Notas em Matemática Aplicada e-ISSN 2236-5915

Volume 61, 2012

Editores

Cassio Machiaveli Oishi
Universidade Estadual Paulista - UNESP
Presidente Prudente, SP, Brasil

Fernando Rodrigo Rafaeli
Universidade Estadual Paulista - UNESP
São José do Rio Preto, SP, Brasil

Rosana Sueli da Motta Jafelice (Editor Chefe)
Universidade Federal de Uberlândia - UFU
Uberlândia, MG, Brasil

Rubens de Figueiredo Camargo
Universidade Estadual Paulista - UNESP
Bauru, SP, Brasil

Sezimária de Fátima P. Saramago
Universidade Federal de Uberlândia - UFU
Uberlândia, MG, Brasil

Vanessa Avansini Botta Pirani (Editor Adjunto)
Universidade Estadual Paulista - UNESP
Presidente Prudente, SP, Brasil

SBMAC **Sociedade Brasileira de Matemática Aplicada e Computacional**

2012

A Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC publica, desde as primeiras edições do evento, monografias dos cursos que são ministrados nos CNMAC.

Para a comemoração dos 25 anos da SBMAC, que ocorreu durante o XXVI CNMAC em 2003, foi criada a série **Notas em Matemática Aplicada** para publicar as monografias dos minicursos ministrados nos CNMAC, o que permaneceu até o XXXIII CNMAC em 2010.

A partir de 2011, a série passa a publicar, também, livros nas áreas de interesse da SBMAC. Os autores que submeterem textos à série Notas em Matemática Aplicada devem estar cientes de que poderão ser convidados a ministrarem minicursos nos eventos patrocinados pela SBMAC, em especial nos CNMAC, sobre assunto a que se refere o texto.

O livro deve ser preparado em **Latex (compatível com o Miktex versão 2.7), as figuras em eps** e deve ter entre **80 e 150 páginas**. O texto deve ser redigido de forma clara, acompanhado de uma excelente revisão bibliográfica e de **exercícios de verificação de aprendizagem** ao final de cada capítulo.

Veja todos os títulos publicados nesta série na página
http://www.sbmac.org.br/notas.php

SBMAC **Sociedade Brasileira de Matemática Aplicada e Computacional**

2012

Introdução ao Método de Elementos Finitos:

Aplicação em Dinâmica dos Fluidos

Márcio da Silveira Carvalho
msc@puc-rio.br

Departamento de Engenharia Mecânica
Pontifícia Universidade Católica do Rio de Janeiro
Rua Marquês de São Vicente 225
Gávea, Rio de Janeiro, RJ, 22453-900

Juliana Vianna Valério
juvianna@dcc.ufrj.br

Departamento de Ciência da Computação
Instituto de Matemática
Universidade Federal do Rio de Janeiro
Caixa Postal 68.530, Rio de Janeiro, RJ, 21941-590

SBMAC **Sociedade Brasileira de Matemática Aplicada e Computacional**

São Carlos - SP, Brasil
2012

Coordenação Editorial: Elbert Einstein Nehrer Macau

Coordenação Editorial da Série: Rosana Sueli da Motta Jafelice

Editora: SBMAC

Capa: Matheus Botossi Trindade

Patrocínio: SBMAC

Catalogação elaborada pela Biblioteca do IBILCE/UNESP
Bibliotecária: Maria Luiza Fernandes Jardim Froner

Carvalho, Márcio S.
Introdução ao Método de Elementos Finitos:
Aplicação em Dinâmica dos Fluidos
- São Carlos, SP :
SBMAC, 2012, 121 p., 20.5 cm - (Notas em Matemática Aplicada; v. 61)

e-ISBN 978-85-8215-012-2

1. Fundamentos do método de elementos finitos
2. Escoamento incompressível

I. Carvalho, Márcio S. II. Valério, Juliana V. III. Título. IV. Série

CDD - 51

Agradecimentos

Somos gratos a todos os alunos que cursaram a disciplina Elementos Finitos em Fluidos do programa de pós-graduação em Engenharia Mecânica da Pontifícia Universidade Católica do Rio de Janeiro por seus comentários em relação às notas de aulas do curso. As críticas, sugestões e erros encontrados nas notas de aula foram de grande importância na elaboração deste livro.

Agradecemos ao Prof.Diego Campana da Universidad Nacional del Litoral pela cuidadosa revisão da versão final do livro.

Conteúdo

Prefácio

Este livro é baseado nas notas de aula da disciplina Elementos Finitos em Fluidos do programa de pós-graduação em Engenharia Mecânica da Pontifícia Universidade Católica do Rio de Janeiro, ministrada pelo Prof. Márcio Carvalho desde 1998. Não tem como objetivo substituir as diversas referências clássicas sobre o Método de Elementos Finitos, mas o de apresentar o mínimo necessário para o entendimento básico do método, os aspectos particulares de sua aplicação na solução de escoamentos de fluidos incompressíveis e detalhes de sua implementação numérica . Desta forma, a nossa proposta é que o livro seja usado como uma primeira leitura antes de um estudo mais aprofundado do assunto, para o qual sugerimos o livro Incompressible Flow and The Finite Element Method, por P. Gresho e R. Sani. O público alvo são alunos de ciências exatas (graduação e pós-graduação), que já tenham cursado cálculo a várias variáveis, equações diferencias (básico), métodos numéricos (básico) e álgebra linear.

Por ser uma leitura inicial, não apresentamos as diferentes formulações e métodos utilizados nos diferentes grupos de pesquisa na solução da equação de Navier-Stokes pelo Método de Elementos Finitos. Fazemos a apresentação dos conceitos fundamentais utilizando uma formulação mista, onde os campos de velocidade e pressão são projetados em espaços de funções distintos, e totalmente acoplada, i.e. as variáveis relacionadas à velocidade e pressão são resolvidas simultaneamente. As não-linearidades do problema são tratadas pelo método de Newton, que garante uma convergência quadrática quando perto da solução. Esta metodologia leva a um procedimento bastante robusto e eficiente.

Alguns detalhes do método e de sua implementação numérica são apresentados através de um código para solução de escoamento bidimensional de fluido incompressível, desenvolvido em ambiente MatLab. Vale ressaltar que os códigos apresentados não foram escritos buscando uma maior eficiência computacional, mas sim a clareza dos conceitos apresentados e discutidos ao longo deste trabalho. Desta forma, várias operações poderiam ser realizadas de forma muito mais eficiente utilizando recursos computacionais específicos do MatLab. O embasamento teórico e o código são apresentados de forma que um aluno no final da graduação de qualquer curso de ciências exatas seja capaz de reproduzí-lo e utilizá-lo para simular um escoamento bidimensional. O código pode ser facilmente modificado para o estudo de escoamentos em outras geometrias e com diferentes condições de contorno e podem ser encontrados nos endereços:

$$http://carvalho.usuarios.rdc.puc-rio.br/elementosfinitos$$

$$http://www.dcc.ufrj.br/\ juvianna$$

Rio de Janeiro, 01 de março de 2012.

Márcio da Silveira Carvalho
Juliana Vianna Valério

Capítulo 1

Introdução

Com o avanço da computação digital, diferentes métodos numéricos foram desenvolvidos para a obtenção de soluções aproximadas de equações diferenciais que descrevem diferentes fenômenos físicos. Em qualquer método numérico, o processo de discretização de uma equação diferencial sempre leva a um sistema de equações algébricas.

1.1 Método de Diferenças Finitas

O método de diferenças finitas tornou-se o mais explorado inicialmente devido a sua simplicidade e a facilidade de implementação. O processo de discretização consiste essencialmente na aproximação das derivadas da função pela diferença entre valores que a função assume em certos pontos do domínio, chamados de nós. Assim a equação diferencial dá origem a um sistema de equações algébricas onde as incógnitas são os valores da função nos pontos escolhidos para representar as derivadas. As idéias básicas de métodos de diferenças finitas são mostradas no exemplo a seguir.

Exemplo 1

Resolver

$$\begin{cases} -\dfrac{d^2u}{dx^2} = f\left(x, u(x), \frac{du}{dx}\right), \quad \forall x \in (0,1); \\ u(0) = 0 \quad \text{e} \quad \left.\dfrac{du}{dx}\right|_{x=1} = 1. \end{cases}$$

O problema de valor de contorno acima é de segunda ordem e linear. A equação diferencial com as condições de contorno descrevem a distribuição de temperatura ao longo de uma barra com uma fonte de calor f. Uma das extremidades da barra está submetida a um fluxo de calor prescrito e a outra, a uma temperatura fixa.

A solução será aproximada através do cálculo do valor da função em um número finito de pontos do domínio.

Se o domínio for dividido em $N-1$ intervalos através de N pontos (nós), deve-se obter o valor aproximado de $u(x)$ em cada nó, ou seja, um problema com N incógnitas. As N equações necessárias para resolver este problema são obtidas aproximando a derivada em função dos valores nodais desconhecidos nas posições dos N nós.

Nesse caso, o valor da variável $u(x)$ em um dos extremos é conhecida, sendo então o número de incógnitas $N-1$.

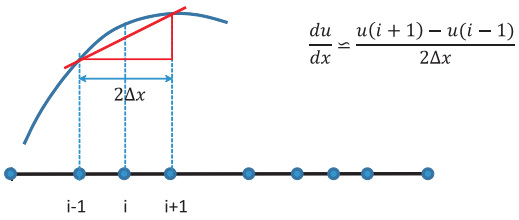

Figura 1.1: Aproximação da derivada da função por diferenças finitas.

A equação diferencial dá origem a um sistema de equações algébricas, uma equação para cada ponto de malha. Para isso, a derivada tem que ser escrita em função dos valores da função nos pontos. Dentre outras opções nas quais as aproximações se confirmam quando $\Delta x \to 0$, comumente aproxima-se as derivadas de primeira (ilustrada na figura

1.1) e segunda ordem como nas equações (1.1.1).

$$\begin{aligned} \frac{du}{dx} &\approx \frac{u(x+\Delta x) - u(x-\Delta x)}{2\Delta x} \\ \frac{d^2u}{dx^2} &\approx \frac{u(x+\Delta x) - 2u(x) + u(x-\Delta x)}{\Delta x^2} \end{aligned} \tag{1.1.1}$$

Em $x_1 = 0$ a equação não precisa ser aproximada uma vez que $u(0) = 0$ pela condição de contorno.
Em x_i, com i percorrendo todos os nós internos do domínio e aproximando a derivada, a equação diferencial é escrita como:

$$-\frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{\Delta x^2} = f(x_i)$$

Em $x_N = 1$ a equação a ser aproximada, $\dfrac{du}{dx} = 1$, novamente segue a condição de contorno.

$$\frac{u(x_N) - u(x_{N-1})}{\Delta x} = 1$$

Por não haver ponto que interesse depois de x_N, a aproximação escolhida para a primeira derivada foi a diferença para trás. Observe que essa escolha é menos precisa que a diferença central apresentada na equação (1.1.1).

Resolvendo o exemplo para $f = u - x^2$, a solução analítica é:

$$u(x) = \frac{2\cos(x-1) - \sin(x)}{\cos(1)} + x^2 - 2$$

Organizando as equações e usando que $\Delta x = h$, o sistema linear que aproxima a equação diferencial pode ser escrito matricialmente como:

$$\mathbf{M} = \begin{bmatrix} 2-h^2 & -1 & 0 & \dots & 0 \\ -1 & 2-h^2 & -1 & 0 & \vdots \\ 0 & & \ddots & & 0 \\ \vdots & 0 & -1 & 2-h^2 & -1 \\ 0 & & & -1 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} h^2x_1^2 \\ h^2x_2^2 \\ \vdots \\ h^2x_{N-1}^2 \\ h \end{bmatrix}$$

Ao resolver o sistema linear $\mathbf{Mc} = \mathbf{b}$ encontra-se $\mathbf{c}$ que é um vetor contendo o valor aproximado de u em cada ponto do domínio.

A figura 1.2 apresenta os gráficos das soluções numérica e analítica com apenas 15 intervalos, essa solução tem um erro absoluto ($||\mathbf{c} - u(x_i)||_\infty$) de 0.0075. Interessante observar que a aproximação atinge uma precisão da ordem de 10^{-5} somente para mais de 1000 intervalos.

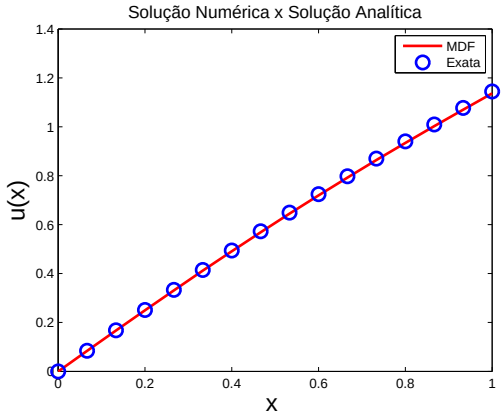

Figura 1.2: Solução **Exemplo 1**: Método de Diferenças Finitas × Exata.

O método de diferenças finitas é simples em geometrias regulares, porém, ao tratar problemas reais de engenharia, envolvendo domínios irregulares, o método de diferenças finitas perde muitas de suas vantagens e simplicidade.

Uma alternativa muito atrativa para domínios complexos é o método de elementos finitos. O método de elementos finitos faz parte de uma classe de métodos variacionais desenvolvidos nos séculos XIX e XX por pesquisadores como Rayleigh (1896), Ritz (1908) e Galerkin (1915).

1.2 Métodos Variacionais

Nos métodos variacionais, busca-se uma solução aproximada de uma equação diferencial em um espaço de funções apropriado de dimensão finita. A solução aproximada é uma projeção da solução exata nesse

subespaço e pode ser representada como uma combinação linear de um número finito de funções conhecidas, que formam a base deste subespaço. As incógnitas do sistema de equações algébricas resultante do processo de discretização são os coeficientes desta expansão linear. O que faz o método de elementos finitos especialmente atrativo é o fato dessas funções base serem diferentes de zero em apenas em uma pequena parte do domínio e descrever sem dificuldades adicionais a solução aproximada em geometrias variadas. A robustez e eficiência do método dos elementos finitos o tornou bastante popular nos vários problemas modelados por equações diferenciais, incluindo dinâmica dos fluidos.

Nos métodos variacionais a solução aproximada é escrita como combinação linear de funções base apropriadamente escolhidas:

$$u(x) = \sum_{i=1}^{N} c_i \phi_i.$$

Os coeficientes c_i são obtidos de forma que a equação diferencial seja satisfeita. Para obter a solução aproximada através de um método variacional, deve-se reescrever o problema em sua forma fraca (integral ou variacional). A necessidade do uso da formulação fraca é ilustrada no Exemplo 2. Os diferentes métodos variacionais utilizam diferentes formas integrais, funções base e funções peso.

A simples substituição da aproximação $u(x) = \sum_{i=1}^{N} c_i \phi_i$ na equação diferencial não garante a obtenção de N (número de coeficientes c_i) equações algébricas linearmente independentes, por isso em métodos variacionais utiliza-se a forma integral do problema, como mostrado no exemplo a seguir.

Exemplo 2

Usando o mesmo problema do exemplo anterior, dado que conhecemos a solução analítica e podemos calcular o erro da aproximação. Resolver

$$\begin{cases} -\dfrac{d^2u}{dx^2} = u - x^2, \quad \text{para } 0 \leq x \leq 1 \ ; \\ u(0) = 0 \quad \text{e} \quad \left.\dfrac{du}{dx}\right|_{x=1} = 1. \end{cases}$$

Solução aproximada na forma:

$$u(x) = \phi_0(x) + \sum_{i=1}^{2} c_i \phi_i(x),$$

onde as funções base escolhidas são:

$\phi_0(x) = x,$

$\phi_1(x) = x^2 - 2x,$

$\phi_2(x) = x^3 - 3x.$

Precisamos encontrar os coeficientes c_1 e c_2, o problema possui 2 incógnitas. Então substituindo ϕ_0, ϕ_1 e ϕ_2, a solução aproximada torna-se:

$$u(x) = x + c_1[x^2 - 2x] + c_2[x^3 - 3x], \text{ derivando}$$
$$\frac{du}{dx}(x) = 1 + 2c_1x - 2c_1 + 3c_2x^2 - 3c_2, \text{ novamente}$$
$$\frac{d^2u}{dx^2}(x) = 2c_1 + 6c_2x.$$

Observe que as condições de contorno são satisfeitas para qualquer valor de c_1 e c_2, que devem ser determinados de forma que a aproximação satisfaça a equação diferencial em algum sentido.

Obrigando a solução aproximada a satisfazer a equação diferencial no sentido exato, isto é, em todos os pontos do domínio:

$$-(2c_1 + 6c_2x) - \big(x + c_1(x^2 - 2x) + c_2(x^3 - 3x)\big) + x^2 = 0,$$
$$\Rightarrow\ -2c_1 + (2c_1 - 3c_2 - 1)\,x + (1 - c_1)x^2 + c_2x^3 = 0.$$

Para essa expressão ser válida para qualquer x, o coeficiente de cada potência de x deve ser nulo:

$$\begin{cases} -2c_1 = 0, \\ 2c_1 - 3c_2 - 1 = 0, \\ 1 - c_1 = 0, \\ c_2 = 0. \end{cases}$$

Esse sistema não tem solução! Vamos obrigar a solução aproximada a satisfazer a equação diferencial num sentido integral ao invés de no sentido exato.

Podemos obrigá-la a satisfazer a seguinte integral ponderada:

$$\int_0^1 w(x) \left[-\frac{d^2u}{dx^2} - u + x^2 \right] dx = 0,$$

onde $w(x)$ é a função peso.

Esta integral pode representar uma medida do erro na aproximação. O número de equações linearmente independentes que envolvam c_1 e c_2 será igual ao número de funções peso utilizadas (também linearmente independentes). No exemplo escolhemos 2 funções peso linearmente independentes: $w = 1$ e $w = x$, teremos:

$$\int_0^1 1 \left[-2c_1 + (2c_1 - 3c_2 - 1)\,x + (1 - c_1)x^2 + c_2x^3 \right] dx = 0,$$
$$\int_0^1 x \left[-2c_1 + (2c_1 - 3c_2 - 1)\,x + (1 - c_1)x^2 + c_2x^3 \right] dx = 0.$$

Então:

$$\begin{cases} -\dfrac{4}{3}c_1 - \dfrac{7}{4}c_2 = \dfrac{1}{6}, \\ -\dfrac{7}{12}c_1 - \dfrac{6}{5}c_2 = \dfrac{1}{12}, \end{cases}$$

e finalmente, $c_1 = -\dfrac{13}{139}$ e $c_2 = -\dfrac{10}{417}$. O erro absoluto comparando a solução aproximada e a exata é 0.0081.

As funções bases utilizadas e a solução aproximada estão ilustradas na figura 1.3:

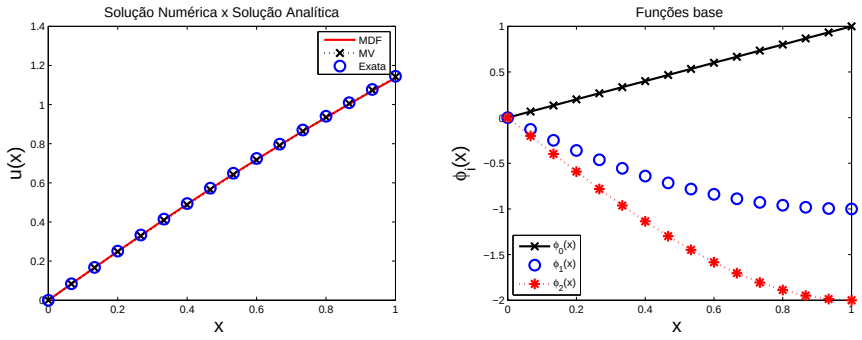

Figura 1.3: Esquerda: solução **Exemplo 2**: Método Diferenças Finitas (MDF)× Método Variacional (MV) × Exata. Direita: funções base $\phi_0(x), \phi_1(x), \phi_2(x)$ utilizadas no Método Variacional.

Código em MatLab referente aos exemplos 1 e 2.

```
%Exemplo 1 - Método de Diferenças Finitas

clear clc  % limpa memória e console

N = 15; % número de intervalos entre 0 e 1
h = 1/N; % distância de um no a outro no domínio
M = zeros(N-1,N-1);% alocando espaço para matriz
b = zeros(N-1,1); % alocando espaço para o vetor independente.
%M e b compõem o sistema linear a ser resolvido:  M c = b.
% em x = 0, u = 0 pela condição de contorno.
x = [h:h:1]; % vetor que inicia em h e vai até 1 de h em h.

i = 1
M(i,i) = 2 - h^2;
M(i,i+1) = -1;
b(i,1) = -h^2 * x(i)^2;
for i = 2:N-1
    M(i,i-1) = -1;
    M(i,i) = 2 - h^2;
    M(i,i+1) = -1;
```

```
    b(i,1) = -h^2 * x(i)^2;
end
i = N;
M(i,i-1) = -1;
M(i,i) = 1;
b(i,1) = h;

%resolvendo o sistema linear via MatLab
c = M\b;

% Colocando o zero no vetor x e no vetor resolvido c (que representa u nos
pontos nodais) para fazer os gráficos
xp = [0:h:1]; cp = [0;c];
% Solução analítica:
y = (2*cos(xp-1) - sin(xp))/cos(1) + xp.^2 - 2;

% Erro absoluto entre a solução numérica por diferenças finitas e a solução
analítica usando a norma sup
Erro = norm(cp - y', inf)

%Gráfico apresentado na figura
figure;
plot(xp,cp,'-r',xp,y,'bo','LineWidth',2,'MarkerSize',10)
legend('MDF','Exata')
xlabel('x ','FontSize',18)
ylabel('u(x)','FontSize',18)
title ('Solução Numérica x Solução Analítica','FontSize',13)

% Exemplo 2 - Método Variacional

A = [-4/3, -7/4; -7/12, -6/5];   b_v = [1/6; 1/12];
%A e b_v compõem o sistema linear a ser resolvido:  Av = b_v.
v = A\b_v;
v = xp + v(1)*(xp.^2 - 2*xp) + (v(2))*(xp.^3 - 3*xp);

% Erro absoluto entre a solução numérica método variacional e analítica
Erro2 = norm(v - y, inf)

%Gráficos apresentados na figura
figure;
plot(xp,cp,'-r',xp,v,':k+',xp,y,'bo','LineWidth',2,'MarkerSize',10)
legend('MDF','MV','Exata')
xlabel('x ','FontSize',18)
ylabel('u(x)','FontSize',18)
title ('Solução Numérica x Solução Analítica','FontSize',13,'MarkerSize',10)

figure;
plot(xp, xp,' k-', x', xp, (xp.^2 - 2*xp),' bo', xp, (xp.^3 - 3*xp),'r:*','LineWidth',2)
legend('phi_0(x)',' phi_1(x)',' phi_2(x)', 3)
title ('Funçõe base','FontSize',13)
xlabel('x ','FontSize',18)
ylabel('phi_i(x)','FontSize',18)
```

1.3 Exercícios

1. Seja $f(x)$ uma função contínua com derivadas contínuas no intervalo (a, b). Denotaremos $\frac{d^k f}{dx^k}(x)$ por $f^{(k)}(x)$ a k-ésima derivada de f. Do teorema de Taylor $f(x)$ pode ser escrita na vizinhança de $x \in (x-h, x+h), h << 1$, da seguinte forma:

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \cdots + \frac{h^k}{k!}f^{(k)}(x) + \cdots$$

A partir disso, deduza uma aproximação para primeira derivada de f:
a) utilizando x e $x+h$ - diferença para frente (progressiva),
b) utilizando x e $x-h$ - diferença para trás (regressiva),
c) utilizando $x-h$ e $x+h$ - diferença central.
Compare a ordem do erro em cada uma das aproximações.

2. Deduza a partir do teorema de Taylor a aproximação apresentada na equação (1.1.1) para segunda derivada de f. Qual é a ordem do erro.

3. Para encontrar a solução do exemplo 2, dada a quantidade e complexidade das funções base ϕ_j, foi possível chegar ao sistema linear algebricamente. Monte um esquema numérico para que, quando necessário, as contas possam ser todas feitas computacionalmente. Discuta pelo menos três possibilidades de algoritmo para integral numérica.

4. Um problema aplicado: Qual é a distribuição de temperatura em uma barra de tamanho 10 cm sujeita a temperatura nos extremos fixa, $T(0) = 40$ e $T(10) = 200$, e a uma fonte de calor constante $f(x) = 10$. Resolva: $\frac{d^2T}{dx^2} = -f(x)$ com $T(0) = 40$ e $T(10) = 200$,

a) por diferenças finitas - aproximação central para a derivada.
b) variacionalmente - mesmas funções base e peso do exemplo 2.
c) calcule o erro absoluto e relativo comparando com a solução analítica.

Capítulo 2

Formulação Integral

Um problema simples foi escolhido como modelo para que os conceitos e definições sobre a formulação forte e fraca sejam explorados e também para mostrar a equivalência entre as duas formulações. Trabalhando ainda em cima do mesmo problema, o método dos Resíduos Ponderados vai ser apresentado juntamente com um exemplo numérico.

2.1 Formulação Forte e Fraca de um dado Problema

Vamos explorar o conceito de formulação forte e fraca e sua equivalência em um determinado problema de valor de contorno:

Determine $u : [0,1] \to \mathbb{R}$, diferenciável (com pelo menos até segunda derivada), tal que

$$(S)\begin{cases} \dfrac{d^2u}{dx^2} + f = 0, & \text{em } (0,1); \\ u(1) = g & \to \text{c.c. de Dirichlet;} \\ \dfrac{du(0)}{dx} = -h & \to \text{c.c. de Neumann.} \end{cases}$$

Na formulação forte (S) a equação diferencial é imposta em todos os pontos $x \in (0,1)$. A formulação fraca (W) equivalente a (S) é definida

como:

Seja Ω um aberto do $\mathbb{R}^n$ com fronteira suave, denominada Γ, determine $u \in U$, tal que

$$(W) \int_0^1 \frac{dw}{dx}\frac{du}{dx}dx = \int_0^1 fwdx + w(0)h\,; \quad \text{para qualquer } w \in V.$$

U é o conjunto de funções base definido como:

$$U = \left\{ u : \Omega \to \mathbb{R} \middle| u(1) = g \text{ e} \int_0^1 \left(\frac{du}{dx}\right)^2 dx < \infty \right\}$$

V é o conjunto de funções peso e é definido como:

$$V = \left\{ \underbrace{w : \Omega \to \mathbb{R} \middle| w(1) = 0}_{**} \text{ e} \int_0^1 \left(\frac{dw}{dx}\right)^2 dx < \infty \right\}$$

** A função peso vale zero quando aplicada no contorno onde a condição é essencial. Isso porque neste ponto a função já é conhecida.

Antes de continuarmos com a definição de formulação fraca e equivalência entre as duas formulações precisaremos das definições:

• Uma função $f : \Omega \to \mathbb{R}$ pertence ao espaço $C^k(\Omega)$ se $\dfrac{d^j f}{dx^j}$ é contínua, com $0 \leqslant j \leqslant k$. C^0 é o espaço das funções contínuas.

• O espaço Sobolev de funções $H^k(\Omega)$ é definido como:

$$H^k(\Omega) = \left\{ w \middle| w \in L_2;\ \frac{dw}{dx} \in L_2; ...; \frac{d^k w}{dx^k} \in L_2 \right\}, \text{onde}$$
$$L_2(\Omega) = \left\{ f \middle| \int_\Omega f^2 d\Omega < \infty \right\}.$$

• Um subespaço importante do espaço $H^1(\Omega)$ é o espaço $H_0^1(\Omega)$, definido pelas funções $v \in H^1(\Omega)$ que se anulam na fronteira Γ de Ω.

Logo podemos escrever U e V como:

$$U = \left\{ u : \Omega \to \mathbb{R} \middle| u \in H^1 \text{ e } u(1) = g \right\},$$

$$V = \left\{ w : \Omega \to \mathbb{R} \middle| w \in H^1 \text{e } w(1) = 0 \right\}.$$

Notas:

• Se $f \in H^1 \Rightarrow f$ é contínua e limitada.

• A condição de contorno de Dirichlet é imposta na definição do espaço de funções base $\rightarrow$ c.c. essencial.

• A condição de contorno de Neumann é imposta naturalmente na formulação $\rightarrow$ c.c. natural.

Vamos mostrar, de maneira pouco rigorosa, que as duas formulações são equivalentes, fazendo:

(i) Se u é solução de $(S) \Rightarrow u$ é solução de (W).

(ii) Se u é solução de $(W) \Rightarrow u$ é solução de (S).

A demonstração da parte (i), na realidade, corresponde ao método de achar a formulação fraca equivalente a uma dada formulação forte. Se u é solução de (S), então u satisfaz a equação diferencial em todos os pontos $x \in (0,1) \Rightarrow$

$$\Rightarrow \int_0^1 w \left[\frac{d^2u}{dx^2} + f \right] dx = 0 \quad \left(w \in H^1(0,1) \right)$$

Fazendo uma integração por partes:

$$\Rightarrow - \int_0^1 \frac{dw}{dx} \frac{du}{dx} dx + w(1) \frac{du(1)}{dx} - w(0) \frac{du(0)}{dx} + \int_0^1 wf dx = 0.$$

Como $w \in V \Rightarrow w(1) = 0$.

Como u é solução de $(S) \Rightarrow \dfrac{du(0)}{dx} = -h$.

$$\Rightarrow \int_0^1 \frac{dw}{dx} \frac{du}{dx} dx = w(0)h + \int_0^1 wf dx, \ \ \forall w \in V$$

Logo, se u é solução de (S), u satisfaz (W).

Demonstração da parte (ii):
Se u é solução de (W), então $u \in U \Rightarrow u(1) = g$ e

$$\int_0^1 \frac{dw}{dx}\frac{du}{dx}dx = w(0)h + \int_0^1 wf dx, \ \ \forall w \in V$$

Integrando por partes obtém-se:

$$-\int_0^1 w\frac{d^2u}{dx^2}dx + w(1)\frac{du(1)}{dx} - w(0)\frac{du(0)}{dx} = \int_0^1 wf dx + w(0)h \Rightarrow$$

$$\Rightarrow \int_0^1 w\Big[\frac{d^2u}{dx^2} + f\Big]dx - w(1)\frac{du(1)}{dx} + w(0)\Big[\frac{du(0)}{dx} + h\Big] = 0.$$

Como $w \in V \Rightarrow w(1) = 0$

$$\Rightarrow \int_0^1 w\Big[\frac{d^2u}{dx^2} + f\Big]dx + w(0)\left[\frac{du(0)}{dx} + h\right] = 0 \ ; \ \forall w \in V. \qquad (2.1.1)$$

Para provar que u é solução de (S), temos que mostrar que:
(a) $\dfrac{d^2u}{dx^2} + f = 0$ para qualquer $x \in (0,1)$ e (b) $\dfrac{du(0)}{dx} = -h$.

Como a equação (2.1.1) é válida para qualquer $w \in V$, vamos escolher um w de forma a concluirmos que $\dfrac{d^2u}{dx^2} + f = 0$.

Seja $w(x) = \phi(x)\Big[\dfrac{d^2u}{dx^2} + f\Big]$ com $\phi(0) = \phi(1) = 0$ e $\phi(x) > 0$, $x \in (0,1)$. Assim, $w(0) = 0$, e a equação (2.1.1) , torna-se:
$\displaystyle\int_0^1 \phi(x)\Big[\frac{d^2u}{dx^2} + f\Big]^2 dx = 0$. Por definição $\phi(x) > 0$ e $\Big[\dfrac{d^2u}{dx^2} + f\Big]^2 \geq 0$ então a integral será zero se e somente se $\Big[\dfrac{d^2u}{dx^2} + f\Big]^2 = 0 \Rightarrow \dfrac{d^2u}{dx^2} + f = 0$.
Com isso temos (a).

Usando (a) na expressão (2.1.1) temos (b) simplesmente:

$$w(0)\Big[\frac{du(0)}{dx} + h\Big] = 0 \ ; \ \forall w \in V.$$

Como o espaço V não restringe o valor de $w(0)$, então

$$w(0)\Big[\frac{du(0)}{dx} + h\Big] = 0 \Leftrightarrow \frac{du(0)}{dx} + h = 0 \Rightarrow \frac{du(0)}{dx} = -h$$

Com (i) e (ii) verificados, temos que se u é solução de (W) então é também solução de (S).

Como mencionado anteriormente, o método de Elementos Finitos é baseado na formulação fraca do problema. Busca-se uma solução aproximada de uma equação diferencial em um espaço apropriado de dimensão finita. A solução aproximada é uma projeção da solução exata nesse subespaço e pode ser representada como uma combinação linear de um número finito de funções conhecidas, que formam a base deste subespaço. Isto implica que qualquer $u \in U$ pode ser escrito como:

$$u(x) = \sum_{i=1}^{N} c_i \phi_i,$$

onde os $\phi_i's$ formam uma base de U.

A fim de simplificar, é comum seguir a notação:

$$a(u, w) = \int_0^1 \frac{dw}{dx}\frac{du}{dx} \quad \text{e} \quad (f, w) = \int_0^1 f w dx,$$

logo, a formulação torna-se:
Achar $u \in U$, tal que:

$$a(u, w) = (f, w) + w(0)h\,;\ \forall w \in V.$$

Repara que essa maneira de representar explicita o produto interno de funções, dado que a interpretação de produto interno é exatamente esta. Nos livros cuja a teoria dos métodos variacionais é encontrada é mais comum a notação simplificada, enquanto que nos livros aplicados, a mais comum é a que apresenta explicitamente a integral.

2.2 Método dos Resíduos Ponderados

Vamos novamente analisar um problema específico com o objetivo de introduzir o Método dos Resíduos Ponderados (Weighted Residuals).

Encontrar uma aproximação u_h para a solução do problema u:

$$\begin{cases} \dfrac{d^2u}{dx^2} = -f, \forall x \in (0,1), \\ u(1) = g = 0 \qquad\qquad \text{considerar } g = 0, \text{ apenas para simplificar.} \\ \dfrac{du(0)}{dx} = -h, \end{cases}$$

onde u_h vai pertencer a um espaço de dimensão finita U_h, logo podemos escrevê-lo como:

$$u_h(x) = \sum_{i=1}^{N} c_i \phi_i,$$

sendo N é a dimensão do espaço e $\phi_i's$ as funções base de U_h. Queremos também que $U_h \subset U$ logo, $u_h(1) = 0$.

Como u_h é apenas uma aproximação de u, $\dfrac{d^2u_h}{dx^2} \neq -f$

Vamos chamar de resíduo (R) da equação, a medida de quanto u_h aproxima adequadamente o problema.

$$R = \frac{d^2u_h}{dx^2} + f,$$

Observe que se $u_h = u$ então, $R = 0$.

Métodos dos Resíduos Ponderados são obtidos através da seguinte integral:

$$\int_0^1 wR \; dx = 0$$

Observe que integrando o resíduo com o peso w, estamos fazendo com que sua média ponderada seja 0. Por isso a condição é dita **fraca**, pois não estamos obrigando ponto a ponto que $u = u_h$ e sim na média (na integral).

Substituindo R, temos: $\int_0^1 w\left[\frac{d^2u_h}{dx^2}+f\right]dx=0$ e integrando por partes: $-\int_0^1 \frac{dw}{dx}\frac{du_h}{dx}dx+\int_0^1 fwdx+w(0)h=0$, equivalente à formulação variacional.

A integração por partes alivia as restrições impostas à função aproximada u_h. A expressão passa a depender da primeira derivada de u_h, $\frac{du_h}{dx}$ e não mais da segunda $\frac{d^2u_h}{dx^2}$. Logo, a restrição sobre u_h deverá ser que sua derivada pertença a L_2, $\frac{du_h}{dx}\in L_2$, isto é, o quadrado de $\frac{du_h}{dx}$ seja integrável. Observe que até funções contínuas com descontinuidades nas derivadas podem satisfazer a estas condições, como ilustrada na figura(2.1):

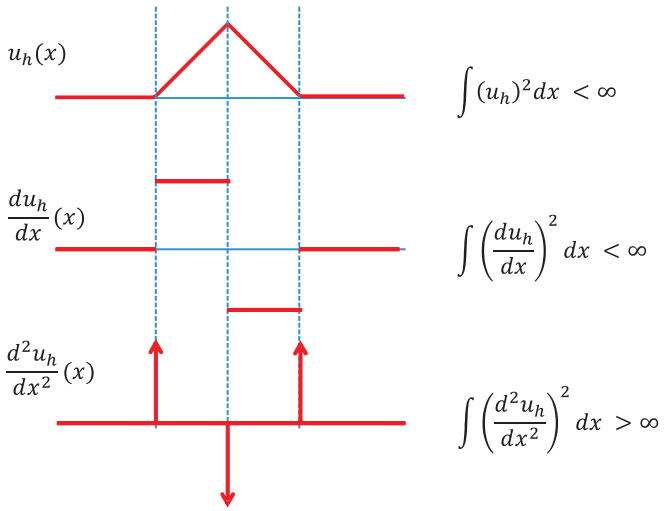

Figura 2.1: Funções contínuas com derivadas descontínuas.

Como $u_h \in U_h \Rightarrow u_h(x)=\sum_{i=1}^{N} c_i\phi_i$, então, $\frac{du_h}{dx}=\sum_{i=1}^{N} c_i\frac{d\phi_i}{dx}$.

Sendo o espaço U_h de dimensão N precisaremos de N equações linearmente independentes a fim de determinar os coeficientes c_i.

As funções peso w devem pertencer a um espaço finito $V_h \subset V$. Esse espaço pode ser representado pela base ψ_i. Cada equação está relacionada com as funções peso ψ_i.

Deste modo as N equações linearmente independentes para determinar os c_i são:

$$R_i = -\int_0^1 \frac{d\psi_i}{dx}\left(\sum_{j=1}^{N} c_j \frac{d\phi_j}{dx}\right) dx + \int_0^1 f\psi_i dx + \psi_i(0)h = 0 \Rightarrow$$

$$\Rightarrow \sum_{j=1}^{N} \underbrace{\left\{\int_0^1 \frac{d\psi_i}{dx}\frac{d\phi_j}{dx}dx\right\}}_{A_{ij}} c_j = \underbrace{\int_0^1 f\psi_i dx + \psi_i(0)h}_{b_i} \; ; \; i = 1, ..., N.$$

repare que a equação que descreve o problema foi denominada R_i, ou seja: $R_i = \int_0^1 wR\, dx$ no qual desejamos encontrar a aproximação de forma a obtermos $R_i = 0$.

Matricialmente temos $\boxed{\mathbf{Ac} = \mathbf{b}}$ algumas referências chamam $\mathbf{A}$ de *matriz de rigidez*.

Como já mencionado, em qualquer método numérico, o processo de discretização de uma equação diferencial sempre leva a um sistema de equações algébricas. Sabendo que tanto o custo computacional como a acurácia das soluções estão diretamente ligadas ao condicionamento da matriz, muitas vezes é importante considerar a forma em que a matriz vai se apresentar para escolher maneiras de discretização.

O Método dos Resíduos Ponderados recebe diferentes denominações de acordo com as funções base ϕ_i e peso ψ_i.

Método de Galerkin

A solução aproximada é uma projeção ortogonal da solução exata no subespaço de dimensão finita. O espaço das funções peso é idêntico ao espaço das funções base: $\psi_i = \phi_i$. Com isso a matriz $\mathbf{A}$ é simétrica.

Método de Petrov-Galerkin

A projeção não é mais ortogonal, $\psi_i \neq \phi_i \Rightarrow$ Matriz $\mathbf{A}$ não é simétrica.

Método dos Mínimos Quadrados

Os coeficientes c_i da aproximação $u_h(x) = \sum_{i=1}^{N} c_i \phi_i$ são obtidos através da minimização da integral do quadrado dos resíduos:

$$\frac{\partial}{\partial c_i} \int_0^1 R^2 dx = 0 \;\Rightarrow\; \int_0^1 \frac{\partial R}{\partial c_i} R dx = 0, \quad \text{assim} \quad \phi_{\mathrm{i}} = \frac{\partial \mathrm{R}}{\partial \mathrm{c_i}}.$$

Método de Colocação

O resíduo R é identicamente nulo em N pontos do domínio.

$$R(\mathbf{x_i}) = 0.$$

Também é um caso particular de Método dos Resíduos Ponderados.

$$w_i = \delta(\mathbf{x} - \mathbf{x_i}) \Rightarrow \int_0^1 \delta(\mathbf{x} - \mathbf{x_i}) R dx = 0 \;\Rightarrow R(\mathbf{x_i}) = 0.$$

Exemplo 1

Considere novamente o problema modelo:

$$\begin{cases} -\dfrac{d^2u}{dx^2} - u + x^2 = 0, \quad \text{onde } 0 \leq x \leq 1; \\ u(0) = 0, \\ u'(1) = 1. \end{cases}$$

Admitindo que uma solução aproximada u_h possa ser escrita como:

$$u(x) = \sum_{i=1}^{N} c_i \phi_i \qquad \text{onde, } \phi_i(0) = 0\,;$$

$$\phi_1(x) = x\,, \quad \phi_2(x) = x^2\,, \quad \phi_3(x) = x^3\,.$$

Encontrar a solução para o problema utilizando o método de Galerkin, ou seja , $\phi_i = \psi_i$.

$$R(x) = -\frac{d^2 u_h}{dx^2} - u_h + x^2,$$

queremos que:

$$\int_0^1 w(x) R(x) dx = 0, \qquad \text{onde } w(x) \text{ é a função peso, combinação}$$

linear das funções ψ_i.

$$\int_0^1 \psi_i \left[-\frac{d^2 u_h}{dx^2} - u_h + x^2 \right] dx = 0, \quad i = 1 \cdots N,$$

$$-\int_0^1 \psi_i \frac{d^2 u_h}{dx^2}\, dx - \int_0^1 \psi_i\, u_h\, dx + \int_0^1 \psi_i\, x^2\, dx = 0, \quad i = 1 \cdots N;$$

integrando por partes, evitando assim a segunda derivada,temos:

$$\frac{du_h}{dx}\psi_i \Big|_0^1 - \int_0^1 \left(\frac{d\psi_i}{dx} \frac{du_h}{dx} \right) dx - \int_0^1 \psi_i\, u_h\, dx + \int_0^1 \psi_i\, x^2\, dx = 0$$

usando que: $u_h'(1) = 1$ e $\psi_i(0) = 0$:

$$-\psi_i(1) + \int_0^1 \left(\frac{d\psi_i}{dx} \frac{du_h}{dx} \right) dx - \int_0^1 \psi_i\, u_h\, dx + \int_0^1 \psi_i\, x^2\, dx = 0,$$

como $u(x) = \sum_{i=1}^{N} c_j \phi_j$, então $\frac{du}{dx} = \sum_{i=1}^{N} c_j \frac{d\phi_j}{dx}$; substituindo temos:

$$\sum_{i=1}^{N} c_j \underbrace{\int_0^1 \left[\frac{d\psi_i}{dx}\frac{d\phi_j}{dx} - \psi_i \phi_j \right] dx}_{A_{ij}} = \underbrace{\psi_i(1) - \int_0^1 \psi_i \; x^2 \; dx}_{b_i}, \quad i = 1, \cdots N;$$

pelo método de Galerkin $\phi_i = \psi_i$; então,

$$A_{ij} = \int_0^1 \left[\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} - \phi_i \phi_j \right] dx \text{ e } b_i = \phi_i(1) - \int_0^1 \phi_i \; x^2 \; dx$$

assim, usando $\phi_1(x) = x \Rightarrow \frac{d\phi_1}{dx} = 1$, $\phi_2(x) = x^2 \Rightarrow \frac{d\phi_2}{dx} = 2x$, $\phi_3(x) = x^3 \Rightarrow \frac{d\phi_3}{dx} = 3x^2$:

$$\mathbf{A} = \begin{bmatrix} \frac{2}{3} & \frac{3}{4} & \frac{4}{5} \\ \\ \frac{3}{4} & \frac{17}{15} & \frac{4}{3} \\ \\ \frac{4}{5} & \frac{4}{3} & \frac{58}{35} \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} \frac{3}{4} \\ \\ \frac{4}{5} \\ \\ \frac{5}{6} \end{bmatrix}$$

Chegamos ao sistema linear $\mathbf{Ac} = \mathbf{b}$, que pode ser resolvido por mais de um método.

Encontramos $c(1) = \frac{2280}{1777}$; $c(2) = \frac{-203}{1777}$; $c(3) = \frac{-175}{7108}$, que quando substituído em u_h e comparado à solução analítica revela um erro absoluto máximo de 0.0013 utilizando-se apenas de três funções base. O gráfico da solução aproximada pelo método de Galerkin e da solução exata é mostrado na figura 2.2.

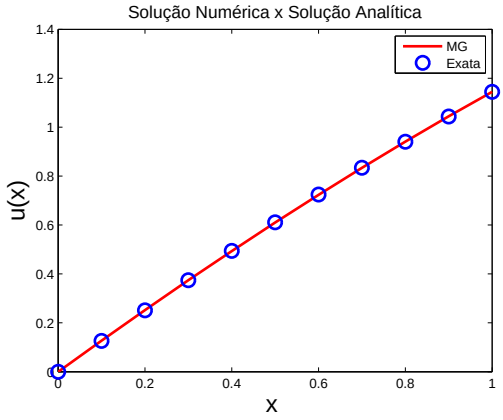

Figura 2.2: Solução **Exemplo 1**: Método de Galerkin (MG)× Exata.

2.3 Exercícios

1. Considere o problema a ser resolvido pelo método de Galerkin:

$$\begin{cases} -\dfrac{d^2u}{dx^2} + u(x) - f(x) = 0, \quad \forall x \in (0,1); \\ u(0) = u(1) = 0. \end{cases}$$

   Utilizando $\phi_j(x) = \text{sen}(j\pi x), j = 1, \cdots, m$, conclua:
   a) a matriz $\mathbf{A}$ é diagonal,
   b) a solução do problema aproximado corresponde a série truncada da solução clássica do problema pelo método de série de Fourier: $u(x) = \displaystyle\sum_{j=1}^{\infty} \frac{\hat{f}_j}{1 + (j\pi)^2} \text{ sen}(j\pi x)$.
   c) explicite $\hat{f}_j$, vindo da solução por série de Fourier, na solução via método de Galerkin.

2. Seja o problema de valor de contorno:

$$\begin{cases} -\dfrac{d^2u}{dx^2} + u(x) - x = 0, \quad \forall x \in (0,1); \\ u(0) = u(1) = 0, \end{cases}$$

a) faça a formulação fraca e indique a matriz $\mathbf{A}$ e o vetor $\mathbf{b}$, com ϕ_i funções base e peso ψ_j,
b) ao substituir a condição de contorno essencial em $x = 0$ onde vai ser alterada a formulação? Substitua $u(0) = 0$ por $\dfrac{du}{dx}(0) = 0$,
c) e se ambas as condições de contorno forem naturais. Substitua $u(0) = u(1) = 0$ por $\dfrac{du}{dx}(0) = 0$ e $\dfrac{du}{dx}(1) = 0$. Como a formulação é alterada?

3. Seja o problema de Sturm-Liouville:

$$\begin{cases} -\left(p(x)u'\right)' + q(x)u(x) = f, \quad \forall x \in (0,1); \\ u(0) = 0, \quad u'(1) = 0, \end{cases}$$

onde u' denota $\dfrac{du}{dx}$. Mostre que a formulação fraca do problema é dada por: $\displaystyle\int_0^1 (pu'v' + quv)dx = \int_0^1 fv\ dx, \forall v \in V$.

Capítulo 3

Introdução ao Método de Elementos Finitos

Podemos perceber que o cálculo dos elementos da matriz de coeficientes **A** resultante do processo de discretização por um método variacional envolve a integração do produto de funções base e peso (e/ou suas derivadas) em todo o domínio.

O método dos elementos finitos consiste na escolha apropriada de funções ϕ_i e ψ_i, de tal forma que as funções e suas derivadas são diferentes de zero em uma pequena parte do domínio. Isto traz dois grandes benefícios no cálculo:

1. A integral no domínio fica reduzida a uma integral na parte do domínio onde as funções (ou suas derivadas) são diferentes de zero simultaneamente, simplificando o processo de integração;

2. A grande maioria dos coeficientes da matriz **A** é igual a zero, levando a uma matriz esparsa.

3.1 Ponto de Vista Global

A figura 3.1 apresenta um exemplo de um domínio unidimensional com o uso de funções base / peso que são diferentes de zero em pequenas

porções deste domínio. Por conveniência, as funções escolhidas são contínuas e lineares por partes.

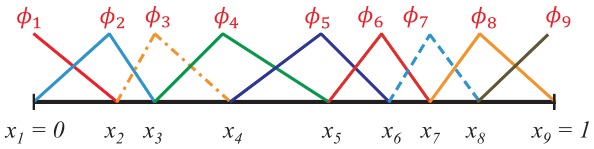

Figura 3.1: Exemplo de função base diferente de zero apenas em uma pequena porção do domínio.

Cada função é não nula em diferentes trechos do domínio. A integral ao longo de todo o domínio fica reduzida ao intervalo no qual as duas funções que aparecem no cálculo são diferentes de zero simultaneamente. Como exemplo, mostramos o cálculo de diferentes elementos da matriz **A** para o problema discutido no capítulo anterior:

$$A(2,3) = \int_0^1 \frac{d\phi_2}{dx}\frac{d\phi_3}{dx}dx = \int_{x_2}^{x_3} \frac{d\phi_2}{dx}\frac{d\phi_3}{dx}dx$$

$$A(2,6) = \int_0^1 \frac{d\phi_2}{dx}\frac{d\phi_6}{dx}dx = 0$$

A estrutura da matriz **A** do exemplo anterior usando as funções definidas na figura 3.1 é tridiagonal:

$$A = \begin{bmatrix} \times & \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \times & \times & \times & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \times & \times & \times & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \times & \times & \times & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Para evitar o cálculo desnecessário de elementos da matriz que são nulos, é importante saber antecipadamente os intervalos do domínio onde cada função é não nula. Isto pode ser feito com certa facilidade através da introdução do conceito de elementos, apresentado na próxima seção.

3.2 Ponto de Vista Elementar - Elemento de Primeira Ordem

No método de elementos finitos, o domínio é dividido em elementos. No caso de um problema unidimensional como do exemplo da figura 3.1, as fronteiras dos elementos são definidas por pontos nodais (nós). Na figura 3.2, o domínio definido entre $0 \leq x \leq 1$ foi dividido em 8 elementos, cujas fronteiras são definidas por 9 nós.

Cada função é diferente de zero ao longo de poucos elementos. Uma classe simples de funções base/peso, usada aqui como exemplo, são os polinômios de **Lagrange**, onde cada polinômio é associado a um nó do domínio de tal forma que

$$\phi_i(x_j) = \delta_{ij} \Rightarrow \phi_i(x_j) = \begin{cases} 1 & \text{se} \quad i = j, \\ 0 & \text{se} \quad i \neq j. \end{cases}$$

Pela própria definição, cada polinômio é não nulo apenas ao longo dos elementos que contém aquele nó. Desta forma, o polinômio $\phi_3(x)$ é diferente de zero apenas ao longo dos elementos 2 ($x_2 < x < x_3$) e 3 ($x_3 < x < x_4$), que são os únicos elementos do domínio que contém o nó 3.

Desta forma o número de funções base (dimensão do espaço) está associado ao número de nós da malha. Quanto mais refinada a malha, maior será a dimensão do espaço de funções e desta forma, mais precisa será a solução aproximada.

Quando a expansão linear é feita com polinômios de Lagrange, os coeficientes c_i passam a ter um significado especial:

$$u_h(x_i) = \sum_{j=1}^{N} c_j \underbrace{\phi_j(x_i)}_{\delta_{ij}} = \sum_{j=1}^{N} c_j \delta_{ij} = c_i$$

$$\Rightarrow c_i = u_h(x_i).$$

O coeficiente c_i é igual ao valor da função u_h no ponto x_i.

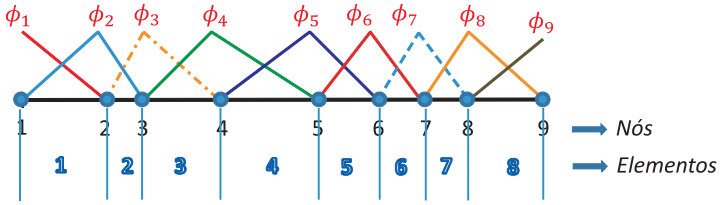

Figura 3.2: O domínio foi dividido em 8 elementos usando 9 nós

No exemplo da figura 3.2, existem duas funções diferentes de zero em cada elemento e uma mesma função é diferente de zero em dois elementos (desconsiderando as funções localizadas na fronteira do domínio), conforme mostrado na tabela 3.1.

Tabela 3.1: Elementos em que ϕ_i não se anula

elemento	nó inicial	nó final	$\phi_i's \neq 0$
1	1	2	ϕ_1 e ϕ_2
2	2	3	ϕ_2 e ϕ_3
3	3	4	ϕ_3 e ϕ_4
4	4	5	ϕ_4 e ϕ_5
5	5	6	ϕ_5 e ϕ_6
6	6	7	ϕ_6 e ϕ_7
7	7	8	ϕ_7 e ϕ_8
8	8	9	ϕ_8 e ϕ_9

O cálculo dos componentes da matriz $\mathbf{A}$, que consiste no cálculo de integrais ao longo do domínio, pode ser feito elemento por elemento, considerando somente os termos da matriz que contém funções não

nulas em cada elemento. Esta procedimento de cálculo leva naturalmente a definição de matriz elementar $\mathbf{A}^{(e)}$.

Como no exemplo em questão só existem duas funções $\phi_i's \neq 0$ em cada elemento, a matriz elementar $\mathbf{A}^{(e)}$ será uma matriz 2×2. Por exemplo, a matriz elementar 3 $\mathbf{A}^{(3)}$ é dada por:

$$\mathbf{A}^{(3)} = \begin{bmatrix} \int_{x_3}^{x_4} \frac{d\phi_3}{dx}\frac{d\phi_3}{dx}dx & \int_{x_3}^{x_4} \frac{d\phi_3}{dx}\frac{d\phi_4}{dx}dx \\ \int_{x_3}^{x_4} \frac{d\phi_4}{dx}\frac{d\phi_3}{dx}dx & \int_{x_3}^{x_4} \frac{d\phi_4}{dx}\frac{d\phi_4}{dx}dx \end{bmatrix}$$

Nesta matriz, só aparecem as funções ϕ_3 e ϕ_4, as únicas não nulas ao longo do elemento 3.

A matriz global $\mathbf{A}$ pode ser escrita como a união de matrizes elementares:

$$\boxed{\mathbf{A} = \bigcup_{e=1}^{8} \mathbf{A}^{(e)}}$$

Vale observar que a forma de cada função nos elementos é sempre a mesma, independentemente do elemento em questão. Isto sugere que, em um sistema de coordenadas elementar, todas as funções globais podem ser descritas por duas únicas funções elementares.

Podemos definir um sistema de coordenadas elementar de forma que o elemento se extenda de $\xi = -1$ a $\xi = +1$ e possua dois nós (1 e 2) como mostrado na figura 3.3.

As funções base no sistema de coordenadas elementar são:

$$\boxed{\phi_1^{(e)}(\xi) = \frac{1-\xi}{2} \quad ; \qquad \phi_2^{(e)}(\xi) = \frac{1+\xi}{2}}$$

e suas derivadas:

$$\boxed{\frac{d\phi_1^{(e)}}{d\xi} = -\frac{1}{2} \quad ; \qquad \frac{d\phi_2^{(e)}}{d\xi} = \frac{1}{2}}$$

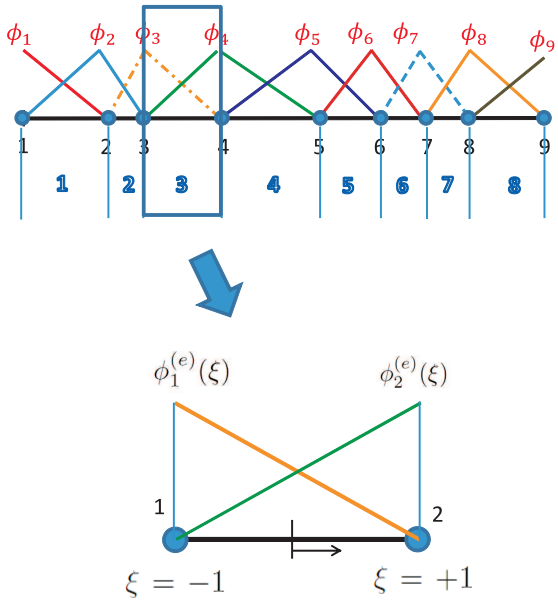

Figura 3.3: Funções base lineares no sistema de coordenadas local.

Todos os elementos do domínio podem ser mapeados para esse elemento padrão.

Por exemplo, o mapeamento do elemento 3, que vai de x_3 a x_4, para o elemento padrão vai de $\xi = -1$ a $\xi = 1$ e pode ser escrito como:

$$x = x_3 \longleftrightarrow \xi = -1,$$

$$\rightarrow \begin{cases} \xi(x) = \dfrac{2x - (x_3 + x_4)}{x_4 - x_3}, \\ x(\xi) = \dfrac{(x_4 - x_3)\,\xi + x_3 + x_4}{2}, \end{cases}$$

$$x = x_4 \longleftrightarrow \xi = 1.$$

As integrais são reescritas no sistema de coordenadas elementar como:

$$A_{12}^{(3)} = \int_{x_3}^{x_4} \frac{d\phi_3}{dx}\frac{d\phi_4}{dx}dx = \int_{-1}^{1} \frac{d\phi_1^{(e)}}{d\xi}\frac{d\xi}{dx}\,\frac{d\phi_2^{(e)}}{d\xi}\frac{d\xi}{dx}\frac{dx}{d\xi}d\xi =$$
$$= \int_{-1}^{1} \frac{d\phi_1^{(e)}}{d\xi}\frac{d\phi_2^{(e)}}{d\xi} \underbrace{\frac{d\xi}{dx}}_{=\frac{2}{x_4-x_3}=\frac{2}{h^{(e)}}} d\xi,$$

onde $h^{(e)}$ é o tamanho do elemento $\Rightarrow \frac{2}{h^{(e)}} \underbrace{\int_{-1}^{1} \frac{d\phi_1^{(e)}}{d\xi}\frac{d\phi_2^{(e)}}{d\xi}d\xi}_{=\text{para todos os elementos}}.$

A matriz do terceiro elemento então fica:

$$\mathbf{A}^{(3)} = \frac{2}{h^{(3)}} \begin{bmatrix} \int_{-1}^{1} \frac{d\phi_1}{d\xi}\frac{d\phi_1}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_1}{d\xi}\frac{d\phi_2}{d\xi}d\xi \\ \int_{-1}^{1} \frac{d\phi_2}{d\xi}\frac{d\phi_1}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_2}{d\xi}\frac{d\phi_2}{d\xi}d\xi \end{bmatrix},$$

onde os $A_{ij}^{(3)}$ são:

$$A_{11}^{(3)} = \frac{2}{h^{(3)}} \int_{-1}^{1} \left(\frac{-1}{2}\right)\left(\frac{-1}{2}\right) d\xi = \frac{2}{h^{(3)}}\frac{1}{4}2 = \frac{1}{h^{(3)}}$$
$$A_{12}^{(3)} = \frac{2}{h^{(3)}} \int_{-1}^{1} \left(\frac{-1}{2}\right)\left(\frac{1}{2}\right) d\xi = \frac{-1}{h^{(3)}} = A_{21}^{(3)}$$
$$A_{22}^{(3)} = \frac{2}{h^{(3)}} \int_{-1}^{1} \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) d\xi = \frac{1}{h^{(3)}}$$

$$\mathbf{A}^{(3)} = \frac{1}{h^{(3)}} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

A matriz global é obtida a partir de um processo de montagem de todas as matrizes elementares. Para este processo de montagem, é necessário saber a correspondência entre a numeração local dos nós no elemento

com a numeração global dos nós no domínio. Esta correspondência é armazenada na forma de uma tabela (matriz de incidência dos nós), definida como:
`DomNodeID(ilnode,iele)` = número global do nó local `ilnode` do elemento `iele`. Para o exemplo que vem sendo discutido, a matriz de incidência é dada em 3.2

Tabela 3.2: DomNodeID

`ilnode`\`iele`	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9

A matriz global é montada segundo a correspondência entre a numeração local e a global. Por exemplo, a matriz local do elemento `3` (2×2) é "montada" na seguinte posição da matriz global:

$$\mathbf{A} = \begin{bmatrix} & & & \\ & \times & \times & \\ & \times & \times & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

$$\texttt{DomNodeID}(1,3) = 3, \quad \Rightarrow \begin{cases} A_{11}^{(3)} & \rightarrow A_{33} \\ A_{12}^{(3)} & \rightarrow A_{34} \\ A_{21}^{(3)} & \rightarrow A_{43} \\ A_{22}^{(3)} & \rightarrow A_{44} \end{cases}$$

$$\texttt{DomNodeID}(2,3) = 4.$$

O processo de montagem também serve para o vetor $\mathbf{f}$

$$f_1^{(3)} \rightarrow f_3 \quad \text{e} \quad f_2^{(3)} \rightarrow f_4$$

Um algoritmo geral para a solução por elementos finitos pode ser escrito como:

```
for iele = 1:NELE      % NELE → # de elementos
  [Aelem,felem] = getelem(iele, ...)
  for ilnode = 1:NLOCALNODES     % NLOCALNODES → # de nós locais
    ignode = DomNodeID(ilnode,iele) % ignode → # global do nó ilnode
    for jlnode = 1, NLOCALNODES
      jgnode = DomNodeID(jlnode,iele)
        A(ignode,jgnode) = A(ignode,jgnode) + Aelem(ilnode,jlnode)
    end
  end
end
```

É importante observar que o componente A_{55} da matriz global tem contribuição de 2 elementos (elementos 4, 5).

$$A_{55} = A_{22}^{(4)} + A_{11}^{(5)}$$

Isto ocorre, pois:

$$A_{55} = \int_0^1 \frac{d\phi_5}{dx}\frac{d\phi_5}{dx}dx = \int_{x_4}^{x_5} \frac{d\phi_5}{dx}\frac{d\phi_5}{dx}dx + \int_{x_5}^{x_6} \frac{d\phi_5}{dx}\frac{d\phi_5}{dx}dx = A_{22}^{(4)} + A_{11}^{(5)}$$

3.3 Elemento de Segunda Ordem

No exemplo anterior, cada elemento possuía 2 nós e a cada nó estava associada uma função base, ou seja um polinômio de Lagrange. Desta forma, as funções elementares eram lineares.

O espaço de funções pode ser enriquecido através da utilização de polinômios de maior grau. Em tese, isto leva a melhores aproximações da solução da equação diferencial. Uma aproximação comum é utilizar funções base quadráticas dentro de cada elemento.

No caso dos polinômios de Lagrange de segundo grau, cada elemento deve possuir 3 nós (3 pontos definem uma parábola), conforme mostrado na figura 3.4. As funções base apresentadas na figura continuam a serem definidas de forma que $\phi_i(x_j) = \delta_{ij}$.

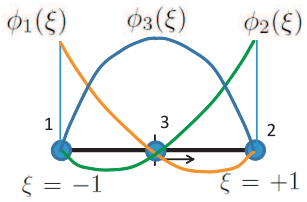

Figura 3.4: Elemento Quadrático.

Nas coordenadas elementares, as funções base quadráticas são os polinômios de Lagrange definidos como:

$$\boxed{\phi_1(\xi) = \frac{\xi(\xi-1)}{2} \quad ; \quad \phi_2(\xi) = \frac{\xi(\xi+1)}{2} \quad ; \quad \phi_3(\xi) = -(\xi+1)(\xi-1).}$$

As suas derivadas são:

$$\boxed{\frac{d\phi_1}{d\xi} = \xi - \frac{1}{2} \quad ; \qquad \frac{d\phi_2}{d\xi} = \xi + \frac{1}{2} \quad ; \qquad \frac{d\phi_3}{d\xi} = -2\xi.}$$

As matrizes elementares passam a ser 3×3, já que cada elemento possui 3 funções bases e 3 coeficientes:

$$\mathbf{A}^{(e)} = \frac{2}{h^{(e)}} \begin{bmatrix} \int_{-1}^{1} \frac{d\phi_1}{d\xi}\frac{d\phi_1}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_1}{d\xi}\frac{d\phi_2}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_1}{d\xi}\frac{d\phi_3}{d\xi}d\xi \\ \int_{-1}^{1} \frac{d\phi_2}{d\xi}\frac{d\phi_1}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_2}{d\xi}\frac{d\phi_2}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_2}{d\xi}\frac{d\phi_3}{d\xi}d\xi \\ \int_{-1}^{1} \frac{d\phi_3}{d\xi}\frac{d\phi_1}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_3}{d\xi}\frac{d\phi_2}{d\xi}d\xi & \int_{-1}^{1} \frac{d\phi_3}{d\xi}\frac{d\phi_3}{d\xi}d\xi \end{bmatrix}$$

O processo de montagem da matriz global a partir das matrizes elementares é exatamente igual ao caso de elementos lineares. A montagem é feita a partir da relação da numeração local e global dos nós.

Para ilustrar este processo, usamos o exemplo apresentado na figura 3.5, onde o domínio foi dividido em 6 elementos (quadráticos), com 13 nós.

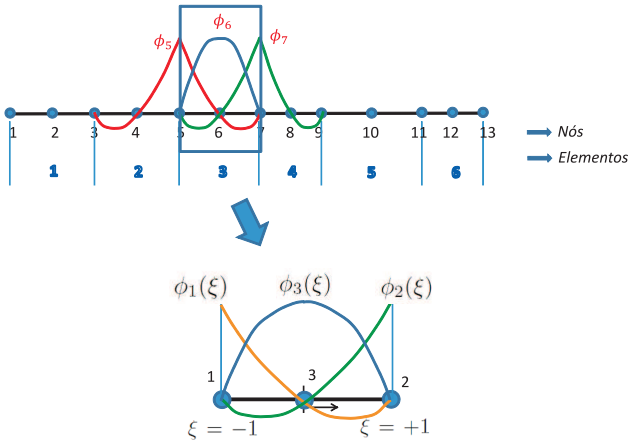

Figura 3.5: Domínio dividido em 6 elementos quadráticos com um total de 13 nós.

Como no caso de elementos lineares, a correspondência entre a numeração local e global dos nós é dada pela tabela 3.3, apresentada para o exemplo da figura 3.5.

Tabela 3.3: DomNodeID

`ilnode\iele`	1	2	3	4	5	6
1	1	3	5	7	9	11
2	3	5	7	9	11	13
3	2	4	6	8	10	12

A matriz elementar do terceiro elemento $\mathbf{A}^{(3)}$ é montada na matriz

global na seguinte posição:

$$\mathbf{A} = \begin{bmatrix} & & & & \\ & \times & \times & \times & \\ & \times & \times & \times & \\ & \times & \times & \times & \\ & & & & \end{bmatrix}$$

$$A_{11}^{(3)} \to A_{55} \quad A_{12}^{(3)} \to A_{57} \quad A_{13}^{(3)} \to A_{56} \quad \cdots$$

3.4 Exercícios

1. Considere o problema de valor de contorno (exercício 2 do capítulo anterior):

$$\begin{cases} -\dfrac{d^2u}{dx^2} + u(x) - x = 0, \quad \forall x \in (0,1); \\ u(0) = u(1) = 0, \end{cases}$$

a) deduza a fórmula da matriz e vetor elementar utilizando elementos lineares.
b) quais matrizes elementares são alteradas durante a aplicação das condições de contorno? Como ficam estas matrizes após a aplicação das condições de contorno?
c) considere que o domínio foi dividido uniformemente em 5 elementos (6 nós). Escreva o sistema linear resultante do processo de discretização.
d) resolva este sistema (usando algum procedimento numérico) para obter a solução do problema.

2. No mesmo problema de valor de contorno anterior, responda:
   a) deduza a fórmula da matriz e vetor elementar utilizando elementos quadráticos.
   b) a aplicação das condições de contorno altera as mesmas matrizes elementares ao usar elemento quadrático no lugar de linear?
   c) escreva o sistema linear resultante do processo de discretização, considerando os mesmos 5 elementos do exercício anterior, porém utilizando elementos quadráticos, ou seja um maior número de nós por elemento.
   d) compare as soluções do sistema linear com elementos quadráticos e lineares em relação a precisão, dimensão do sistema a ser resolvido e esforço computacional.

3. Do ponto de vista elementar, o que muda se a malha não for uniforme? Descreva e compare as mudanças no caso de utilizar elementos lineares e quadráticos para uma malha concentrada da seguinte forma: $x_i = 1 - \sqrt{\left(\frac{N-i}{N-1}\right)}$, onde N é o número total de nós. Em que situações a malha concentrada é interessante?

Capítulo 4

Problema Unidimensional Linear

Com o objetivo de apresentar detalhes da implementação numérica, vamos mostrar todos os passos para a solução de um problema de convecção - difusão usando o método dos elementos finitos.

A equação de convecção - difusão descreve o transporte de uma grandeza escalar, que poderia ser temperatura por exemplo. O modelo é descrito por uma equação diferencial que em sua forma adimensional traz o número de Péclet que caracteriza a razão entre os termos convectivos e difusivos e é definido como $Pe = \dfrac{LU}{\alpha}$ sendo L e U comprimento e velocidade característicos e α difusividade térmica.

$$\begin{cases} \dfrac{d^2u}{dx^2} - Pe\,\dfrac{du}{dx} = 0, & 0 < x < 1; \\ \dfrac{du}{dx}(0) = u(0) - 1; & u(1) = 0. \end{cases}$$

4.1 Formulação Fraca (Método de Galerkin)

Seguindo a teoria apresentada, sabemos que a solução aproximada é uma projeção ortogonal da solução exata no subespaço de dimensão finita. Monta-se então o resíduo e o projeta no espaço das funções peso,

que no caso do método de Galerkin é o mesmo espaço das funções base ($U_h = V_h$) já descritos no capítulo anterior:

$$R_i = \int_0^1 \left(\frac{d^2 u_h}{dx^2} - Pe \frac{du_h}{dx} \right) \phi_i \, dx,$$

integrando por partes obtém-se a formulação fraca apresentada no capítulo 2, com isso a segunda derivada da função aproximada é evitada e aparecem os termos de fronteira:

$$R_i = -\int_0^1 \frac{du_h}{dx}\frac{d\phi_i}{dx} dx + \frac{du_h}{dx}(1)\,\phi_i(1) - \frac{du_h}{dx}(0)\phi_i(0) - Pe \int_0^1 \frac{du_h}{dx}\,\phi_i \, dx.$$

O objetivo é que a projeção seja encontrada de forma que o erro seja o menor possível. Fazendo $R_i = 0$, manipulando as equações algebricamente e ainda usando que a função peso é escolhida de forma a garantir as condições de contorno, com isso na fronteira cuja condição é imposta (Dirichlet) ϕ_i vale zero, temos:

$$\Rightarrow \int_0^1 \left(\frac{du_h}{dx}\frac{d\phi_i}{dx} + Pe \frac{du_h}{dx}\,\phi_i \right) dx = \frac{du_h}{dx}(1)\,\phi_i(1) - \underbrace{\frac{du_h}{dx}(0)}_{u_h(0)-1}\,\phi_i(0)$$

$$\Rightarrow \int_0^1 \left(\frac{du_h}{dx}\frac{d\phi_i}{dx} + Pe \frac{du_h}{dx}\,\phi_i \right) dx = \frac{du_h}{dx}(1)\,\underbrace{\phi_i(1)}_{=0} - (u_h(0) - 1)\phi_i(0)$$

$$R_i = \int_0^1 \left(\frac{du_h}{dx}\frac{d\phi_i}{dx} + Pe \frac{du_h}{dx}\,\phi_i \right) dx + (u_h(0) - 1)\phi_i(0) = 0. \quad (4.1.1)$$

4.2 Método dos Elementos Finitos

Seguindo o método de Galerkin, o campo aproximado $u_h(x)$ vai ser escrito como combinação linear das funções ϕ_i, como na equação (4.2.2).

$$u_h(x) = \sum_{i=1}^{N} U_i \phi_i. \quad (4.2.2)$$

Dividindo o domínio $\{x \in \mathbb{R} | 0 \leq x \leq 1\}$ em oito elementos e nove nós, como mostra a figura 4.1. Lembrando que o método de elementos finitos justamente sugere escolhas apropriadas de ϕ_i de tal forma que as funções e suas derivadas só são diferentes de zero em uma pequena parte do domínio.

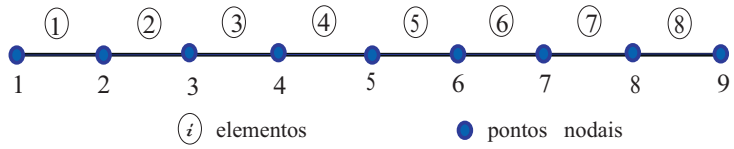

Figura 4.1: Discretização do domínio em oito elementos.

Usando a aproximação (4.2.2) no resíduo (4.1.1) para $i = 1 \cdots N$, temos:

$$R_i = \sum_{j=1}^{N} \left\{ \int_0^1 \left(\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + Pe\frac{d\phi_j}{dx}\phi_i \right) dx \right\} U_j + \left[\sum_{j=1}^{N} U_j\phi_j(0) - 1 \right] \phi_i(0);$$

observe que $x = 0$ é o nó 1, então $\phi_1(0) = 1$ e $\phi_j(0) = 0, \forall j \neq 1$.
Assim:

$$R_i = \sum_{j=1}^{N} \left\{ \int_0^1 \left(\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + Pe\frac{d\phi_j}{dx}\,\phi_i \right) dx \right\} U_j + [U_1 - 1]\phi_i(0).$$

Identificando R_i para cada i : c.c. essencial $\Rightarrow u(1) = 0 \Rightarrow \sum_{j=1}^{N} U_j\phi_j(1)$,
lembrando que no extremo do domínio,

$$x = 1 \text{ é o nó } N \Rightarrow U_N\phi_N(1) = 0.$$

$$
\begin{aligned}
&i = 1 \Rightarrow \\
&R_1 \Rightarrow \sum_{j=1}^{N} \left\{ \int_0^1 (\frac{d\phi_1}{dx}\frac{d\phi_j}{dx} + Pe\frac{d\phi_j}{dx}\,\phi_1)dx \right\} U_j = -[U_1 - 1], \\
&i = 2, ..., N-1 \Rightarrow \\
&R_i \Rightarrow \sum_{j=1}^{N} \left\{ \int_0^1 (\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + Pe\frac{d\phi_j}{dx}\,\phi_i)dx \right\} U_j = 0, \\
&i = N \Rightarrow \\
&R_N \Rightarrow U_N = 0.
\end{aligned}
$$

Matricialmente o problema fica como em 4.2.3 no qual a esparcidade da matriz está diretamente ligada a escolha das funções base/peso. Nesse exemplo vamos usar elementos lineares e com isso a matriz global é bastante esparsa uma vez que a local vai ter dimensão 2×2.

$$
\begin{bmatrix}
\times & 0 & \dots & 0 & \times & \\
 & & 0 & & & \\
\times & & & & & 0 \\
\vdots & & \ddots & & & \vdots \\
 & & & 0 & & \\
\times & & & & & \times \\
 & & & & & \\
\times & & \times & 0 & & \\
0 & \dots & & & 0 & 1
\end{bmatrix}
\begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \\ U_7 \\ U_8 \\ U_9 \end{bmatrix}
=
\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}
\tag{4.2.3}
$$

4.3 Ponto de Vista Local

O desafio agora é desenvolver um esquema de montagem dos vetores e matrizes elementares a partir da formulação:

$$
\sum_{j=1}^{N} \left\{ \int_0^1 \left(\frac{d\phi_i}{dx}\frac{d\phi_j}{dx} + Pe\frac{d\phi_j}{dx}\,\phi_i \right) dx \right\} U_j = \underbrace{-[U_1 - 1]\phi_i(0)}_{\text{condição de contorno.}} .
$$

$\rightarrow$ O ideal é montar as matrizes e vetores elementares sem se preocupar com as condições de contorno e depois modificá-las em função destas.

Escolhendo elementos de primeira ordem como apresentados na seção 3.2 a matriz e vetor elementares ficam:

$$\mathbf{A^{(e)}} = \begin{bmatrix} A_{11}^{(e)} & A_{12}^{(e)} \\ A_{21}^{(e)} & A_{22}^{(e)} \end{bmatrix}; \qquad \mathbf{f^{(e)}} = \begin{bmatrix} f_1^{(e)} \\ f_2^{(e)} \end{bmatrix}; \qquad \frac{d\xi}{dx} = \frac{2}{h^{(e)}}.$$

$$A_{ij}^{(e)} = \frac{2}{h^{(e)}} \int_{-1}^{1} \frac{d\phi_j}{d\xi} \frac{d\phi_i}{d\xi} d\xi + Pe \int_{-1}^{1} \frac{d\phi_j}{d\xi} \phi_i d\xi; \qquad f_i^{(e)} = 0. \quad (4.3.4)$$

onde $\phi_1^{(e)}(\xi) = \dfrac{1-\xi}{2} \rightarrow \dfrac{d\phi_1}{d\xi} = \dfrac{-1}{2}$ e $\phi_2^{(e)}(\xi) = \dfrac{1+\xi}{2} \rightarrow \dfrac{d\phi_2}{d\xi} = \dfrac{1}{2}$.
Substituindo na expressão de (4.3.4) e fazendo a conta, temos

$$\begin{aligned} A_{11}^{(e)} &= \frac{2}{h^{(e)}} \int_{-1}^{1} (-\frac{1}{2}) (-\frac{1}{2})\, d\xi \; + Pe \int_{-1}^{1} (-\frac{1}{2}) (\frac{1-\xi}{2})\, d\xi \\ &= \frac{2}{h^{(e)}} \int_{-1}^{1} \frac{1}{4}\, d\xi \; - \int_{-1}^{1} \frac{Pe}{4} (1-\xi)\, d\xi \\ &= \frac{1}{h^{(e)}} - \frac{Pe}{2}, \end{aligned}$$

e assim por diante...

Após obter as matrizes elementares, temos que nos preocupar com as condições de contorno. Os termos correspondentes às condições de contorno são nulos a menos no primeiro e no último, ou seja $\phi_i(0) \neq 0$, somente para $i = 1$ e $\phi_i(1) \neq 0$, apenas para $i = N$.

No primeiro elemento, temos que acrescentar:

$$\int_0^1 (\quad) = -[U_1 - 1] \quad \Rightarrow \quad \sum_{j=1}^{N} (\quad)\, U_j + U_1 = +1.$$

$$A_{1,1}^{(1)} = A_{1,1}^{(1)} + 1 \quad \text{e} \quad f_1^{(1)} = f_1^{(1)} + 1.$$

Enquanto que no último elemento temos uma condição de contorno essencial. Logo, impõe-se $U_9 = 0$.

$$A_{2,1}^{(8)} \leftarrow 0 \quad A_{2,2}^{(8)} \leftarrow 1 \quad \text{e} \quad f_2^{(8)} \leftarrow 0,$$

se a condição de contorno fosse $u(1) = g$ então, $f_2^{(8)} \leftarrow g$.

Integração Numérica

Para calcularmos os elementos de cada matriz, temos que integrar funções ao longo do elemento. De um modo geral, calculamos $\int_{-1}^{1} F(\xi)d\xi$.

Para um problema simples, podemos calcular a integral analiticamente. Porém, geralmente devemos utilizar algum método de integração numérica. Em códigos de elementos finitos, o método de integração mais utilizado e o método de Quadratura Gaussiana.

$$\int_{-1}^{1} F(\xi)d\xi = \sum_{i=1}^{NGP} F(\xi)W_i,$$

onde NGP é o número de pontos utilizados para a integração. Quanto maior o número de pontos, melhor a aproximação.

A tabela 4.1 fornece o valor de ξ_i e W_i para diferentes NGP.

Tabela 4.1: Pontos e pesos de Gauss

NGP	ξ_i	W_i
1	0.0	2.0
2	-0.57735	1.0
	+0.57735	1.0
3	-0.77459	0.555
	0.0	0.888
	+0.77459	0.555

4.4 Exemplo de Implementação do Código

O problema apresentado ao longo deste capítulo será usado como exemplo para apresentação do código. A solução obtida é comparada com a solução analítica da equação diferencial:

$$u(x) = \frac{e^{Pex} - e^{Pe}}{1 - Pe - e^{Pe}}$$

O script principal `Main1D` é dividido em diferentes funções, e é apresentado abaixo. As funções `GlobalPointer` e `Mesh` correspondem ao pré-processamento do problema. A primeira calcula a matriz `DomNodeID` que relaciona a numeração local e global e a segunda fornece dados da malha (coordenadas nodais e tamanho dos elementos).
`Main1D`

```
%=========================================================================
% Codigo Baseado no MEF - problema 1D
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
clear; clc;
%=========================================================================
% PREPROCESSAMENTO
% Entrada de Dados
Pe = 5;
L = 1;
Nele = 20;
% Calcula nnodes e DomNodeID
[nnodes, DomNodeID] = GlobalPointer(Nele);
% Geracao de Malha
[x, dx] = Mesh(L, nnodes, Nele, DomNodeID);
%=========================================================================
% RESOLUCAO DO PROBLEMA
% Vetor com a solucao aproximada
[U] = Solution(Pe, nnodes, Nele, DomNodeID, dx)
%=========================================================================
% POSPRECESSAMENTO
% Vetor com a solucao exata
[Uex] = Exata(Pe,x, nnodes)
% Calculo do erro da solucao
Erro = (U - Uex)./(U + 1e-10);
figure; plot(x,U,'bo',x,Uex,'r')
xlabel('x'); ylabel('U(x)'); title('Solucao')
figure; plot(x,Erro)
xlabel('x'); ylabel('Erro relativo(x)'); title('Erro relativo')
%=========================================================================
```

GlobalPointer

```
%=========================================================================
% Funcao Mapeamento Local / Global
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
%=========================================================================
function [nnodes, DomNodeID] = GlobalPointer(Nele)
nnodes = Nele + 1;
DomNodeID = zeros(Nele, 2);
for i = 1:Nele
    DomNodeID(i,1) = i;
    DomNodeID(i,2) = i+1;
end
end
%=========================================================================
```

Mesh

```
%=========================================================================
% Funcao para calculo da malha problema 1D
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
%=========================================================================
function [x, dx] = Mesh(L, nnodes, Nele, DomNodeID)
x = zeros(nnodes,1);
dx = zeros(Nele,1);
for i = 1:nnodes
    x(i) = L*((i-1)/(nnodes-1));
end
% calculo do comprimento dos elementos
for iele = 1:Nele
    dx(iele) = x(DomNodeID(iele,2)) - x(DomNodeID(iele,1));
end
end
%=========================================================================
```

A solução pelo método de elementos finitos é implementada na função `Solution`, apresentada a seguir. A matriz e vetor elementar são calculados na função `GetElemAb`.

Solution

```
%=========================================================================
```

```
% Funcao para solucao pelo metodo de EF
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
%=========================================================================
function [U] = Solution(Pe, nnodes, Nele, DomNodeID, dx)
A = zeros(nnodes, nnodes);
b = zeros(nnodes, 1);
for iele = 1:Nele
    %Calcula a Matriz Elementar
    [Aelem, belem] = GetElemAb(iele, Nele, Pe, dx(iele));
    nlocalnodes = 2;
    for ilnode = 1:nlocalnodes
        ignode = DomNodeID(iele, ilnode);
        for jlnode = 1:nlocalnodes
            jgnode = DomNodeID(iele, jlnode);
            A(ignode, jgnode) = A(ignode, jgnode) + Aelem(ilnode, jlnode);
        end
        b(ignode) = b(ignode) + belem(ilnode);
    end
end
U = A;
end
%=========================================================================
```

GetElemAb

```
%=========================================================================
% Funcao para calculo da matriz e vetor elementar
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
%=========================================================================
function [Aelem, belem] = GetElemAb(iele,Nele,Pe,dx)
nlocalnodes = 2;
ngp = 2;
XIGP = [-0.57735 0.57735];
WGP = [1.0 1.0];
Aelem = zeros(nlocalnodes,nlocalnodes);
belem = zeros(nlocalnodes, 1);
for igp = 1:ngp
    XI = XIGP(igp);
    W = WGP(igp);
    [Phi, GradPhi] = BasisFunc(XI);
    for ilnode = 1:  nlocalnodes
        for jlnode = 1:nlocalnodes
            Aelem(ilnode, jlnode) = Aelem(ilnode, jlnode) + ...
            W*(GradPhi(ilnode) * GradPhi(jlnode)* 2/dx + ...
            Pe * Phi(ilnode)*GradPhi(jlnode));
        end
        belem(ilnode) = 0;
```

```
    end
end
if (iele == 1)
    Aelem(1,1) = Aelem(1,1) + 1.0;
    belem(1) = 1.0;
elseif (iele == Nele)
    Aelem(2,1) = 0.0;
    Aelem(2,2) = 1.0;
end
end
%=========================================================================
```

As funções bases e suas derivadas em cada ponto de Gauss são calculadas pela função `BasisFunc`, mostrada a seguir.
`BasisFunc`

```
%=========================================================================
% Funcao para calculo das funcoes base
% Data:  Marco de 2012
% Autores:  Marcio Carvalho, Juliana Valerio e Aline Amaral Abdu
%=========================================================================
function [Phi, GradPhi] = BasisFunc(X)
Phi(1) = (1.0 - X)/2;
Phi(2) = (1.0 + X)/2;
GradPhi(1) = -0.5;
GradPhi(2) = 0.5;
end
%=========================================================================
```

A solução do problema e apresentada na figura 4.2.

4.5 Exercícios

1. Utilize o programa apresentado e obtenha a solução do problema para os seguintes casos:

   (a) malha de 10 elementos lineares uniformemente espaçados com $Pe = 5,\ Pe = 10,\ Pe = 30$. Explique o que ocorre.

   (b) implemente no código apresentado a possibilidade de usar um elemento quadrático. Resolva o problema com uma malha

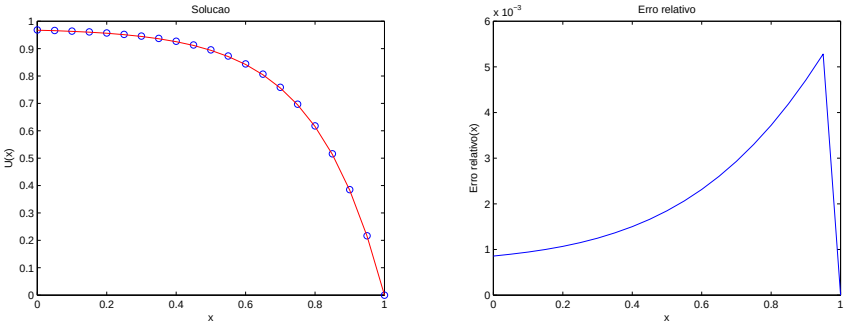

Figura 4.2: Comparação da solução pelo MEF com a solução analítica.

de 5 elementos quadráticos uniformemente espaçados com $Pe = 5,\ Pe = 10,\ Pe = 30$. Explique o que ocorre.

(c) implemente no código apresentado a possibilidade de usar uma malha com elementos não uniformemente espaçados. Utilize a seguinte função para gerar a malha concentrada:

$$x_i = 1 - \left[\frac{(NNODES - i)}{(NNODES - 1)}\right]^a.$$

$a \rightarrow$ parâmetro que regula a concentração da malha. Resolva o problema utilizando uma malha não uniforme de 10 elementos para $Pe = 30$. Determine o efeito do parâmetro de concentração de malha na solução do problema. Explique o que ocorre.

2. Verifique que a função

$$u(x) = (1 - x)\left[\tan^{-1}\alpha(x - x_0) + \tan^{-1}(\alpha x_0)\right]$$

é solução da equação diferencial, onde u' denota $\dfrac{du}{dx}$:

$$\begin{cases} -(k(x)u'(x))' = f(x), \quad 0 < x < 1; \\ u(0) = 0; \quad u(1) = 0, \end{cases}$$

onde

$$k(x) = \frac{1}{\alpha} + \alpha(x - x_0)^2 \quad \text{e}$$

$$f(x) = 2\Big[1 + \alpha(x - x_0)\left[\tan^{-1}\alpha(x - x_0) + \tan^{-1}(\alpha x_0)\right]\Big].$$

Observe que as escolhas de α e x_0 fazem com que a função u, definida acima, seja bastante suave ou quase descontínua próximo de x_0.

Modifique o programa apresentado nesse capítulo de forma a resolver esse novo problema. Faça um estudo da suavidade de u em relação a acurácia do método.

Capítulo 5

Problema Bidimensional Linear

Assim como no capítulo anterior, vamos aplicar os conceitos discutidos e apresentar a formulação do problema discreto utilizando um exemplo, só que neste caso um problema de condução de calor bidimensional. O campo de temperatura em Ω, cuja fronteira é $\Gamma_1 \bigcup \Gamma_2$ representadas na figura 5.1, com condições de contorno diferentes para cada $\Gamma_i (i = 1, 2)$, é descrito pela equação (5.0.1).

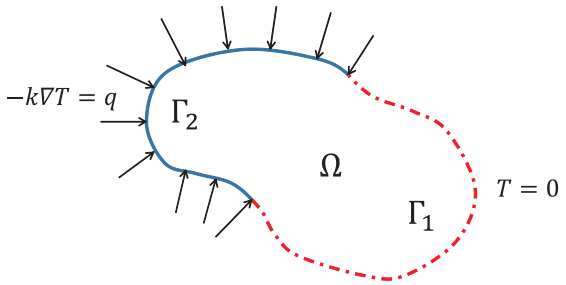

Figura 5.1: Problema bidimensional.

$$\begin{cases} \nabla \cdot (k\nabla T) = 0 \text{ em } \Omega, \\ T = 0 \text{ em } \Gamma_1, \\ \mathbf{n} \cdot \nabla T = q \text{ em } \Gamma_2. \end{cases} \tag{5.0.1}$$

onde k é a condutividade térmica, que sendo constante $\Rightarrow \nabla^2 T = 0$.

A condição de contorno $T\big|_{\Gamma_1} = 0$, de Dirichlet, prescreve a temperatura na fronteira. Enquanto que na condição de Newman, $\mathbf{n} \cdot \nabla T = q \Rightarrow \dfrac{\partial T}{\partial \mathbf{n}}\big|_{\Gamma_2} = q$, o fluxo é prescrito.

5.1 Formulação Fraca

Ache $T \in U$ tal que:

$$\int_\Omega w \nabla^2 T d\Omega = 0 \ , \ \forall w \in V;$$

onde:

$$U = \{T \mid \int_\Omega |\nabla T|^2 d\Omega < \infty \ ; \ T(\Gamma_1) = 0\},$$

$$V = \{w \mid \int_\Omega |\nabla w|^2 d\Omega < \infty \ ; \ w(\Gamma_1) = 0\}.$$

Para desenvolver a expressão do resíduo ponderado, usa-se a relação vinda da derivada do produto de duas funções, que para uma variável pode ser visto na equação (5.1.2):

$$\frac{d}{dx}\left[w \ \frac{dT}{dx}\right] = \frac{dw}{dx}\frac{dT}{dx} + w\frac{d^2T}{dx^2},$$

$$w\frac{d^2T}{dx^2} = -\frac{dw}{dx}\frac{dT}{dx} + \frac{d}{dx}\left[w \ \frac{dT}{dx}\right], \tag{5.1.2}$$

e para várias variáveis tem-se a equação (5.1.3):

$$\int_\Omega w \left[\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right] dxdy =$$

$$-\int_\Omega \left[\frac{\partial w}{\partial x}\frac{\partial T}{\partial x}+\frac{\partial w}{\partial y}\frac{\partial T}{\partial y}\right]dxdy+\int_\Omega \frac{\partial}{\partial x}\left[w\frac{\partial T}{\partial x}\right]+\frac{\partial}{\partial y}\left[w\frac{\partial T}{\partial y}\right]dxdy,$$

$$\text{vetorialmente, } \int_\Omega w\nabla^2 T d\Omega = -\int_\Omega \nabla w\cdot\nabla T d\Omega+\int_\Omega \nabla\cdot(w\nabla T)d\Omega. \tag{5.1.3}$$

Usando o teorema da divergência, que diz:

$$\int_\Omega \nabla\cdot\mathbf{f}d\Omega=\int_\Gamma \mathbf{n}\cdot\mathbf{f}d\Gamma,$$

a formulação fraca fica:

$$\int_\Omega w\nabla^2 T d\Omega = -\int_\Omega \nabla w\cdot\nabla T d\Omega+\int_\Gamma \mathbf{n}\cdot(w\nabla T)d\Gamma.$$

Dividindo o contorno em dois pedaços e como $w|_{\Gamma_1}=0$ (a função peso deve ser identicamente nula ao longo do contorno onde é imposta uma condição de contorno de Dirichlet), temos

$$\Rightarrow \int_\Gamma w\mathbf{n}\cdot\nabla T d\Gamma=\int_{\Gamma_1}\underbrace{w\mathbf{n}\cdot\nabla T}_{0}d\Gamma_1+\int_{\Gamma_2} w\underbrace{\mathbf{n}\cdot\nabla T}_{q}d\Gamma_2.$$

Finalmente chega-se a formulação fraca:

$$\boxed{-\int_\Omega \nabla w\ \cdot\nabla T\ d\Omega+\int_{\Gamma_2} w\ q\ d\Gamma_2=0.}$$

Procurando um espaço de dimensão finita que descreva de forma satisfatória a variável T, tem-se $U_h\subset U$ onde U_h tem dimensão N e T_h é escrito como combinação linear de funções base desse espaço, ou seja

$$T_h=\sum_{j=1}^{N} C_j\phi_j,$$

sendo $C_j's$ as N incógnitas.

As N equações linearmente independentes serão obtidas escolhendo-se N funções peso também linearmente independentes :

$$R_i = -\int_\Omega \nabla\psi_i \ \cdot \ \nabla T \ d\Omega + \int_{\Gamma_2} \psi_i \ q \ d\Gamma_2 = 0,$$

sendo $T_h = \sum_{j=1}^{N} C_j\phi_j$, então, $\nabla T = \sum_{j=1}^{N} C_j \nabla\phi_j$:

$$\int_\Omega \nabla\psi_i \cdot \nabla(\sum_{j=1}^{N} C_j\phi_j) \ d\Omega = \int_{\Gamma_2} \psi_i \ q \ d\Gamma_2 \Rightarrow$$

$$\sum_{j=1}^{N} C_j \underbrace{\left\{\int_\Omega \nabla\psi_i \cdot \nabla\phi_j \ d\Omega\right\}}_{A_{ij}} = \underbrace{\int_{\Gamma_2} \psi_i \ q \ d\Gamma_2}_{b_i}.$$

$$\boxed{A_{ij}c_j = b_i}$$

$$A_{ij} = \int_\Omega \nabla\psi_i \cdot \nabla\phi_j \ d\Omega \quad \text{e} \quad \mathrm{b_i} = \int_{\Gamma_2} \psi_\mathrm{i} \ \mathrm{q} \ \mathrm{d}\Gamma_2.$$

5.2 Método dos Elementos Finitos

Escolher ϕ_i, como representado na figura 5.2, de forma que

$$\phi(x_j) = \delta_{ij} \Rightarrow \phi_i(x_j) = \begin{cases} 1 & \text{se} \quad i = j \\ 0 & \text{se} \quad i \neq j \end{cases}$$

Ponto de Vista Elementar

Elemento bilinear, ilustrado na figura 5.3:

- $\phi_1(\xi,\eta) = \dfrac{(\eta-1)(\xi-1)}{4}$,
- $\phi_2(\xi,\eta) = -\dfrac{(\eta-1)(\xi+1)}{4}$,
- $\phi_3(\xi,\eta) = \dfrac{(\eta+1)(\xi+1)}{4}$,
- $\phi_4(\xi,\eta) = -\dfrac{(\eta+1)(\xi-1)}{4}$.

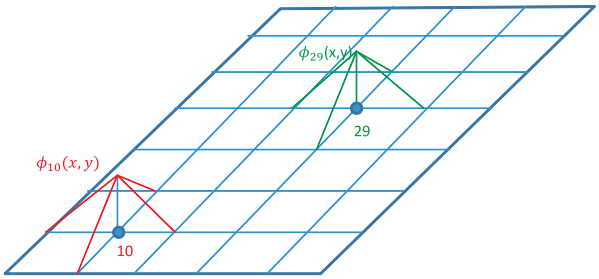

Figura 5.2: Função base bidimensional $\phi(x_j) = \delta_{ij}$

$$A_{ij}^{(e)} = \int_{\Omega_{(e)}} \nabla\phi_i \cdot \nabla\phi_j \; d\Omega,$$

$$A_{ij}^{(e)} = \int_{\Omega_{(e)}} \left[\frac{\partial\phi_i}{\partial x}\frac{\partial\phi_j}{\partial x} + \frac{\partial\phi_i}{\partial y}\frac{\partial\phi_j}{\partial y} \right] d\Omega.$$

A matriz $\mathbf{A^{(e)}}$ é uma matriz 4×4 no caso de elementos bilineares.

Precisamos conhecer as derivadas das funções base, $\dfrac{\partial\phi_i}{\partial x}$; $\dfrac{\partial\phi_i}{\partial y}$, para $i = 1 \cdots 4$, já que os $\phi_i's$ são definidos em termos de ξ e η.

Da mesma forma que no caso unidimensional, precisamos de um mapeamento do sistema de coordenada local (ξ, η) para o sistema de coordenadas global (x, y).

Uma vez obtida as matrizes elementares, temos que "montar" a matriz global. O processo de montagem é análogo ao caso unidimensional. Precisamos da matriz

DomNodeID$(ilnode, iele)$ = # global do nó local ilnode do elemento iele.

Esta matriz é obtida a partir da numeração dos elementos e nós usados para dividir o domínio de cálculo. A numeração tanto dos nós como dos elementos é uma escolha. Como essa numeração está

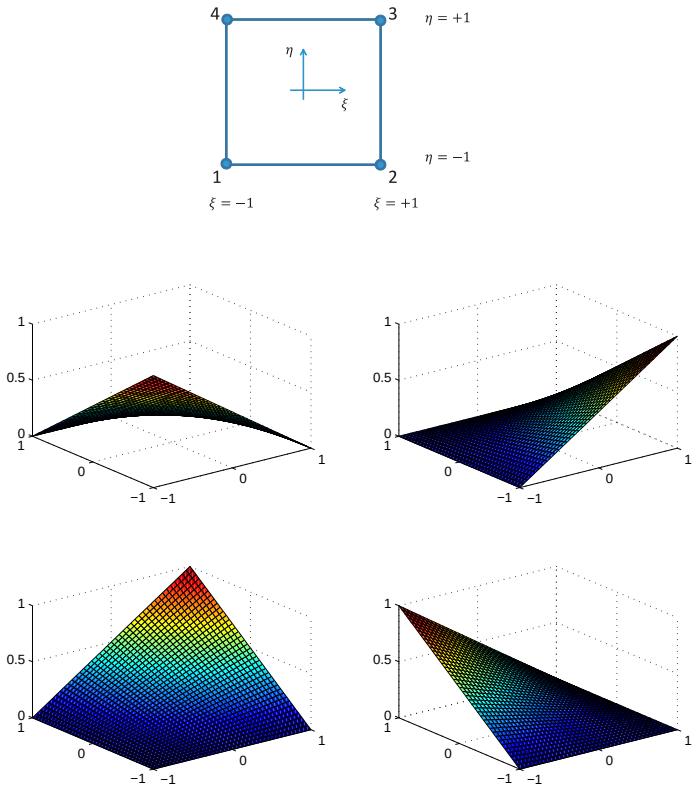

Figura 5.3: Função base de elemento bilinear.

diretamente relacionada com a estrutura da matriz global, é interessante usar uma numeração otimizada, que minimize a banda da matriz global ou que valorize alguma estrutura específica de matriz.

A figura 5.4 mostra duas diferentes numerações representando matrizes diferentes. A estrutura da matriz global é função direta da numeração global dos nós, como já foi dito. Repare a diferença da banda da matriz global para as diferentes numerações na figura 5.4. Existem diversos algoritmos para otimizar a numeração dos nós, que não serão tratados aqui.

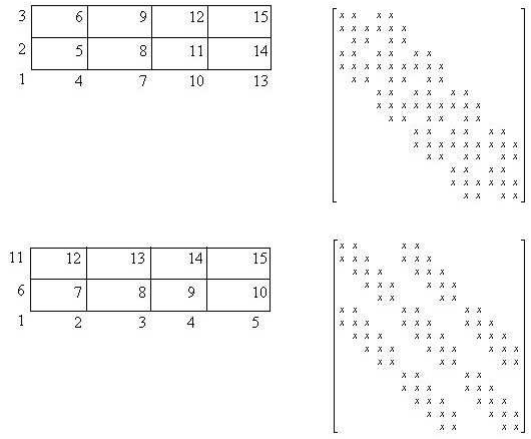

Figura 5.4: Dois esquemas de numeração de nós e suas respectivas matriz global.

A matriz `DomNodeID` para este problema é dada na tabela 5.1
A matriz $\mathbf{A}^{(5)}$ é montada na seguinte posição:

$$\mathbf{A} = \begin{bmatrix} & & & & \\ & \times & \times & \times & \times & \\ & \times & \times & \times & \times & \\ & & & & & \\ & \times & \times & \times & \times & \\ & \times & \times & \times & \times & \\ & & & & & \end{bmatrix}$$

Tabela 5.1: DomNodeID

ilnode\iele	1	2	3	4	5	6	7	8	9
1	1	2	3	5	6	7	9	10	11
2	5	6	7	9	10	11	13	14	15
3	6	7	8	10	11	12	14	15	16
4	2	3	4	6	7	8	10	11	12

onde $A^{(5)}_{1\,1} \rightarrow A_{6\,6}$; $A^{(5)}_{1\,2} \rightarrow A_{6\,10}$; $A^{(5)}_{1\,3} \rightarrow A_{6\,11}$; $A^{(5)}_{1\,4} \rightarrow A_{6\,7}$ $\cdots$

Para calcular elementos da matriz elementar, precisamos calcular as derivadas das funções base com relação a x e y:

$$A^{(e)}_{ij} = \int_{\Omega_{(e)}} \left[\frac{\partial \phi_i}{\partial x}\frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial y}\frac{\partial \phi_j}{\partial y} \right] d\Omega$$

Mas, a função $\phi(\xi, \eta)$ está definida em termos de ξ e η, logo temos que mudar de coordenadas e para isso vamos usar um mapeamento que escreva x e y como função de ξ e η: • $x = x(\xi, \eta)$ e • $y = y(\xi, \eta)$.

Mapeamento Isoparamétrico

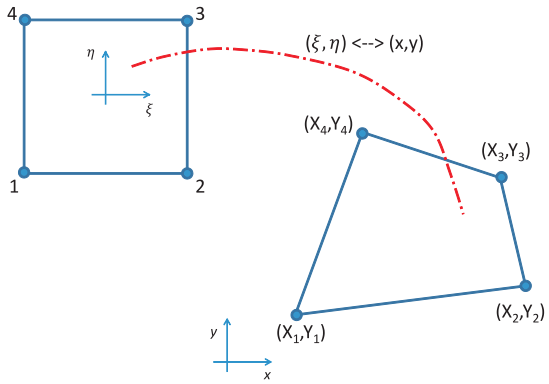

Figura 5.5: Mapeamento isoparamétrico

Requisitos para mapeamento isoparamétrico que é ilustrado na figura 5.5:

→ Linhas de contorno do elemento são mapeadas em linhas de contorno.

→ Nós globais são mapeados em nós locais.

Um exemplo de mapeamento muito utilizado é definido por:

$$\begin{cases} x(\xi,\eta) = \sum_{i=1}^{4} X_i \ \phi_i(\xi,\eta), \\ y(\xi,\eta) = \sum_{i=1}^{4} Y_i \ \phi_i(\xi,\eta), \end{cases}$$

note que :
$x_1 = X_1 \underbrace{\phi_1(-1,-1)}_{=1} + X_2 \underbrace{\phi_2(-1,-1)}_{=0} + X_3 \underbrace{\phi_3(-1,-1)}_{=0} + X_4 \underbrace{\phi_4(-1,-1)}_{=0} =$
X_1,
e assim em diante...

A função interpolação utilizada no mapeamento é a mesma função interpolação utilizada para aproximar a função T (elemento isoparamétrico).

Uma vez conhecendo o mapeamento entre as coordenadas, pode-se determinar : $\dfrac{\partial \phi_i}{\partial x}$ e $\dfrac{\partial \phi_i}{\partial y}$:

$$\begin{cases} \dfrac{\partial \phi_i}{\partial \xi} = \dfrac{\partial \phi_i}{\partial x}\dfrac{\partial x}{\partial \xi} + \dfrac{\partial \phi_i}{\partial y}\dfrac{\partial y}{\partial \xi}, \\ \dfrac{\partial \phi_i}{\partial \eta} = \dfrac{\partial \phi_i}{\partial x}\dfrac{\partial x}{\partial \eta} + \dfrac{\partial \phi_i}{\partial y}\dfrac{\partial y}{\partial \eta}. \end{cases}$$

O sistema linear pode ser escrito vetorialmente:

$$\begin{bmatrix} \dfrac{\partial \phi_i}{\partial \xi} \\ \dfrac{\partial \phi_i}{\partial \eta} \end{bmatrix} = \underbrace{\begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix}}_{\text{Jacobiano da Transformação}} \begin{bmatrix} \dfrac{\partial \phi_i}{\partial x} \\ \dfrac{\partial \phi_i}{\partial y} \end{bmatrix},$$

sendo a transformação inversível, temos:

$$\begin{bmatrix} \dfrac{\partial \phi_i}{\partial x} \\ \dfrac{\partial \phi_i}{\partial y} \end{bmatrix} = \mathbf{J^{-1}} \begin{bmatrix} \dfrac{\partial \phi_i}{\partial \xi} \\ \dfrac{\partial \phi_i}{\partial \eta} \end{bmatrix} \quad \text{onde} \quad \mathbf{J^{-1}} = \frac{1}{|\mathbf{J}|} \begin{bmatrix} \dfrac{\partial y}{\partial \eta} & -\dfrac{\partial y}{\partial \xi} \\ -\dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \xi} \end{bmatrix}$$

e $|\mathbf{J}| = \left(\dfrac{\partial x}{\partial \xi}\dfrac{\partial y}{\partial \eta} - \dfrac{\partial y}{\partial \xi}\dfrac{\partial x}{\partial \eta} \right)$.

Assim:

$$\frac{\partial \phi_i}{\partial x} = \frac{1}{\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}} \left[\frac{\partial y}{\partial \eta}\frac{\partial \phi_i}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial \phi_i}{\partial \eta} \right],$$

$$\frac{\partial \phi_i}{\partial y} = \frac{1}{\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}} \left[-\frac{\partial x}{\partial \eta}\frac{\partial \phi_i}{\partial \xi} + \frac{\partial x}{\partial \xi}\frac{\partial \phi_i}{\partial \eta} \right].$$

Para um mapeamento isoparamétrico temos que:

$\dfrac{\partial x}{\partial \xi} = \displaystyle\sum_{i=1}^{N} X_i \frac{\partial \phi_i}{\partial \xi}$; $\dfrac{\partial x}{\partial \eta} = \displaystyle\sum_{i=1}^{N} X_i \frac{\partial \phi_i}{\partial \eta}$; $\ldots$

Assim a integral elementar $A_{ij}^{(e)}$ pode ser reescrita em função das variáveis locais:

$$A_{ij}^{(e)} = \int_{\Omega_{(e)}} \nabla \phi_i \cdot \nabla \phi_j \; dxdy = \int_{-1}^{1}\int_{-1}^{1} \left(\frac{\partial \phi_i}{\partial x}\frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial y}\frac{\partial \phi_j}{\partial y} \right) |\mathbf{J}| \; d\xi \; d\eta$$

$$A_{ij}^{(e)} = \int_{-1}^{1}\int_{-1}^{1} \left\{ \left(\frac{\partial y}{\partial \eta}\frac{\partial \phi_i}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial \phi_i}{\partial \eta} \right) \left(\frac{\partial y}{\partial \eta}\frac{\partial \phi_j}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial \phi_j}{\partial \eta} \right) + \right.$$

$$+\left(-\frac{\partial x}{\partial \eta}\frac{\partial \phi_i}{\partial \xi}+\frac{\partial x}{\partial \xi}\frac{\partial \phi_i}{\partial \eta}\right)\left(-\frac{\partial x}{\partial \eta}\frac{\partial \phi_j}{\partial \xi}+\frac{\partial x}{\partial \xi}\frac{\partial \phi_j}{\partial \eta}\right)\Bigg\}\ \frac{1}{|\mathbf{J}|}d\xi d\eta$$

Repare que a expressão para o cálculo da matriz elementar contém o determinante da matriz Jacobiana, $|\mathbf{J}|$, no denominador, por isso é importante que esse determinante não seja muito pequeno, $|\mathbf{J}| \nrightarrow 0$. Se o elemento ficar muito distorcido isto pode acontecer e a precisão do cálculo de $A_{ij}^{(e)}$ fica comprometida, como exemplificado na figura 5.6.

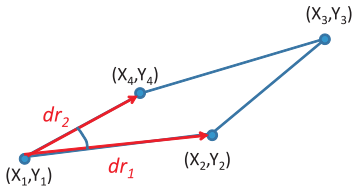

Figura 5.6: Elemento distorcido.

$$dA = |\mathbf{J}|d\xi d\eta = |d\mathbf{r_1}||d\mathbf{r_2}|\text{sen}\alpha$$

$$|\mathbf{J}| = \frac{|d\mathbf{r_1}||d\mathbf{r_2}|\text{sen}\alpha}{d\xi d\eta}$$

Quando $\alpha \to 0$ ou $\alpha \to 180$ graus , $|\mathbf{J}| \to 0$. Deve-se monitorar o valor de $|\mathbf{J}|$. Se ficar negativo ou muito pequeno, os cálculos devem ser interrompidos e uma nova malha deve ser criada.

Integração Numérica

Novamente o método escolhido para a integração numérica é quadratura Gaussiana: $\int_{-1}^{1}\int_{-1}^{1} F(\xi,\eta)d\xi d\eta = \sum_{igp=1}^{NGPI}\sum_{jgp=1}^{NGPJ} F(\xi_i,\eta_j)\ w_I\ w_J.$

A localização dos pontos está representada na figura 5.7 e os pontos e pesos podem ser encontrados na Tabela 5.2.

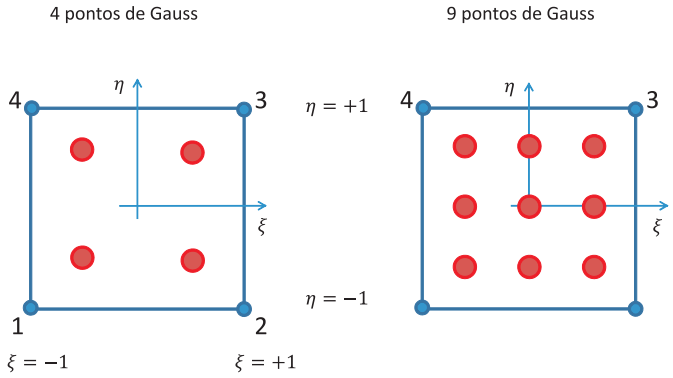

Figura 5.7: Localização dos Pontos de Gauss.

5.3 Algoritmo para o Cálculo da Matriz Elementar $\mathbf{A^{(e)}}$

O cálculo da matriz elementar representa a parte principal de um código de elementos finitos. Nesta seção apresentamos um algoritmo de cálculo da matriz elementar do problema bidimensional discutido neste capítulo.

Para facilitar a implementação, vamos considerar a estrutura de dados apresentada a seguir.

Os valores das coordenadas nodais do elemento são armazenadas em forma matricial, como

$$\mathrm{XY}(i,j) = \begin{bmatrix} X_1 & X_2 & X_3 & X_4 \\ Y_1 & Y_2 & Y_3 & Y_4 \end{bmatrix}.$$

Da mesma forma, os valores das funções base e suas derivadas sao armazenadas da seguinte forma:

$$\mathrm{PHI}(i) = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{bmatrix} \qquad \mathrm{GradPHI}(i,j) = \begin{bmatrix} \dfrac{\partial \phi_1}{\partial \xi} & \dfrac{\partial \phi_2}{\partial \xi} & \dfrac{\partial \phi_3}{\partial \xi} & \dfrac{\partial \phi_4}{\partial \xi} \\ \dfrac{\partial \phi_1}{\partial \eta} & \dfrac{\partial \phi_2}{\partial \eta} & \dfrac{\partial \phi_3}{\partial \eta} & \dfrac{\partial \phi_4}{\partial \eta} \end{bmatrix}$$

Tabela 5.2: Pontos e pesos para quadratura Gaussiana

Elemento	Num.pontos	Pesos	Coordenadas	
	INTEGRAÇÃO	w	ξ	η
linear	$2 \times 2 = 4$	1.0×1.0	-0.57735	-0.57735
		1.0×1.0	+0.57735	-0.57735
		1.0×1.0	-0.57735	+0.57735
		1.0×1.0	+0.57735	+0.57735
quadrático	$3 \times 3 = 9$	0.555×0.555	-0.77459	-0.77459
		0.888×0.555	0.0	-0.77459
		0.555×0.555	+0.77459	-0.77459
		0.555×0.888	-0.77459	0.0
		0.888×0.888	0.0	0.0
		0.555×0.888	+0.77459	0.0
		0.555×0.555	-0.77459	+0.77459
		0.888×0.555	0.0	+0.77459
		0.555×0.555	+0.77459	+0.77459

Observe que desta forma, a matriz Jacobiana da transformação de coordenadas pode ser facilmente calculada como

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \phi_1}{\partial \xi} & \dfrac{\partial \phi_2}{\partial \xi} & \dfrac{\partial \phi_3}{\partial \xi} & \dfrac{\partial \phi_4}{\partial \xi} \\ \\ \dfrac{\partial \phi_1}{\partial \eta} & \dfrac{\partial \phi_2}{\partial \eta} & \dfrac{\partial \phi_3}{\partial \eta} & \dfrac{\partial \phi_4}{\partial \eta} \end{bmatrix} \cdot \begin{bmatrix} X_1 & Y_1 \\ X_2 & Y_2 \\ X_3 & Y_3 \\ X_4 & Y_4 \end{bmatrix} \Rightarrow$$

$$\mathrm{JAC} = \mathrm{GradPHI} \cdot \mathrm{XY}^T$$

A matriz Jacobiana inversa $\mathbf{J^{-1}}$ pode ser facilmente calculada a partir dos elementos da matriz Jacobiana.

As derivadas das funções base em relação às coordenadas globais são armazenadas em forma matricial como:

$$\mathrm{GradPHIXY}(i,j) = \begin{bmatrix} \dfrac{\partial \phi_1}{\partial x} & \dfrac{\partial \phi_2}{\partial x} & \dfrac{\partial \phi_3}{\partial x} & \dfrac{\partial \phi_4}{\partial x} \\ \\ \dfrac{\partial \phi_1}{\partial y} & \dfrac{\partial \phi_2}{\partial y} & \dfrac{\partial \phi_3}{\partial y} & \dfrac{\partial \phi_4}{\partial y} \end{bmatrix}$$

.

Os valores podem ser calculados por um produto de matrizes:

$$\begin{bmatrix} \frac{\partial \phi_1}{\partial x} & \frac{\partial \phi_2}{\partial x} & \frac{\partial \phi_3}{\partial x} & \frac{\partial \phi_4}{\partial x} \\ \frac{\partial \phi_1}{\partial y} & \frac{\partial \phi_2}{\partial y} & \frac{\partial \phi_3}{\partial y} & \frac{\partial \phi_4}{\partial y} \end{bmatrix} = \mathbf{J^{-1}} \cdot \begin{bmatrix} \frac{\partial \phi_1}{\partial \xi} & \frac{\partial \phi_2}{\partial \xi} & \frac{\partial \phi_3}{\partial \xi} & \frac{\partial \phi_4}{\partial \xi} \\ \frac{\partial \phi_1}{\partial \eta} & \frac{\partial \phi_2}{\partial \eta} & \frac{\partial \phi_3}{\partial \eta} & \frac{\partial \phi_4}{\partial \eta} \end{bmatrix}$$

$$\text{GradPHIXY} = \mathbf{J^{-1}} \cdot \text{GradPHI}$$

A matriz elementar pode ser calculada através do algoritmo apresentado a seguir.

`getelemA`

```
%=========================================================================
% Calculo da matriz elementar
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [elemA] = getelemA(iele, DomNodeID, X, Y);
% Função que cria a matriz elementar
elemA = zeros(4,4);
% Número de pontos de Gauss
NGP=2;
% Peso para cada ponto de Gauss
WGP=[1.0, 1.0];
% Posição dos números de Gauss
XGP = [ -0.57735, 0.57735];
% Valor elementar das coordenadas dos nos
for ilnode = 1:4
    ignode = DomNodeID(ilnode,iele);
    XY(1,ilnode) = X(ignode);
    XY(2,ilnode) = Y(ignode);
end
% Loop nos pontos de Gauss
for igp = 1:NGP
    XI = XGP(igp);
    WI = WGP(igp);
    for jgp = 1:NGP
        NETA = XGP(jgp);
        WJ =WGP(jgp);
        W = WI*WJ;
        [Phi,GradPhi] = basisfunc(XI,NETA);
        [JACinv, detJ] = mapping(XY, GradPhi)
        % calculando o GradPhi-xy
        GradPhixy = JACinv*GradPhi;
```

```
        for ilnode = 1:4
            for jlnode =1:4
                elemA(ilnode,jlnode) = elemA(ilnode,jlnode) + W*detJ * ...
                (GradPhixy(1,ilnode)*GradPhixy(1,jlnode) + ...
                GradPhixy(2,ilnode)*GradPhixy(2,jlnode));

            end
        end
    end
end
%=========================================================================
```

5.4 Exercícios

1. Mostre que o mapeamento isoparamétrico satisfaz a condição que a imagem da linha $\eta = 1$ é mapeada no contorno do elemento entre os nós 3 e 4.

2. Escreva a rotina `basisfunc` para elementos bilineares.

3. Escreva a rotina `mapping` indicada no algoritmo acima.

4. Escreva a expressão para os elementos da matriz elementar para o seguinte problema bidimensional (considere coordenadas cartesianas).

$$\mathbf{v} \cdot \nabla T - \nabla^2 T = 0 \text{ em } \Omega$$

Capítulo 6

Solução da Equação de Navier-Stokes pelo Método de Elementos Finitos

6.1 Formulação Forte: Equações de Conservação de Massa e Quantidade de Movimento

O escoamento de fluidos é regido pelos princípios de conservação de massa e de quantidade de movimento.

Para um escoamento em regime permanente (campos não variam no tempo), a equação de conservação de massa é dada pela equação (6.1.1):

$$\nabla \cdot (\rho \mathbf{u}) = 0, \tag{6.1.1}$$

onde ρ é a massa específica do fluido e $\mathbf{u}$ é o campo de velocidade. Se consideramos o fluido incompressível, isto é a sua massa específica não varia com a pressão, a equação é simplificada e escrita como a equação (6.1.2):

$$\nabla \cdot \mathbf{u} = 0. \tag{6.1.2}$$

A equação de Navier-Stokes descreve a conservação de quantidade

de movimento para fluidos incompressíveis. Para um escoamento em regime permanente, é dada pela equação (6.1.3):

$$\rho\ \mathbf{u} \cdot \nabla \mathbf{u} = \nabla \cdot \mathbf{T}, \tag{6.1.3}$$

onde $\mathbf{T}$ é o tensor das tensões. Para um fluido Newtoniano, o tensor das tensões varia linearmente com a taxa de deformação do fluido

$$\mathbf{T} = -p\mathbf{I} + \mu[\nabla \mathbf{u} + \nabla \mathbf{u}^T].$$

A equação de Navier-Stokes é não-linear (termo $\mathbf{u}\cdot\nabla\mathbf{u}$) e de segunda ordem (termo $\nabla \cdot (\nabla \mathbf{u})$). Os campos de velocidade $\mathbf{u}$ e pressão p são obtidos pela solução simultânea das equações 6.1.2 e 6.1.3.

Condições de contorno são necessárias para a solução do problema. Como a equação de Navier-Stokes é de segunda ordem, condições de contorno devem ser especificadas em todas as fronteiras do domínio. Dois tipos de informações podem ser especificadas ao longo da fronteira: velocidade ou força agindo no contorno. A condição de contorno a ser utilizada em cada parte da fronteira depende da física do problema. Como ilustração, a figura 6.1 apresenta o esquema de um escoamento através de uma contração. A fronteira pode ser claramente dividida em quatro partes:

1. Parede sólida: Ao longo de uma parede, considera-se as condições de impermeabilidade e não-deslizamento. Para o caso de uma parede estacionária, isto equivale a especificar uma velocidade nula ao longo da fronteira 1.

$$\mathbf{u} = 0.$$

2. Linha de simetria: Não há fluxo através da linha de simetria, fronteira 2, e a tensão cisalhante é nula:

$$\mathbf{n} \cdot \mathbf{u} = v = 0 \quad ; \qquad \mathbf{t} \cdot (\mathbf{n} \cdot \mathbf{T}) = 0$$

3. Seção de entrada do escoamento: Esta não é uma fronteira real do escoamento, mas um contorno fictício para limitar o domínio de interesse. Normalmente, alguma hipótese é feita sobre o campo

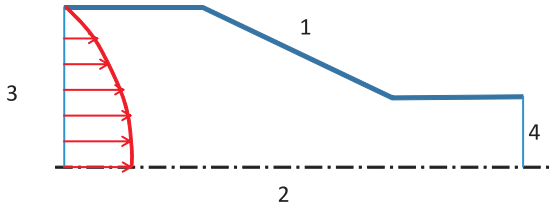

Figura 6.1: Domínio para problema de escoamento.

de velocidade, como por exemplo considerar, na fronteira 3, que o escoamento é desenvolvido: a componente vertical da velocidade é nula e a componente horizontal é uma função quadrática da coordenada y:

$$\mathbf{u} = (u = u(y), v = 0).$$

4. Seção de saída do escoamento: Novamente, não é uma fronteira real do escoamento, mas um contorno fictício definido para limitar o domínio de interesse. A hipótese mais utilizada é novamente assumir que o plano encontra-se longe o suficiente da contração de forma que o escoamento esteja plenamente desenvolvido. As condições de contorno normalmente impostas, na fronteira 4, são:

$$\mathbf{n} \cdot (\nabla \cdot \mathbf{u}) = 0 \quad ; \qquad p = p_{\text{ saida}}.$$

6.2 Formulação Fraca: Método dos Resíduos Ponderados

Para resolver o sistema de equações diferenciais parciais pelo método dos resíduos ponderados, temos que multiplicar o resíduo da aproximação de cada equação por um função peso e forçar a integral ao longo de todo o domínio Ω a ser nula.

Como a equação de consevação da quantidade de movimento (6.1.3) é uma equação vetorial, o resíduo ponderado correspondente a esta equação será calculado através do produto interno do resíduo da aproximação com um função peso vetorial $\mathbf{W}$. A equação da continuidade

é uma equação escalar e desta forma, a função peso utilizada w é uma função escalar, como as discutidas nos capítulos anteriores.

$$R_m = \int_\Omega \left[\rho\ \mathbf{u}\cdot\nabla\mathbf{u} - \nabla\cdot\mathbf{T}\right]\cdot\mathbf{W}\ d\Omega = 0; \tag{6.2.4}$$

$$R_c = \int_\Omega \left[\nabla\cdot\mathbf{u}\right] w\ d\Omega = 0. \tag{6.2.5}$$

6.2.1 Resíduo Ponderado da Equação de Quantidade de Movimento

O resíduo ponderado da equação de conservação de quantidade de movimento, equação (6.2.4), será desenvolvido termo a termo:

$$R_m = \int_\Omega \rho\ (\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{W} d\Omega - \int_\Omega (\nabla\cdot\mathbf{T})\cdot\mathbf{W}\ d\Omega.$$

O termo $\nabla\cdot\mathbf{T}$ apresenta segunda derivadas da velocidade (variável primitiva do problema). Vamos integrar por partes para transferir a derivada para a função peso, conforme já feito anteriormente, usando a seguinte igualdade tensorial:

$$\boxed{\mathbf{T} : \nabla\mathbf{W} = \nabla\cdot(\mathbf{T}\cdot\mathbf{W}) - (\nabla\cdot\mathbf{T})\cdot\mathbf{W}}$$

As operações envolvendo vetores e tensores podem ser encontradas em [1]. Desta forma,

$$\int_\Omega (\nabla\cdot\mathbf{T})\cdot\mathbf{W}\ d\Omega = -\int_\Omega \mathbf{T} : \nabla\mathbf{W} + \int_\Omega \nabla\cdot(\mathbf{T}\cdot\mathbf{W})\ d\Omega.$$

Usando o teorema de Gauss podemos escrever:

$$\int_\Omega \nabla\cdot(\mathbf{T}\cdot\mathbf{W})\ d\Omega = \int_\Gamma \mathbf{n}\cdot(\mathbf{T}\cdot\mathbf{W})\ d\Gamma.$$

O resíduo ponderado da equação de conservação de quantidade de movimento é escrito em uma forma onde não aparecem termos de

segunda derivadas das funções base (usadas na expansão dos campos desconhecidos) ou peso:

$$R_m = \int_\Omega \rho\ (\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{W} + \int_\Omega \mathbf{T} : \nabla\mathbf{W} - \int_\Gamma (\mathbf{n}\cdot\mathbf{T})\cdot\mathbf{W}\ d\Gamma.$$

A função peso vetorial $\mathbf{W}$ pode ser escrita em termos de suas componentes: $\mathbf{W} = [W_1, W_2]$ e cada termo do resíduo da equação de conservação da quantidade de movimento escrito em termos de suas componentes. No caso particular de sistema cartesiano, cada termo do resíduo ponderado é escrito como:

1) $(\mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{W} = W_1\left(u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y}\right) + W_2\left(u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y}\right).$

2) $\mathbf{T} : \nabla\mathbf{W} = \dfrac{\partial W_1}{\partial x}\underbrace{\left[-p + 2\mu\dfrac{\partial u}{\partial x}\right]}_{T_{xx}} + \dfrac{\partial W_1}{\partial y}\underbrace{\left[\mu(\dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x})\right]}_{T_{xy}} +$

$$+ \frac{\partial W_2}{\partial x}\underbrace{\left[\mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})\right]}_{T_{yx}=T_{xy}} + \frac{\partial W_2}{\partial y}\underbrace{\left[-p + 2\mu\frac{\partial v}{\partial y}\right]}_{T_{yy}}.$$

3) $(\mathbf{n}\cdot\mathbf{T})\cdot\mathbf{W} = f_x\ W_1 + f_y\ W_2.$

Cada componente da função peso vetorial pode ser escrita como combinação linear de funções base escalar ψ_i. Se cada componente W_1 e W_2 pertence a um espaço vetorial de dimensão n, a função peso vetorial $\mathbf{W}$ pertence a um espaço de dimensão $2n$ que é gerado pela seguinte base de funções:

$$\underbrace{\mathbf{W} = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}}_{\text{Espaço dim}=2n}$$

$$\left\langle \underbrace{\begin{bmatrix} \psi_1 \\ 0 \end{bmatrix}, \begin{bmatrix} \psi_2 \\ 0 \end{bmatrix}, \cdots, \begin{bmatrix} \psi_n \\ 0 \end{bmatrix}}_{n \text{ funções}}, \underbrace{\begin{bmatrix} 0 \\ \psi_1 \end{bmatrix}, \begin{bmatrix} 0 \\ \psi_2 \end{bmatrix}, \cdots, \begin{bmatrix} 0 \\ \psi_n \end{bmatrix}}_{n \text{ funções}} \right\rangle.$$

Definindo o espaço das funções peso desta forma, as componentes da equação de conservação podem ser desacopladas, pois para as primeiras n funções base aparecem termos correspondentes à primeira componente da velocidade, já que $W_2 = 0$ e para as últimas n funções base aparecem temos correspondentes à segunda componente da velocidade, já que para estas funções base $W_1 = 0$.

Substituindo a expressão do tensor das tensões para fluidos Newtonianos, $\mathbf{T} = -p\mathbf{I} + \mu[\nabla\mathbf{u} + \nabla\mathbf{u}^T]$, as $2n$ equações algébricas associadas à equação de conservação da quantidade de movimento são escritas como:

$$R^i_{mx} = \int_\Omega \rho\ \psi_i \left[u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right] + \frac{\partial \psi_i}{\partial x}\left[-p + 2\mu\frac{\partial u}{\partial x}\right] + \tag{6.2.6}$$
$$+ \frac{\partial \psi_i}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] d\Omega - \int_\Gamma \psi_i\ f_x\ d\Gamma\ ; \quad i = 1, \cdots, n.$$

$$R^i_{my} = \int_\Omega \rho\ \psi_i \left[u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right] + \frac{\partial \psi_i}{\partial x}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \tag{6.2.7}$$
$$+ \frac{\partial \psi_i}{\partial y}\left[-p + 2\mu\frac{\partial v}{\partial y}\right] d\Omega - \int_\Gamma \psi_i\ f_y\ d\Gamma\ ; \quad i = 1, \cdots, n.$$

6.2.2 Resíduo Ponderado da Equação de Conservação de Massa

O resíduo ponderado da equação de conservação de massa (6.2.5) não apresenta nenhum termo com segunda derivadas dos campos de velocidade e pressão. Desta forma, não é necessária nenhuma manipulação para remover estes termos.

$$Rc = \int_{\Omega} \left[\nabla \cdot \mathbf{u}\right] \; w \; d\Omega = 0,$$

onde w é uma função escalar que pertence ao espaço de funções gerado pelas seguintes funções peso: $\{\zeta_1, \zeta_2, \cdots, \zeta_m\}$. As m equações algébricas associadas á equação da continuidade são escritas como:

$$R_c^i = \int_{\Omega} \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) \; \zeta_i \; d\Omega \; ; \; i = 1, \cdots, m. \tag{6.2.8}$$

6.2.3 Expansão dos Campos Desconhecidos

Os campos de velocidade e pressão devem ser escritos como uma expansão linear das funções base dos espaços de cada um dos campos, ϕ_i e χ_i

Os n resíduos ponderados associados a cada componente da equação de conservação de quantidade de movimento contém termos com derivadas de velocidade e pressão. Para que estes termos tenham a mesma precisão de discretização, as funções bases utilizadas para pressão não precisam ser da mesma ordem que as usadas para o campo de velocidade. Quando os dois campos de variáveis pertencem a espaços diferentes, diz-se que a Formulação é Mista (*Mixed Finite Element*).

$$\mathbf{u} = \left[\begin{array}{c} u \\ v \end{array}\right] = \left[\begin{array}{c} \sum_{i=1}^{n} U_i \phi_i \\ \sum_{i=1}^{n} V_i \phi_i \end{array}\right] \Rightarrow \; 2 \; n \text{ incógnitas;}$$

$$p = \sum_{i=1}^{m} P_i \; \chi_i \Rightarrow \; m \text{ incógnitas.}$$

As incógnitas do problema discretizado são os coeficientes destas expansões lineares, U_i e V_i $(i = 1, \ldots, N)$ e P_i $(i = 1, \ldots, m)$. O número de incógnitas $(2n + m)$ é igual ao número de equações algébricas.

Como o problema discreto não contém derivadas da função base do espaço de pressão χ_i, este espaço pode ser de menor dimensão que

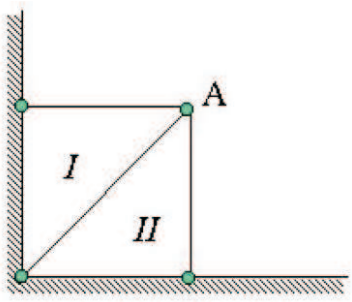

Figura 6.2: Exemplo de formulação instável - malha truncada.

o espaço usado para o campo de velocidade, $m < N$. Além disto, a formulação fraca permite que as funções χ_i sejam descontínuas. Observe que as funções usadas como base do campo de velocidade ϕ_i não podem ser descontínuas, já que a formulação apresenta termos do tipo $\int |\nabla \phi_i|^2$, que desta forma deve ser integrável.

Não é qualquer combinação arbitrária de funções base $\phi_i's$ e $\chi_i's$ que leva a formulações com bom desempenho numérico. Uma escolha errada de funções base pode levar a formulações instáveis.

Um exemplo simples que ilustra o tipo de problema que pode ocorrer com uma escolha errada de combinação de espaços é discutido a seguir. Para facilitar a interpretação geométrica, vamos analisar um problema de elasticidade linear de materiais incompressíveis, onde $\mathbf{u}$ é o campo de deslocamento e $\mathbf{T}$ o tensor das tensões, função da pressão e do gradiente de deformação. A primeira equação corresponde ao equilíbrio de forças e a segunda à condição de incompressibilidade.

$$\nabla \cdot \mathbf{T} = 0, \qquad \nabla \cdot \mathbf{u} = 0.$$

Vamos considerar a situação de dois elementos triangulares, como mostrado na figura 6.2 e funções base ϕ_i para o campo de deslocamento, linear $n = 2$, e χ_i para o campo de pressão, constante $m = 1$.

A formulação fraca da condição de incompressibilidade é escrita

como

$$\int_{\Omega^{(e)}} (\nabla \cdot \mathbf{u}) \underbrace{\chi_i}_{\text{cte}} \, d\Omega = 0 \quad \Rightarrow \qquad \int_{\Omega^{(e)}} \nabla \cdot \mathbf{u} \, d\Omega = 0.$$

Pelo teorema de Gauss, a integral de área do divergente do campo de deformação é igual a integral ao longo da fronteira da deformação normal a superfície, isto é representa a variação da área de cada elemento.

A incompressibilidade do material com χ_i constante leva a uma condição de área constante em todos os elementos. Desta forma, quando esta condição de área constante é aplicada ao elemento triangular `I` da figura, o produto da base pela altura do elemento não pode variar, logo o nó A só pode deslocar-se na vertical. Aplicando o mesmo raciocínio para o elemento `II` da figura, chega-se a conclusão que o nó A só pode se deslocar na direção horizontal. A escolha de função constante para o campo de pressão em um elemento triangular leva a uma deformação nula do ponto A não relacionada à física do problema, apenas a uma instabilidade numérica da formulação discreta.

A figura 6.3, adaptada do livro de Gresho e Sani, mostra as diferentes combinações de funções bases mais usadas para elementos quadrangulares e triangulares.

6.2.4 Elemento Biquadrático (velocidade) e Linear Descontínuo (pressão)

No programa apresentado ao final deste capitulo, vamos adotar como exemplo elementos quadrangulares com funções biquadráticas para a velocidade e linear descontínua para pressão. A figura 6.4 mostra uma representação esquemática do elemento escolhido com 9 nós e a numeração local dos nós adotada aqui. Neste elemento, cada componente da velocidade possui 9 graus de liberdade e o campo de pressão é representado por 3 graus de liberdade. Cada elemento possui 21 graus de liberdade.

Dentro de cada elemento, os campos de velocidade e pressão são escritos como:

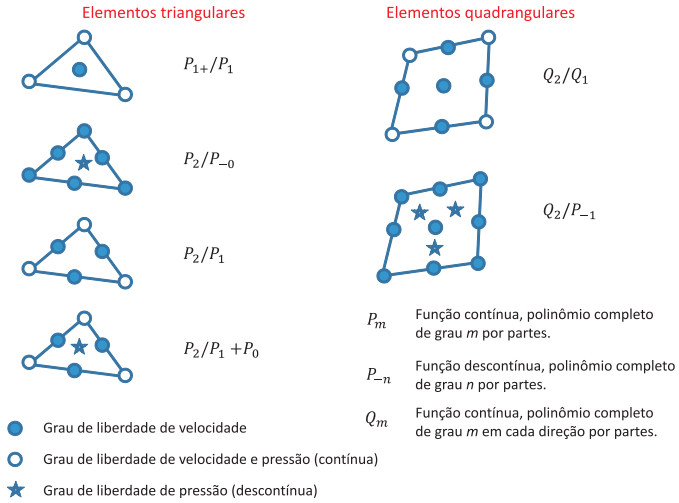

Figura 6.3: Elementos triangulares e quadrangulares mais utilizados na literatura.

$$u = \sum_{i=1}^{9} U_i \phi_i \quad ; \quad \sum_{i=1}^{9} V_i \phi_i \quad ; \quad p = \sum_{i=1}^{3} P_i \ \chi_i. \tag{6.2.9}$$

As funções base biquadráticas ϕ_i utilizadas para expandir o campo de velocidade serão Lagrangeanas, isto é:

$$\phi_i(X_j) = \delta_{ij},$$

cada função base é igual a 1 nó associado a função e nula nos demais. Cada coeficiente U_j e V_j representa o valor da velocidade u e v no nó j. Em termos das coordenadas locais, as funções base para a velocidade são:

$$\phi_1(\xi,\eta) = \frac{\xi(\xi-1)\eta(\eta-1)}{4}; \qquad \phi_2(\xi,\eta) = \frac{\xi(\xi+1)\eta(\eta-1)}{4};$$

$$\phi_3(\xi,\eta) = \frac{\xi(\xi+1)\eta(\eta+1)}{4}; \qquad \phi_4(\xi,\eta) = \frac{\xi(\xi-1)\eta(\eta+1)}{4};$$

$$\phi_5(\xi,\eta) = \frac{(1-\xi^2)\eta(\eta-1)}{2}; \qquad \phi_6(\xi,\eta) = \frac{\xi(\xi+1)(1-\eta^2)}{2};$$

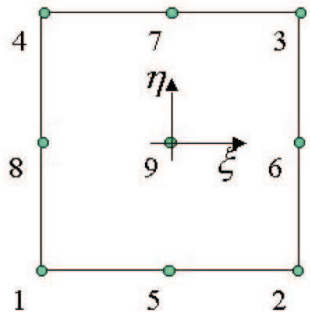

Figura 6.4: Elemento biquadrático de 9 nós.

$$\phi_7(\xi,\eta) = \frac{(1-\xi^2)\eta(\eta+1)}{2}; \qquad \phi_8(\xi,\eta) = \frac{\xi(\xi-1)(1-\eta^2)}{2};$$
$$\phi_9(\xi,\eta) = (1-\xi^2)(1-\eta^2).$$

A figura 6.5 ilustra a forma de algumas destas funções base.

As funções base utilizadas para expandir o campo de pressão não são Lagrangeanas. Para este elemento, elas são escolhidas de forma que o primeiro grau de liberdade de pressão P_1 represente o valor da pressão no centro do elemento; o segundo grau de liberdade P_2, a derivada da pressão na direção η; e o terceiro grau de liberdade P_3, a derivada da pressão na direção ξ. Para isto, a variação da pressão em cada elemento deve ser escrita como:

$p = P_1 + P_2\ \eta + P_3\ \xi$, consequentemente, as função base ψ_i são:

$$\psi_1(\xi,\eta) = 1 \quad ; \quad \psi_2(\xi,\eta) = \eta \quad \text{e} \quad \psi_3(\xi,\eta) = \xi.$$

Nos capítulos anteriores, a numeração dos graus de liberdade dos elementos se confundia com a própria numeração dos nós. O processo de montagem da matriz global era feito utilizando a relação entre a numeração elementar e global dos nós, armazenada na variável `DomNodeID`.

Como agora estamos resolvendo um sistema de equações diferenciais parciais para os campos de velocidade (campo vetorial) e pressão

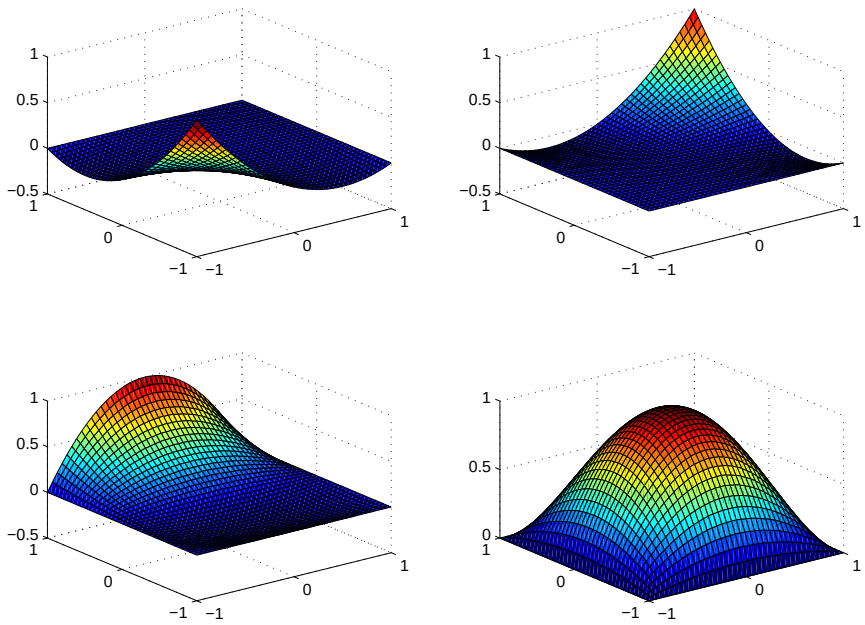

Figura 6.5: Exemplo de funções base biquadráticas: ϕ_1, ϕ_3, ϕ_4 e ϕ_9.

Tabela 6.1: Grau de liberdade

#	grau de liberdade
1	U_1
2	U_2
⋮	⋮
8	U_8
9	U_9
10	V_1
11	V_2
⋮	⋮
17	V_8
18	V_9
19	P_1
20	P_2
21	P_3

(escalar), o número de graus de liberdade de cada elemento é maior do que o número de nós do elemento. O elemento que estamos trabalhando possui 9 nós e 21 graus de liberdade. São eles: $U_1, ..., U_9; V_1, ..., V_9; P_1, P_2, P_3$. A montagem da matriz global não pode ser feita através da relação entre as numerações dos nós, mas sim através de uma nova tabela que relacione a numeração elementar e global dos graus de liberdade do problema. Esta relação será armazenada na variável `AssMtrx`:

`AssMtrx(ildof,iele)` = numero global do grau de liberdade local `ildof` do elemento `iele`.

A numeração dos graus de liberdade elementar é totalmente arbitrária. A numeração local adotada no exemplo aqui apresentado é mostrada na tabela 6.1 e esquematizada no elemento da figura 6.6.

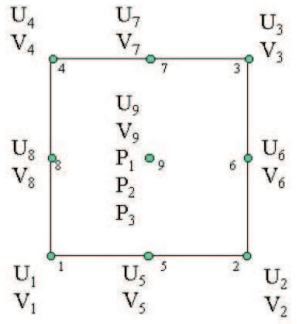

Figura 6.6: Numeração local dos graus de liberdade.

A apresentação do processo de montagem da matriz e vetor global será feita através de um exemplo simples de uma malha com 4 elementos quadrangulares (com 25 nós), mostrada na figura 6.7. A figura também apresenta a numeração global dos nós e dos graus de liberdade. A matriz `DomNodeID` que relaciona a numeração dos nós é apresentada na tabela 6.2.

Tabela 6.2: DomNodeID

ilnode \ iele	1	2	3	4
1	1	4	2	3
2	2	3	16	17
3	3	10	17	22
4	4	11	3	10
5	5	7	18	20
6	6	12	19	23
7	7	13	20	24
8	8	14	6	12
9	9	15	21	25

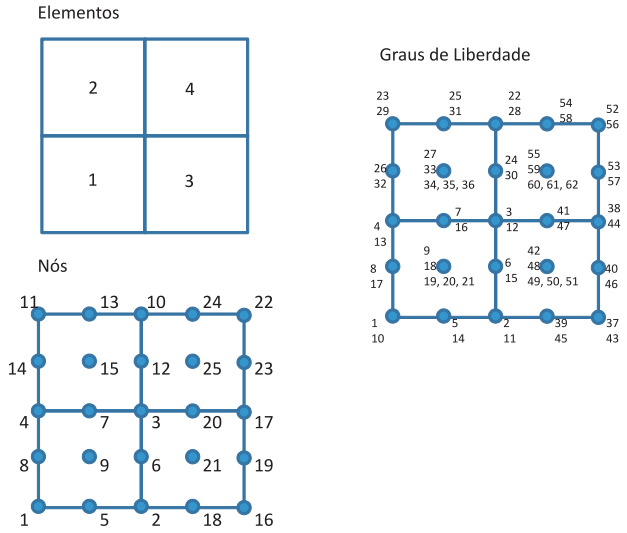

Figura 6.7: Exemplo apresentando a numeração de elementos, nós e graus de liberdade.

A matriz `AssMtrx` que relaciona a numeração dos graus de liberdade é apresentada na tabela 6.3.

Como já mencionado, o processo de montagem é feito utilizando esta última matriz. O algoritmo esquemático para o cálculo da matriz

Tabela 6.3: AssMtrx

ildof\iele	1	2	3	4
1	1	4	2	3
2	2	3	37	38
3	3	22	38	52
4	4	23	3	22
5	5	7	39	41
6	6	24	40	53
7	7	25	41	54
8	8	26	6	24
9	9	27	42	55
10	10	13	11	12
11	11	12	43	44
12	12	28	44	56
13	13	29	12	28
14	14	16	45	47
15	15	30	46	57
16	16	31	47	58
17	17	32	15	30
18	18	33	48	59
19	19	34	49	60
20	20	35	50	61
21	21	36	51	62

global é apresentado a seguir.

```
for iele = 1:NELE      % NELE → # de elementos
  [Aelem,felem] = getelem(iele, ...)
  for ildof = 1:NLOCALDoF      % NLOCALDoF → # de graus de liberdade local
    igdof = AssMtrx(ildof,iele) % igdof → # global do gdl ildof
    for jldof = 1, NLOCALDoF
      jgdof = AssMtrx(jldof,iele)
        A(igdof,jgdof) = A(igdof,jgdof) + Aelem(ildof,jldof)
    end
  end
end
```

Substituindo as expansões dos campos de velocidade e pressão, (6.2.9), nas equações (6.2.6), (6.2.7) e (6.2.8) dos resíduos ponderados, obtém-se um sistema de $2n + m$ equações algébricas não lineares. As não linearidades vêm dos termos convectivos da equação de Navier-Stokes.

A próxima seção apresenta a solução deste sistema não linear pelo método de Newton.

6.3 Solução pelo Método de Newton

Vamos apresentar um código no qual o sistema de equações algébricas não linear resultante do processo de discretização será resolvido pelo método de Newton. O sistema de equações pode ser representado como:

$$\mathbf{R}(\mathbf{c}) = 0,$$

onde $\mathbf{c} = [U_1, ..., U_n; V_1, ..., V_n; P_1, ..., P_m]^T$ é o vetor de incógnitas do problema e $\mathbf{R}$ é o vetor contendo os resíduos ponderados. No método de Newton, a solução é obtida pelo seguinte processo iterativo:

$\mathbf{c} = \mathbf{c}_0$
`while` $\|\mathbf{R}(\mathbf{c})| > \epsilon$
  $\mathbf{J}\boldsymbol{\Delta}\mathbf{c} = -\mathbf{R}$
  $\mathbf{c} = \mathbf{c} + \boldsymbol{\Delta}\mathbf{c}$
`end`

onde a matriz Jacobiana,**J**, representa a sensibilidade de cada equação em relação a cada incógnita: $J_{ij} = \dfrac{\partial R_i}{\partial c_j}$.

O vetor resíduo elementar será um vetor com 21 componentes, seguindo a numeração local dos graus de liberdade apresentada anteriormente. A expressão dos elementos deste vetor é apresentada a seguir:

Resíduo ponderado de R^i_{mx}:

$$R^i_{mx} = \int_\Omega Re\ \psi_i \left[u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right] + \frac{\partial \psi_i}{\partial x}\left[-p + 2\mu\frac{\partial u}{\partial x}\right] + \\ + \frac{\partial \psi_i}{\partial y}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] d\Omega.$$

Resíduo ponderado de R^i_{my}:

$$R^i_{my} = \int_\Omega Re\ \psi_i \left[u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right] + \frac{\partial \psi_i}{\partial x}\left[\mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right] + \\ + \frac{\partial \psi_i}{\partial y}\left[-p + 2\mu\frac{\partial v}{\partial y}\right] d\Omega.$$

Resíduo ponderado de R^i_c:

$$R^i_c = \int_\Omega \left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial x}\right)\ \chi_i\ d\Omega.$$

Como discutido no capítulo anterior, mapeamento isoparamétrico é usado para transformar as coordenadas global x, y em coordenadas local ξ, η, de forma que as integrais ao longo do elemento sejam escritas como:

$$\int_\Omega F(x,y)\ dxdy = \int_{\overline{\Omega}} \overline{F}(\xi,\eta)\ |\mathbf{J}|\ d\xi d\eta.$$

As integrais são calculadas usando o método de integração de Quadratura Gaussiana:

$$\int_{\overline{\Omega}} \overline{F}(\xi,\eta)\ d\xi d\eta = \sum_{igp}\sum_{jgp} \overline{F}(\xi_{igp},\eta_{jgp})\ W_{igp}W_{jgp}.$$

De forma análoga, a matriz Jacobiana elementar será uma matriz 21×21 com a seguinte estrutura de blocos:

$$\mathbf{J}^{(e)} = \begin{bmatrix} \dfrac{\partial R^i_{mx}}{\partial U_j} & \dfrac{\partial R^i_{mx}}{\partial V_j} & \dfrac{\partial R^i_{mx}}{\partial P_j} \\ \dfrac{\partial R^i_{my}}{\partial U_j} & \dfrac{\partial R^i_{my}}{\partial V_j} & \dfrac{\partial R^i_{my}}{\partial P_j} \\ \dfrac{\partial R^i_{c}}{\partial U_j} & \dfrac{\partial R^i_{c}}{\partial V_j} & \dfrac{\partial R^i_{c}}{\partial P_j} \end{bmatrix}.$$

As primeiras 9 linhas da matriz Jacobiana elementar correspondem aos 9 resíduos da componente x da equação de Navier-Stokes; as 9 linhas seguintes, aos 9 resíduos da componente y; e as 3 últimas linhas, aos 3 resíduos da equação de continuidade. Da mesma forma, as 9 primeiras colunas estão associadas aos coeficientes U_j; as 9 seguintes, aos coeficientes V_j; e as 3 últimas colunas, aos 3 coeficientes P_j.

A expressão de cada cada bloco desta estrutura é apresentada a seguir:

Jacobiano de R^i_{mx}:

$$\frac{\partial R^i_{mx}}{\partial U_j} = \int_{\Omega_e} \left\{ \rho\ \psi_i \left[\phi_j \frac{\partial u}{\partial x} + u \frac{\partial \phi_j}{\partial x} + v \frac{\partial \phi_j}{\partial y} \right] + \right.$$
$$\left. + \frac{\partial \psi_i}{\partial x}\ 2\ \mu\ \frac{\partial \phi_j}{\partial x} + \ \mu\ \frac{\partial \psi_i}{\partial y}\ \frac{\partial \phi_j}{\partial y} \right\}\ d\Omega_e;$$

$$\frac{\partial R^i_{mx}}{\partial V_j} = \int_{\Omega_e} \left\{ \rho\ \psi_i\ \phi_j\ \frac{\partial u}{\partial y}\ +\ \mu\ \frac{\partial \psi_i}{\partial y}\ \frac{\partial \phi_j}{\partial x} \right\}\ d\Omega_e;$$

$$\frac{\partial R^i_{mx}}{\partial P_j} = \int_{\Omega_e} -\ \frac{\partial \psi_i}{\partial x}\ \chi_i\ d\Omega_e.$$

Jacobiano de R^i_{my}:

$$\frac{\partial R^i_{my}}{\partial U_j} = \int_{\Omega_e} \left\{ \rho\ \psi_i\ \phi_j\ \frac{\partial v}{\partial x}\ +\ \mu\ \frac{\partial \psi_i}{\partial x}\ \frac{\partial \phi_j}{\partial y} \right\}\ d\Omega_e;$$

$$\frac{\partial R^i_{my}}{\partial V_j} = \int_{\Omega_e} \left\{ \rho\ \psi_i \left[u \frac{\partial \phi_j}{\partial x} + \phi_j \frac{\partial v}{\partial y} + v \frac{\partial \phi_j}{\partial y} \right] + \right.$$
$$\left. + \mu\ \frac{\partial \psi_i}{\partial x}\ \frac{\partial \phi_j}{\partial x} + \ 2\ \mu\ \frac{\partial \psi_i}{\partial y}\ \frac{\partial \phi_j}{\partial y} \right\}\ d\Omega_e;$$

$$\frac{\partial R^i_{my}}{\partial P_j} = \int_{\Omega_e} -\ \frac{\partial \psi_i}{\partial y}\ \chi_i\ d\Omega_e.$$

Jacobiano de R^i_c:

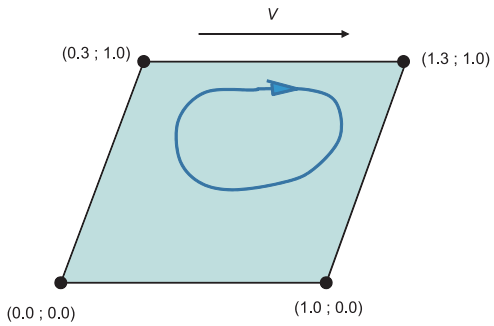

Figura 6.8: Cavidade com tampa móvel.

$$\frac{\partial R_c^i}{\partial U_j} = \int_{\Omega_e} \frac{\partial \phi_j}{\partial x} \chi_i \; d\Omega_e;$$

$$\frac{\partial R_c^i}{\partial V_j} = \int_{\Omega_e} \frac{\partial \phi_j}{\partial y} \chi_i \; d\Omega_e;$$

$$\frac{\partial R_c^i}{\partial P_j} = 0.$$

6.4 Exemplo de Implementação do Código

O problema do escoamento em uma cavidade inclinada com tampa móvel, figura (6.8), será usado como exemplo para apresentação do código. Todos as fronteiras do domínio são paredes onde a velocidade é prescrita.

Nesta seção discutimos alguns detalhes da estrutura de dados usada e da implementação do código em ambiente MatLab. Os códigos apresentados não foram escritos buscando uma maior eficiência computacional, mas sim a clareza dos conceitos apresentados e discutidos ao longo desse trabalho. Desta forma, várias operações poderiam ser

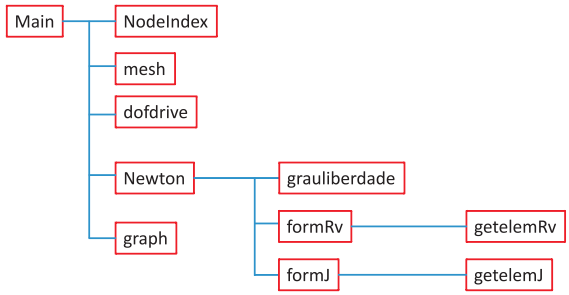

Figura 6.9: Estrutura do codigo apresentado.

realizadas de forma muito mais eficiente utilizando recursos computacionais específicos do MatLab.

O código apresentado como exemplo segue a estrutura apresentada na figura 6.9. O *script* `Main`, apresentado a seguir, é divido em 3 partes:

1. Pré-processamento: Consiste na definição dos parâmetros do problema (propriedades, geometria, dados do escoamento e malha), determinação da relação entre as numerações elementar e global dos nós (função `NodeIndex`) e graus de liberdade (função `dofdrive`) e cálculo das coordenadas nodais (função `mesh`).

2. Solução: Solução do problema não linear pelo método de Newton (função `Newton`).

3. Pós-processamento: Construção do gráfico vetorial com o campo de velocidade (função `graph`)

`Main`

```
%=========================================================================
% Codigo Baseado no MEF para solucao de escoamentos incompressiveis
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
clear; clc;
%=========================================================================
% PREPROCESSAMENTO
% Entrada de Dados
%   Propriedades
```

```
ro = 10; mi = 1;
%   Condicoes do escoamento Vel = 1;
%   Geometria
L = 1; H = 1;
%   Malha
NEX = 10; NEY = 10;
% Calculo do numero de elementos e nos
NELE = NEX * NEY; NODES = (2*NEX+1) * (2*NEY+1);
% Relacao da numeracao local e global dos nos
[DomNodeID] = NodeIndex(NELE,NEX,NEY);
% Calculo das coordenadas dos pontos nodais
[X,Y] = mesh(L, H, NEX, NEY, DomNodeID);
% Relacao da numeracao local e global dos graus de liberdade
[NDoF, ASSMtrx] = dofdrive(NELE, NODES, DomNodeID);
%=========================================================================
% RESOLUCAO DO PROBLEMA
% Solucao pelo metodo de Newton
[C] = Newton(NELE, NDoF, NEX, NEY, ASSMtrx, DomNodeID, X, Y, ro, mi, Vel);
%=========================================================================
% POSPRECESSAMENTO
graph(NELE, DomNodeID, ASSMtrx, X, Y, C);
%=========================================================================
```

As funções relacionadas com o pré-processamento são apresentadas a seguir. Considerou-se uma malha regular. Os elementos foram numerados de baixo para cima, da esquerda para direita. Os nós e graus de liberdade foram numerados seguindo a sequência dos elementos.
`NodeIndex`

```
%=========================================================================
% Calculo da matriz que relaciona a numeracao elementar e global
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [DomNodeID] = DomNodeID(NELE,NEX,NEY);
DomNodeID = zeros(9,NELE);
NodeCount = 1;
for iele = 1:NELE
    iWestEle = 0;
    iSouthEle = 0;
    % find West element neighbour
    if (iele > NEY)
        iWestEle = iele - NEY;
    end
    % find South element neighbour
    iSouth = mod(iele-1,NEY);
    if (iSouth == 0)
        iSouthEle = 0;
```

```
    else
        iSouthEle = iele-1;
    end
    % Set DomNodeID for nodes 1,4,8 if element has West Neighbour
    if (iWestEle  = 0)
        DomNodeID(1,iele) = DomNodeID(2,iWestEle);
        DomNodeID(4,iele) = DomNodeID(3,iWestEle);
        DomNodeID(8,iele) = DomNodeID(6,iWestEle);
    end
    % Set DomNodeID for nodes 1,5,2 if element has South Neighbour
    if (iSouthEle  = 0)
        DomNodeID(1,iele) = DomNodeID(4,iSouthEle);
        DomNodeID(2,iele) = DomNodeID(3,iSouthEle);
        DomNodeID(5,iele) = DomNodeID(7,iSouthEle);
    end
    for ilnode = 1:9
        if (DomNodeID(ilnode,iele) == 0)
        DomNodeID(ilnode,iele) = NodeCount;
        NodeCount = NodeCount + 1;
        end
    end
end
%=========================================================================
```

mesh

```
%=========================================================================
% Calculo das coordenadas nodais
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function[X,Y] = mesh(L, H, NEX, NEY, DomNodeID);
tanALFA=1/0.3; DX=L/NEX; DY=H/NEY; for ix=1:NEX
    for jy=1:NEY
        iele=(ix-1)*NEY+jy;
        xloc(1)=(ix-1)*DX+(jy-1)*DY/(tanALFA);
        xloc(2)=xloc(1)+DX;
        xloc(3)=xloc(2)+DY/(tanALFA);
        xloc(4)=xloc(1)+DY/(tanALFA);
        xloc(5)=(xloc(1)+xloc(2))/2;
        xloc(6)=(xloc(2)+xloc(3))/2;
        xloc(7)=(xloc(3)+xloc(4))/2;
        xloc(8)=(xloc(1)+xloc(4))/2;
        xloc(9)=(xloc(6)+xloc(8))/2;
        yloc(1)=(jy-1)*DY;
        yloc(2)=yloc(1);
        yloc(3)=yloc(2)+DY;
        yloc(4)=yloc(3);
        yloc(5)=yloc(1);
```

```
        yloc(6)=yloc(2)+DY/2;
        yloc(7)=yloc(3);
        yloc(8)=yloc(6);
        yloc(9)=yloc(6);
        for ilnode=1:9
            ignode=DomNodeID(ilnode,iele);
            X(ignode)=xloc(ilnode);
            Y(ignode)=yloc(ilnode);
        end
    end
end
%=========================================================================
```

dofdrive

```
%=========================================================================
% Calculo da matriz de graus de liberdade
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [NDoF, ASSMtrx] = dofdrive(NELE, NODES, DomNodeID);
ASSMtrx = zeros(21,NELE); GlobalDoFID = zeros(5,NODES);
% Definition of the array GlobalDoFID(ldof,ignode)
igdofcount = 1;
for iele = 1:NELE
    nadd = 0;
    icount = igdofcount;
    ildofcount = 0;
    for ilnode = 1:9
        ignode = DomNodeID(ilnode,iele);
        if ( GlobalDoFID(1,ignode) == 0 )
            nadd = nadd + 1;
        end
    end
    for ilnode = 1:9
        ignode = DomNodeID(ilnode,iele);
        if ( GlobalDoFID(1,ignode) == 0 )
            GlobalDoFID(1,ignode) = icount;
            GlobalDoFID(2,ignode) = icount+nadd;
            icount = icount + 1;
            ildofcount = ildofcount + 2;
        end
    end
    ignode = DomNodeID(9,iele);
    GlobalDoFID(3,ignode) = GlobalDoFID(2,ignode) + 1;
    GlobalDoFID(4,ignode) = GlobalDoFID(3,ignode) + 1;
    GlobalDoFID(5,ignode) = GlobalDoFID(4,ignode) + 1;
    ildofcount = ildofcount + 3;
    igdofcount = igdofcount + ildofcount;
```

```
end
NDoF = igdofcount - 1;
% Definition of Assembly Matrix
for iele = 1:NELE
    ildof = 0;
    for ilnode = 1:9
        ignode = DomNodeID(ilnode,iele);
        ildof = ildof + 1;
        ASSMtrx(ildof,iele) = GlobalDoFID(1,ignode);
    end
    for ilnode = 1:9
        ignode = DomNodeID(ilnode,iele);
        ildof = ildof + 1;
        ASSMtrx(ildof,iele) = GlobalDoFID(2,ignode);
    end
    ignode = DomNodeID(9,iele);
    ildof = ildof + 1;
    ASSMtrx(ildof,iele) = GlobalDoFID(3,ignode);
    ildof = ildof + 1;
    ASSMtrx(ildof,iele) = GlobalDoFID(4,ignode);
    ildof = ildof + 1;
    ASSMtrx(ildof,iele) = GlobalDoFID(5,ignode);
end
%=========================================================================
```

A função `Newton` realiza o processo iterativo para o cálculo da solução do problema baseado no método de Newton. O vetor solução **C** contém os coeficientes das expansões em funções bases dos campos de velocidade e pressão. O método requer o cálculo do vetor global de resíduos ponderados (função `formRv`) e matriz Jacobiana global (função `formJ`). O sistema linear a cada passo de Newton foi resolvido por decomposição LU, como indicado na função, apresentada a seguir.
`Newton`

```
%=========================================================================
% Metodo de Newton para solucao de sistema nao linear
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [C] = Newton(NELE, NDoF, NEX, NEY, ASSMtrx, DomNodeID, X, ...
        Y, ro, mi, Vel)
E = 0.0001; itermax = 10;
[iuldof,ivldof,ipldof] = grauliberdade;
C = zeros(NDoF,1);
DC = zeros(NDoF,1);
% Calculo do vetor residuo global
```

```
[Rv] = formRv(NELE, NDoF, NEX, NEY, DomNodeID, ASSMtrx, ...,
        iuldof,ivldof,ipldof, X, Y, ro, mi, Vel, C);
rvnorm = norm(Rv);
iter = 0;
iter rvnorm
itervec(1) = 1; rvnormvec(1) = rvnorm;
while ( (rvnorm > E) & (iter < itermax) )
    iter = iter + 1;
    % Calculo da matriz jacobiana global
    [J] = formJ(NELE, NDoF, NEX, NEY, DomNodeID, ASSMtrx, ...
        iuldof,ivldof,ipldof, X, Y, ro, mi, Vel, C);
    b=-Rv;
    [LSP,USP]=lu(J);
    YLU = LSP \ b;
    DC = USP \ YLU;
    C = DC + C;
    % Calculo do vetor residuo global
    [Rv] = formRv(NELE, NDoF, NEX, NEY, DomNodeID, ASSMtrx, ...
        iuldof,ivldof,ipldof, X, Y, ro, mi, Vel, C);
    rvnorm = norm(Rv);
    iiter
    rvnorm
    itervec(iter+1) = iter+1;
    rvnormvec(iter+1) = rvnorm;
end
if (norm(Rv) > E)
    display('Did not converge ');
    stop;
end
%=========================================================================
```

A função `formRv` realiza a montagem do vetor resíduo ponderado global a partir dos vetores resíduos elementares (variável `elemRv`). O cálculo dos vetores elementares é feito na função `getelemRv` e o processo de montagem utiliza a matriz `ASSMtrx` que relaciona a numeração elementar e global dos graus de liberdade.

`formRv`

```
%=========================================================================
% Calculo do vetor residuo global
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [Rv] = formRv(NELE, NDoF, NEX, NEY, DomNodeID, ASSMtrx, ...
        iuldof,ivldof,ipldof, X, Y, ro, mi, Vel, C);
NdoFele = 21;
Rv = zeros(NDoF,1);
```

```
for iele= 1:NELE
    [elemRv] = getelemRv(iele, NEX, NEY, DomNodeID, ASSMtrx, ...
            iuldof, ivldof, ipldof, X, Y, ro, mi, Vel, C);
    for ildof = 1:NdoFele
        igdof = ASSMtrx(ildof,iele);
        Rv(igdof) = Rv(igdof) + elemRv(ildof);
    end
end
%=========================================================================
```

O cálculo dos vetores e matrizes elementares representam a parte central do código.

Os valores das coordenadas dos nós do elemento e dos valores dos graus de liberdade de velocidade (valores nodais) e pressão são obtidos a partir dos vetores com as coordenadas nodais global e do vetor solução **C**. Estes valores são armazenados nas seguintes variáveis:

$$\begin{matrix}\mathrm{XY}(i,j)\\ i=1,2.\\ j=1,\ldots,9.\end{matrix} \qquad \mathrm{XY} = \begin{bmatrix} X_1 & \ldots & X_9 \\ Y_1 & \ldots & Y_9 \end{bmatrix},$$

$$\begin{matrix}\mathrm{velocity}(i,j)\\ i=1,2.\\ j=1,\ldots,9.\end{matrix} \qquad \mathrm{velocity} = \begin{bmatrix} U_1 & \ldots & U_9 \\ V_1 & \ldots & V_9 \end{bmatrix},$$

$$\begin{matrix}\mathrm{pressure}(i)\\ i=1,2,3.\end{matrix} \qquad \mathrm{pressure} = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}.$$

As integrais são calculadas pelo método da quadratura Gaussiana. Desta forma, os *loops* mais externos incluem o cálculo do valor das velocidade u e v, de suas derivadas em relação a x e y e da pressão p nos pontos de Gauss. Para tal, é necessário calcular o valor das funções base e duas derivadas no ponto de Gauss (`XI,NETA`). A função `basifunc` retorna um vetor contendo o valor das 9 funções bases de velocidade `Phi`, uma matriz contendo as suas derivadas em relação às coordenadas elementares `GradPhi` e um vetor contendo o valor das 3 funções base de pressão `Psi`, no ponto de Gauss. A estrutura destes vetores e matriz é mostrada a seguir:

$$\begin{matrix}\text{PHI}(i)\\ i=1,\ldots,9.\end{matrix} \qquad \text{PHI} = \begin{bmatrix}\phi_1\\ \phi_2\\ \vdots\\ \phi_9\end{bmatrix},$$

$$\begin{matrix}\text{GradPHI}(i,j)\\ i=1,2.\\ j=1,\ldots,9.\end{matrix} \qquad \text{GradPHI} = \begin{bmatrix}\dfrac{\partial\phi_1}{\partial\xi} & \cdots & \dfrac{\partial\phi_9}{\partial\xi}\\ \\ \dfrac{\partial\phi_1}{\partial\eta} & \cdots & \dfrac{\partial\phi_9}{\partial\eta}\end{bmatrix},$$

$$\begin{matrix}\text{PSI}(i)\\ i=1,2,3\end{matrix} \qquad \text{PSI} = \begin{bmatrix}\chi_1\\ \chi_2\\ \chi_3\end{bmatrix}.$$

As velocidades u e v e pressão p no ponto de Gauss são calculadas através da expansão destes campos usando as respectivas funções bases:

$$u(\xi_{igp},\eta_{igp}) = \sum_{i=1}^{9} U_i\phi_i(\xi_{igp},\eta_{igp}), \qquad v(\xi_{igp},\eta_{igp}) = \sum_{i=1}^{9} V_i\phi_i(\xi_{igp},\eta_{igp}),$$

$$p(\xi_{igp},\eta_{igp}) = \sum_{i=1}^{3} P_i\chi_i(\xi_{igp},\eta_{igp}).$$

Na função apresentada abaixo, estes somatórios são convenientemente calculados através de uma multiplicação de matrizes (vetores):

$$\text{UV} = \begin{bmatrix}u\\ v\end{bmatrix} = \begin{bmatrix}U_1 & \ldots & U_9\\ V_1 & \ldots & V_9\end{bmatrix} \cdot \begin{bmatrix}\phi_1\\ \vdots\\ \phi_9\end{bmatrix},$$

$$\text{Pre} = \begin{bmatrix}\text{ Pre }\end{bmatrix} = \begin{bmatrix}P_1 & P_2 & P_3\end{bmatrix} \cdot \begin{bmatrix}\chi_1\\ \chi_2\\ \chi_3\end{bmatrix}.$$

O cálculo dos termos relacionados ao gradiente de velocidade também é feito através de multiplicação de matrizes:

$$\mathrm{dUV} = \begin{bmatrix} \dfrac{\partial u}{\partial x} & \dfrac{\partial v}{\partial x} \\ \dfrac{\partial u}{\partial y} & \dfrac{\partial v}{\partial y} \end{bmatrix} = \begin{bmatrix} U_1 & \cdots & U_9 \\ V_1 & \cdots & V_9 \end{bmatrix} \cdot \begin{bmatrix} \dfrac{\partial \phi_1}{\partial x} & \dfrac{\partial \phi_1}{\partial y} \\ \vdots & \vdots \\ \dfrac{\partial \phi_9}{\partial x} & \dfrac{\partial \phi_9}{\partial y} \end{bmatrix}.$$

As derivadas das funções bases em relação às coordenadas globais x e y são calculadas, como discutido no capítulo anterior, usando o Jacobiano da transformação do mapeamento isoparamétrico (variável `JAC` na função apresentada a seguir). Estes cálculos também são realizados através de multiplicação de matrizes:

$$\mathrm{JAC} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \phi_1}{\partial \xi} & \cdots & \dfrac{\partial \phi_9}{\partial \xi} \\ \dfrac{\partial \phi_1}{\partial \eta} & \cdots & \dfrac{\partial \phi_9}{\partial \eta} \end{bmatrix} \cdot \begin{bmatrix} X_1 & Y_1 \\ \vdots & \vdots \\ X_9 & Y_9 \end{bmatrix},$$

$$\mathrm{GradPhixy} = \begin{bmatrix} \dfrac{\partial \phi_1}{\partial x} & \cdots & \dfrac{\partial \phi_9}{\partial x} \\ \dfrac{\partial \phi_1}{\partial y} & \cdots & \dfrac{\partial \phi_9}{\partial y} \end{bmatrix} = \mathbf{J^{-1}}. \begin{bmatrix} \dfrac{\partial \phi_1}{\partial \xi} & \cdots & \dfrac{\partial \phi_9}{\partial \xi} \\ \dfrac{\partial \phi_1}{\partial \eta} & \cdots & \dfrac{\partial \phi_9}{\partial \eta} \end{bmatrix}$$

Após o cálculo do resíduo elementar, a função teste se o elemento sendo calculado está na fronteira do domínio. Neste problema simples, com a utilização de uma malha regular, é muito simples saber se um elemento está na fronteira ou não. A numeração dos elementos adotada neste exemplo seguiu uma ordem da esquerda para direita, e de baixo para cima. Desta forma, o elemento 1 está localizado no canto inferior do domínio. Uma simples operação com o número do

elemento permite verificar se ele pertence a uma determinada fronteira ou não:

Elemento $\in$ fronteira Oeste $\Longleftrightarrow iele \leq NEY$.

Elemento $\in$ fronteira Leste $\Longleftrightarrow NELE - NEY < iele \leq NELE$.

Elemento $\in$ fronteira Norte $\Longleftrightarrow mod(iele, NEY) = 0$ (mod $\equiv$ resto da divisão por NEY).

Elemento $\in$ fronteira Sul $\Longleftrightarrow mod(iele, NEY) = 1$.

Os resíduos ponderados associados aos nós ao longo da fronteira são descartados e o valor da velocidade é imposta de forma essencial. A equação algébrica associada ao grau de liberdade passa a ser:

$$R^i_{mx} = U_i - U_{\text{parede}} \quad \text{e} \quad R^i_{my} = V_i - V_{\text{parede}}.$$

O último aspecto a ser apresentado da função `getelemRv` é o estabelecimento do nível de pressão do escoamento. Observe que o campo de pressão que satisfaz a equação de Navier-Stokes está determinado a menos de uma constante. Desta forma, um nível de pressão do escoamento deve ser especificado através de uma condição de contorno ou arbitrariamente. Neste problema, todos os contornos são paredes, onde o valor da velocidade é imposto de forma essencial. Desta forma, devemos estabelecer um nível de pressão arbitrário, no caso a pressão do elemento 1 foi fixada igual a 0. Como na especificação de uma condição de contorno essencial, o resíduo ponderado associado ao primeiro grau de liberdade de pressão do elemento 1 é descartado e substituído por:

$$R^1_c = P_1 - 0.$$

`getelemRv`

```
%=====================================================================
% Calculo do vetor residuo elementar
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=====================================================================
function [elemRv] = getelemRv(iele, NEX, NEY, DomNodeID, ASSMtrx, ...
       iuldof, ivldof, ipldof, X, Y, ro, mi, Vel, C);
% inicializacao
elemRV = zeros(21,1);
% Número de pontos de Gauss
NGP=3;
% Peso para cada ponto de Gauss
```

```
WGP=[0.555,0.888,0.555];
% Posição dos números de Gauss
XGP = [ -0.7745966, 0.0, 0.7745966];
% Matriz auxiliar para localizar nos nas fronteiras
iaux = [1 2 4 1;5 6 7 8;2 3 3 4];
% Valor elementar das coordenadas dos nos, velocidade e pressao
for ilnode = 1:9
    ignode = DomNodeID(ilnode,iele);
    XY(1,ilnode) = X(ignode);
    XY(2,ilnode) = Y(ignode);
end
for ilnode=1:9
    velocity(1,ilnode) = C(ASSMtrx(ilnode,iele));
    velocity(2,ilnode) = C(ASSMtrx(ilnode+9,iele));
end
for ipress = 1:3
    pressure(ipress) = C(ASSMtrx(18+ipress,iele));
end
% Loop nos pontos de Gauss
for igp = 1:NGP
    XI = XGP(igp);
    WI = WGP(igp);
    for jgp = 1:NGP
        NETA = XGP(jgp);
        WJ =WGP(jgp);
        W = WI*WJ;
        [Phi,GradPhi,Psi]= basifunc (XI,NETA);
        % calculando a jacobiana da mudança de variáveis:
        JAC = GradPhi * XY';
        % JACinv = inversa da Jacobiana.
        JACinv = inv(JAC);
        % JACdet = determinante da Jacobiana.
        JACdet = det(JAC);
        % calculando o GradPhixy
        GradPhixy = JACinv*GradPhi;
        % Calcular u,v (velocides) e pre(pressão)
        UV = velocity* Phi;
        Pre = pressure* Psi;
        % Derivada da velo em relação a xy
        dUV = velocity * GradPhixy';
        for ilnode = 1:9
            iu = iuldof(ilnode);
            iv = ivldof(ilnode);
            elemRv(iu) = elemRv(iu) + W*JACdet *(ro* Phi(ilnode)*...
            (UV(1)*dUV(1,1) + UV(2)*dUV(1,2))+ GradPhixy(1,ilnode)*...
            ( 2*mi*dUV(1,1) - Pre ) + GradPhixy(2,ilnode)*(dUV(1,2) + dUV(2,1)));
            elemRv(iv) = elemRv(iv) + W*JACdet *(ro* Phi(ilnode)*...
            (UV(1)*dUV(2,1) + UV(2)*dUV(2,2))+ GradPhixy(2,ilnode)*...
            ( 2*mi*dUV(2,2) - Pre ) + GradPhixy(1,ilnode)*(dUV(1,2) + dUV(2,1)));
        end

        for ipress=1:3
```

```
            ip = ipldof(ipress);
            elemRv(ip) = elemRv(ip)+ W* JACdet*(dUV(1,1)+dUV(2,2))*Psi(ipress);
        end
    end
end
% Condicoes de contorno
if (mod(iele,NEY) == 0)
    for isidenode = 1:3
        ilnode = iaux(isidenode,3);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        elemRv(iu) = ( velocity(1,ilnode) - Vel);
        elemRv(iv) = ( velocity(2,ilnode) - 0.0);
    end
end
if (iele <= NEY)
    for isidenode = 1:3
        ilnode = iaux(isidenode,4);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        elemRv(iu) = ( velocity(1,ilnode) - 0.0);
        elemRv(iv) = ( velocity(2,ilnode) - 0.0);
    end
end
if (mod(iele,NEY) == 1)
    for isidenode = 1:3
        ilnode = iaux(isidenode,1);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        elemRv(iu) = ( velocity(1,ilnode) - 0.0);
        elemRv(iv) = ( velocity(2,ilnode) - 0.0);
    end
end
if (iele > (NEX-1)*NEY)
    for isidenode = 1:3
        ilnode = iaux(isidenode,2);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        elemRv(iu) = ( velocity(1,ilnode) - 0.0);
        elemRv(iv) = ( velocity(2,ilnode) - 0.0);
    end
end
% Fixando o nivel de pressao
if (iele == 1)
    elemRv(19) = (pressure(1) - 0.0);
end
%=========================================================================
```

O processo de montagem da matriz Jacobiana global e o cálculo das matrizes elementares é semelhante a montagem e cálculo dos vetores resíduos ponderados. As funções `FormJ` e `getelemJ` realizam estas etapas.

`FormJ`

```
%=========================================================================
% Calculo da matriz jacobiana global
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [J] = formJ(NELE, NDoF, NEX, NEY, DomNodeID, ASSMtrx, ...
    iuldof,ivldof,ipldof, X, Y, ro, mi, Vel, C);
NdoFele = 21;
NDIM = NdoFele * NELE;
Jval = zeros(NDIM,1); irow = zeros(NDIM,1); jcol = zeros(NDIM,1);
icont = 1;
for iele= 1:NELE
    [elemJ] = getelemJ(iele, NEX, NEY, DomNodeID, ASSMtrx, ...
        iuldof, ivldof, ipldof, X, Y, ro, mi, Vel, C);
    for ildof = 1:NdoFele
        igdof = ASSMtrx(ildof,iele);
        for jldof = 1:NdoFele
            jgdof = ASSMtrx(jldof,iele);
            irow(icont) = igdof;
            jcol(icont) = jgdof;
            Jval(icont) = elemJ(ildof,jldof);
            icont = icont+1;
        end
    end
end
J = sparse(irow,jcol,Jval,NDoF,NDoF);
spy(J)
%=========================================================================
```

`getelemJ`

```
%=========================================================================
% Calculo da matriz jacobiana elementar
% Data:  Marco de 2012
% Autores:  Marcio Carvalho e Juliana Valerio
%=========================================================================
function [elemJ] = getelemJ(iele, NEX, NEY, DomNodeID, ASSMtrx, ...
        iuldof, ivldof, ipldof, X, Y, ro, mi, Vel, C);
% Função que cria a matriz elementar
elemJ = zeros(21,21);
% Número de pontos de Gauss
NGP=3;
```

```
% Peso para cada ponto de Gauss
WGP=[0.555,0.888,0.555];
% Posição dos números de Gauss
XGP = [ -0.7745966, 0.0, 0.7745966];
% Matriz auxiliar para localizar nos nos lados
iaux = [1 2 4 1;5 6 7 8;2 3 3 4];
% Valor elementar das coordenadas dos nos, velocidade e pressao
for ilnode = 1:9
    ignode = DomNodeID(ilnode,iele);
    XY(1,ilnode) = X(ignode);
    XY(2,ilnode) = Y(ignode);
end
for ilnode=1:9
    velocity(1,ilnode) = C(ASSMtrx(ilnode,iele));
    velocity(2,ilnode) = C(ASSMtrx(ilnode+9,iele));
end
for ipress = 1:3
    pressure(ipress) = C(ASSMtrx(18+ipress,iele));
end
% Loop nos pontos de Gauss
for igp = 1:NGP
    XI = XGP(igp);
    WI = WGP(igp);
    for jgp = 1:NGP
        NETA = XGP(jgp);
        WJ =WGP(jgp);
        W = WI*WJ;
        [Phi,GradPhi,Psi]= basifunc (XI,NETA);
        % calculando a jacobiana da mudança de variáveis:
        JAC = GradPhi * XY';
        % JACinv = inversa da Jacobiana.
        JACinv = inv(JAC);
        % JACdet = determinante da Jacobiana.
        JACdet = det(JAC);
        % calculando o GradPhi-xy
        GradPhixy = JACinv*GradPhi;
        % Calcular u,v (velocides) e pre(pressão)
        UV = velocity* Phi;
        Pre = pressure* Psi;
        % Derivada da velo em relação a xy
        dUV = velocity * GradPhixy';
        for ilnode = 1:9
            iu = iuldof(ilnode);
            iv = ivldof(ilnode);
            for jlnode =1:9
                ju = iuldof(jlnode);
                jv = ivldof(jlnode);
                elemJ(iu,ju) = elemJ(iu,ju) + W*JACdet *(ro* Phi(ilnode)*...
                (Phi(jlnode)*dUV(1,1) + UV(1)*GradPhixy(1,jlnode)+ ...
                UV(2)*GradPhixy(2,jlnode))+...
                ( 2*mi*GradPhixy(1,ilnode)*GradPhixy(1,jlnode)+ ...
```

```
                GradPhixy(2,ilnode)*GradPhixy(2,jlnode)));

                elemJ(iu,jv) = elemJ(iu,jv) + W*JACdet *(ro* Phi(ilnode)*...
                Phi(jlnode)*dUV(1,2)+ GradPhixy(2,ilnode)*GradPhixy(1,jlnode));

                elemJ(iv,ju) = elemJ(iv,ju) + W*JACdet *(ro* Phi(ilnode)*...
                Phi(jlnode)*dUV(2,1) + GradPhixy(1,ilnode)*GradPhixy(2,jlnode));

                elemJ(iv,jv) = elemJ(iv,jv) + W*JACdet *(ro* Phi(ilnode)*...
                (Phi(jlnode)*dUV(2,2) + UV(1)*GradPhixy(1,jlnode)+ ...
                UV(2)*GradPhixy(2,jlnode)) + ( 2*mi*GradPhixy(2,ilnode)*...
                GradPhixy(2,jlnode)+ GradPhixy(1,ilnode)*GradPhixy(1,jlnode)));
            end
            for jpress = 1:3
                jp = ipldof(jpress);
                elemJ(iu,jp) = elemJ(iu,jp) + W*JACdet * ...
                (-GradPhixy(1,ilnode)*Psi(jpress) );

                elemJ(iv,jp) = elemJ(iv,jp) + W*JACdet * ...
                (-GradPhixy(2,ilnode)*Psi(jpress) );
            end
        end
        for ipress=1:3
            ip = ipldof(ipress);
            for jlnode = 1:9
                ju = iuldof(jlnode);
                jv = ivldof(jlnode);
                elemJ(ip,ju) = elemJ(ip,ju)+ W* JACdet*...
                (GradPhixy(1,jlnode)*Psi(ipress));

                elemJ(ip,jv) = elemJ(ip,jv)+ W* JACdet*...
                (GradPhixy(2,jlnode)*Psi(ipress));
            end
        end
    end
end
% Condicoes de contorno
if (mod(iele,NEY) == 0)
    for isidenode = 1:3
        ilnode = iaux(isidenode,3);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        for j = 1:21
            elemJ(iu,j) = 0.;
            elemJ(iv,j) = 0.;
        end
        elemJ(iu,iu) = 1.;
        elemJ(iv,iv) = 1.;
    end
end
if (iele <= NEY)
    for isidenode = 1:3
```

```
        ilnode = iaux(isidenode,4);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        for j = 1:21
            elemJ(iu,j) = 0.;
            elemJ(iv,j) = 0.;
        end
        elemJ(iu,iu) = 1.;
        elemJ(iv,iv) = 1.;
    end
end
if (mod(iele,NEY) == 1)
    for isidenode = 1:3
        ilnode = iaux(isidenode,1);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        for j = 1:21
            elemJ(iu,j) = 0.;
            elemJ(iv,j) = 0.;
        end
        elemJ(iu,iu) = 1.;
        elemJ(iv,iv) = 1.;
    end
end
if (iele > (NEX-1)*NEY)
    for isidenode = 1:3
        ilnode = iaux(isidenode,2);
        iu = iuldof(ilnode);
        iv = ivldof(ilnode);
        for j = 1:21
            elemJ(iu,j) = 0.;
            elemJ(iv,j) = 0.;
        end
        elemJ(iu,iu) = 1.;
        elemJ(iv,iv) = 1.;
    end
end
% Fixando o nivel de pressao
if (iele == 1)
    ip = ipldof(1);
    for j = 1:21
        elemJ(ip,j) = 0.;
    end
    elemJ(ip,ip) = 1.;
end
%=========================================================================
```

O resultado obtido para uma malha 10×10 é apresentado na figura 6.10. O chute inicial para o método de Newton foi um campo de

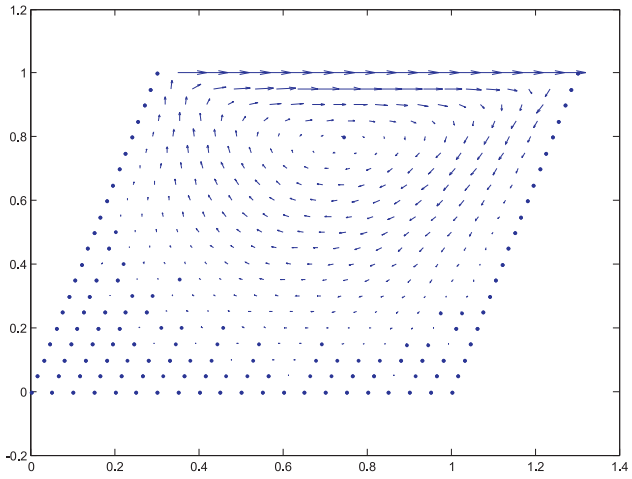

Figura 6.10: Campo de velocidade.

velocidade e pressão identicamente nulo. O problema convergiu quadraticamente para a solução em 3 iterações com a norma do resíduo menor do que 10^{-9}.

6.5 Exercícios

1. Modifique o programa apresentado neste capítulo para resolver o escoamento através de uma contração, figura 6.1. Utilize as condições de contorno apresentadas no início do capítulo para este problema. Adaptar para a solução deste problema: `Mesh`, `getelemRv` e `getelemJ`.

2. O campo de velocidade $\mathbf{v}$, com $\nabla \cdot \mathbf{v} = 0$, admite uma função de corrente Ψ satisfazendo $\frac{\partial \Psi}{\partial y} = u$ e $\frac{\partial \Psi}{\partial x} = -v$. Nesse caso, a vorticidade é definida como $\mathbf{w} = \nabla \times \mathbf{u}$.
   a) A partir de $\mathbf{u} \cdot \nabla \mathbf{u} = -\left(\frac{1}{\rho}\right) \nabla p + \left(\frac{\mu}{\rho}\right) \nabla^2 \mathbf{u}$, usando as condições acima e a definição de vorticidade, deduza $\mathbf{u} \cdot \nabla \mathbf{w} = \left(\frac{\mu}{\rho}\right) \nabla^2 \mathbf{w}$.

b) Modifique o código apresentado nesse capítulo para resolver um escoamento utilizando a vorticidade no lugar da velocidade.

Bibliografia

[1] R. Aris " Vectors, Tensors and the Basic Equations of Fluid Mechanics", Courier Dover Publications, 1990.

[2] E. B. Becker, Graham F. Carey e J. Tinsley Oden. " Finite Elements: An Introduction", Prentice Hall, vol 1, 1981.

[3] B. A. Finlayson e L. E. Scriven, "The Method of Weighted Residuals - A Review", *Applied Mechanics Reviews*,**19(9)** (1966), 735-748.

[4] R. Glowinski e O. Pironneau, "Finite Element Methods for Navier-Stokes Equations", *Annual Review of Fluid Mechanics*,**24** (1992), 167-204.

[5] P. M. Gresho e R. L. Sani, "Incompressible Flow and the Finite Element Method", John Wiley and Sons, 1998.

[6] P. M. Gresho, "Some Current CFD Issues Relevant to the Incompressible Navier-Stokes Equations", *Computer Methods in Applied Mechanics and Engineering*,**87** (1991), 201-252.

[7] T. J. R. Hughes, "The Finite Element Method: Linear Static and Dynamic Finite Element Analysis", Dover Publications, 2000.

[8] C. Johnson, "Numerical Solution of Partial Differential Equations by the Finite Element Method", Cambridge University Press, 1987.

[9] P. Knabner e L. Angermann, "Numrical Method for Elliptic and Parabolic Partial Differential Equations", Springer, 2000.

[10] I-Shih Lui e M. A. Rincon, "Introdução ao Método de Elementos Finitos", 3. Ed. UFRJ/IM, 2011.

[11] B. D. Reddy, "Introductory Functional Analysis : With Applications to Boundary Value Problems and Finite Elements", Springer, 1998.

[12] G. Strang e G. Fix, "An Analysis of The Finite Element Method", Prentice Hall, 1973.

[13] V. Thomee, "From finite differences to finite elements: A short history of numerical analysis of partial differential equations", *Journal of Computational and Applied Mathematics*,**128(1-2)** (2001), 1-54.

Índice