobi, com 0, 1, 50 e 1000 iterações	40
6.3	Gráfico do speedup do Gauss-Jacobi.	43
6.4	Gráfico do speedup do Gauss-Seidel.	44

LISTA DE TABELAS

6.1	Comparação de métodos iterativos de Gauss-Seidel e Gauss-Jacobi	39
6.2	Análise da eficiência do método Multimalha - Gauss-Jacobi.	41
6.3	Análise da eficiência do método Multimalha - Gauss-Seidel.	42
6.4	Speedup do Gauss-Jacobi.	43
6.5	Speedup do Gauss-Seidel.	43
6.6	Ordem do Gauss-Jacobi.	46
6.7	Ordem do Gauss-Seidel.	46
6.8	Speedup do Gauss-Jacobi com método multimalha de 4 níveis.	46
6.9	Speedup do Gauss-Seidel com método multimalha de 4 níveis.	47

1. INTRODUÇÃO

Há problemas que estão presentes em nosso cotidiano que podem ser modelados como um sistema de equações lineares. Métodos para a solução de sistemas lineares podem ser classificados em métodos diretos e métodos iterativos.

Para identificar o melhor método a ser utilizado, é preciso levar em consideração questões como a sensibilidade do sistema e do método na propagação de erros de arredondamento e a estrutura do sistema de forma a melhor administrar a memória do computador a ser alocada. Matrizes esparsas são aquelas que possuem o número de elementos nulos na ordem de 90% do total e podem ser alocadas em estruturas especiais que contenham somente aqueles elementos não nulos.

Soluções analíticas para os sistemas lineares resultantes do processo de modelagem em questão são sempre desejáveis, mas a abordagem analítica é complexa e muitas das vezes impossível, uma vez que, em boa parte, esses sistemas são de grande porte, ou seja, há uma grande quantidade de incógnitas e equações.

Nesse contexto, o trabalho irá focar na utilização dos métodos iterativos. Esta técnica consiste em aplicar esquemas iterativos de forma a se obter uma solução aproximada, gerando uma sequência de vetores com soluções aproximadas a partir de um vetor aproximação inicial e, a cada iteração que vai sendo executada, espera-se que as aproximações convirjam à solução do problema. Apesar de haver vantagens na aplicação de métodos iterativos, como utilizar memória de forma eficiente e possuírem mais resistência quanto à propagação aos erros de arredondamento, há também restrições que devem ser obedecidas para garantir a convergência para o vetor solução.

A garantia de convergência de um método numérico para solução do sistema linear está associada a análise do raio espectral da matriz iteração do método numérico, que deve ser menor do que um. Corriqueiramente, essa matriz de iteração é de grande porte e realizar o cálculo do raio espectral se torna menos vantajoso. Devido a isso, busca-se analisar a condição suficiente à convergência e ela está relacionada a garantia que, para alguma norma, a norma da matriz iteração seja menor que um.

Métodos numéricos possuem convergência garantida se a matriz dos coeficientes for estritamente diagonal dominante e para qualquer norma for menor do que 1. No entanto, mesmo convergentes, esses métodos conhecidos como suavizadores possuem convergência lenta por agirem efetivamente apenas em altas frequências do erro. De forma a contornar o problema em questão e acelerar o processo de obtenção de uma solução aproximada, o problema pode ser

resolvido considerando diferentes níveis de refinamento em métodos denominados multimalha.

Fedorenko (1964)[6] foi o pioneiro a estudar a técnica que visa melhorar a taxa de convergência dos métodos iterativos tradicionais, mas a princípio seria voltado para resolver a equação de Poisson em um domínio retangular. Entretanto, com o trabalho de Brandt (1977)[2], que analisou os componentes sobre problemas de advecção e advecção-difusão, foi notória a vasta aplicação do método multigrid na área da Dinâmica Computacional de Fluidos. Logo, como fruto positivo da aplicação na área, possuímos os resultados dos trabalhos publicados por Stüben (1999)[17], Wesseling e Oosterlee (2001)[20], principalmente quando o problema modelado é de grande porte.

A ideia do método multimalha é agir quando os erros estão suaves e os métodos iterativos como Jacobi e Gauss-Seidel, se tornam ineficazes. Logo, utiliza-se de malhas auxiliares subsequentes a fim de tornar os erros oscilatórios novamente ao transportar a informação do resíduo por meio do operador de restrição para malhas menos refinadas. Posteriormente, resolve-se para cada nível de malha o sistema de equação do resíduo com o método iterativo, denominado como solvers, até que o nível desejado seja atingido. Os solvers possuem a finalidade de suavizar os erros oscilatórios e, após nivelada a restrição, a solução é corrigida por meio do operador de prolongamento.

Os algoritmos multimalha apresentam-se em dois esquemas diferentes, sendo eles CS (*Correction Scheme*) e FAS (*Full Approximation Scheme*), podendo ter estruturas na forma de ciclo V, ciclo W, ciclo F dentre outros (Briggs et al., 2000; Trottenberg et al., 2001)[4], além de se dividirem em duas classes, como multimalha geométrico e multimalha algébrico. Para a primeira, multimalha geométrico, (Wesseling e Oosterlee, 2001)[20] é explorado em problemas que envolvem geometrias regulares. Já para problemas que abrangem as geometrias irregulares (Stüben, 1999)[17], opta-se pelo multigrid algébrico.

De acordo com Zhang (1996[22], 1997[23]); Spitaleri (2000)[16] e Gupta et al. (1997)[10], o método multimalha ganha destaque em relação a eficiência computacional na resolução de problemas que envolvem equações parciais elípticas discretizadas em N pontos, com complexidade da ordem do número de pontos. A eficiência do método se deve, de acordo com Velde(1994), na combinação dos métodos iterativos em malhas com espaçamentos diferentes.

Neste trabalho serão analisados problemas em que a geometria se apresenta de forma regular e o multimalha será utilizado para resolver a equação de Laplace, que é o modelo de uma situação de distribuição de temperatura em uma placa quadrada, no qual estima-se encontrar os valores no interior do objeto de estudo.

Trottenberg et al. (2001)[19] acredita que alguns parâmetros como escolha dos suavizadores, ciclos, malhas, número de passos, operadores de restrição e prolongamento dentre outros, podem apresentar uma grande influência na taxa de convergência do algoritmo. Isso se deve ao fato de que todos os algoritmos do método tem a dependência de parâmetros que influenciam no tempo de CPU - tempo no qual o CPU consome operando, descartando o tempo de espera e saída de dados, além do tempo de execução de outros programas.

De forma geral, o principal foco é analisar a eficiência do método multimalha na resolução

de sistemas lineares resultantes do processo de discretização associado a equação de Laplace.

A estrutura deste estudo está organizado da seguinte forma: para a secção 2 é apresentado as investigações de equações diferenciais parciais. Na secção 3, apresentam-se as etapas para obtenção da solução de equações diferenciais parciais do tipo elíptica. A secção 4 é apontado a respeito do método multimalha, explicando de forma detalhada todo o seu processo, que posteriormente é definido a matriz do problema e para o ambiente de resultados e discussões é realizado a análise e inferência sobre o método estudado. Por fim, o tópico de conclusão responsável por apresentar a síntese de todo o trabalho desenvolvido.

2. EQUAÇÃO DIFERENCIAL PARCIAL

Uma Equação Diferencial Parcial (EDP) é caracterizada como sendo uma equação que contém derivadas parciais de uma função com incógnita de duas ou mais variáveis independentes. O estudo dessa teoria iniciou-se no século XVIII, nos quais os pioneiros foram Euler, d'Alembert, Lagrange e Laplace, de modo que, engloba uma ampla gama de áreas do conhecimento científico. Além do mais a sua investigação auxilia na compreensão de diversos fenômenos físicos que são descritos e modelados por meio de Equações Diferenciais Parciais.

A seguir, encontra-se a representação de uma EDP que envolve n variáveis e possui grau k .

$$F\left(x_1, x_2, \dots, x_n, u, \frac{\partial u}{\partial x_1}, \dots, \frac{\partial u}{\partial x_n}, \frac{\partial^2 u}{\partial x_1^2}, \frac{\partial^2 u}{\partial x_1 \partial x_2}, \dots, \frac{\partial^k u}{\partial x_n^k}\right) = 0 \quad (2.1)$$

Considere $x = (x_1, \dots, x_n) \in \Omega$, onde Ω é um conjunto conexo de $\mathbb{R}^n$. A função F é uma função dada e $u = u(x)$ é a função incógnita (Iorio, 1989)[11].

Devido a vasta aplicação no campo da física e ciências naturais, podemos destacar diversas áreas de estudo, tais como a acústica, a elasticidade, a eletrodinâmica, a dinâmica dos fluidos, a difusão do calor a propagação de ondas e muito mais. Isso faz com que as Equações Diferenciais Parciais (EDP) ocupem um lugar significativo nas pesquisas científicas e o estudo sobre técnicas de resoluções se tornam relevantes.

2.1 CLASSIFICAÇÃO DAS EQUAÇÕES DIFERENCIAIS PARCIAIS

As EDP's podem ser classificadas quanto a sua ordem, à linearidade e pela homogeneidade:

ORDEM – A ordem de uma equação diferencial parcial é definida pelo grau da derivada mais alta que aparece na equação.

LINEARIDADE – A linearidade de uma equação diferencial parcial é caracterizada pela dependência linear das funções desconhecidas e de suas derivadas.

HOMOGENEIDADE – A homogeneidade de uma equação diferencial parcial é estabelecida quando o termo que não depende das funções incógnitas e de suas derivadas é nulo em todas as instâncias.

2.1.1 CLASSIFICAÇÃO DAS EQUAÇÕES DIFERENCIAIS PARCIAIS DE 2ª ORDEM

As Equações Diferenciais Parciais de segunda ordem destacam-se entre as outras por serem consideradas as mais importantes. Isso se deve ao fato de que muitos dos problemas físicos encontrados na natureza são modelados por essas equações, desempenhando um papel fundamental no avanço tecnológico ao auxiliar em áreas como na otimização e, também, na engenharia civil, atuando crucialmente no projeto de estruturas como pontes, permitindo o cálculo das deformações.

Essas equações podem ser classificadas em três categorias principais: parabólicas, hiperbólicas e elípticas. A abordagem de Magalhães (2016)[12] tem suas raízes na classificação das cônicas na geometria analítica, portanto, é aplicável somente a equações diferenciais parciais lineares de segunda ordem com duas variáveis independentes. A forma geral de uma equação desse tipo se manifestam como

$$A(x, y) \frac{\partial^2 u}{\partial x^2} + B(x, y) \frac{\partial^2 u}{\partial x \partial y} + C(x, y) \frac{\partial^2 u}{\partial y^2} + D(x, y, u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}) = 0 \quad (2.2)$$

em que os coeficientes A, B, C determinam a categoria que a respectiva equação pertence, de acordo com o valor da expressão:

$$B^2 - 4AC = x$$

- Equações Hiperbólicas

$$x > 0$$

- Equações Parabólicas

$$x = 0$$

- Equações Elípticas

$$x < 0$$

2.2 EXEMPLOS DE EQUAÇÕES DIFERENCIAIS PARCIAIS BIDIMENSIONAIS

A seguir, serão apresentados exemplos de Equações Diferenciais Parciais bidimensionais, que desempenham um papel significativo na realização deste trabalho.

2.2.1 EQUAÇÃO DO CALOR

A condução de calor é um fenômeno físico que ocorre de forma com que a agitação das partículas, sejam elas atômicas e/ou moleculares, provoque uma transferência de energia entre as

mais energéticas para as menos energéticas ao serem expostas a uma colisão. Essa transferência de calor pode ser quantificada por meio das equações que calculam a quantidade de energia por unidade de tempo, que são denominadas como sendo equações das taxas de transferência de calor.

A equação do calor é um exemplo de equação diferencial parcial que se enquadra na categoria das equações parabólicas, descrevendo fenômenos difusivos. No contexto de coordenadas cartesianas e de ordem 2, a sua representação em duas dimensões (2D) assume a forma:

$$\frac{\partial \varphi}{\partial t} = \alpha^2 \left(\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} \right). \quad (2.3)$$

Dentro do contexto desta equação, a letra grega α desempenha o papel de uma difusibilidade térmica. Nesse cenário, encontramos três variáveis independentes: as coordenadas espaciais x e y e o tempo.

2.2.2 EQUAÇÃO DA ONDA

A onda se caracteriza com sendo uma perturbação que se propaga pelo espaço a partir do ponto onde é gerada, carregando consigo energia e momento linear, porém sem transportar matéria. Ao lançar uma pedra em um lago que está inicialmente em repouso, ocorrerá a formação de ondas circulares a partir da perturbação causada pelo objeto, que se espalharão em todas as direções. Elas podem necessitar ou não de um meio de propagação. As que precisam são denominadas com sendo Ondas Mecânicas e os meios de propagação são de diferentes natureza referente aos estados da matéria como o sólido, líquido ou gasoso. Para exemplificar, pode-se falar das ondas sonoras e as ondas que são produzidas em uma corda. Ademais, aquelas que não utilizam de um meio são nomeadas como Ondas Eletromagnéticas e um exemplo é a luz.

A equação da onda é o exemplo mais simples de equação diferencial parcial que se enquadra na categoria das equações hiperbólicas, logo trata-se de fenômenos ondulatórios. A sua representação em duas dimensões (2D) em coordenadas cartesianas e de 2ª ordem assume a forma:

$$\frac{\partial^2 \varphi}{\partial t^2} = v^2 \left(\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} \right). \quad (2.4)$$

Para esta equação, v denota uma constante que representa a velocidade de propagação. Assim, temos três variáveis independentes: as coordenadas espaciais x e y e o tempo t .

2.2.3 EQUAÇÃO DE POISSON

As equações de Poisson representam problemas estacionários (regime permanente), ou seja, as propriedades do fluido permanecem constantes em cada ponto ao longo do tempo. Demonstra ser de grande utilidade nos campos da eletrostática, engenharia mecânica e física. No contexto do Eletromagnetismo, a equação de Poisson é utilizada para determinar os valores do potencial

elétrico em regiões onde esse potencial é desconhecido.

A equação de Poisson é um exemplo de equação diferencial parcial caracterizada como equação elíptica, que descrevem fenômenos em regime permanente e ela pode ser definida em duas dimensões (2D) ou em três dimensões (3D). O perfil da equação em 2D toma a forma em coordenadas cartesianas como sendo:

$$\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} = f(x, y). \quad (2.5)$$

2.2.4 EQUAÇÃO DE LAPLACE

A equação de Laplace é um caso particular da equação de Poisson, em que $f(x, y) = 0$ e pode ser encontrado modelando vários problemas relacionados a astronomia, ao eletromagnetismo e também a mecânica dos fluidos, sendo frequentemente usada para descrever e modelar as funções potenciais gravitacionais, elétricas, fluidas e outras aplicações similares.

Esta é uma equação diferencial do tipo elíptica, consequentemente descreve fenômenos em regime permanente e a representação da equação em duas dimensões (2D) assume a seguinte configuração quando expressa em coordenadas cartesianas:

$$\frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} = 0. \quad (2.6)$$

3. SOLUÇÃO DE EQUAÇÕES DIFERENCIAIS PARCIAIS ELÍPTICA

A solução de Equações Diferenciais pode ser obtida por abordagens, métodos analíticos e métodos numéricos. Nos cursos de engenharia, física e áreas afins, o enfoque são resoluções de problemas em que são possíveis a obtenção das soluções analíticas. Devido a isso, existe uma gama de estudos dessas técnicas e os métodos numéricos caíram em desuso, sendo abordado apenas em disciplinas específicas como Cálculo Numérico e Algoritmos e Programação, de forma sucinta e descontextualizada.

A relevância do estudo das técnicas numéricas se evidencia em virtude da maioria dos problemas modelados apresentarem frequente ocorrência de complexidades nas condições de contorno e nas geometrias propostas. Tais complexidades resultam na extrema dificuldade, e em casos particulares, na impossibilidade, de se obter soluções analíticas para essas equações. Na busca por uma solução para uma equação diferencial é importante uma investigação a respeito da existência e unicidade de soluções, ou seja, se uma solução existir ou não e, quando existir, ela poder ser única ou não.

A abordagem do método numérico consiste em resolver equações diferenciais substituindo as derivadas nelas presentes por expressões algébricas que envolvem a função desconhecida. Diferentemente do método analítico, que permite o cálculo das variáveis dependentes em um número infinito de pontos.

As soluções de problemas que envolvem equações diferenciais da natureza elíptica vista na Eq. 2.5 e Eq. 2.6, são obtidas por meio da discretização do domínio, de modo a subdividir a região viável em malhas, seja elas estruturadas ou não estruturadas. Esse processo de discretização frequentemente emprega o método das diferenças finitas, no qual a função é definida nos vértices da malha, possibilitando a obtenção de uma solução aproximada com o uso de uma unidade computacional denominada "estêncil". Os métodos de diferenças finitas podem ser divididos em três classes, sendo ela: diferença centrada, diferença regressiva e diferença progressiva. A escolha da melhor abordagem dependerá da ordem a ser atingida e da natureza do problema. Essas aproximações por diferenças finitas são baseadas na expansão em série de Taylor, apresentada para uma função de duas variáveis na Eq. 3.1.

O exemplo ilustrativo dessa discretização na equação de Poisson, caracterizada como uma

equação diferencial elíptica de segunda ordem por diferenças finitas, pode ser encontrado em Fortuna(2000) [8], como descrito abaixo Eq. 3.1.

$$\begin{aligned}
 f(x_i + \Delta x, y_j + \Delta y) &= f(x_i, y_j) + \\
 &+ (\Delta x) \frac{\partial f}{\partial x} \Big|_{i,j} + (\Delta y) \frac{\partial f}{\partial y} \Big|_{i,j} + \\
 &+ \frac{(\Delta x)^2}{2!} \frac{\partial^2 f}{\partial x^2} \Big|_{i,j} + \frac{2\Delta x \Delta y}{2!} \frac{\partial^2 f}{\partial x \partial y} \Big|_{i,j} + \frac{(\Delta y)^2}{2!} \frac{\partial^2 f}{\partial y^2} \Big|_{i,j} + R_N,
 \end{aligned} \tag{3.1}$$

onde R_N é o resto.

Após o processo de discretização, a Equação Diferencial Parcial (EDP) previamente contínua assume uma forma discreta. Consequentemente, a tarefa de resolução da EDP evolui de uma solução única para a solução de um sistema linear composto por múltiplas equações. Para alcançar a resolução desse sistema, recorremos a métodos que podem ser classificados em dois grupos principais: métodos exatos e métodos numéricos iterativos.

Em suma, as etapas para obtenção da solução numérica são apresentadas na Fig. 3.1

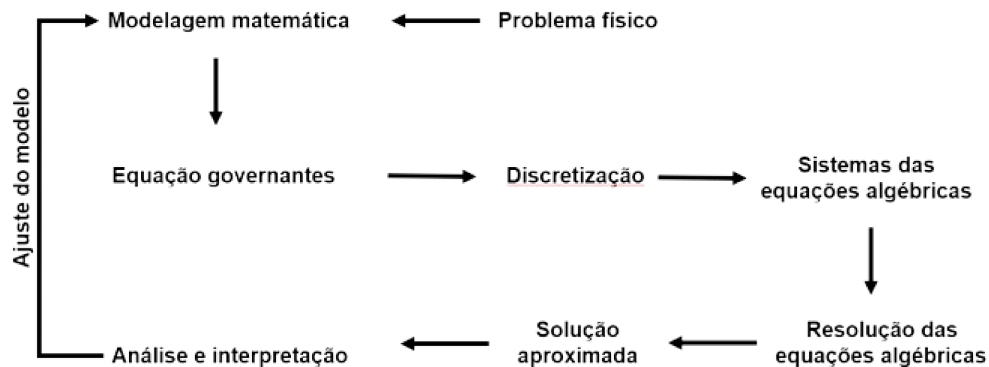


Figura 3.1: Etapas para obtenção da solução numérica. Fonte: A Autora baseado em Fortuna (2000, p.31)[8].

3.1 DISCRETIZAÇÃO

A discretização é um processo fundamental na modelagem de problemas apresentados de forma contínua. Ela dispõe da finalidade de dividir o domínio do problema em partes menores denominados como subdomínios, transformando assim equações diferenciais contínuas em equações discretas. Essa discretização pode ser realizada utilizando os Métodos Numéricos e os Métodos Analíticos. A escolha do método dependerá da natureza do problema em questão, caso seja possível encontrar a solução exata é ideal que se utilize dos métodos analíticos, já se a

solução for aproximada opta-se pelos métodos numéricos. O resultado do processo de discretização é um conjunto de expressões algébricas. Nesse momento, aborda-se o estabelecimento das condições auxiliares, que podem se manifestar como condições de contorno (PVC) ou condições iniciais (PVI) conforme a especificação do problema em foco (FORTUNA, 2000, p.30)[8].

3.1.1 MÉTODOS NÚMERICOS

Os métodos numéricos se fundamentam na aplicação finita de operações aritméticas, com o propósito de estimar a solução de um problema complexo que não pode ser resolvido de maneira analítica direta. A aproximação numérica fornece a solução em um conjunto discreto de pontos, "pontos nodais ou nós", determinado pela configuração da malha computacional conforme explicado por Mesquita (2000)[13]. A quantidade de pontos têm influência direta na qualidade da solução, uma vez que, quanto mais refinada for a malha, mais próxima a solução estará da exata, ou seja, o resultado numérico obtido será mais compatível ao modelo (FORTUNA, 2000, p. 30)[8]. Os principais métodos são MDF, MVF e MEF.

MÉTODO DE DIFERENÇAS FINITAS (MDF)

O Método de Diferenças Finitas detêm como principal objetivo converter as equações contínuas em discretas através da expansão em série de Taylor, ou seja, transformar um problema modelado por Equações Diferenciais em um conjunto de Equações Algébricas.

O primeiro passo dessa metodologia recebe o nome de discretização do domínio da variável independente. Essa etapa consiste em dividir o domínio de cálculo inicial em certas quantidades de subdomínios gerando uma malha, sendo essa uma representação do domínio discreto de $[a,b]$. Na Fig. 3.2 encontra-se uma discretização de uma região contínua R em duas dimensões.

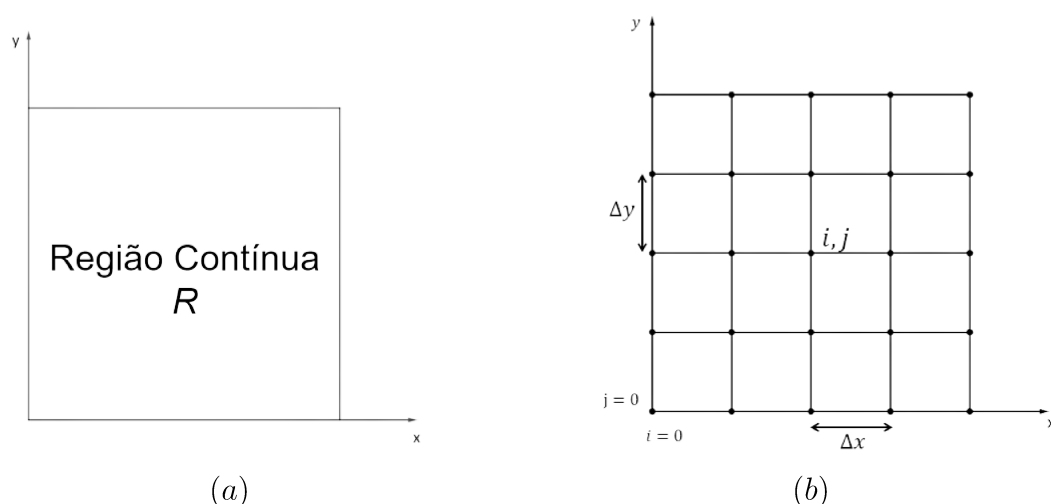


Figura 3.2: Discretização do domínio de uma região contínua. Fonte: A Autora baseado em Fortuna (2000)[8].

Os pontos que mostram na Fig. 3.2 a serem calculados estão localizados na intersecção das retas paralelas ao eixo X com as retas paralelas ao eixo Y , eles se separam a uma distância

Δx e Δy respectivamente, que não precisam ser necessariamente iguais. Cada ponto (i, j) terá coordenadas do tipo:

$$(x_0 + i\Delta x, y_0 + j\Delta y)$$

No caso acima, têm-se que para o problema em X , a origem do sistema de coordenadas é $(0, 0)$, pois a origem do sistema de coordenadas corresponde a (x_0, y_0) (FORTUNA, 2000)[8].

A escolha do tipo de malha dependerá do problema a ser resolvido. Existem dois tipos: malha uniforme e malha não uniforme.

- **Malhas Uniformes** As malhas uniformes são caracterizadas por distribuírem os pontos igualmente espaçados, tanto na direção do eixo X , quanto na do eixo Y . Neste caso, tem-se que $(\Delta x = \Delta y)$:

1. Uma dimensão - 1D

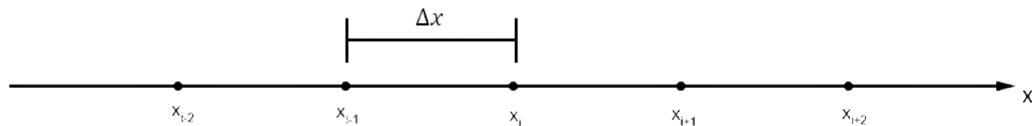


Figura 3.3: Representação unidimensional da malha uniforme. Fonte: A Autora baseado em Almeida (2017) [7].

2. Duas dimensões - 2D

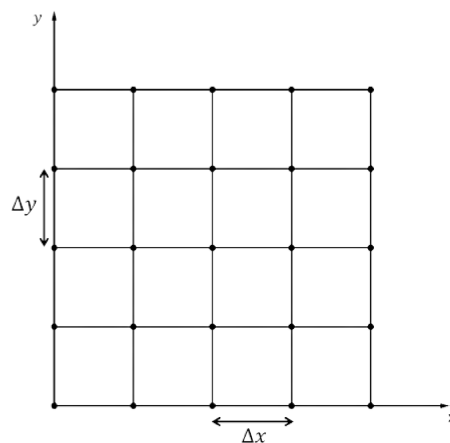


Figura 3.4: Representação bidimensional da malha uniforme. Fonte: A Autora baseado em Almeida (2017)[7].

- **Malhas Não Uniformes**

Malhas não uniformes são definidas pela distribuição não equidistante dos pontos nas direções dos eixos X e Y . Nesse contexto, as distâncias são diferentes, ou seja, $(\Delta x \neq \Delta y)$.



Figura 3.5: Representação unidimensional da malha não uniforme. Fonte: A Autora.

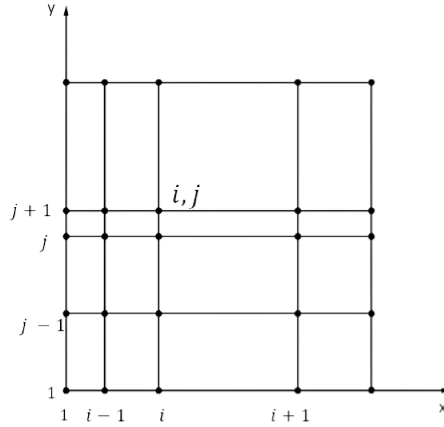


Figura 3.6: Representação bidimensional da malha não uniforme. Fonte: A Autora.

1. Uma dimensão - 1D
2. Duas dimensões - 2D

No próximo passo, a fim de encontrar a solução numérica da EDP, é preciso identificar as expressões algébricas que são escritas em função dos pontos da malha obtidos ao realizar o passo anterior, para que assim, os computadores possa fazer os cálculos de acordo com as operações aritméticas. Pois segundo Fortuna(2000, p. 74)[8]: "devemos representar os diferenciais da EDP por expressões algébricas, ou seja, discretizar a EDP", pois com as expressões algébricas, os cálculos se tornam mais fáceis.

As equações algébricas nada mais são do que equações de diferenças finitas (EDF), que quando aplicadas em cada ponto da malha que se deseja encontrar a solução do problema, geram um sistema de equações lineares. Ao resolver esse sistema que foi gerado, encontra-se uma solução aproximada para o problema inicial da EDP. Essa solução não pode ser exata por consequência da utilização de métodos baseados na aproximação, que por muitas vezes carregam erros de diferentes naturezas.

Para a discretização das derivadas parciais utiliza-se da expansão em série de Taylor, que relaciona o valor de uma função u e suas derivadas em ponto qualquer x com os valores de u nos pontos vizinhos do valor escolhido de x (CUMINATO; JUNIOR, 2013)[5]. Por exemplo, para a função u com derivadas de até ordem n , compreendida em $[a, b]$ com $x, x_0 \in [a, b]$, u em expansão por serie de Taylor será:

$$u(x) = u(x_0) + \Delta x \frac{\partial u}{\partial x} \Big|_{x_0} + \frac{(\Delta x)^2}{2!} \frac{\partial^2 u}{\partial x^2} \Big|_{x_0} + \frac{(\Delta x)^3}{3!} \frac{\partial^3 u}{\partial x^3} \Big|_{x_0} + \dots + R_n, \quad (3.2)$$

em que $\Delta x = x - x_0$ e R_n é o resto.

$$R_n = \frac{(\Delta x)^n}{n!} \frac{\partial^n u(\xi)}{\partial x^n}, \xi \in [a, b] \quad (3.3)$$

Considerando a malha unidimensional que possui pontos espaçados de maneira uniforme, conforme visto na Fig. 3.2, em que $\Delta x = x_i - x_{i-1}$, têm-se que a discretização da primeira derivada de uma função u no ponto $x_i = i\Delta x$, que pode ser rescrita como $\frac{\partial u}{\partial x}|_i$ pelo método de diferença finitas, deve-se fazer a expansão em série de Taylor da função u em torno do ponto x_i , de modo a obter a seguinte expressão:

$$u(x_i + \Delta x) = u(x_i) + \Delta x \frac{\partial u}{\partial x}\bigg|_i + \frac{(\Delta x)^2}{2!} \frac{\partial^2 u}{\partial x^2}\bigg|_i + \frac{(\Delta x)^3}{3!} \frac{\partial^3 u}{\partial x^3}\bigg|_i + \dots + R_n. \quad (3.4)$$

Isolando a primeira derivada na Eq. 3.4, chegamos em:

$$\frac{\partial u}{\partial x}\bigg|_i = \frac{u(x_i + \Delta x) - u(x_i)}{\Delta x} + \left[-\frac{\Delta x}{2!} \frac{\partial^2 u}{\partial x^2}\bigg|_i - \frac{(\Delta x)^2}{3!} \frac{\partial^3 u}{\partial x^3}\bigg|_i - \dots - R_n \right] \quad (3.5)$$

Logo, pela Eq. 3.5 a primeira derivada pode ser reescrita em termos algébricos como

$$\frac{\partial u}{\partial x}\bigg|_i = \frac{u_{i+1} - u_i}{\Delta x}.$$

Os termos que pertence a série de Taylor até o R_n são denominados como erro local de truncamento (ELT).

Esse erro estabelece a diferença entre o valor exato da derivada e sua aproximação numérica. Ele indica que, com o refinamento da malha, é possível ver que a variação dessa diferença (FORTUNA, 2000)[8].

A aproximação da derivada por diferença finita pode ser feita de três formas: progressiva, regressiva e centrada. A seguir, encontra-se exemplos de cada uma delas.

1. Diferença Progressiva

A estratégia da diferença progressiva é que, dado a discretização do domínio, a aproximação da deriva no ponto x_i fará o uso da informação do ponto subsequente, ou seja, em $x_i + \Delta x$. A seguir, um exemplo de diferença progressiva de primeira ordem para a primeira derivada:

$$\frac{\partial u}{\partial x}\bigg|_{x=x_i} = \frac{u_{i+1} - u_i}{\Delta x}.$$

2. Diferença Regressiva

A abordagem da diferença regressiva consiste em, que quando dada a discretização do domínio, aproxima-se a derivada no ponto x_i fazendo uso da informação do ponto precedente, ou seja, em $x_i - \Delta x$. A seguir, um exemplo de diferença regressiva de primeira ordem para a primeira derivada:

$$\left. \frac{\partial u}{\partial x} \right|_{x=x_i} = \frac{u_i - u_{i-1}}{\Delta x}.$$

3. Diferença Centrada

O procedimento da diferença centrada compreende-se em dada a discretização do domínio, aproximar a derivada no ponto x_i fazendo uso da informação do ponto precedente e subsequente, ou seja, em $x_i - \Delta x$ e $x_i + \Delta x$. A seguir, um exemplo de diferença centrada de primeira ordem para a primeira derivada:

$$\left. \frac{\partial u}{\partial x} \right|_{x=x_i} = \frac{u_{i+1} - u_{i-1}}{2\Delta x}$$

MÉTODO DE VOLUMES FINITOS (MVF)

Assim como no MDF, o método de volumes finitos compartilha objetivos semelhantes e eles só se caracterizam se as equações aproximadas obtidas após a discretização, forem feitas através de um balanço no volume de controle finito. Ganha destaque em problemas que envolvem a conservação de propriedades.

O processo inicial é idêntico, iniciando com a discretização da geometria do problema e construindo a malha, cada ponto (nó) gerado recebe o nome de volumes finitos (volumes de controles).

Posteriormente, deve-se escrever as ED de governo em termos de equações integrais de volume em cada volume de controle. Para a obtenção da aproximação dessas integrais, utilizam-se de métodos numéricos, como a regra do ponto médio ou a regra dos trapézios, representando as propriedades em cada volume. Gerando assim, o sistema de equações que devem ser resolvidos para obtenção da solução aproximadas da ED requerida.

MÉTODO DE ELEMENTOS FINITOS (MEF)

A discretização do corpo do problema formará subdomínios que serão denominados como elementos finitos e eles podem tomar formas triangulares. Métodos como Método Variacional, Método Direto e Métodos dos Resíduos Ponderados, auxiliam na obtenção das equações algébricas associadas as ED.

Para tornar a abordagem numérica mais apropriada, é necessário considerar a complexidade das condições de contorno associadas ao problema a ser solucionado.

3.1.2 MÉTODOS ANALÍTICOS

Os Métodos Analíticos são aplicados em problemas que envolvem ED, quando for possível a obtenção da solução exata, que por vezes são expressas como polinômios, exponenciais e outros. Alguns desses Métodos são Método de Transformada de Laplace, Método de Separação de Variáveis e Método de Serie de Potência.

3.2 CONDIÇÕES AUXILIARES

Na resolução de problemas relacionados a equações diferenciais parciais, é fundamental impor restrições específicas ao problema. Essas restrições podem se manifestar como condições de contorno, que se aplicam às fronteiras da geometria envolvida, ou como condições iniciais, que descrevem o estado inicial do processo transiente. As condições auxiliares serão então o conjunto das condições de contorno com as condições iniciais. A existência e a correta especificação dessas condições auxiliares são cruciais para garantir a solução do problema.

A seguir, encontra-se detalhado a condição inicial e três condições de contorno que são frequentemente empregadas nos problemas.

3.2.1 CONDIÇÕES INICIAIS

As condições iniciais desempenham um papel de extrema importância na resolução de problemas que envolvem equações diferenciais. Elas fornecem informações cruciais que são necessárias para determinar uma solução única e para compreender a evolução do sistema ao longo do tempo ou em um domínio específico. Caso contrário, obtém-se uma família de soluções, cada uma com valores diferentes em cada etapa do processo. Portanto, é importante que os dados iniciais sejam fornecidos de maneira precisa para todas as variáveis e em toda a geometria, juntamente com as condições de contorno. Uma variável independente será mantida constante e os valores, tanto das soluções, quanto das derivadas parciais em relação a esse parâmetro fixo, será imposta em função de outras variáveis. Sendo do tipo:

$$\begin{aligned}u(x, 0) &= f(x), \\ \frac{\partial}{\partial t}u(x, 0) &= g(x)\end{aligned}$$

de modo que, $f(x)$ e $g(x)$ são funções dadas.

3.2.2 CONDIÇÕES DE CONTORNO

Após a formulação de um sistema de equações a ser resolvido, designamos como condições de contorno ou condições de fronteira todas as especificações adicionais ou restrições que devem ser atendidas na fronteira do domínio de um problema. Tais condições são de relevância primordial para a definição integral de um problema e para garantir a obtenção de uma solução única e substancial. As principais condições são do tipo:

CONTORNO DE DIRICHLET

Nomeada em honra a Peter Gustav Lejeune Dirichlet (1805-1859), essa condição, ao ser aplicada a uma equação diferencial, estipula os valores que a solução deve assumir na fronteira do domínio da seguinte forma:

$$u(x) = f(x),$$

em que $u(x)$ é a função desconhecida e $f(x)$ é a função dos valores de u definida na fronteira do domínio.

CONTORNO DE NEUMANN

Recebe esse nome devido ao John Von Neumann (1903-1957), em que, ao ser aplicada a uma equação diferencial, seja ela ordinária ou parcial, a condição estipula os valores que a derivada da solução deve ter na fronteira do domínio, expressada como:

$$\frac{\partial u}{\partial n} = f(x),$$

sabendo que f é uma função conhecida definida no contorno que estipula o valor de u nas bordas.

CONTORNO DE ROBIN

Nomeado graças a Victor Gustave Robin (1855-1897). Em sua aplicação na equação diferencial, a condição é considerada mista por combinar a condição de contorno de Dirichlet com a de Neumann e ela pode ser escrita como sendo a combinação linear da solução com a derivada na fronteira. Elas são do tipo:

$$\alpha \frac{\partial u}{\partial n}(x) + \beta u(x) = f(x),$$

onde α e β são constantes dadas, $f(x)$ é uma função conhecida definida no contorno.

Note que a partir da condição de contorno de Robin é possível observar alguns casos especiais. Quando $\alpha = 0$, nota-se que corresponde a condição de Dirichlet. Por outro lado se $\beta = 0$ têm-se a condição de Neumann. Por fim, se $\alpha, \beta \neq 0$, temos o cenário misto, que representa justamente a condição de contorno de Robin.

3.3 SOLUÇÃO DE SISTEMAS LINEARES

Para a resolução de sistemas lineares, existem dois tipos de abordagens numéricas, sendo elas: a aplicação de Métodos Diretos ou a aplicação dos Métodos Iterativos.

3.3.1 MÉTODOS DIRETOS

São técnicas que, ao ser aplicado um número finito de operações sobre as equações do sistema linear, teoricamente levam à obtenção da solução exata. No entanto, na prática, essa solução não será precisa devido ao acúmulo de erros que podem ocorrer dependendo das características do sistema e de cargas nas operações aritméticas. Esse acúmulo acarreta soluções divergentes em relação à solução exata. Contudo, não é possível a aplicação desse método para todos os tipos de problemas que envolvam a resolução de sistemas lineares, uma vez que, os erros são inevitáveis e, para isso surge o desenvolvimento de estratégias que trazem como intuito principal

amenizar os efeitos dos erros. Os principais métodos diretos são o Método de Eliminação de Gauss (MEG) e Decomposição em LU.

OPERAÇÕES ELEMENTARES - LINHA

1. Multiplicação de uma linha por uma constante não-nula

exemplo: $L_i \leftarrow c \cdot L_i$, $c \in \mathbb{R}$, $c \neq 0$, $i = 1, 2, \dots, n$.

2. Troca de posição entre duas linhas

exemplo: $L_i \rightleftharpoons L_j$; $i, j = 1, 2, \dots, n$; $i \neq j$.

3. Adição de um múltiplo de uma linha a outra linha

exemplo: $L_i \leftarrow L_i + c \cdot L_j$, $c \in \mathbb{R}$, $c \neq 0$, $i, j = 1, 2, \dots, n$; $i \neq j$.

Matrizes equivalentes

Duas matrizes são ditas equivalentes entre si quando for possível transformar uma na outra por meio de um número finito de transformações elementares.

Sistema de equações equivalentes

São sistemas de equações equivalentes quando o vetor solução for o mesmo um do outro.

Matriz triangular

As matrizes triangulares podem ser classificadas em:

a) **Inferior**: Matriz quadrada em que todos os elementos acima da diagonal principal são nulos.

$$L = \begin{bmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & \cdots & 0 \\ l_{31} & l_{32} & l_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{bmatrix}$$

b) **Superior**: Matriz quadrada em que todos os elementos abaixo da diagonal principal são nulos.

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & \cdots & u_{2n} \\ 0 & 0 & u_{33} & \cdots & u_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{bmatrix}$$

MÉTODO DE ELIMINAÇÃO DE GAUSS (MEG)

A técnica de Eliminação de Gauss consistem em aplicar sobre as equações do sistema linear operações elementares com a finalidade de as transformarem em um sistema linear equivalente,

$$Ax = b \quad \Rightarrow \quad \tilde{A}x = \tilde{b},$$

em que $\tilde{A}x$ é uma matriz triangular superior, logo, o sistema fica fácil de ser resolvido e a solução encontrada será a mesma solução do sistema original. Devido a erros de arredondamentos da máquina operacional e o nível de sensibilidade do sistema linear a perturbações em seus dados de entrada, o resultado obtido no processo é uma aproximação. Mas cabe ressaltar que, dependendo dessas características e do método numérico utilizado, essa aproximação pode ser descartada devido a alta carga de erros. Para o método de Eliminação de Gauss, podemos separar o processo em duas fases, em que a primeira é a eliminação e a segunda é a substituição.

- 1ª FASE - Eliminação

Consiste em realizar operações elementares nas linhas da matriz aumentada de um sistema de equação do tipo $Ax = b$ para obter um sistema de equação triangular superior equivalente $\tilde{A}x = \tilde{b}$, após (n-1) passos, logo o sobrescrito representa o passo em que é realizado as operações elementares no sistema, até que ele atinge o formato de triangular superior.

A matriz ampliada do sistema $Ax = b$ se caracteriza como sendo a junção da matriz dos coeficientes A com o vetor dos termos independentes b . As operações aplicadas a essa matriz devem ser feitas respeitando uma ordem bem definida, a fim de minimizar os custos computacionais por se tratar de métodos numéricos.

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} & b_n \end{array} \right] \xrightarrow{\text{TRANSFORMAÇÃO ELEMENTAR}} \left[\begin{array}{cccc|c} a_{11}^{(0)} & a_{12}^{(0)} & a_{13}^{(0)} & \dots & a_{1n}^{(0)} & b_1^{(0)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} & b_2^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \dots & a_{3n}^{(2)} & b_3^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn}^{(n-1)} & b_n^{(n-1)} \end{array} \right]$$

- 2ª FASE - Substituição

Essa fase consiste em encontrar a solução por meio do sistema linear triangularizado no qual foi obtido na etapa anterior. Para isso, opta-se por utilizar a técnica de substituições retroativas, quando a partir de x_n vai se obtendo regressivamente todas as outras, isto é, $x_{n-1}, x_{n-2}, \dots, x_1$.

DECOMPOSIÇÃO EM LU

A Decomposição em LU consiste em dado uma matriz, transformá-la como produto de outras duas matrizes conhecidas ou mais simples, com o propósito de auxiliar nos cálculos.

$$Ax = b \quad \Rightarrow \quad LUx = b.$$

Teorema 4.2 Dada uma matriz quadrada A , de ordem n , seja A_k a matriz constituída das primeiras k linhas e colunas de A . Suponha que $\det(A_k) \neq 0$ para $k = 1, 2, \dots, (n - 1)$. Então, existe uma única matriz triangular inferior $L = (l_{ij})$, com $l_{11} = l_{22} = \dots = l_{nn} = 1$, e uma única matriz triangular superior $U = (u_{ij})$, tal que $L \cdot U = A$. Além disso, $\det(A) = u_{11} \cdot u_{22} \dots u_{nn}$.

As matrizes L e U podem ser determinadas utilizando o conceito fundamental do Método de Eliminação de Gauss supracitado. A matriz U é triangular superior, obtida na etapa final do processo, enquanto L é matriz triangular inferior em que os elementos da diagonal principal são unitários, ou seja, tem o valor sendo 1 e, os elementos abaixo são multiplicadores com o sinal trocado da K etapa da fase do método utilizado para encontrar a matriz U .

Para obtenção da solução do sistema $Ax = b$ utilizando o Método Decomposição em LU, basta executar as seguintes etapas:

- 1ª ETAPA - Obter a fatoração de A como sendo $A = L \cdot U$, que implica em resolver um sistema sendo $LUx = b$.
- 2ª ETAPA - Fazer $Ux = y$, que implicará em $Ly = b$.
- 3ª ETAPA - Resolver o sistema triangular inferior $Ly = b$.
- 4ª ETAPA - Resolver o sistema triangular superior $Ux = y$, que será a solução do sistema de equação $Ax = b$.

3.3.2 MÉTODOS ITERATIVOS

Os Método Iterativos são técnicas numéricas que consistem na aplicação de esquemas iterativos com a finalidade obter a solução do sistema de equação, gerando a partir de um vetor de chute inicial $x^{(0)}$, uma sequência de vetores $x^{(K)}$ que converge para a solução exata.

Um fato interessante dos métodos iterativos é que eles podem ser aplicados em sistemas lineares de grande porte e também em esparsos, pois conseguem economizar memória ao utilizar a estratégia de armazenamento de informações em vetores, aplicando técnicas de multiplicação de matriz - vetor.

SISTEMAS LINEARES ESPARSOS

Sistema Lineares Esparsos são aqueles no qual a matriz A dos coeficientes possuem maior quantidade elementos nulos em relação aos elementos não-nulos.

Exemplo:

$$\begin{bmatrix} a_{11} & 0 & a_{13} & \dots & 0 \\ 0 & a_{22} & 0 & \dots & 0 \\ a_{31} & 0 & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & 0 & 0 & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$

Um sistema linear do tipo $Ax = b$, em que o $\det(A) \neq 0$, $A \in \mathbb{R}^{n \times n}$ e $x, b \in \mathbb{R}^{n \times 1}$, pode ser reescrito como:

$$b - Ax = 0. \quad (3.6)$$

A Eq. 3.6 é o mesmo que resolver um problema de encontrar zeros de funções.

Dentre os métodos clássico para a resolução de sistemas lineares, citamos os métodos de Jacobi e Gauss-Seidel, ambos com finalidade de encontrar a solução aproximada do sistema linear de equações.

JACOBI

Considerando que o problema a ser resolvido seja do tipo $Ax = b$, podemos decompor a matriz A dos coeficientes de um sistema linear como sendo a soma de três, sendo elas L, U e D . A L (*Lower*) é a matriz triangular inferior que consiste exclusivamente nos elementos da matriz A localizados abaixo da diagonal principal. A matriz U (*Upper*) é triangular superior, formada apenas pela parte da matriz A acima da diagonal principal. Por fim, D (*Diagonal*) é a matriz diagonal constituída unicamente pelos elementos que compõem a diagonal da matriz A . Podendo ser apresentada da seguinte forma:

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}}_A = \underbrace{\begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ a_{21} & 0 & 0 & \dots & 0 \\ a_{31} & a_{32} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & 0 \end{bmatrix}}_L + \underbrace{\begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ 0 & a_{22} & 0 & \dots & 0 \\ 0 & 0 & a_{33} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_{nn} \end{bmatrix}}_D + \underbrace{\begin{bmatrix} 0 & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & 0 & a_{23} & \dots & a_{2n} \\ 0 & 0 & 0 & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}}_U \quad (3.7)$$

Logo

$$Ax = b \Rightarrow (L+D+U)x = b \Rightarrow Dx = -(L+U)x + b \Rightarrow x = -D^{-1}(L+U)x + D^{-1}b.$$

A solução pelo processo iterativo do Método de Jacobi será dado por:

$$x^{(k+1)} = \underbrace{-D^{-1}(L+U)}_{B_J} x^{(k)} + \underbrace{D^{-1}b}_g \quad \text{para um } x^{(0)} \text{ e } k \geq 0,$$

em que B_J é a matriz iteração de Jacobi.

Outra forma de deduzir o método é isolando o x no sistema linear. Seja $Ax = b$, um sistema possível e determinado, com n equações e incógnitas como o sistema Eq. 5.3, basta isolar x_i , com $i = 1, 2, 3, \dots, n$, e assim o método iterativo ficará da seguinte forma:

$$\begin{aligned} x_1^{(k+1)} &= \frac{b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - a_{14}x_4^{(k)} - \dots - a_{1n}x_n^{(k)}}{a_{11}}, \\ x_2^{(k+1)} &= \frac{b_2 - a_{21}x_1^{(k)} - a_{23}x_3^{(k)} - a_{24}x_4^{(k)} - \dots - a_{2n}x_n^{(k)}}{a_{22}}, \\ x_3^{(k+1)} &= \frac{b_3 - a_{31}x_1^{(k)} - a_{32}x_2^{(k)} - a_{34}x_4^{(k)} - \dots - a_{3n}x_n^{(k)}}{a_{33}}, \\ &\vdots \\ x_n^{(k+1)} &= \frac{b_n - a_{n1}x_1^{(k)} - a_{n2}x_2^{(k)} - a_{n3}x_3^{(k)} - \dots - a_{nn-1}x_{n-1}^{(k)}}{a_{nn}}, \end{aligned}$$

onde k indica a iteração anterior e $k + 1$ a iteração atual.

Logo, $x^{(k+1)} = B_J x^k + g$, no qual:

$$B_J = \begin{bmatrix} 0 & -\frac{a_{12}}{a_{11}} & -\frac{a_{13}}{a_{11}} & \dots & -\frac{a_{1n}}{a_{11}} \\ -\frac{a_{21}}{a_{22}} & 0 & -\frac{a_{23}}{a_{22}} & \dots & -\frac{a_{2n}}{a_{22}} \\ -\frac{a_{31}}{a_{33}} & -\frac{a_{32}}{a_{33}} & 0 & \dots & -\frac{a_{3n}}{a_{33}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{a_{n1}}{a_{nn}} & -\frac{a_{n2}}{a_{nn}} & \dots & -\frac{a_{n,n-1}}{a_{nn}} & 0 \end{bmatrix} \text{ e } g = \begin{bmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} \\ \frac{b_3}{a_{33}} \\ \vdots \\ \frac{b_n}{a_{nn}} \end{bmatrix}.$$

GAUSS-SEIDEL

De mesma forma ao método de Jacobi, o Método de Gauss-Seidel emprega abordagens iterativas na busca da solução de sistemas lineares fazendo do uso da partição encontrada Eq. 3.7.

$$Ax = b \Rightarrow (L + D + U)x = b \Rightarrow (D + L)x = -Ux + b \Rightarrow x = -(D + L)^{-1}Ux + (D + L)^{-1}b$$

O esquema iterativo pelo Método de Gauss- Seidel ficará da seguinte maneira:

$$x^{(k+1)} = \underbrace{-(D + L)^{-1}Ux^{(k)}}_{B_{GS}} + \underbrace{(D + L)^{-1}b}_g \quad \text{para um } x^{(0)} \text{ e } k \geq 0.$$

Ou pode ser escrito como:

$$x^{(k+1)} = -D^{-1}Lx^{(k+1)} - D^{-1}Ux^{(k)} + D^{-1}b.$$

Em notação de sistema linear, tem-se:

$$\begin{aligned}
x_1^{(k+1)} &= \frac{b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - a_{14}x_4^{(k)} - \cdots - a_{1n}x_n^{(k)}}{a_{11}}, \\
x_2^{(k+1)} &= \frac{b_2 - a_{21}x_1^{(k+1)} - a_{23}x_3^{(k)} - a_{24}x_4^{(k)} - \cdots - a_{2n}x_n^{(k)}}{a_{22}}, \\
x_3^{(k+1)} &= \frac{b_3 - a_{31}x_1^{(k+1)} - a_{32}x_2^{(k+1)} - a_{34}x_4^{(k)} - \cdots - a_{3n}x_n^{(k)}}{a_{33}}, \\
&\vdots \\
x_n^{(k+1)} &= \frac{b_n - a_{n1}x_1^{(k+1)} - a_{n2}x_2^{(k+1)} - a_{n3}x_3^{(k+1)} - \cdots - a_{nn-1}x_{n-1}^{(k+1)}}{a_{nn}},
\end{aligned}$$

onde k indica a iteração anterior e $k + 1$ a iteração atual.

Nota-se que o Método de Gauss-Seidel se beneficia do uso das informações mais atualizadas. A utilização de dados recentes torna o esquema mais eficaz, pois geralmente leva a uma convergência mais rápida. No entanto, é importante ressaltar que a convergência não é garantida e depende da satisfação dos critérios de convergência.

Embora o método seja eficaz, ele sofre influência sobre a escolha inicial das estimativas, ou seja, dependendo do chute inicial, o método poderá divergir da solução esperada. Por isso, para a seleção do método apropriado, deve-se levar em conta as características do sistema linear a ser resolvido.

3.3.3 CRITÉRIOS DE CONVERGÊNCIA PARA MÉTODOS ITERATIVOS

É possível estabelecer critérios que permitem determinar se a solução do sistema linear após a aplicação do método iterativo, irá convergir para a solução esperada. Seja $x^{(0)} \in \mathbb{R}^n$ condição inicial para o método iterativo, calcula-se:

$$x^{(k+1)} = Bx^{(k)} + g; \quad k \geq 0 \quad (3.8)$$

em que $B \in \mathbb{R}^{n \times n}$ é a matriz iteração e $g \in \mathbb{R}^n$. Dado uma sequência $x^{(k)}$ obtida pela Eq. 5.3, têm-se que a sequência x^k converge para x se:

$$\lim_{k \rightarrow \infty} x^{(k)} = x$$

A condição tanto necessária, quanto suficiente para a convergência do processo iterativo definido é de que

$$\rho(B) = \max_{i=1, \dots, n} |\lambda_i| < 1$$

em que λ representa os autovalores da matriz B e ρ é o raio espectral.

Pelo critério geral temos como condição suficiente de convergência que para qualquer norma de matrizes

$$\|B\| < 1.$$

Neste trabalho não será apresentado a demonstração, mas elas podem ser encontradas em [1].

3.3.4 NORMA

Para calcular a distância entre dois vetores ou entre duas matrizes, utiliza-se das normas. Elas são aplicadas como critério de parada dos métodos iterativos, de modo que, fornece a informação da proximidade do valor estimado $\tilde{x}$ do valor exato x denominado como erro.

Define-se norma de vetor $x \in \mathbb{R}^n$, adotando a simbologia $\|x\|$, um número real que satisfaz as seguintes propriedades:

- $\|x\| \geq 0$ e $\|x\| = 0$ se, e somente se, $x = 0$ (vetor nulo);
- $\|\lambda x\| = |\lambda| \|x\|$, $\forall \lambda \in \mathbb{R}$;
- $\|x + y\| \leq \|x\| + \|y\|$ (desigualdade triangular).

A seguir, encontra-se alguns exemplos de normas de vetores:

1. Norma infinita ou norma do máximo:

$$\|x\|_{\infty} = \max_{1 \leq i \leq n} |x_i|;$$

2. Norma 1 ou norma da soma dos módulos:

$$\|x\|_1 = \sum_{i=1}^n |x_i|;$$

3. Norma 2 ou norma euclidiana:

$$\|x\|_2 = \|x\| = \sqrt{\sum_{i=1}^n x_i^2}.$$

EQUIVALÊNCIA DE NORMA

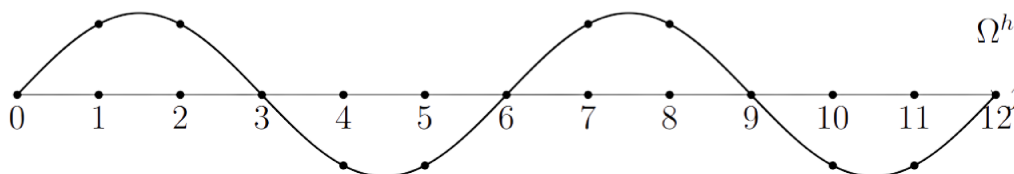
Dizemos que duas normas $\|\cdot\|_a$ e $\|\cdot\|_b$ são equivalentes se existem constantes k_1 e k_2 tais que

$$k_1 \|x\|_a \leq \|x\|_b \leq k_2 \|x\|_a, \quad \forall x \in \mathbb{R}^n.$$

4. MÉTODO MULTIMALHA

O método Multimalha originalmente proposto por Fedorenko (1964)[6], pertence ao grupo dos métodos iterativos. Assim como os outros supracitados no capítulo anterior, possui como finalidade a obtenção da solução aproximada do sistema linear. Esse método pode ser aplicado nos sistemas $Ax = b$ obtidos após o processo de discretização do problema inicialmente contínuo do tipo $\nabla^2 u = f$. Na busca por uma solução numérica para equação diferencial elíptica (vista como modelo para um problema real a ser discretizado), quanto mais refinada for a malha, a solução estará mais próxima da exata. Isso gera sistemas de equação muito grandes e torna-se necessário a utilização de métodos computacionais eficientes. Métodos multimalha são desenvolvidos de forma a acelerar a convergência de esquemas iterativos aplicados, mantendo o custo operacional ou até reduzindo consideravelmente ao efetuar menos iterações. Com isso, demonstra-se mais eficiente na velocidade de convergência do que os métodos iterativos aplicados puramente sem fazer do uso de varias malhas.

O objetivo principal do multimalha é acelerar a convergência (TANNEHILL et al., 1997)[18] e a sua dinâmica consiste em aplicar métodos iterativos básicos como os métodos de Jacobi e Gauss-Seidel no problema $Ax = b$ para encontrar a aproximação inicial da solução do problema. Essa etapa é denominada como relaxação, que nada mais é do que utilizar de suavizadores com a finalidade aproximar a solução diminuindo a oscilação dos erros. Os métodos iterativos do tipo suavizador - como os métodos de Jacobi e Gauss-Seidel irão agir de forma eficiente até que a frequência dos erros se tornem suaves, pois, as primeiras iterações obtidas com a aplicação dos métodos geram muitas oscilações do erro. Com o passar do tempo, a convergência que antes eram rápidas, vão se tornando lentas e, conseqüentemente, a suavidade nos espectro do erro que torna a convergência lenta ficam suaves. A essência de método multimalha está relacionada a um transferência do problema do domínio original para um novo, onde os métodos iterativos escolhidos possam atuar de forma mais efetiva. Ao fazer essa transferência dos componentes do erro suavizado para a malha mais grossa, ele se torna mais oscilatório e os métodos iterativos mostram-se eficientes novamente, como mostra a Fig. 4.1.



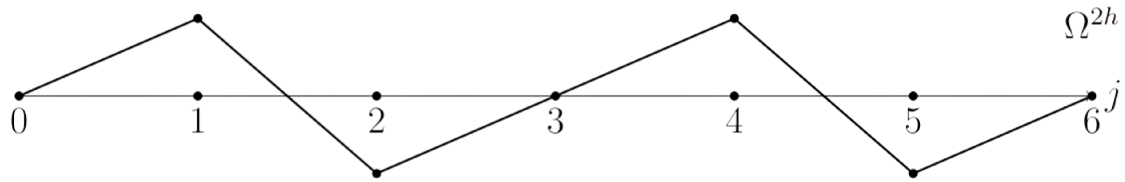


Figura 4.1: Número de ondas sobre uma malha fina com $N = 12$ pontos e sobre uma malha grossa com $N = 6$ pontos (BRIGGS et al., 2000, p. 32).

Mostrando como o multimalha age, transformando os modos suaves em oscilatórios ao submetê-los em malhas grosseiras.

A solução do novo sistema linear gerado pela restrição sofrerá o efeito de prolongamento e será acrescido na solução inicial, a fim de corrigir os erros gerados nas malhas mais finas. O processo pode ser repetido até atingir a malha adequada. Portanto, resume-se o esquema multimalha em dois níveis do tipo CS (*Correction Scheme*):

1. Considerando a discretização em uma malha ω^h , obtém-se a solução aproximada inicial aplicando os métodos iterativos.
2. Calcula-se o resíduo e aplica-se a restrição em uma malha mais grossa, por exemplo ω^{2h} .
3. Utiliza-se dos suavizadores para obtenção da solução do novo sistema linear gerado.
4. A solução proveniente da etapa anterior sofre o prolongamento retornado para a malha inicial. Após a interpolação, a primeira solução é acrescida da obtida na presente etapa, resultando na nova aproximação do problema após correções do erro de baixa frequência.

4.1 RESÍDUO

Considere um sistema $Ax = b$ com x sendo a solução exata e $\tilde{x}$ a solução aproximada. Temos que o resíduo r é dado por:

$$r = b - A\tilde{x}.$$

O erro e é definido como $e = x - \tilde{x}$. Têm-se que

$$e = x - \tilde{x} \Rightarrow Ae = Ax - A\tilde{x} \Rightarrow Ae = b - A\tilde{x} \Rightarrow Ae = r. \quad (4.1)$$

Dessa forma, a matriz que relaciona termo independente e solução do sistema é a mesma que relaciona resíduo e erro ao ser aplicado o processo iterativo.

4.2 ELEMENTOS DE UM MÉTODO MULTIGRID: SUAVIZADORES

O multimalha desempenha um papel muito importante, influenciando no comportamento de convergência dos métodos iterativos estacionários como Jacobi e Gauss - Seidel. Esses métodos são destaques na eliminação dos componentes dos erros de alta frequência, no entanto, revelam-se menos eficazes na remoção dos componentes suaves. Portanto, é por esse motivo que recebem o nome de suavizadores, atuando na relaxação dos componentes em frequências altas. Pelo fato de não serem eficientes em baixas frequências, opta-se pela correção em malhas (grids) mais grosseiras.

Quanto ao comportamento dos erros, Briggs et al. (2000)[4] afirma que empregar o sistema homogêneo $Au = 0$ é suficiente. Como a solução exata para o problema ($u = 0$) é conhecida, o erro da aproximação fornecida por v é simplesmente $-v$. Com o intuito de simplificar os cálculos, considere o cenário unidimensional da equação de Poisson com condições de contorno de Dirichlet e uma estimativa inicial, que pode ser representada em termos de modos. Utiliza-se dos modos de Fourier representados por

$$v_j^{(0)} = \sin\left(\frac{jk\pi}{n}\right) \quad 0 \leq j \leq n, \quad 1 \leq k \leq n-1.$$

em que k denota a frequência, ou número de ondas, indicando a metade do número de períodos da função seno no domínio do problema, $v_j^{(0)}$ indica a j -ésima componente do vetor $v^{(0)}$. Com k indicando quantas meia-senóides existem no domínio, temos algumas definições como

- Se $1 \leq k \leq N/2$, os modos de Fourier são denominados **suaves**.
- Se $N/2 \leq k \leq N-1$, os modos de Fourier são denominados **oscilatórios**.

Na fig. 4.2 encontra-se a representação gráfica da aproximação inicial da solução.

No geral, médias e altas frequências do erro estão relacionadas a autovalores próximos a zero em métodos iterativos do tipo suavizador o que torna a aplicação do suavizador eficiente na eliminação desses componentes de erro.

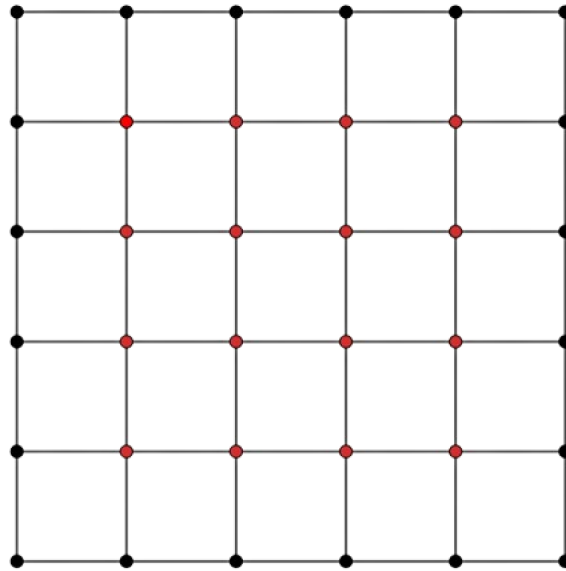


Figura 4.2: Representação gráfica da aproximação inicial da solução. Fonte: (Briggs et al., 1999)[3].

4.3 MÉTODO DE CORREÇÃO E CICLOS

4.3.1 MÉTODO DE CORREÇÃO

O método multigrid pode ser utilizado com dois tipos de sistemas de correção (Briggs et al., 2000)[4], sendo eles: o esquema de correção (*Correction Scheme, CS*) e o esquema de aproximação completa (*Full Approximation Scheme, FAS*).

CS

O esquema de correção CS define-se em resolver a equação $Ax = b$ na malha mais fina e o resíduo gerado por essa mesma equação é transferido para a malha grossa subsequente e a equação residual $Ae = r$ é resolvida. Na malha grosseira, o erro é interpolado com a finalidade de corrigir a aproximação da malha fina consecutiva. De acordo com (Briggs et al., 2000)[4], o esquema CS é mais indicado para problemas lineares.

FAS

Em um esquema de correção FAS, além do resíduo, transfere-se para a malha mais grossa próxima a aproximação da solução obtida da equação $Ax = b$, nessa mesma malha é solucionado equações com aproximações para a solução e não a equação residual como é feito no esquema CS. Posteriormente, ocorre a interpolação da correção e o resultado é usado para aproximar a solução na malha fina subsequente.

4.3.2 CICLOS

Wesseling (1992)[21], Briggs et al. (2000)[4] e Trottenberg et al. (2001)[19] apresentam-se como parte da família dos ciclos nomeado como μ -ciclo a generalização dos ciclos V , F e W . Esses ciclos de suavização são caracterizados pelos percursos entre as malhas, logo abaixo encontra-se o perfil de cada um dos modelos citados anteriormente.

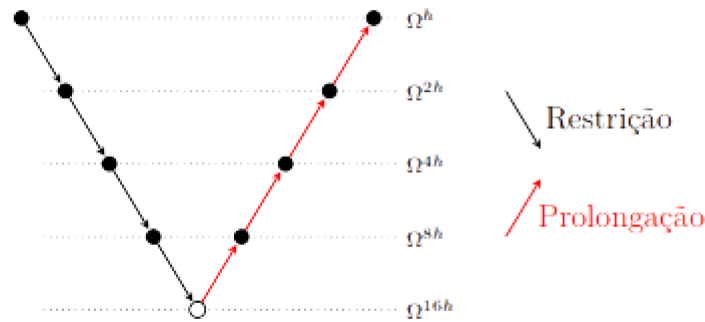


Figura 4.3: Esquema de representação do ciclo V. Fonte: Franco[9].

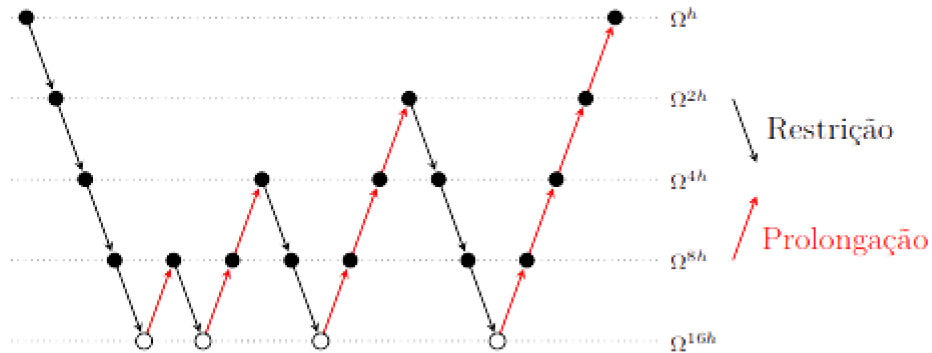


Figura 4.4: Esquema de representação do ciclo F. Fonte: Franco[9].

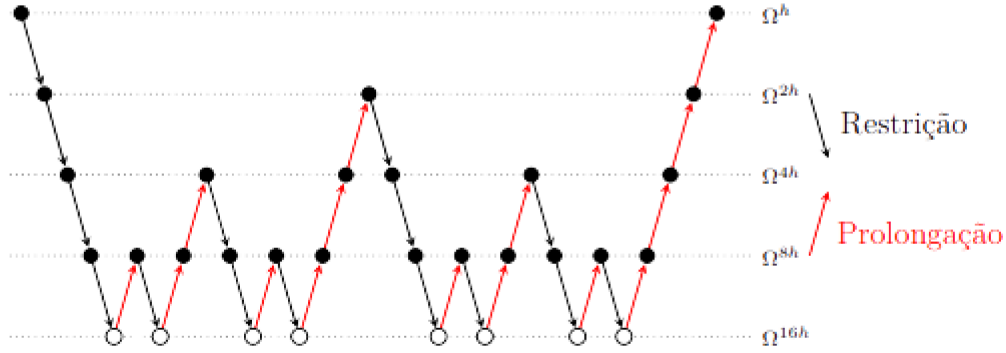


Figura 4.5: Esquema de representação do ciclo W. Fonte: Franco[9].

4.4 PROCESSO DE RESTRIÇÃO

A dinâmica desse processo é que a partir de um operador de restrição (I_h^{2h}) ocorre a transferência da informação de uma malha mais fina ω^h , que tem discretização de malhas de espaçamento h em todas as direções espaciais, para uma malha mais grossa ω^{2h} , com $2h$. A técnica mais simples é a injeção (Inj), mas existem outras como a meia ponderação/*half weighting* (HW) e ponderação completa/*full weighting* (FW) (BRIGGS et al., 2000; TROTTERBERG et al., 2001; HORTON; VANDEWALLE, 1995).

1. Injeção (Inj)

O sistema de injeção faz com que o operador $(I_h^{2h})_{Inj}$ transfira os valores da malha mais fina, que estão na mesma posição da malha mais grossa, para a última citada. Para casos 1D representado pela Fig. 4.6 e 2D pela Fig. 4.7, podem ser vistos de forma geométrica como abaixo

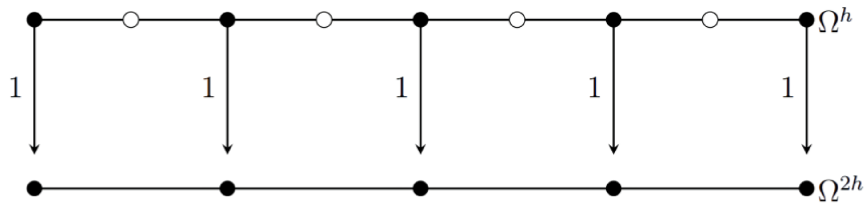


Figura 4.6: Operação de restrição para o caso unidimensional. Fonte: Franco [9]

Considerando o caso 2D, o resíduo r_{2h} calculado pelo operador $(I_h^{2h})_{Inj}$ na malha mais grossa é dado por

$$\begin{aligned} r_{2h}(x, y) &= (I_h^{2h})_{Inj} r_h(x, y) \\ &= r_h(x, y). \end{aligned} \tag{4.2}$$

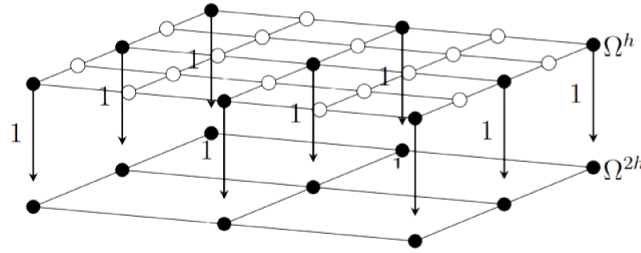


Figura 4.7: Operação de restrição para o caso bidimensional. Fonte: Franco [9]

2. Meia Ponderação (HW)

O operador de meia ponderação $(I_h^{2h})_{HW}$ no caso 2D, transporta as informações dos quatro pontos mais próximos ao ponto calculado na grade mais fina para o ponto correspondente na grade mais grossa, como pode ser visto na Fig. 4.8

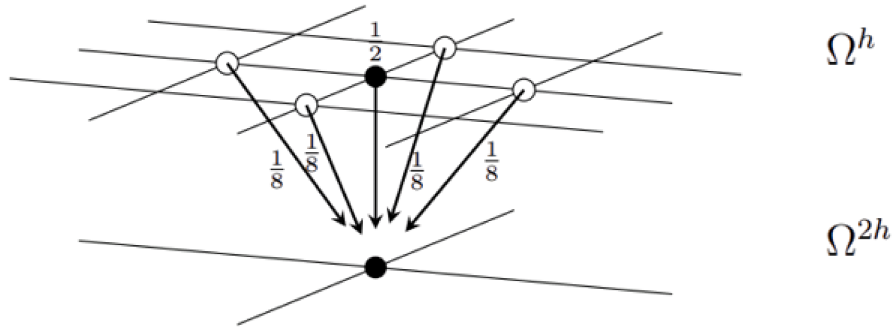


Figura 4.8: Operação de restrição para o caso meia ponderação (HW). Fonte: Franco [9].

Considerando o caso 2D, o resíduo r_h calculado pelo operador $(I_h^{2h})_{HW}$ na malha mais grossa é dado por

$$\begin{aligned}
 r_{2h}(x, y) &= (I_h^{2h})_{HW} r_h(x, y) \\
 &= \frac{1}{8} [4r_h(x, y) + r_h(x, y - h) + r_h(x - h, y) \\
 &\quad + r_h(x + h, y) + r_h(x, y + h)].
 \end{aligned} \tag{4.3}$$

3. Ponderação Completa (FW)

Para a estratégia utilizando o operador de ponderação completa $(I_h^{2h})_{FW}$, têm-se que utiliza de todos os pontos vizinhos o ponto calculado na grade mais fina para o ponto correspondente na grade mais grossa. A representação para os casos de 1D e 2D, se encontram nas Figs. 4.9 4.10 respectivamente:

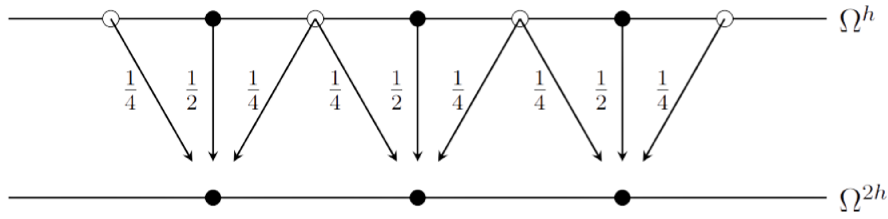


Figura 4.9: Operação de restrição para o caso ponderação completa (FW), unidimensional. Fonte: Franco[9]

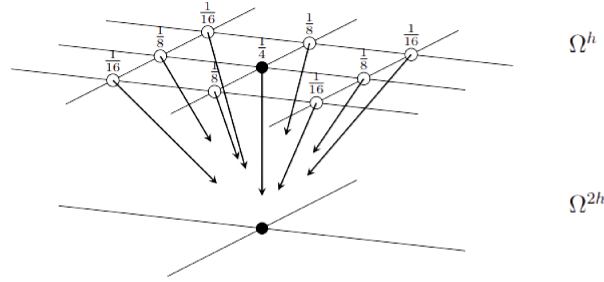


Figura 4.10: Operação de restrição para o caso ponderação completa (FW), bidimensional. Fonte: Franco[9]

Considerando o caso 2D, o resíduo r_h calculado pelo operador $(I_h^{2h})_{FW}$ na malha mais grossa é dado por

$$\begin{aligned}
 r_{2h}(x, y) &= (I_h^{2h})_{FW} r_h(x, y) \\
 &= \frac{1}{16} \{ 4r_h(x, y) + 2[r_h(x, y-h) + r_h(x-h, y) + r_h(x+h, y) \\
 &\quad + r_h(x, y+h)] + r_h(x-h, y-h) + r_h(x+h, y-h) + r_h(x-h, y+h) \\
 &\quad + r_h(x+h, y+h) \}. \tag{4.4}
 \end{aligned}$$

4.5 PROCESSO DE PROLONGAMENTO

O prolongamento faz o processo contrário da restrição, logo, ele será responsável por transferir os dados da correção de uma malha mais grosseira ω^{2h} , com a discretização em malha de espaçamento em todas as direções espaciais $2h$, para a malha mais fina ω^h , com h . Essa transposição é responsabilidade do fator de prolongação, conhecido também como fator de interpolação. As abordagens de prolongação que podem ser utilizadas são por interpolação linear (para o caso 1D), bilinear (para o caso 2D), dentre outras.

O caso da interpolação linear da correção do erro, empregando (I_h^{2h}) saindo da malha grossa e chegando na mais tênue em 1D e 2D, é representado pelas Fig. 4.11 e Fig. 4.12 respectivamente:

Para o caso 2D, utiliza-se um operador de interpolação bilinear I_h^{2h} . Neste cenário, a inter-

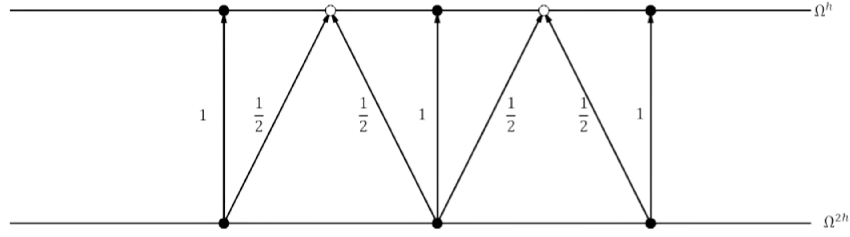


Figura 4.11: Operação de prolongação para o caso linear. Fonte: Franco[9]

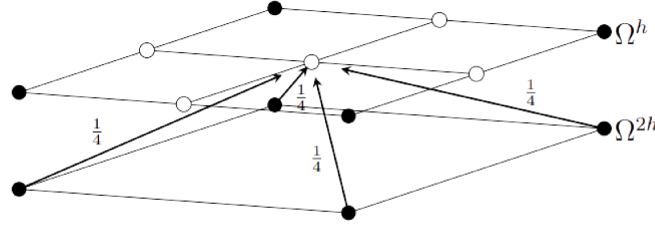


Figura 4.12: Operação de prolongação para o caso bilinear. Fonte: Franco[9]

polação bilinear da correção pelo operador I_h^{2h} da malha ω^{2h} para a malha ω^h é dada por

$$v_h(x, y) = I_{2h}^h v_{2h}(x, y),$$

$$v_h(x, y) = \begin{cases} v_{2h}(x, y) \\ \frac{1}{2}[v_{2h}(x-h, y) + v_{2h}(x+h, y)], \\ \frac{1}{2}[v_{2h}(x, y-h) + v_{2h}(x, y+h)], \\ \frac{1}{4}[v_{2h}(x-h, y) + v_{2h}(x+h, y) + v_{2h}(x, y-h) + v_{2h}(x, y+h)]. \end{cases}$$

5. MATRIZ DO PROBLEMA

O processo de condução térmica consiste na propagação do calor ao longo de um corpo elevando as temperaturas, que por conseguinte forma-se gradientes térmicos. Essa transferência pode ser por condução, convecção ou radiação, podendo também ser classificado como regime transiente ou regime permanente. Sendo o primeiro citado, um processo em que a temperatura oscila em vários pontos do sistema e em função do tempo, de modo que o fluxo de calor seja transitório. Para o regime permanente tem-se que a temperatura varia para cada ponto do sistema, mas o calor permanece constante.

O estudo de problemas que envolvem a distribuição de calor em um corpo se reconstitui há muitos séculos devido a sua relevância em solucionar os cenários encontrados no cotidiano, que são modelados por este processo. Situações que envolvem a transferência de calor possuem uma grande aplicação em áreas como a engenharia mecânica, química, nuclear e muitos outros campos. Objetiva-se analisar o comportamento da distribuição de temperatura em meio a condições impostas na fronteira, ou seja, determinar os campos de temperatura resultante dessas condições da matéria.

A utilização de modelos pautados em estratégias numéricas para resolver problemas de fluxo de calor oferece uma série de benefícios, mostrando ter uma vantagem em relação a métodos experimentais. O estudo incide sobre a condução térmica, considerando como um fenômeno bidimensional com ausência de geração interna de calor e em estado estacionário em uma placa quadrada de dimensão 2×2 .

O problema de condução de calor pode ser modelado pela equação diferencial parcial de Laplace e, para o trabalho, será estudado o caso 2D, que se apresenta na Eq. 5.1 em coordenadas cartesianas como

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0, \quad (5.1)$$

em que T é a temperatura.

Para a solução numérica desse problema físico, é empregado o Método das Diferenças Finitas (MDF) na resolução da Eq. 5.1 como modelagem discreta, com a finalidade de transformar em um problema de coeficiente algébricos, de modo que o que era contínuo, passe a ser discreto. Logo, a discretização da Eq. 5.1 pelo MDF para uma placa quadrada em que o $\Delta x = \Delta y$ será dada por:

$$\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) \Big|_{i,j} = \frac{T_{i+1,j} + T_{i-1,j} - 4T_{i,j} + T_{i,j+1} + T_{i,j-1}}{\Delta x^2} = 0 \quad (5.2)$$

Para cada ponto (nós) gerado pelo processo da discretização do domínio da placa, obtém-se uma equação linear dado por: $4T_{i,j} - T_{i+1,j} - T_{i-1,j} - T_{i,j+1} - T_{i,j-1} = 0$. Por isso, gera-se um sistema linear, que será resolvido utilizando os métodos iterativos, que serão o Método de Jacobi e Gauss-Seidel. A condição de contorno empregada no problema da placa neste estudo, é a de Dirichlet, no qual foi estipulado a mesma temperatura em toda a borda de $10^\circ C$ (graus Celsius) O problema é interessante como comparativo, pois admite solução igual a 10.

A visualização geométrica do problema é apresentada na Fig. 5.1

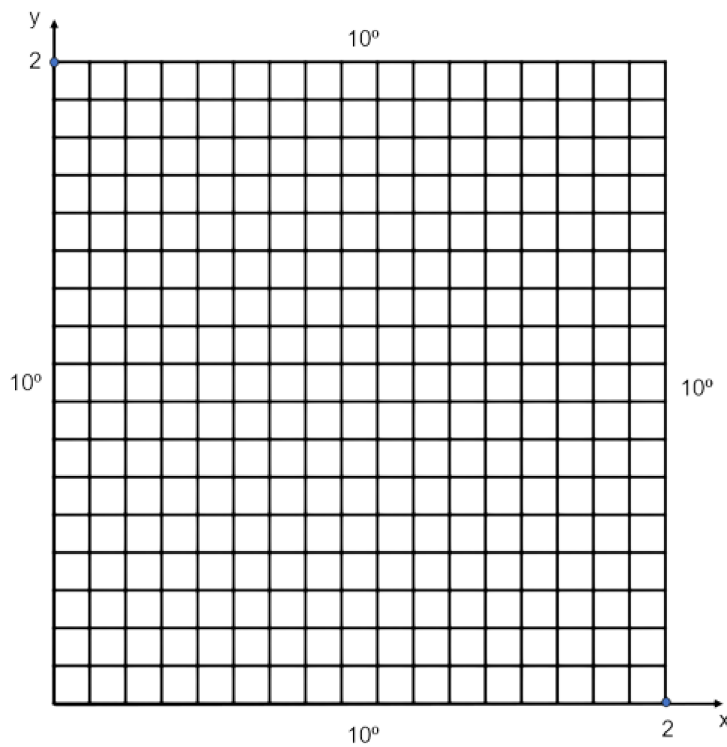


Figura 5.1: Discretização de uma placa quadrada 2×2 com condição de contorno de $10^\circ C$

5.1 SOLUÇÕES DE SISTEMAS DE EQUAÇÕES

Com os passos da discretização, que visam transformar a ED em equações algébricas, obtém-se um sistema dessas equações, pois para cada ponto da malha há um equação associada. Buscando encontrar a solução aproximada da ED inicial, o resultado será a solução do sistema de equações associados ao problema. Um sistema possível e determinado (SPD) $Ax = b$ de m equações, n incógnitas, descreve como m equações lineares e cada uma delas com n variáveis. Apresenta-se como:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2, \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m. \end{cases} \quad (5.3)$$

Em notação matricial é representado por:

$$\underbrace{\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}}_A \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}}_x = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}}_b \quad (5.4)$$

onde :

$A \rightarrow$ Matriz dos Coeficientes

$x \rightarrow$ Vetor Solução

$b \rightarrow$ Vetor dos Termos Independentes

Um sistema do tipo SPD possui uma única solução. A quantidade de soluções de um sistema pode ser verificada com o cálculo do determinante da matriz dos coeficientes associada. No caso se for diferente de zero, isso indicará que a solução do sistema é única, que é a situação que ocorre com os SPD. Essa solução se apresenta na forma de vetor, de modo a satisfazer, simultaneamente, todas as equações do sistema.

Considere o problema com a geometria discretizada conforme a Fig. 5.2.

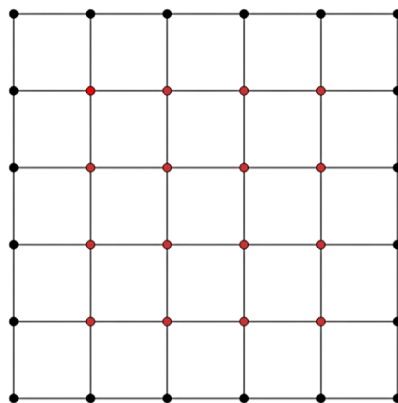


Figura 5.2: Discretização da região e ênfase nos pontos internos. Fonte: A Autora.

Usando uma discretização de pontos internos 4×4 para exemplificar a forma matricial do problema abordado neste trabalho, levando em consideração a condição de contorno, observa-se

que a representação gera uma matriz penta diagonal da maneira que é apresentada na Eq. 5.5:

$$\underbrace{\begin{bmatrix}
 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 & -1 & 0 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 & 0 & -1 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & 0 & 0 & 0 & -1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 4 & -1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 & 0 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4 & -1 \\
 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 4
 \end{bmatrix}}_A = \underbrace{\begin{bmatrix}
 T_{2,2} \\
 T_{2,3} \\
 T_{2,4} \\
 T_{2,5} \\
 T_{3,2} \\
 T_{3,3} \\
 T_{3,4} \\
 T_{3,5} \\
 T_{4,2} \\
 T_{4,3} \\
 T_{4,4} \\
 T_{4,5} \\
 T_{5,2} \\
 T_{5,3} \\
 T_{5,4} \\
 T_{5,5}
 \end{bmatrix}}_x = \underbrace{\begin{bmatrix}
 20 \\
 10 \\
 10 \\
 20 \\
 10 \\
 0 \\
 0 \\
 10 \\
 10 \\
 0 \\
 0 \\
 10 \\
 20 \\
 10 \\
 10 \\
 20
 \end{bmatrix}}_b \quad (5.5)$$

$$Ax = b$$

Os sistemas lineares que aparecem na prática modelados pelas ED, normalmente são de grande porte, ou seja, consideráveis quantidades de equações e incógnitas. Por isso, inviabiliza qualquer tipo de abordagem que não recorra a esquemas numéricos para obtenção da solução. Além do objetivo principal de encontrar uma solução aproximada ou pontual, é crucial minimizar o erro, que é a diferença entre a solução aproximada e a solução exata, para assegurar a validade do método.

6. RESULTADOS E DISCUSSÕES

Neste espaço encontra-se os resultados obtidos durante a execução dos algoritmos computacionais dos métodos juntamente com uma análise alinhada aos estudos teóricos sobre a temática. A estratégia utilizada para organização deste tópico foi, à priori, apresentar os dados de maneira clara e concisa, utilizando de tabelas e gráficos para dispor da melhor forma as principais observações. Consecutivamente, os resultados serão discutidos em paralelo com a literatura utilizada no trabalho.

A discussão será dividida em várias seções temáticas, começando com o resultado obtido do problema proposto na placa, em que os métodos de Gauss-Jacobi e Gauss-Seidel são analisados. Após, é realizada a análise do erro, mostrando a ação do método em suavizar as oscilações do erro. Por fim, o tópico que contempla o método multigrid, de forma em que sua eficiência será exposta em relação ao demais métodos.

Objetiva-se em não apenas apresentar os dados na forma pura, mas também em fornecer uma visão mais abrangente das implicações teóricas alinhadas com as práticas, visando contribuir para o avanço do conhecimento científico e servindo de inspirações para trabalhos futuros.

6.1 PRELIMINARES

O multimalha será explorado a fim de identificar a eficiência quanto a velocidade de convergência, auxiliando manter a qualidade dos métodos de Jacobi e Gauss-Seidel. A estratégia utilizada consistirá em confrontar os dados obtidos durante o ensaio com os resultado advindos da solução analítica.

O estudo do multimalha foi em torno do ciclo V com diferentes níveis, sendo 2 níveis, 3 níveis e 4 níveis, com a finalidade de identificar a influência desses no aproximação da solução. Ademais, fez-se o estudo aplicando somente os métodos de Jacobi e Gauss-Seidel na forma mais pura. Posteriormente, desenvolveu-se o método multimalha e para as etapas de restrição e prolongamento, optou-se pelas estratégias de ponderação completa e interpolação bilinear, respectivamente.

Ressalta-se que a tolerância utilizada foi 10^{-6} para todos os casos, mas esse valor poder ser variado e isso implicará em mais ou menos iterações para aproximar a solução com a tolerância desejada. Para o chute inicial, optou-se pelo vetor nulo.

Os números de iterações foram atribuídos para dois níveis consecutivos, de forma que o menos refinado possua o dobro de iterações do mais refinado. No primeiro nível aplicou-se duas

iterrações e nos pr3ximos foram duplicados ou seja, o segundo n3vel ser3o quatro iterrações e assim por diante, at3 atingir o 4º n3vel. Para o processo de prolongamento, fixou-se para todas as malhas o valor de 0 iterrações.

A seleção da malha para a implementaça3o do m3todo multimalha partiu-se da mais fina, dimensionada como 129×129 , com o aumento posterior sendo o dobro da anterior.

Em relaça3o a parte computacional, a implementaça3o dos c3digos foi feita utilizando o software *GNU octave* na vers3o 8.3.0, uma plataforma livre e compat3vel com o MATLAB que oferece uma grande quantidade de comandos convenientes para resolver problemas numericamente lineares e n3o-lineares. Referente as configuraç3es de hardware trata-se de um processador Intel Core i7-1255U de 12ª Geraça3o, com frequ3ncia de 1,70GHz, mem3ria RAM de 8GB e sistema operacional de 64 bits. Na Fig.6.1 encontra-se o esquema do m3todo multimalha do tipo CS em 4 n3veis pelo ciclo V.

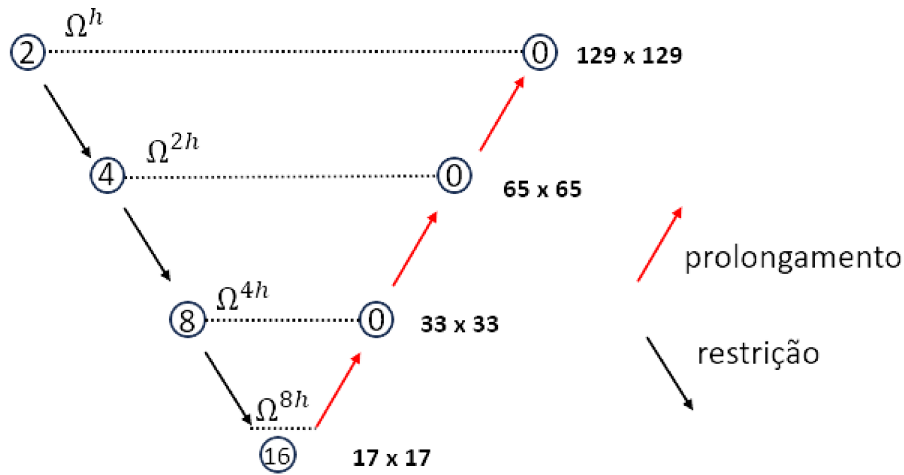


Figura 6.1: Ciclo V no M3todo Multimalha.

6.2 RESULTADO DO PROBLEMA

Considerando a quest3o abordada no cap3tulo anterior, 3 vi3vel avaliar a efic3cia dos m3todos iterativos e realizar uma comparaça3o entre eles. A soluça3o exata para esse problema 3 trivial, uma vez que, ela se manifesta como a pr3pria condiça3o de contorno. Em outras palavras, se a temperatura ao longo de toda a borda da placa for uniforme, por exemplo, $10^\circ C$, ent3o cada ponto no interior apresentar3 a mesma temperatura.

A Tabela 6.1 apresenta os dados resultantes da aplicaça3o dos algoritmos de Gauss-Jacobi e Gauss-Seidel, permitindo uma an3lise comparativa em relaça3o ao n3mero de iterrações e ao tempo m3dio para diferentes conjunto de malhas. O tempo m3dio foi obtido atrav3s da m3dia de quatro mediç3es temporais. Vale ressaltar que a an3lise desses dados contribui para a compreens3o da efici3ncia e desempenho desses m3todos na resoluça3o do problema em quest3o.

Nota-se que, em todos os casos analisados para uma malha de mesma dimens3o, apesar do m3todo Gauss-Jacobi ser a estrat3gia mais simples, o m3todo de Gauss-Seidel revelou-se

Tabela 6.1: Comparação de métodos iterativos de Gauss-Seidel e Gauss-Jacobi

Método	Dimensão	Número de Iterações	Tempo Médio [s]
GJ	17×17	654	2,43885
GS	17×17	345	0,83188
GJ	33×33	2336	42,1029
GS	33×33	1240	13,9402
GJ	65×65	8198	463,6565
GS	65×65	4388	210,285

Fonte: A autora

mais eficiente, tanto em termos do número de iterações, quanto ao tempo de CPU. No entanto, mesmo possuindo notável eficácia, o número de iterações permanece elevado e o tempo de CPU é significativo. Vale ressaltar que, no contexto do problema descrito, só foi possível observar resultados favoráveis devido à natureza esparsa da matriz. Essa propriedade da matriz contribui significativamente para a eficiência de métodos como esses.

6.3 ANÁLISE DO PROCESSO DE SUAVIZAÇÃO DO ERRO

Os métodos do tipo suavizadores apresentam uma taxa de progresso mais lenta devido à demora em atenuar baixa frequência de erro. Com o propósito de elucidar tal comportamento e esclarecer a característica de que métodos iterativos são eficazes na redução dos erros associados a altas frequências, será abordado a equação de Laplace bidimensional apresentada no capítulo anterior, com as mesmas dimensões e condições de contorno.

Note que a solução analítica do problema descrito da equação de Laplace (5.1) é dada pela condição, ou seja, $T = 10^\circ C$. Ao aproximar essa equação pelo método de diferenças finitas centradas de segunda ordem, com a malha discretizada em dimensão 17×17 , resulta-se no sistema linear de equações. Esse sistema pode numericamente ser resolvido pelo método de Jacobi. Para os testes, considere uma condição inicial dada por

$$T_0(x, y) = \sin(10x) + \cos(5y). \quad (6.1)$$

A Fig.6.2 representa a evolução do módulo do erro após 0, 1, 50 e 1000 iterações. O erro e é definido como

$$e = |T^k - 10|,$$

em que k representa as iterações.

É notório que, com um número determinado de iterações, o método é capaz de suavizar as oscilações do erro (passagem de 0 a 1 e 1 a 50). mas quando esse erro já se encontra em baixas frequências, o Jacobi não consegue atuar com eficiência, que é explicitado na passagem de 50 a 1000 iterações, pois não é visto com formas similares a gaussianas com erro máximo observado no centro do domínio. Demonstrando a capacidade efetiva que o método tem de agir efetivamente em altas frequências de erro, podendo ser considerado como suavizador.

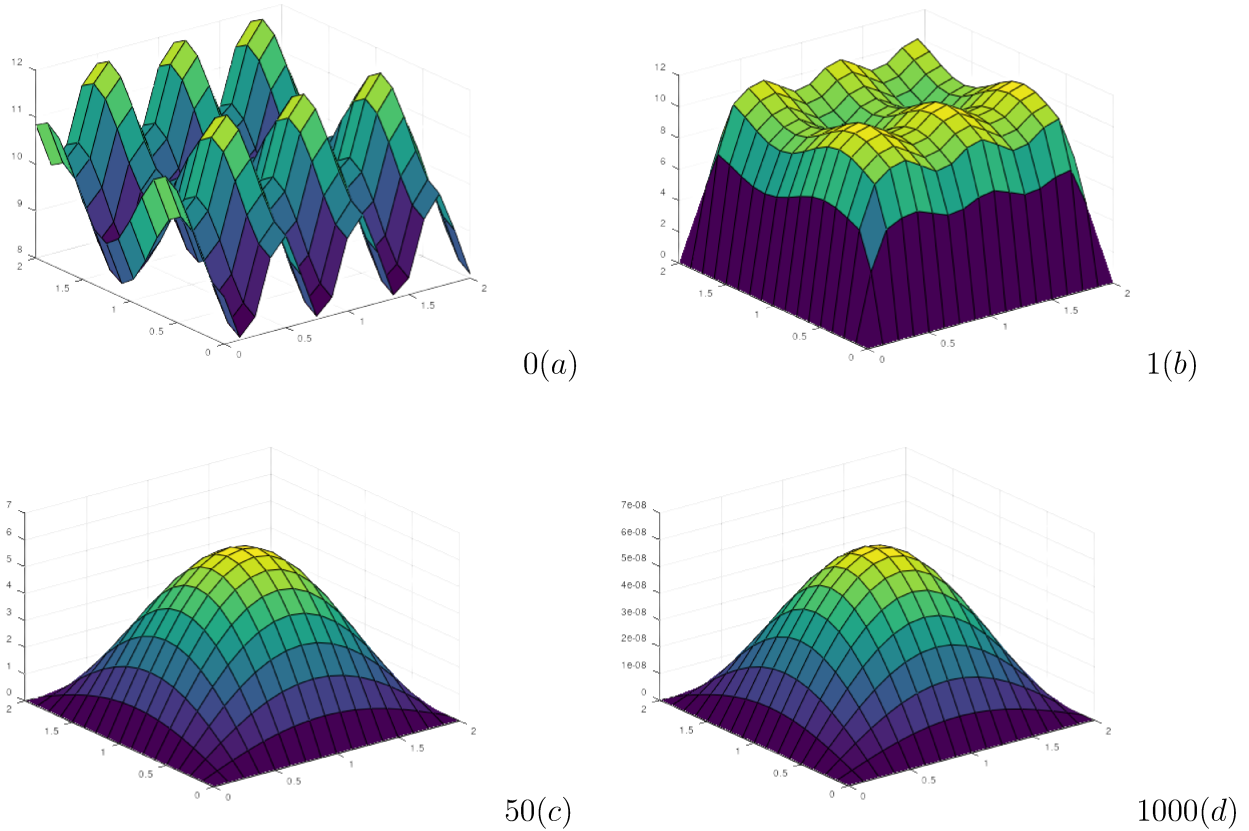


Figura 6.2: Suavização do erro pelo método Gauss-Jacobi, com 0, 1, 50 e 1000 iterações

Com isso, é possível inferir que quando o método estiver submetido a grandes oscilações ele será eficaz. Visando atingir tais resultados positivos, o método multimalha tem o intuito de usar um conjunto de malhas em que os espaçamentos mudam de acordo com o refinamento. Como mostra a Fig.4.1, nota-se que baixas frequências associada a uma malha ω^h , quando submetidas a malha ω^{2h} , assumem caracteres oscilatórios. Essa estratégia, portanto, permite ao método multimalha adaptar-se e garantir que os métodos iterativos não percam a eficiência. Além de aumentar a taxa de convergência, uma vez que, há uma redução de erros em todas as frequências, independentemente da malha.

6.4 APLICAÇÃO DO MÉTODO MULTIMALHA

Para o presente trabalho um multimalha CS na forma de ciclo V foi implementado. Estudos para essa estrutura foram conduzidos assumindo o uso de dois, três ou quatro níveis de refinamento. Testes foram realizados considerando dois tipos de suavizadores: o método de Jacobi e o método de Gauss-Seidel. De acordo com (TANNEHILL et al., 1997[18]; PINTO; MARCHI, 2007[14]) para alcançar um desempenho eficaz do método multimalha, é recomendável utilizar múltiplos níveis de malhas.

Considerando o suavizador definido como sendo o método de Gauss-Jacobi, a primeira simulação foi entorno do método de Gauss-Jacobi, e os resultados obtidos estão detalhados na Tab.6.2, a qual já foi previamente apresentada na Tabela 6.1. No entanto, para fins de exame

comparativo, o mesmo método foi adicionado como suavizador em simulações de multimalha em níveis de 2 a 4, conforme apresentado nos resultados.

Tabela 6.2: Análise da eficiência do método Multimalha - Gauss-Jacobi.

Método	Dimensão	Número de Iterações	Tempo Médio [s]
GJ	17×17	654	2,43885
GJM2	17×17	38	0,70987
GJM3	17×17	11	0,1523
GJM4	17×17	12	0,4409
GJ	33×33	2336	42,1029
GJM2	33×33	131	10,4947
GJM3	33×33	19	1,15507
GJM4	33×33	12	1,6816
GJ	65×65	8198	463,6565
GJM2	65×65	457	151,9540
GJM3	65×65	59	15,2623
GJM4	65×65	16	6,8984
GJ	129×129	> 6h	> 6h
GJM2	129×129	> 6h	> 6h
GJM3	129×129	196	282,8660
GJM4	129×129	26	45,4955

Fonte: A autora

Nota-se que para uma mesma dimensão, o método multimalha se mostrou mais eficaz tanto na velocidade de convergência, quanto no número de iterações. Porém, o número de níveis em uma mesma malha não implica em resultados melhores, isso pode ser visto com os dados do multimalha de nível 3 e nível 4. O que pode-se inferir é que para problemas com um número pequeno de variáveis, o custo adicional em processamento para transferência de informação entre malhas é superior ao tempo que o método malha única demanda para atingir a tolerância definida. Mas já para malhas de dimensões maiores como 65×65 e 129×129 ao serem submetidas a camadas superiores, tem ganhos significativos.

Contudo, o nível é um fator que influencia a eficácia do multimalha. Por exemplo, ao considerar uma malha de dimensões 129×129 , percebe-se que não é vantajoso aguardar a convergência quando submetida ao método em uma única malha, pois o tempo de CPU é consideravelmente maior, o que se mantém para duas malhas. No entanto, ao lidar com problemas de maior dimensão, método demonstra resultados significativos. Essa observação é particularmente útil, pois ao enfrentar desafios dessa magnitude, a escolha por níveis mais elevados é preferível, dada a grande quantidade de pontos em níveis inferiores, resultando em uma eficiência aprimorada dos suavizadores.

Independentemente do tamanho da malha, o resultado foi satisfatório para todos os níveis. Dessa forma, o método multigrid revela-se ser uma escolha mais eficiente e rápida para resolver o problema em questão, comparado a utilizar o Jacobi em um único malha.

Como foi visto pela tab.6.1 o método Gauss-Seidel apresentou ser mais rentável que o Gauss-

Jacobi. Para verificar se essa vantagem permanece ao fazer o uso da metodologia do multimalha, realizou-se os mesmo ensaios do suavizador apresentado anteriormente e gerou-se a Tab. 6.3

Tabela 6.3: Análise da eficiência do método Multimalha - Gauss-Seidel.

Método	Dimensão	Número de Iterações	Tempo Médio (segundos)
GS	17×17	345	0,83188
GSM2	17×17	23	0,27435
GSM3	17×17	9	0,201724
GSM4	17×17	11	0,247794
GS	33×33	1240	13,9402
GSM2	33×33	78	4,1074
GSM3	33×33	11	1,05882
GSM4	33×33	18	1,8144
GS	65×65	4388	210,285
GSM2	65×65	274	69,24148
GSM3	65×65	32	1,2731
GSM4	65×65	20	8,3316
GS	129×129	> 6h	> 6h
GSM2	129×129	> 6h	> 6h
GSM3	129×129	107	203,320
GSM4	129×129	20	33,7799

Fonte: A autora

Analizando a Tab.6.3, é possível fazer as mesmas avaliações que foram observadas no método de Jacobi. No entanto, ao comparar a Tab.6.2 com a Tab.6.3 nota-se uma grande discrepância nos resultados, destacando o desempenho superior do método Gauss-Seidel em relação ao Jacobi. Essa superioridade era esperada, uma vez que, os dados obtidos corroboram com a literatura, em que a eficácia do Gauss-Seidel em suavizar os erros é maior do que o Jacobi. Assim como visto no problema anterior, as malhas 17×17 33×33 não obtém um bom desempenho em níveis superiores a 3, tendo em vista que o custo operacional, que é o que visa minimizar, fique maior.

6.5 SPEEDUP

A eficiência do método malha única comparada ao multimalha, pode ser observada com os resultado do speedup. Em linhas gerais, ele é uma ferramenta que expressa a medida do ganho de desempenho obtido na comparação do tempo de execução método de referência com o método otimizado. Sua utilização está em várias áreas e contextos, entretanto possui uma grande aplicação na análise de desempenhos software, dos algoritmos e afins. No caso deste trabalho, será comparado o tempo de CPU dos métodos malha única (método de referência) com o multimalha (método otimizado).

O speedup pode ser definido como

$$S = \frac{\text{Tempo de execução do singlegrid}}{\text{Tempo de execução do multigrid}}. \quad (6.2)$$

Quando o valor encontrado é superior a 1, indica que o método otimizado mostrou-se mais eficaz e, caso contrário, o tempo de execução do método de referência foi menor. Isso indica que o método apontado como otimizado não conseguiu atingir resultados melhores.

Na Tab.6.4 e na Fig. 6.3 encontra-se os resultados de speedup dos tempos médios de Jacobi e dos tempos médios do multimalha utilizando Jacobi.

Tabela 6.4: Speedup do Gauss-Jacobi.

Dimensão	Nível 2	Nível 3	Nível 4
17×17	3,43563	16,0153	5,53135
33×33	4,01184	36,4505	25,0378
65×65	3,05129	30,3793	67,2126
129×129	-	-	-

Fonte: A autora

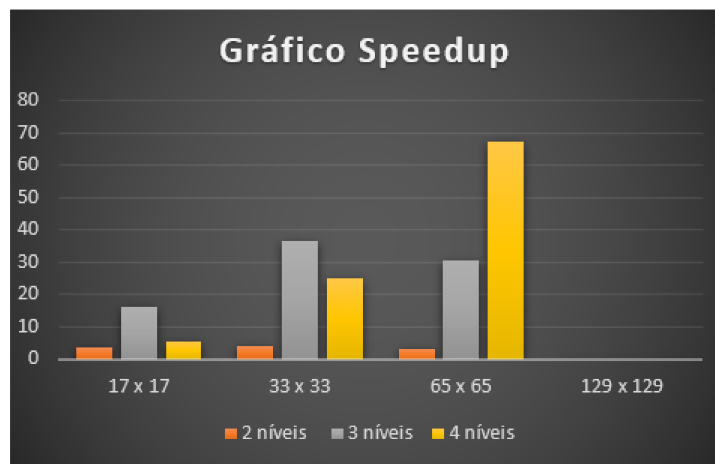


Figura 6.3: Gráfico do speedup do Gauss-Jacobi.

Os resultados na malha 129×129 não foram representados na tabela e no gráfico devido a justificativa citada anteriormente. Sem informação do tempo médio na malha única, não é possível encontrar os ganhos pelo speedup.

Pela análise, nota-se que o nível três obteve o melhor ganho quando submetido a malhas mais grossas, quando a dimensão aumenta, o nível quatro fica muito mais eficaz. Mais uma vez, os resultados estão alinhados com a literatura que diz que grande número de pontos requerem níveis maiores.

Visando analisar o método de Gauss-Seidel, a Fig.6.4 e a Tab. 6.5, expõe os resultados:

Tabela 6.5: Speedup do Gauss-Seidel.

Dimensão	Nível 2	Nível 3	Nível 4
17×17	3,032149	4,123852	3,357143
33×33	3,393934	13,16573	7,683003
65×65	3,03698	165,1723	25,23942
129×129	-	-	-

Fonte: A autora

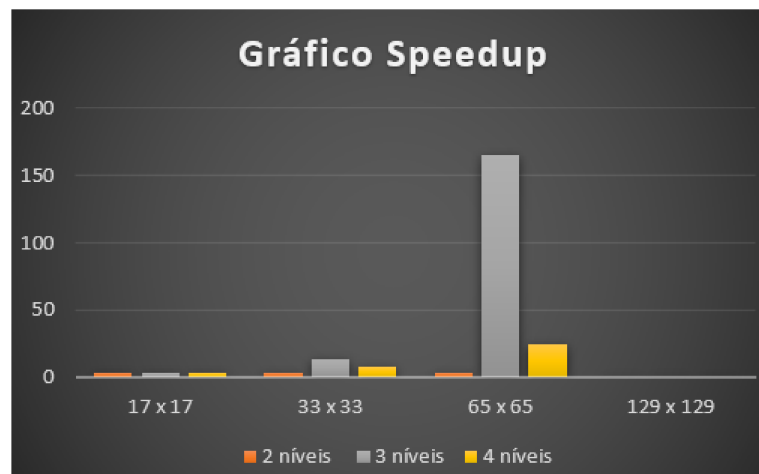


Figura 6.4: Gráfico do speedup do Gauss-Seidel.

O maior ganho do método de Gauss-Seidel foi na malha de 65×65 , utilizando-se de três níveis. O restante não apresentou tanta discrepância, um dos fatores ocasionais pode ser que como o Gauss-Seidel é superior ao Jacobi e já possui uma velocidade de convergência grande, o multimalha não melhora tanto quanto é visto em Fig.6.3, onde as melhorias são mais evidentes.

6.6 SOLUÇÃO DA EQUAÇÃO DE POISSON

Em um cenário de abordagem numérico, os passos fundamentais que devem ser realizados ao utilizar um código computacional, que visa a obtenção de soluções numéricas para o problema de difusão térmica, é a verificação e validação do mesmo.

O processo de validação de uma metodologia utilizada para obtenção de soluções, se dá pela comparação dos resultados obtidos ao fazer o uso do código computacionais, baseando-se em soluções analíticas, numéricas ou experimentais. No caso, todos os dados apresentados em tabelas e as discussões em torno dos mesmos, realizados anteriormente, são exemplos desse procedimento. Contudo, a conclusão desta fase não assegura que a solução obtida atenda às expectativas, uma vez que podem ocorrer acumulações de erros oriundos da própria implementação do algoritmo. Para garantir essa validação, opta-se por executar a etapa de verificação.

Segundo (Moreira, 2011)[15] o processo de verificação constitui em um estudo que tem como objetivo assegurar a resolução correta das equações que representa o problema físico, de forma que garanta a ausência de erros na implementação e no algoritmo. Essa verificação pode não estar associado ao um problema físico realmente, sendo assim, apenas uma estratégia matemática (Moreira, 2011)[15].

Com o intuito de verificar as conformidades dos códigos implementados e garantir um desempenho satisfatório, foi empregado o método das Soluções Manufaturadas para a Equação de Laplace.

O método das Soluções Manufaturadas é um estratégia que permite a geração de soluções analíticas para modelos diferenciais, com a finalidade de obter uma solução segura com o propósito de fazer a verificação dos algoritmos. A ideia do método é a partir de uma solução

analítica verossímil determinar o termo fonte, aplicando essa solução na equação diferencial. Sendo assim, o problema diferencial permite verificar as metodologias investigando os erros de implementações do algoritmo.

Considerando o mesmo problema da condução térmica apresentada no secção 5, a proposta deste trabalho será utilizar solução analítica definida em Eq. 6.3

$$T(x, y) = \cos(x)\sin(y) \quad (6.3)$$

o termo-fonte f gerado da solução analítica, é obtido fazendo a aplicação na equação de Laplace, ficando da forma com que é apresentado na Eq.6.4

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = -2\cos(x)\sin(y) \quad (6.4)$$

Por tanto, ao fazer esse processo o operador de Laplace se "transforma" em uma equação de Poisson que está representada pela Eq.6.4, com a condição de contorno determinada usando a solução dada pela Eq. 6.3. A partir desse novo problema que se fará o estudo da ordem de convergência, neste caso deverá ser 2, pois usou-se o método de diferenças finitas de 2ª ordem, validando ou não o algoritmo dos métodos aplicados para resolver a Eq.5.2. ordem,

A fim de obter essa ordem utiliza-se a expressão dada por.

$$O = \left| \frac{\log\left(\frac{Erro_{\frac{\Delta h}{2}}}{Erro_{\Delta h}}\right)}{\log(2)} \right|,$$

em que

$$Erro_{\Delta h} = ||T_{aprox} - T_{exato}||_{\infty}$$

Para verificar se o erro está na ordem de h^2 realizou-se o estudo de ordem de convergência em todas as malhas utilizadas no problema da secção 5 e os métodos. A seguir, encontra-se os dados obtidos, organizados em tabela.

Com base nos dados, pode-se inferir que os algoritmos implementados demonstram eficácia na busca da solução aproximada do problema descrito na seção 5.

Na Tab. 6.8 e Tab. 6.9 encontra-se o Speedup do multimalha de 4 níveis tanto utilizando Gauss-Jacobi, como Gauss-Seidel.

Nas malhas mais grosseiras o método malha única de Jacobi mostrou-se mais eficaz, uma vez que, usando o método multimalha de 4 níveis é necessário mais tempo para percorrer todos os níveis, ultrapassando assim o tempo demandado pelo método malha única para atingir a tolerância definida. Por outro lado, o método multimalha é eficiente quando o problema apresenta uma grande quantidade de variáveis, que é o caso da malha 65×65 . Para o método de Gauss-Seidel é visto pela Tab.6.9 que o método multimalha foi mais vantajoso independentemente da malha em 4 níveis e, que o ganho se mostrou superior ao método de Gauss-Jacobi.

Tabela 6.6: Ordem do Gauss-Jacobi.

Método	Dimensão	Erro	Ordem
GJ	17×17	3,4923e-04	–
GJ	33×33	8,8146e-05	1,9862
GJ	65×65	2,1977e-05	2,0039
GJ	129×129	> 6h	> 6h
GJM2	17×17	3,4904e-04	–
GJM2	33×33	8,8376e-05	1,9817
GJM2	65×65	2,2047e-05	2,0031
GJM2	129×129	> 6h	> 6h
GJM3	17×17	3,4924e-04	–
GJM3	33×33	8,8163e-05	1,9860
GJM3	65×65	2,2047e-05	1,9996
GJM3	129×129	5,5172e-06	1,9986
GJM4	17×17	3,4924e-04	–
GJM4	33×33	8,8163e-05	1,9860
GJM4	65×65	2,2051e-05	1,9993
GJM4	129×129	5,5172e-06	1,9988

Tabela 6.7: Ordem do Gauss-Seidel.

Método	Dimensão	Erro	Ordem
GS	17×17	3,4900e-04	–
GS	33×33	8,7289e-05	1,9994
GS	65×65	1,8631e-05	2,2281
GS	129×129	> 6h	> 6h
GSM2	17×17	3,4916e-04	–
GSM2	33×33	8,7498e-05	1,9966
GSM2	65×65	1,8813e-05	2,2175
GSM2	129×129	> 6h	> 6h
GSM3	17×17	3,4925e-04	–
GSM3	33×33	8,5723e-05	2,0265
GSM3	65×65	2,0414e-05	2,0701
GSM3	129×129	5,3977e-06	1,9191
GSM4	17×17	3,4941e-04	–
GSM4	33×33	9,0805e-05	1,9441
GSM4	65×65	2,3369e-05	1,9582
GSM4	129×129	5,5254e-06	2,0804

Tabela 6.8: Speedup do Gauss-Jacobi com método multimalha de 4 níveis.

Dimensão	Speedup
17×17	0,710091104
33×33	0,806737834
65×65	1,271299293
129×129	> 6h

Tabela 6.9: Speedup do Gauss-Seidel com método multimalha de 4 níveis.

Dimensão	Speedup
17×17	4,411714593
33×33	18,48862804
65×65	26,71097
129×129	> 6h

7. CONCLUSÕES

Problemas reais muita vezes podem ser modelados através de um conjunto de equações diferenciais parciais. A resolução numérica dessas equações, com a equação de Laplace, muitas vezes exigem a resolução de sistemas lineares. De forma a resolver sistemas lineares eficientemente, o presente estudo foi realizado dando ênfase ao método multigrid como alternativa de melhorar a velocidade de convergência dos métodos iterativos, objetivando-se resolver as equações lineares obtidas do processo de discretização.

O estudo sobre as equações diferenciais existentes, os métodos de discretizações e os métodos de resoluções de sistemas lineares, permitiu uma visão mais ampla e aprofundada do tema tratado.

Durante o processo de investigação, foi possível identificar que o método multigrid tem um grande potencial na resolução de problemas que envolvem as Equações Diferenciais Elípticas se comparado com a aplicação dos suavizadores clássicos. Métodos do tipo suavizadores são efetivos na redução de altas frequências do erro, mas são ineficientes na atenuação de frequências mais baixas. Para caracterizar os elementos de método multimalha CS, adotou-se uma estrutura na forma de ciclo V. Para o conjunto de malhas utilizadas, métodos multimalha foram mais eficientes em métricas de tempo de execução e número de iterações.

Esses resultados destacam a importância do estudo e de simulações a respeito do método multigrid, pois fornecem soluções para problemas da indústria de grande magnitude em tempo hábil.

Ao perpassar por literaturas importantíssimas, que contribuíram muito para o avanço da temática, foi possível confirmar a relevância das teorias estudadas. Contudo, nota-se que ainda há muito o que se explorar como por exemplo: utilizar de números de iterações distintas, aumentar o nível de malha em problemas mais refinados, verificar as melhores combinações dos suavizadores nos níveis e por fim, analisar os parâmetros que mais impacta no desempenho do método e encontrar meios de ajustá-los de forma otimizada. Com a finalidade de aprimorar as técnicas e aperfeiçoar os resultados da aplicação do multigrid.

Sugere-se como perspectivas para trabalhos futuros o estudo aprofundado de simulações aplicadas a equações diferenciais parabólicas. Adicionalmente, propõe-se explorar de maneira mais detalhada os desafios associados à difusão térmica, considerando variações temperatura com o tempo.

Em geral, o estudo foi capaz de contribuir de forma significativa para o entendimento a respeito do método multigrid, representando um passo importante na busca pelo conhecimento

da área de matemática computacional. Os desafios encontrados ao serem superados pelos estudos contínuos, mostrou a importância da pesquisa e a busca por soluções.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] *Métodos Iterativos - Sistemas Lineares*, 2023. <https://www.math.tecnico.ulisboa.pt/~calves/cursos/SisLin-Iter.htm>, Acessado em: 15/11.
- [2] Brandt, A.: *Multi-level adaptive solutions to boundary-value problems*. Mathematics of Computation, 31(138), 1977.
- [3] Briggs, W. L.; Henson, V. E. M. S. F.: *A Multigrid Tutorial*. Philadelphia: SIAM, 2^a ed., 1999.
- [4] Briggs, W. L.; Henson, V. E. M. S. F.: *A Multigrid Tutorial*, vol. 2. Philadelphia: SIAM, 3^a ed., 2000.
- [5] Cuminato, J. A. e Junior, M. M.: *Discretização de equações diferenciais parciais*. SBM, Rio de Janeiro, 2013.
- [6] Fedorenko, R.: *On the Speed of Convergence of an Iteration Process*. Comput. Math. and Math. Phys, 4(3), 1964.
- [7] Ferreira, J. N.: *Notas de Cálculo Numérico*. notas de aula utilizados na disciplina Cálculo Numérico, 2015.
- [8] Fortuna, A. d. O.: *Técnicas computacionais para dinâmica dos fluídos: conceitos básicos e aplicações*. USP, 2000.
- [9] Franco, S. R.: *Métodos Multigrid Espaço-Tempo Para Resolver as Equações do Calor e da Poroelasticidade*. Tese de Doutorado, 2017.
- [10] Gupta, M. M., Kouatchou, J. e Zhang, J.: *Comparison of second- and fourth-order discretizations for multigrid Poisson solvers*. Journal of Computational Physics, 132:226–232, 1997.
- [11] Iório, V.: *EDP: um curso de graduação*. IMPA, Rio de Janeiro, 1989.
- [12] Magalhães, P. M. D. d.: *Introdução as Equações Diferenciais Parciais*, 2016. <http://professor.ufop.br/sites/default/files/freud/files/edp-cap1.pdf>.
- [13] Mesquita, M. S.: *Solução Numérica de Escoamentos bidimensionais Não-isotérmicos usando o Método Multigrid*. Tese de Mestrado, Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, SP, 2000.

- [14] Pinto, M. A. V. e Marchi, C. H.: *Optimum Parameters of a Geometric Multigrid for the Two-Dimensional Laplace's Equation*. Em *Proceedings of COBEM*, 2007.
- [15] Queiroz Moreira, L. de: *Modelagem Matemática de Jatos em Desenvolvimento Espacial Usando a Metodologia Pseudo Espectral de Fourier*. Tese de Doutorado, 2011.
- [16] Spitaleri, R. M.: *Full-FAS multigrid grid generation algorithms*. *Applied Numerical Mathematics*, 32:483–494, 2000.
- [17] Stüben, K.: *Algebraic Multigrid (AMG): an introduction with applications*. 1999.
- [18] Tannehill, J. C., Anderson, D. A. e Pletcher, R. H.: *Computational Fluid Mechanics and Heat Transfer*. Taylor & Francis, Philadelphia, 1997.
- [19] Trottenberg, U., Oosterlee, C. e Schüller, A.: *Multigrid*. Academic Press, San Diego, 2001.
- [20] Wesseling, P.; Oosterlee, C.: *Multigrid with applications to computational fluid dynamics*. *Comput. Appl. Math.*, 128:311–334, 2001. <http://dx.doi.org/10.1016/j.compfluid.2017.06.012>.
- [21] Wesseling, P.: *An Introduction to Multigrid Methods*. John Wiley & Sons, Chichester, 1992.
- [22] Zhang, J.: *Acceleration of five-point red-black Gauss-Seidel in multigrid for Poisson equation*. *Applied Mathematics and Computation*, 80:73–93, 1996.
- [23] Zhang, J.: *Residual scaling techniques in multigrid, I: equivalence proof*. *Applied Mathematics and Computation*, 80:283–303, 1997.

A. APÊNDICE

A.1 OPERADORES

A.1.1 RESÍDUO

```
function r = residuo(A, b, dx)
    [N, N] = size(A);
    r = zeros(N, N);

    % Loop pelos elementos internos da matriz

    for i = 2:N-1
        for j = 2:N-1
            % Cálculo do resíduo
            r(i, j) = b(i, j) - (-4 * A(i, j) + (A(i, j - 1) +
            A(i, j + 1) + A(i - 1, j) + A(i + 1, j))) / (dx * dx);
        endfor
    endfor
endfunction
```

A.1.2 OPERADOR DE RESTRIÇÃO

```
% operação de restrição para o multigrid
function A1 = restricao(A0)

[nl,nc] = size(A0);
for i = 3 : 2 : nl-2
    for j = 3 : 2 : nc-2
        A1((i+1)/2,(j+1)/2) = (4*A0(i,j)+2*(A0(i-1,j)+A0(i+1,j)+
        A0(i,j-1)+A0(i,j+1))+A0(i-1,j-1)+A0(i-1,j+1)+A0(i+1,j+1)+A0(i+1,j-1))/16;
    endfor
endfor
```

```
endfunction
```

A.1.3 OPERADOR DE PROLONGAÇÃO

```
function A0 = prolongamento(A1)

    % prolongamento

    % dimensão malha grossa
    [nl,nc] = size(A1);

    % injeção direta malha grossa (A1) para malha fina (A0)
    for i = 1 : nl
        for j = 1 : nc
            A0(2*i-1,2*j-1) = A1(i,j);
        endfor
    endfor

    % dimensão malha fina com fator 2
    nl = 2*nl - 1;
    nc = 2*nc - 1;

    % interpolação na direção y
    for i = 2 : 2 : nl-1
        for j = 1 : 2 : nc
            A0(i,j) = ( A0(i-1,j)+A0(i+1,j) ) / 2;
        endfor
    endfor

    % interpolação na direção x
    for j = 2 : 2 : nc-1
        for i = 1 : nl
            A0(i,j) = ( A0(i,j-1)+A0(i,j+1) ) / 2;
        endfor
    endfor
endfunction
```

A.2 GAUSS-JACOBI

```
function [T, erro] = jacobi2(T, b, Niter, k, hy)

% Função de Jacobi para resolver um sistema linear

[N, N] = size(T);
T_new = T;

for iter = 1:Niter
    for i = 2:N-1
        for j = 2:N-1
            % Atualização usando o método de Jacobi
            T_new(i, j) = ((T(i, j - 1) + T(i, j + 1) + k * (T(i - 1, j) +
                T(i + 1, j)) - b(i, j) * (hy^2)) / (2 * k + 2));
        endfor
    endfor

    % Cálculo do erro
    erro = max(max(abs(T_new - T)));

    % Atualização para a próxima iteração
    T = T_new;
endfor

endfunction
```

```
function [T] = jacobi3(T, b, Niter, k, hy)

% Função de Jacobi para resolver um sistema linear

[N, N] = size(T);
T_new = T;

for iter = 1:Niter
    for i = 2:N-1
        for j = 2:N-1
            % Atualização usando o método de Jacobi
            T_new(i, j) = ((T(i, j - 1) + T(i, j + 1) + k * (T(i - 1, j) +
                T(i + 1, j)) - b(i, j) * (hy^2)) / (2 * k + 2));
        endfor
    endfor
endfor
```

```
        endfor
    endfor

    % Atualização para a próxima iteração
    T = T_new;
endfor

endfunction
```

```
% Limpar console e workspace
clc;
clear;

% Parâmetros iniciais
Niter = 1;
Miter = Niter;
N = 65;
x = 2;
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Inicializar matriz de temperatura T
T = zeros(N, N);

% Aplicar condições de contorno
```

```
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

% Inicializar matriz de temperatura temporária
T_new = T;

% Configurar critérios de convergência
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Iniciar temporizador
tic

% Laço principal para método iterativo (Jacobi)
while (erro > eps)
    ciclos = ciclos + 1;

    % Chamar função jacobi2 para atualizar a matriz de temperatura
    [T, erro] = jacobi2(T, b, Niter, k, hy);

end

% Parar temporizador e exibir tempo de execução
toc
```

A.2.1 MULTIGRID COM GAUSS - JACOBI

2 NÍVEIS

```
% Limpar console e workspace
clc;
clear;

% Parâmetros do método de Jacobi com multigrid de 2 níveis
Niter = 1;
```

```
Miter = 20;
N = 129;
x = 2;
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_new = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Inicialização de variáveis
erro2_inicial = zeros(65, 65);
```

```
% Iniciar temporizador
tic

% Laço principal do método de Jacobi com multigrid
while (erro > eps)
    % Iteração do método de Jacobi
    [T, erro] = jacobi2(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Jacobi para o primeiro resíduo
    erro2 = jacobi3(erro2_inicial, A1, 4, k, 2 * hy);

    % Prolongamento e correção do campo de temperatura
    corr1 = prolongamento(erro2);
    T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc
```

3 NÍVEIS

```
% Limpar console e workspace
clc;
clear;

% Iteração do método de Jacobi para o multigrid de 3 níveis
% Parâmetros
Niter = 1;
Miter = 20;
N = 33;
x = 2;
y = 2;
```

```
% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_new = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Inicialização de variáveis
erro2_inicial = zeros(17, 17);
erro3_inicial = zeros(9, 9);

% Iniciar temporizador
tic

% Laço principal do método de Jacobi com multigrid de 3 níveis
```

```
while (erro > eps)
    % Iteração do método de Jacobi
    [T, erro] = jacobi2(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Jacobi para o segundo resíduo
    erro2 = jacobi3(erro2_inicial, A1, 4, k, 2 * hy);

    % Cálculo do resíduo
    r1 = residuo(erro2, A1, 2 * hx);
    A2 = restricao(r1);

    % Iteração do método de Jacobi para o terceiro resíduo
    erro3 = jacobi3(erro3_inicial, A2, 8, k, 4 * hy);

    % Prolongamento e correção do segundo resíduo
    corr2 = prolongamento(erro3);
    erro2 = erro2 + corr2;

    % Prolongamento e correção do campo de temperatura
    corr1 = prolongamento(erro2);
    T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc
```

4 NÍVEIS

```
% Limpar console e workspace
clc;
clear;

% Parâmetros
```

```
Niter = 1;
Miter = 20;
N = 129;
x = 2;
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_new = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Inicialização de variáveis
erro2_inicial = zeros(65, 65);
```

```
erro3_inicial = zeros(33, 33);
erro4_inicial = zeros(17, 17);

% Iniciar temporizador
tic

% Laço principal do método de Jacobi com multigrid de 4 níveis
while (erro > eps)
    % Iteração do método de Jacobi
    [T, erro] = jacobi2(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Jacobi para o segundo resíduo
    erro2 = jacobi3(erro2_inicial, A1, 4, k, 2 * hy);

    % Cálculo do resíduo
    r1 = residuo(erro2, A1, 2 * hx);
    A2 = restricao(r1);

    % Iteração do método de Jacobi para o terceiro resíduo
    erro3 = jacobi3(erro3_inicial, A2, 8, k, 4 * hy);

    % Cálculo do resíduo
    r2 = residuo(erro3, A2, 4 * k);
    A3 = restricao(r2);

    % Iteração do método de Jacobi para o quarto resíduo
    erro4 = jacobi3(erro4_inicial, A3, 16, k, 8 * hy);

    % Prolongamento e correção do terceiro resíduo
    corr3 = prolongamento(erro4);
    erro3 = erro3 + corr3;

    % Prolongamento e correção do segundo resíduo
```

```

    corr2 = prolongamento(erro3);
    erro2 = erro2 + corr2;

    % Prolongamento e correção do campo de temperatura
    corr1 = prolongamento(erro2);
    T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc

```

A.3 GAUSS-SEIDEL

```

function [T, erro] = GaussSeidel(T, b, Niter, k, hy)
    % Função de Gauss-Seidel para resolver um sistema linear

    [N, N] = size(T);
    T_old = T;

    % Iterações
    for iter = 1:Niter
        for i = 2:N-1
            for j = 2:N-1
                % Atualização usando o método de Gauss-Seidel
                T(i, j) = ((T(i, j - 1) + T(i, j + 1) + k * (T(i - 1, j) +
                    T(i + 1, j)) - b(i, j) * (hy^2)) / (2 * k + 2));
            end
        end
    end

    % Cálculo do erro
    erro = max(max(abs(T_old - T)));
    T_old = T;
end
end

```

```

function [T] = GaussSeidel2(T, b, Niter, k, hy)
    % Função de Gauss-Seidel para resolver um sistema linear

```

```
[N, N] = size(T);
T_old = T;

% Iterações
for iter = 1:Niter
    for i = 2:N-1
        for j = 2:N-1
            % Atualização usando o método de Gauss-Seidel
            T(i, j) = ((T(i, j - 1) + T(i, j + 1) + k * (T(i - 1, j)
                + T(i + 1, j)) - b(i, j) * (hy^2)) / (2 * k + 2));
        end
    end

    % Atualização para a próxima iteração
    T_old = T;
end
end
```

```
% Limpar console e workspace
```

```
clc;
```

```
clear;
```

```
% Parâmetros
```

```
Niter = 1;
```

```
N = 65;
```

```
x = 2;
```

```
y = 2;
```

```
% Calcular passos nas direções x e y
```

```
hx = x / (N - 1);
```

```
hy = y / (N - 1);
```

```
% Condições iniciais de temperatura nas bordas
```

```
T_1 = 10;
```

```
T_2 = 10;
```

```
T_3 = 10;
```

```
T_4 = 10;
```

```
% Calcular coeficiente de difusão térmica
```

```
k = hy^2 / hx^2;
```

```
% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_old = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Iniciar temporizador
tic

% Laço principal do método de Gauss-Seidel
while (erro > eps)
    ciclos = ciclos + 1;

    % Iteração do método de Gauss-Seidel
    [T, erro] = GaussSeidel(T, b, Niter, k, hy);

end

% Parar temporizador e exibir tempo de execução
toc
```

A.3.1 MULTIGRID COM GAUSS - SEIDEL

2 NÍVEIS

```
% Limpar console e workspace
clc;
clear;
```

```
% Parâmetros
Niter = 1;
Miter = 20;
N = 129;
x = 2;
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 35;
T_2 = 35;
T_3 = 35;
T_4 = 35;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_old = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;
```

```
% Inicialização de variáveis
erro2_inicial = zeros(65, 65);

% Iniciar temporizador
tic

% Laço principal do método de Gauss-Seidel com multigrid
while (erro > eps)
    % Iteração do método de Gauss-Seidel
    [T, erro] = GaussSeidel(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Gauss-Seidel para o segundo resíduo
    erro2 = GaussSeidel2(erro2_inicial, A1, 4, k, 2 * hy);

    % Prolongamento e correção do campo de temperatura
    corr1 = prolongamento(erro2);
    T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc
```

3 NÍVEIS

```
% Limpar console e workspace
clc;
clear;

% Parâmetros
Niter = 1;
Miter = 20;
N = 129;
x = 2;
```

```
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_old = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;

% Inicialização de variáveis
erro2_inicial = zeros(65, 65);
erro3_inicial = zeros(33, 33);

% Iniciar temporizador
tic
```

```
% Laço principal do método de Gauss-Seidel com multigrid de 3 níveis
while (erro > eps)
    % Iteração do método de Gauss-Seidel
    [T, erro] = GaussSeidel(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Gauss-Seidel para o segundo resíduo
    erro2 = GaussSeidel2(erro2_inicial, A1, 4, k, 2 * hy);

    % Cálculo do resíduo
    r1 = residuo(erro2, A1, 2 * hx);
    A2 = restricao(r1);

    % Iteração do método de Gauss-Seidel para o terceiro resíduo
    erro3 = GaussSeidel2(erro3_inicial, A2, 8, k, 4 * hy);

    % Prolongamento e correção do segundo resíduo
    corr2 = prolongamento(erro3);
    erro2 = erro2 + corr2;

    % Prolongamento e correção do campo de temperatura
    corr1 = prolongamento(erro2);
    T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc
```

4 NÍVEIS

```
% Limpar console e workspace
clc;
clear;
```

```
% Parâmetros
Niter = 1;
Miter = 20;
N = 65;
x = 2;
y = 2;

% Calcular passos nas direções x e y
hx = x / (N - 1);
hy = y / (N - 1);

% Condições iniciais de temperatura nas bordas
T_1 = 10;
T_2 = 10;
T_3 = 10;
T_4 = 10;

% Calcular coeficiente de difusão térmica
k = hy^2 / hx^2;

% Inicializar matriz de fontes de calor
b = zeros(N, N);

% Condições iniciais de temperatura
T = zeros(N, N);
T(:, 1) = T_1;
T(1, :) = T_3;
T(:, N) = T_2;
T(N, :) = T_4;

T_new = T;

% Critérios de parada
erro = 1;
eps = 1e-6;

% Contador de iterações
ciclos = 0;
```

```
% Inicialização de variáveis
erro2_inicial = zeros(33, 33);
erro3_inicial = zeros(17, 17);
erro4_inicial = zeros(9, 9);

% Iniciar temporizador
tic

% Laço principal do método de Gauss-Seidel com multigrid de 4 níveis
while (erro > eps)
    % Iteração do método de Gauss-Seidel
    [T, erro] = GaussSeidel(T, b, 2, k, hy);

    % Contagem de ciclos
    ciclos = ciclos + 1;

    % Cálculo do resíduo
    r = residuo(T, b, hx);
    A1 = restricao(r);

    % Iteração do método de Gauss-Seidel para o segundo resíduo
    erro2 = GaussSeidel2(erro2_inicial, A1, 4, k, 2 * hy);

    % Cálculo do resíduo
    r1 = residuo(erro2, A1, 2 * hx);
    A2 = restricao(r1);

    % Iteração do método de Gauss-Seidel para o terceiro resíduo
    erro3 = GaussSeidel2(erro3_inicial, A2, 8, k, 4 * hy);

    % Cálculo do resíduo
    r2 = residuo(erro3, A2, 4 * k);
    A3 = restricao(r2);

    % Iteração do método de Gauss-Seidel para o quarto resíduo
    erro4 = GaussSeidel2(erro4_inicial, A3, 16, k, 8 * hy);

    % Prolongamento e correção do terceiro resíduo
    corr3 = prolongamento(erro4);
    erro3 = erro3 + corr3;
```

```
% Prolongamento e correção do segundo resíduo
corr2 = prolongamento(erro3);
erro2 = erro2 + corr2;

% Prolongamento e correção do primeiro resíduo
corr1 = prolongamento(erro2);
T = T + corr1;
end

% Parar temporizador e exibir tempo de execução
toc
```