Universidade Federal de Santa Catarina
Pós-Graduação em Engenharia Elétrica

Dissertação submetida à Universidade Federal de Santa Catarina,
Pós-Graduação em Engenharia Elétrica, para preenchimento dos requisitos
parciais para obtenção do Título de Mestre em Engenharia Elétrica.

IMPLEMENTAÇÃO DE UMA BIBLIOTECA INFORMÁTICA PARA DIVERSOS TIPOS DE ELEMENTOS FINITOS EM 2D E 3D

Autor: **Emerson Silveira Serafim**

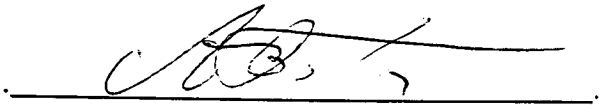
Orientador: **João Pedro Assumpção Bastos**

Florianópolis – Santa Catarina
Março de 1998

Implementação de uma Biblioteca Informática Diversos Tipos de Elementos Finitos em Duas e Três Dimensões.

Emerson Silveira Serafim

Esta dissertação foi julgada adequada para obtenção do título
de Mestre em Engenharia, especialidade Engenharia Elétrica
e aprovada em sua forma final pelo curso de Pós-Graduação

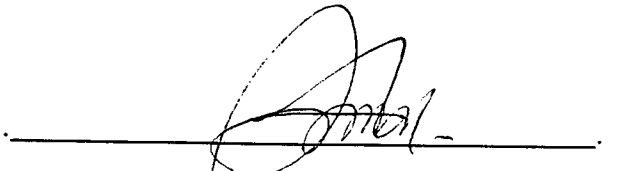


Prof. João Pedro Assumpção Bastos, Dr.
ORIENTADOR

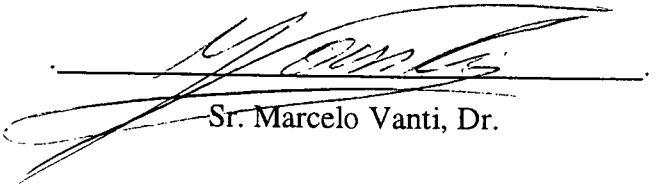
Prof. Adroaldo Raizer, Dr.
Coordenador do Curso de Pós-Graduação
em Engenharia Elétrica

BANCA EXAMINADORA:

Prof. João Pedro Assumpção Bastos, Dr.



Prof. Patrick Kuo-Peng,Dr.



Sr. Marcelo Vanti, Dr.

RESUMO

Neste trabalho, fez-se um estudo e implementação de uma biblioteca informática de funções de forma para diversos tipos de elementos finitos, em duas e três dimensões.

Inicialmente, estudou-se as formulações relativas a aplicação do Método dos Resíduos Ponderados, formulações matriciais e os procedimentos númericos necessários para resolução dos mesmos.

Na sequência, são apresentados os vários tipos de elementos finitos com suas respectivas funções de forma e características.

Finalmente, é descrita a implementação da biblioteca informática.

ABSTRACT

This work is related to the implementation of a numerical library of shape functions for different types of finite elements, in two and three dimensions.

In the firsty part of this work we present the Weighted Residual Method applications, the corresponding matrix formulation and the procedures for the numerical resolution. Then, different types of finite elements with its respective shape functions and general characteristics are presented.

Finally the implementation of the numerical libary is described.

AGRADECIMENTOS

Inicialmente, gostaria de dedicar este trabalho aos meus pais Itamar José Serafim e Zuleide Silveira Serafim, sem os quais jamais teria chegado aonde estou, e também a meus irmãos Jefferson Silveira Serafim e Elaine Silveira Serafim.

Gostaria de agradecer ao meu orientador e amigo Professor João Pedro Assumpção Bastos, pelo apoio e por seus esclarecimentos, sempre objetivando um rumo correto e eficiente, e também pela sua confiança.

Gostaria de agradecer também ao apoio e amizade dos Professores Nelson Sadowski e Patrick Kuo Peng e ao Sr. Marcelo Vanti.

A todos os amigos do GRUCAD (Grupo de Concepção e Análise de Dispositivos Eletromagnéticos) em especial ao Jorge, Ivan, Mauro, André, Maurício, Gianfranco, Claudia, Ana, Paulo, Airton, Pinho, Righi e Guilherme.

À excelente banca de examinadores desta dissertação, pela oportuna contribuição através de suas opiniões e sugestões.

Finalmente, a todos aqueles que direta ou indiretamente tiveram a sua parcela de contribuição ao longo deste trabalho, e a Deus por me propiciar paz e confiança nas horas mais difíceis.

Sumário

SIMBOLOGIA

$< a > = < a_1 \quad a_2 \ldots a_n >$	Parâmetros generalizados da aproximação
$<a_x>,<a_y>,<a_z>$	Coordenadas generalizadas do elemento
$e(x) = c(x) - c_{ex}(x)$	Erro de aproximação
$[J],[\,j], det(J)$	Matriz Jacobiana, sua inversa e seu determinante
n	Número de nós de interpolação
n_d	Número de graus de liberdade de um elemento
n^e	Número de nós de interpolação de um elemento
n_{el}	Número de elementos
$\bar{n}$	Número de nós geométricos
$\bar{n}^e$	Número de nós geométricos de um elemento
$<N(x)>=<N_1(x) \quad N_2(x)...>$	Funções de interpolação nodais sobre o elemento real
$<N(u)>=<N_1(u) \quad N_2(u)...>$	Funções de interpolação sobre o elemento de referência
$<N^*(u)>=<N_1^*(u) \quad N_2^*(u)...>$	Funções de transformação geométrica
$[N],<N>$	Matrizes e vetores das funções de interpolação
$[P_n],[P_n^*]$	Matrizes nodais de interpolação
$<P(x)>=<P_1(x) \quad P_2(x)...>$	Base polinomial sobre o elemento real
$<P(u)>=<P_1(u) \quad P_2(u)...>$	Base polinomial sobre o elemento referência
$<P^*(u)>$	Base polinomial da transformação geométrica
$c(x)$	Funções aproximadas
$c_{ex}(x)$	Funções exatas
$c^e(x)$	Funções aproximadas sobre um elemento
$<c_n>=<c_1 \quad c_2...>$	Variáveis nodais
$\{C_n\}$	Conjunto de todas as variáveis nodais
V	Domínio em estudo
V^e	Domínio do elemento real
V^r	Domínio do elemento de referência
$\boldsymbol{x} = <x \quad y \quad z>$	Coordenadas cartesianas de um ponto
$\boldsymbol{x}_i = <x_i \quad y_i \quad z_i>$	Coordenadas do nó i
$<\boldsymbol{x}_n>$	Coordenadas dos nós de um elemento
$\boldsymbol{u} = <u \quad v \quad p>$	Coordenadas de um ponto do elemento de referência

$<u_n>$	Coordenadas dos nós de um elemento de referência
$<\partial_x>,<\partial_u>$	Operadores diferenciais: $\left\langle \frac{\partial}{\partial x}\frac{\partial}{\partial y}\frac{\partial}{\partial z} \right\rangle$ e $\left\langle \frac{\partial}{\partial u}\frac{\partial}{\partial v}\frac{\partial}{\partial p} \right\rangle$
t^e	Transformação geométrica do elemento e
$\| \|_s \quad \| \|_s$	Máximo e norma ao quadrado das derivadas de ordem s de uma função
E_c	Conjunto de funções admissíveis c
E_ψ	Conjunto de funções ponderadas
f_v, f_s	Vetores de forças sobre o volume e a superfície
$\{f\}$	Vetores de forças elementares
$\{F\}$	Vetor de forças concentradas de um sistema discreto
$[k]$	Matriz de elementos
$\{K^*\}$	Matriz a ser integrada para obter k
$[K]$	Sistema matricial global de um sistema discreto
$[K^e]$	Matriz de elementos expandidas
l,m,n	Componentes de vetores unitários perpendiculares as fronteiras de um domínio
$L(u),C(u)$	Operadores diferenciais definindo as equações governantes e as condições de contorno de um sistema físico
R	Resíduo correspondendo a uma equação diferencial parcial
S_c,S_f	Partes da fronteira de um domínio onde c e f são especificados
c	Variáveis desconhecidas de um sistema físico
$W,W_1,W_2,...$	Formas integrais
W^e	Forma integral do elemento
δc	Primeira variação de uma função
$\{\delta c_n\}$	Variação das variáveis nodais de um elemento
$\psi,\psi_1,\psi_2,...$	Funções ponderadas
$[B]$	Matriz relacionando os gradientes em x e as variáveis nodais
$[B_\delta]$	Matriz relacionando as variações dos gradientes em x e as variações das variáveis nodais
$[B_u]$	Matriz relacionando os gradientes em u e as variáveis nodais

$[D]$	Matriz das propriedades do material
$[Q]$	Matriz de transformação dos gradientes
$[T]$	Matriz de transformação das variáveis nodais
dV^e, dV^r	Volume diferencial do elemento real e do elemento de referência
$\{f^*_V\}, \{f^*_S\}$	Vetores a serem integrados para obter-se f_V e f_S
w_i	Coeficientes ponderados da integração numérica
$y(u)$	Funções a serem integradas numericamente
$\{F'\}$	Vetor F depois da eliminação Gaussiana
$[S]$	Matriz triangular superior da decomposição.

Capítulo 1 - INTRODUÇÃO

1.1- Generalidades sobre o Método de Elementos Finitos

Nos últimos anos, modernos projetos de engenharia tornaram-se extremamente complexos, caros e assunto de rigorosa confiança e segurança. Veículos espaciais, aeronaves e reatores nucleares são exemplos de projetos onde a confiabilidade e a segurança são de crucial importância. Para um melhor entendimento, os projetistas precisavam de modelos matemáticos que pudessem ser usados para simular o comportamento de sistemas físicos complexos, que seriam então usados durante o desenvolvimento dos projetos.

As ciências de engenharia exigem em diversos casos, que sistemas físicos sejam modelados por equações diferenciais parciais. Hoje, o Método de Elementos Finitos (FEM) é o método mais difundido na resolução de tais equações. O método, associado com o desenvolvimento da tecnologia computacional, tem sido aplicado com sucesso na solução de problemas em regime transitório e permanente, em regiões lineares e não-lineares, para domínios em uma, duas ou três dimensões. Podendo ainda, modelar formas geométricas complexas, bem como descontinuidades em um material.

O processo de discretização em elementos finitos, analogamente ao processo de diferenças finitas, transformam as equações diferenciais em equações algébricas. Ele atua em três disciplinas fundamentais em Engenharia:

- física contínua;
- análise numérica;
- programação computacional.

A partir da década de 60, o método de elementos finitos tornou-se amplamente conhecido; pesquisas foram realizadas simultaneamente em várias partes do mundo, e em diferentes direções:

- os métodos variacionais e de resíduos ponderados foram aplicados à técnica numérica dos elementos finitos;
- uma enorme variedade de elementos foram desenvolvidos, incluindo elementos curvilíneos e a introdução do conceito de elementos isoparamétrico;
- o método foi generalizado para solução de equações diferenciais parciais; sua aplicação em problemas de estruturas não-lineares e dinâmicas foram amplamente

desenvolvidas; extensões em outros domínios, como mecânica dos sólidos, dos fluidos, termodinâmica, produziram soluções de problemas de engenharia antes inacessíveis.

A partir de 1967, muitos livros foram escritos sobre o Método de Elementos Finitos, com destaque para o professor Zienkiewicz[2], e também Gallagher[3], Rockey[4], Absi[5] e Imbert[6]. Durante o mesmo período, muitos periódicos apresentaram trabalhos sobre o método.

Um grande número de códigos computacionais utilizando o método foram desenvolvidos com sucesso para uso industrial. O maior problema de muitos, estava na preparação dos dados de entrada e na interpretação de volumosos dados de saída. A ausência de controle durante a execução de várias fases de uma análise com elementos finitos era outro problema, bem como uma visualização do que estava sendo executado pelo computador. Hoje a tendência é desenvolver softwares com facilidades na geração de modelos, interação do programa com o usuário e capacidade de visualização gráfica.

1.2 Objetivos desta Dissertação

Esta dissertação tem por objetivo desenvolver subrotinas computacionais (bibliotecas) relacionadas às funções de forma de diferentes elementos finitos, tais como:

- Elemento Triangular 2D (linear, quadrático e cúbico);
- Elemento Quadrilátero 2D (bilinear, quadrático e cúbico);
- Elemento Tetraédrico 3D (linear, quadrático e cúbico);
- Elemento Hexaédrico 3D (trilinear, quadrático e cúbico),

possibilitando assim, uma padronização na estrutura de formação das mesmas, tornando-as mais objetivas, de fácil entendimento, e mais simples de serem utilizadas pelo pessoal envolvido em programação nos pacotes computacionais desenvolvidos no GRUCAD - Grupo de Concepção e Análise de Dispositivos Eletromagnéticos. Juntamente com as subrotinas das funções de forma, também serão desenvolvidas subrotinas com os pontos de integração de cada elemento citado acima, tornando as integrações numéricas de mais fácil acesso.

No transcorrer dos capítulos serão abordados os conceitos necessários para construção das funções de forma para cada um dos elementos citados acima, e na sequência a biblioteca é implementada.

1.3 Organização dos Capítulos desta Dissertação

Esta dissertação está dividida em sete capítulos.

CAPÍTULO 1

Introdução ao método de elementos através de uma abordagem de sua evolução, objetivos e organização da dissertação.

CAPÍTULO 2

Aplicação do Método dos Resíduos Ponderados na construção de integrais.

CAPÍTULO 3

Aplicação de formulação matricial em sistemas de equações algébricas e as fases do processo de condensação para formação do sistema global de equações.

CAPÍTULO 4

Métodos numéricos para construção e solução de sistemas lineares de equações algébricas.

CAPÍTULO 5

Introdução do elemento de referência e das funções de interpolação para elementos em duas e três dimensões.

CAPÍTULO 6

Apresentação geral da biblioteca, com a declaração das variáveis e nomenclatura para todos os elementos, e estrutura das subrotinas.

CAPÍTULO 7

Conclusões finais sobre as bibliotecas e proposta para trabalhos futuros.

Capítulo 2 – FORMULAÇÃO DE INTEGRAIS PARA SISTEMAS FÍSICOS

2.1 – Introdução

Neste capítulo, serão estudadas formulações integrais de problemas físicos que envolvem sistemas de equações algébricas lineares depois da discretização. O método de elementos finitos é um dos procedimentos de discretização que podem ser utilizados para gerar o sistema de equações algébricas.

2.2 – Classificação dos Problemas Físicos

2.2.1 – Sistemas Contínuos e Discretos

Qualquer sistema pode ser caracterizado por uma série de variáveis que são funções de coordenadas espaciais $\boldsymbol{x} = (x,y,z)$ e do tempo t. O sistema é chamado estacionário se nenhuma de suas variáveis forem dependentes do tempo.

Ao aplicarmos as leis físicas a um sistema que possue um certo número de variáveis conhecidas **d** e outras desconhecidas $\boldsymbol{c}$, teremos um modelo matemático que irá relacionar $\boldsymbol{c}$ e **d**. Estas relações formam um sistema de equações em $\boldsymbol{c}$ para serem resolvidas.

Um sistema discreto é descrito por uma série de equações algébricas. Um sistema contínuo é governado por uma série de equações diferenciais, equações diferenciais parciais e/ou equações íntegro-diferenciais, acompanhada pelas apropriadas condições de contorno espaciais e temporais.

Sistemas algébricos podem ser resolvidos diretamente por métodos numéricos. Por outro lado, sistemas contínuos primeiro são discretizados, isto é, substituídos por sistemas aproximadamente equivalentes às equações algébricas.

2.2.2 – Problemas de Equilíbrio

Para tais problemas, que também são conhecidos como valores de fronteira, devemos descobrir os valores desconhecidos $\boldsymbol{c}$ em um sistema em regime permanente. Para um sistema discreto, as equações governantes podem ser escritas na seguinte forma matricial:

$$[K]\{c\} = \{F\} \tag{2.1a}$$

onde $[K]$ é uma matriz de coeficientes caracterizando o sistema, $\{c\}$ são as variáveis desconhecidas, e $\{F\}$ são as forças aplicadas (condições de contorno impostas).

Sistemas contínuos, por outro lado, são caracterizados por equações diferenciais parciais:

$$L(c) + f_V = 0 \quad \text{em um domínio } V$$
$$C(c) = f_S \quad \text{sobre a fronteira } S \text{ de } V \qquad (2.1b)$$

Onde L e C são operadores diferenciais caracterizando o sistema, c são as funções desconhecidas, e f_V e f_S são funções dadas(vetores força) .

2.3 – Método dos Resíduos Ponderados

2.3.1 – Resíduos

Considere um sistema físico contínuo tendendo ao seguinte sistema de equações diferenciais parciais de ordem m (linear):

$$L(c) + f_V = 0 \quad \text{em um domínio } V \qquad (2.2a)$$

As condições de contorno são:

$$C(c) = f_S \quad \text{sobre a fronteira } S \qquad (2.2b)$$

As variáveis c são funções das coordenadas geométricas x. As soluções são funções c, satisfazendo ambas (2.2a) e (2.2b).

A função resíduo é definida como:

$$\mathbf{R}(c) = L(c) + f_V \qquad (2.3)$$

A função resíduo desaparece quando c for uma solução de (2.2).

2.3.2 – Na Forma de Integrais

O Método de Resíduos Ponderados consiste em descobrir as funções c que satisfazem a seguinte equação integral:

$$W(c) = \int_V < \psi > \{R(c)\} dV = \int_V < \psi > \{L(c) + f_V\} dV = 0 \qquad (2.4)$$

para qualquer função ponderada ψ.

Qualquer solução c satisfazendo (2.2a) e (2.2b), também satisfaz (2.4), não importando qual seja a função ponderada. Embora, uma solução de (2.4) dependa da escolha da função ponderada.

2.4 – Transformações das Integrais

2.4.1 – Integração por Partes

A integração por partes é usada em expressões como (2.4) para reduzir a ordem de derivadas elevadas contidas no integrando.

(a) Duas Dimensões:

$$\int_A \psi \frac{\partial c}{\partial x} dxdy = -\int_A \frac{\partial \psi}{\partial x} cdxdy + \oint_S \psi c dy = -\int_A \frac{\partial \psi}{\partial x} cdxdy + \oint_S \psi L dS \qquad (2.5a)$$

$$\int_A \psi \frac{\partial c}{\partial y} dxdy = -\int_A \frac{\partial \psi}{\partial y} cdxdy - \oint_S \psi c dx = \int_A \frac{\partial \psi}{\partial y} cdxdy + \oint_S \psi cm dS \qquad (2.5b)$$

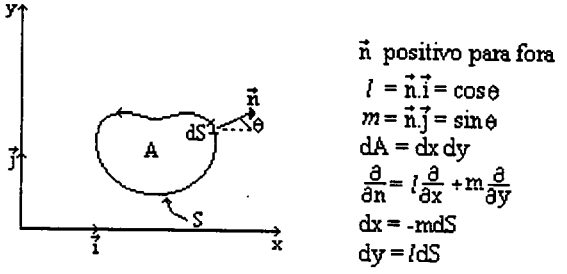


Figura 2.1 - Integração por partes em Duas Dimensões

(c) Três Dimensões:

$$\int_A \psi \frac{\partial c}{\partial x} dxdydz = -\int_V \frac{\partial \psi}{\partial x} cdxdydz + \oint_S \psi cl dS \qquad (2.6)$$

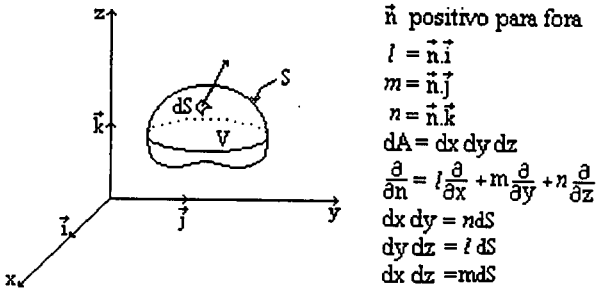


Figura 2.2 - Integração por partes em Três Dimensões

2.4.2 – Forma Fraca da Integral

Uma dada forma integral pode ser transformada para obter, uma então chamada forma fraca, através da integração por partes. Se necessário, podem ser realizadas sucessivas integrações por partes, até que se obtenha a integral desejada. Tais expressões são pretendidas pelas seguintes razões:

- a ordem da derivada superior de c é reduzida; isto então relaxa a condição de continuidade requerida para convergência;
- algumas condições de contorno aparecem na integral fraca, não satisfazendo identicamente c.

Por outro lado, a integração por partes introduz derivadas na função ponderada ψ. As condições de continuidade sobre ψ são então mais difíceis do que antes de tais operações.

2.5 – Discretização de Formas Integrais

2.5.1 – Discretização da Função W

Nos parágrafos 2.3 e 2.4, mostrou-se que a solução direta da equação diferencial (2.2) pode ser substituída pela procura de uma função c que anule a forma integral(2.4):

$$W = -\int_V \left(\frac{\partial \psi}{\partial x}\frac{\partial c}{\partial x} + \frac{\partial \psi}{\partial y}\frac{\partial c}{\partial y} - \psi f_V \right) dV + \int_{S_f} \psi (f_S - \alpha c) dS = 0 \qquad (2.7)$$

para qualquer função ponderada ψ.

Para obter uma solução aproximada c, deve-se transportar (2.4) através da discretização, para dentro de expressões algébricas que possam ser resolvidas para variáveis desconhecidas. Isto deve ser feito em dois passos:

- funções desconhecidas c são aproximadas pela representação de um parâmetro n. Os parâmetros aproximados podem ser valores nodais de c ou não; eles podem estar sobre todo o domínio ou restritos a um sub-domínio. O método de elementos finitos utiliza aproximações nodais sobre o sub-domínio. Em todos os casos c pode ser escrito como:

$$c = c(a_1, a_2, \ldots, a_n) \qquad (2.8)$$

A expressão (2.7) será:

$$W = \int_V \psi(L(c(a_1, a_2, ..., a_n) + f_V))dV = 0 \qquad (2.9)$$

para qualquer ψ.

- n funções ponderadas independentes $\psi_1, \psi_2, ..., \psi_n$ são escolhidas arbitrariamente, mas o número n pode ser o mesmo que o número de parâmetros escolhidos para c (2.8). A escolha das funções ponderadas tendem a uma enorme variedade de métodos, os mais populares são: o método de Galerkin, dos Mínimos Quadrados, etc. A equação (2.9) pode ser escrita como:

$$\begin{aligned} W_1 &= \int_V \psi_1 (L(c(a_1, a_2, ..., a_n) + f_v))dV = 0 \\ W_2 &= \int_V \psi_2 (L(c(a_1, a_2, ..., a_n) + f_v))dV = 0 \\ &..... \\ W_n &= \int_V \psi_n (L(c(a_1, a_2, ..., a_n) + f_v))dV = 0 \end{aligned} \qquad (2.10)$$

Depois da integração de (2.10), um sistema de equações algébricas para determinação dos parâmetros a_i na aproximação de c pode ser escrito como:

$$[K]\{a\}=\{F\} \qquad (2.11)$$

2.5.2 – Escolha das Funções Ponderadas ψ

Aqui será apresentado um método para escolha da função ponderada ψ. No restante desta dissertação, serão aplicadas apenas as aproximações com elementos finitos acopladas com as funções ponderadas do tipo Galerkin, para obter as soluções desejadas.

2.5.2.1 – Método de Galerkin

As funções ponderadas são idênticas as funções de base c. Assim, frequentemente ψ será representada por uma série de variações δu da aproximação c com relação a $\{a\}$.

$$\psi = \delta c = < P > \{\delta a\} \quad \text{para todos } \{\delta a\} \qquad (2.12)$$

onde $\{\delta a\}$ são as primeiras variações de $\{a\}$.

A equação (2.10) torna-se:

$$W = \int_V \delta c(L(c) + f_V)dV = 0 \tag{2.13}$$

$$W = \langle \delta a \rangle \int_V \{P\}(L(< P > \{a\}) + f_V)dV = 0 \tag{2.14}$$

Uma vez que W pode desaparecer para qualquer valor de $\{\delta a\}$, a próxima expressão é equivalente ao seguinte sistema de equações algébricas:

$$\begin{aligned} &W_1(a) = \int_V P_1(L(< P > \{a\}) + f_V)dV = 0 \\ &..... \\ &W_n(a) = \int_V P_n(L(< P > \{a\}) + f_V)dV = 0 \end{aligned} \tag{2.15}$$

Capítulo 3 – FORMULAÇÃO MATRICIAL DO MÉTODO DE ELEMENTOS FINITOS

3.1 – Introdução

Neste capítulo serão descritos os estágios necessários para as análises com elementos finitos. Álgebra matricial será usada para escrever as equações fundamentais, devido ao fato destas expressões serem facilmente transformadas em códigos computacionais.

Primeiramente, o método de elementos finitos será definido como um processo de discretização das formas integrais de Galerkin; que consiste em substituir a forma integral global W pelas formas integrais do elemento W^e. Então, cada forma integral é discretizada usando aproximações com elementos finitos. Isto permite que matrizes elementares possam ser condensadas numa matriz global.

Algumas técnicas de condensação que permitem sobrepor as matrizes elementares dentro das matrizes do sistema global serão descritas neste capítulo. Finalmente, será indicado um método para introduzir as condições de contorno dentro do sistema de equações.

3.2 – Definição do Método de Elementos Finitos

Este método consiste na discretização da forma integral W (parágrafo 2.5) usando uma aproximação por elementos finitos para as variáveis desconhecidas $\boldsymbol{c}$ e resolvendo, o sistema resultará em equações algébricas. Os vários estágios do processo de solução serão brevemente descritos neste parágrafo.

A partir da forma integral de Galerkin(parágrafo 2.5.2.1), em que as funções ponderadas são dadas por $\psi \equiv \delta c$:

$$W = \int_V \delta c \left(L(c) + f_V\right) dV = 0 \qquad (3.1)$$

A integral sobre todo o domínio é substituída por integrais de um sub-domínio(ou domínio de elementos):

$$W = \sum_{e=1}^{n_{el}} W^e = \sum_{e=1}^{n_{el}} \int_V \delta c^e \left(L(c) + f_V\right) dV = 0 \qquad (3.2a)$$

Para cada termo W^e(chamado forma integral elementar) c e δc são substituídos por uma aproximação de elementos finitos limitados pelo domínio do elemento V^e:

$$c^e = <N>\{c_n\}$$
$$\delta c^e = <N>\{\delta c_n\} \qquad (3.2b)$$

Uma vez que $<N>$ é definido nulo fora da região V^e e depende somente dos valores nodais $< c_n>$ pertencentes a V^e, todo o processo computacional é confinado ao domínio do elemento. É esta repetição natural das matrizes elementares que contribui para o sucesso do método. Usando-se (3.2b), W^e torna-se:

$$W^e = \int_V e\, \delta c^e \left(L(c^e) + f_V\right) dV$$
$$W^e = < \delta c_n > \left(\int_V e\, \{N\} L(< N >) dV \{c_n\} + \int_V e\, \{N\} f_V\, dV\right) \qquad (3.2c)$$

Integração por partes é frequentemente usada na equação (3.1) para reduzir a ordem das derivadas superiores contidas na expressão(ver parágrafo 2.4). A forma integral, pode então conter derivadas δc de algumas integrais de contorno suplementares. A formas integrais elementares W^e são então escritas na notação matricial como:

$$W^e = \int_V e \left(\left\langle \delta(\partial c^e) \right\rangle [D] \{\partial c^e\} - \partial c^e . f_V \right) dV - \int_{S_f^e} dc^e . f_S\, dS$$

$$onde: \qquad < \partial c^e > = \left\langle c^e \frac{\partial c^e}{\partial x} \cdots \frac{\partial^2 c^e}{\partial x^2} \right\rangle \qquad (3.3)$$

$$< \delta(\partial c^e) > = \left\langle \delta c^e \delta\left(\frac{\partial c^e}{\partial x}\right) \cdots \delta\left(\frac{\partial^2 c^e}{\partial x^2}\right) \cdots \right\rangle$$

- $[D]$ é a matriz que é independente de c^e e de suas derivadas, para operadores lineares L;
- f_V e f_S são as regiões do corpo e da superfície;
- V^e é o volume do elemento;
- S^e_f é a parte da fronteira sobre V^e, em que a integração por partes atribui formas integrais de contorno adicionais.

Finalmente, usando-se as expressões (3.2b) de c^e e δc^e, bem como expressões similares de ∂c^e e $\delta(\partial c^e)$ em (3.3), obtêm-se a seguinte expressão matricial para a forma integral elementar W^e discretizada:

$$W^e = < \delta c_n > ([k]\{ c_n \} - \{ f \}) \tag{3.4}$$

onde $[k]$ é a matriz com os elementos, e é independente de c_n se L for linear; $\{ f \}$ consiste num arranjo de elementos vetoriais; $\{ c_n \}$ são os valores nodais dos elementos vetoriais; e δc_n é a primeira variação dos valores nodais dos elementos.

A forma integral (3.2a) é construída pelo processo de acoplamento das formas integrais do elemento(3.4):

$$W = \sum_e W^e = \sum_e < \delta c_n > ([k]\{c_n\} - \{f\}) = 0 \tag{3.5a}$$

O resultado do processo de acoplamento é a seguinte forma matricial global total:

$$W = < \delta C_n > ([K]\{ C_n \} - \{ F \}) = 0 \tag{3.5b}$$

onde $[K]$ é o sistema matricial global;$\{ F \}$ é o arranjo vetorial global do lado direito; $\{ C_n \}$ é o vetor global de todos os valores nodais das funções desconhecidas; e $\{\delta c_n\}$ é a variação arbitrária de C_n.

O processo de acoplamento para passar de (3.5a) para (3.5b) é chamado de condensação, e será discutido mais adiante.

Uma vez que (3.5b) é nulo para variações arbitrárias $< \delta C_n >$, tem-se:

$$[K]\{ C_n \} = \{F\} \tag{3.5c}$$

A função resíduo do elemento é definida por:

$$\{r\} = \{f\} - [k] \{ c_n \} \tag{3.6a}$$

A função residual global obtida depois do processo de condensação é:

$$\{R\} = \sum_e \{r\} = \{F\} - [K]\{C_n\} \tag{3.6b}$$

Estes últimos resíduos desaparecem, quando $\{ C_n \}$ torna-se a solução exata de (3.5c).

3.3 – Forma Integral Discretizada W^e

3.3.1 – Expressão Matricial para W^e

Para obter a forma discretizada (3.4) de W^e, serão utilizadas aproximações por elementos finitos para c, δc, e suas derivadas no domínio do elemento e.

Então:

$$\begin{aligned} c &=< N > \{c_n\} \\ \frac{\partial c}{\partial x} &= \left\langle \frac{\partial N}{\partial x} \right\rangle \{c_n\} \\ &\cdots\cdots \\ \delta c &=< N > \{\delta c_n\} \\ \delta\left(\frac{\partial c}{\partial x}\right) &= \left\langle \frac{\partial N}{\partial x} \right\rangle \{\delta c_n\} \\ &\cdots\cdots \end{aligned} \tag{3.7a}$$

Então:

$$\begin{aligned} \{\partial c\} &= \begin{Bmatrix} c \\ \frac{\partial c}{\partial x} \\ \vdots \end{Bmatrix} = \begin{bmatrix} < N > \\ \left\langle \frac{\partial N}{\partial x} \right\rangle \\ \vdots \end{bmatrix} \{c_n\} = [B]\{c_n\} \\ \{\delta(\partial c)\} &= \begin{Bmatrix} \delta c \\ \delta(\frac{\partial c}{\partial x}) \\ \vdots \end{Bmatrix} = \begin{bmatrix} < N > \\ \left\langle \frac{\partial N}{\partial x} \right\rangle \\ \vdots \end{bmatrix} \{\delta c_n\} = [B_\delta]\{\delta c_n\} \end{aligned} \tag{3.7b}$$

Para os operadores auto-adjuntos L:

$$\{\delta(\partial c)\} \equiv \delta(\{\partial c\}); \qquad [B_\delta] \equiv [B]$$

Substituindo-se (3.7a) e (3.7b) em (3.3), obtêm-se:

$$W^e =< \delta c_n > \left(\int_V e [B_\delta]^T [D][B] dV \{c_n\} = \int_V e \{N\} f_V dV - \int_{S^e_f} \{N\} f_S dS \right) \tag{3.8a}$$

Assim, comparando com (3.4):

$$[k] = \int_V e [B_\delta]^T [D][B] dV \tag{3.8b}$$

$$\{f\} = \int_V e \{N\} f_V dV + \int_{S^e_f} \{N\} f_S dS \tag{3.8c}$$

3.3.2 – Forma Integral W^e no Espaço do Elemento de Referência

Neste capítulo, as expressões (3.3) e (3.8) de W^e contêm:

- derivadas de $\boldsymbol{c}$ e $\delta\boldsymbol{c}$ em relação a $\boldsymbol{x}$;

- variáveis nodais $\boldsymbol{c}_n$ e $\delta\boldsymbol{c}_n$, que podem incluir derivadas de $\boldsymbol{c}$ e $\delta\boldsymbol{c}$, em relação a $\boldsymbol{x}$ para os nós;

- integrações no espaço do elemento real V^e.

Todas as derivadas e integrações no espaço $\boldsymbol{x}$ podem ser transformadas no espaço $\boldsymbol{u}$.

3.3.2.1 – Transformação das derivadas com relação a x

As derivadas $c_{,x}$, $c_{,y}$, $c_{,z}$, são transformadas em $c_{,u}$, $c_{,v}$, $c_{,p}$, com a ajuda da uma matriz Jacobiana de transformação geométrica. Por exemplo, em uma dimensão tem-se:

$$c(u) = < N(u) > \{c_n\}$$

$$\frac{dc}{dx} = \frac{du}{dx}\frac{dc}{du} = \frac{du}{dx}\left\langle \frac{dN(u)}{du} \right\rangle \{c_n\}$$

Na expressão (3.7b), a matriz $[B]$ é reescrita como:

$$[B] = [Q][B_u] \tag{3.9}$$

onde $[Q]$ é uma matriz que contêm certos termos de [j] = $[J]^{-1}$; $[B_u]$ é uma matriz similar a $[B]$, que contêm derivadas das funções $N(\boldsymbol{u})$ em termos de u em vez das derivadas da funções $N(\boldsymbol{x})$ com relação a $\boldsymbol{x}$.

3.3.2.2 – Transformação do domínio de integração

A integração sobre o volume do elemento real V^e é substituída por uma integração sobre o volume do elemento de referência V^r:

$$\int_{V^e} ...dV = \int_{V^r} ...\det(J)dudvdp \tag{3.10}$$

Os limites de integração para todos os elementos clássicos de referência descritos anteriormente são:

- Duas dimensões:

Triângulo $\int_{u=0}^{u=1}\int_{v=0}^{v=1-u}...\det(J)dvdu$

Quadrilátero $\int_{u=-1}^{u=1}\int_{v=-1}^{v=1}...\det(J)dvdu$

- Três dimensões:

Tetraédro $\int_{u=0}^{u=1}\int_{v=0}^{v=1-u}\int_{p=0}^{p=1-u-v}...\det(J)dpdvdu$

Hexaédro $\int_{u=0}^{u=1}\int_{v=0}^{v=1-u}\int_{p=-1}^{p=1}...\det(J)dpdvdu$

3.3.2.3 – Transformação das integrais de contorno

(a) Integração linear em duas ou três dimensões.

Integral

$$l = \int_S ...dS$$

é escrita em termos da abscissa curvilínea s sobre a curva S

$$l = \int_{s_1}^{s2} ...J_S ds \tag{3.11a}$$

Em geral, a abscissa s é uma das variáveis u, v ou p. A curva S, geralmente corresponde a uma das arestas do elemento de referência, em que o parâmetro s é definido:

$$x =< N(s) > \{x_n\}, etc...$$
$$J_S = \sqrt{x_{,S}^2 + y_{,S}^2 + z_{,S}^2} \tag{3.11b}$$

(b) Integração superficial em três dimensões.

Integral

$$\int_S ...dS$$

é escrita em termos da coordenadas superficiais s_1 e s_2, que são geralmente, (u,v) ou (u,p) ou (v,p).

$$\int_S \ldots J_S \, ds_1 ds_2 \qquad (3.12a)$$

A superfície S é uma das faces do elemento de referência. Nesta face, um ponto é dado em termos de dois parâmetros s_1 e s_2:

$x =< N(s_1, s_2) > \{x_n\}$ *etc.*

$$J_S = \sqrt{\left(y_{,s_1} z_{,s_2} - z_{,s_1} y_{,s_2}\right)^2 + \left(z_{,s_1} x_{,s_2} - x_{,s_1} z_{,s_2}\right)^2 + \left(x_{,s_1} y_{,s_2} - y_{,s_1} x_{,s_2}\right)^2} \qquad (3.12b)$$

3.3.2.4 – Expressões de $[k]$ e $\{f\}$ no espaço do elemento de referência

As expressões (3.8b) e (3.8c), da matriz $[k]$ e do vetor $\{f\}$, transformadas no espaço de referência V^r, são:

$$[k] = \int_V r \, [B_{\delta u}]^T [Q_\delta]^T [D][Q][B_u] \det(J) \, du dv dp \qquad (3.13a)$$

$$\{f\} = \int_V r \, \{N\} f_V \det(J) \, du dv dp + \int_S r \, \{N\} f_s J_s \, ds_1 ds_2 \qquad (3.13b)$$

onde, $[Q]$ e $[B_u]$ são definidos por (3.9); onde, $[Q_\delta]$ e $[B_{\delta u}]$ são análogos a $[Q]$ e $[B_u]$, exceto pelos operadores não-adjuntos; e J_S é definido (3.11b) ou (3.12b).

As integrais (3.13a) e (3.13b) são geralmente calculadas numericamente, usando os métodos descritos no parágrafo 4.1.

3.4 – Condensação da Forma Discretizada Global W

As operações necessárias para construir o sistema matricial global $[K]$ e o arranjo vetorial $\{F\}$, iniciam pelas matrizes elementares $[k]$ e $\{f\}$.

3.4.1 – Condensação pela Expansão das Matrizes Elementares

Cada forma integral elementar W^e fornece, depois de discretizado:

$$W^e = < \delta c_n > ([k]\{ c_n \} - \{ f \})$$

Os vetores $< \delta c_n >$ e $\{ c_n \}$ são diferentes para cada elemento. Os vetores $\{\delta C_n\}$ e $\{C_n\}$ sendo vetores globais, contêm valores de todos os nós do domínio completo V, apresentados em (3.5b). Os vetores elementares $< \delta c_n >$ e $\{ c_n \}$ estão, assim, contidos nos vetores globais.

Variáveis Globais: $< \delta C_n > = < \delta c_1 \ldots \delta c_i \ldots \delta c_j \ldots \delta c_k \ldots \delta c_n >$

Variáveis Elementares: $< \delta c_n > = < \delta c_i \quad \delta c_j \quad \delta c_k >$

onde: δc_i, δc_j, δc_k, são as variáveis nodais do elemento.

A forma discretizada W é a soma das formas discretizadas elementares W^e (3.5a). Esta operação é chamada processo de condensação:

$$W = \sum_{elementos} W^e$$

$$W = \sum_{elementos} < \delta c_n > ([k]\{c_n\} - \{f\})$$

Inicialmente, esta última expressão será colocada na seguinte forma:

$$W = < \delta C_n > ([K]\{ C_n \} - \{ F \})$$

Para que este objetivo seja alcançado, as formas elementares podem ser reescritas em termos de $\{ C_n \}$ e $< \delta C_n >$:

$$W^e = < \delta C_n > ([K^e]\{ C_n \} - \{F^e\}) \qquad (3.14)$$

A matriz $[K^e]$ é construída pela expansão da matriz $[k]$, e possui a mesma dimensão de $[K]$; similarmente, $\{F^e\}$ é construído pela expansão de $\{f\}$, e possui a mesma dimensão de $[F]$.

(a) Expansão de $[k]$ (ver parágrafo 4.4.1 da referência [1])

$$W^e = < \delta C_n > \begin{bmatrix} 0 & .. & 0 & .. & 0 & .. & 0 \\ : & & : & & : & & : \\ 0 & .. & k_{11} & .. & k_{12} & .. & 0 \\ : & & : & & : & & : \\ 0 & .. & 0 & .. & 0 & .. & 0 \\ : & & : & & : & & : \\ 0 & .. & k_{21} & .. & k_{22} & .. & 0 \\ : & & : & & : & & : \\ 0 & .. & 0 & .. & 0 & .. & 0 \end{bmatrix} \begin{matrix} \\ \\ \leftarrow \textit{linha } i \\ \\ \leftarrow \textit{linha } j \\ \\ \\ \end{matrix} \{C_n\} = < \delta C_n > [K^e]\{C_n\}$$

$$\uparrow \qquad \uparrow \qquad (n \, x \, n)$$

$$\textit{coluna } i \quad \textit{coluna } j \qquad (3.14)$$

(b) Expansão de $\{f\}$ (ver parágrafo 4.4.1 da referência [1])

$$W^e =< \delta C_n > \begin{Bmatrix} 0 \\ 0 \\ 0 \\ f_1 \\ 0 \\ : \\ 0 \\ f_2 \\ 0 \\ : \\ 0 \end{Bmatrix} =< \delta C_n > \{F^e\} \tag{3.15}$$

A forma integral global W é obtida pela sumarização da expressão (3.14):

$$\begin{aligned} W &= \sum_e W^e = \sum_e < \delta C_n > \left([K^e]\{C_n\} - \{F^e\}\right) \\ &=< \delta C_n > \left(\left[\sum_e [K^e]\right]\{C_n\} - \left\{\sum_e \{F^e\}\right\}\right) \\ &=< \delta C_n > ([K]\{C_n\} - \{F\} \end{aligned} \tag{3.16a}$$

onde:

$$\begin{aligned} [K] &= \sum_e [K^e] \\ \{F\} &= \sum_e \{F^e\} \end{aligned} \tag{3.16b}$$

A matriz global $[K]$ é então a soma das matrizes elementares expandidas $[K^e]$; o mesmo ocorre com $\{F\}$ e $\{F^e\}$.

3.4.2 – Fases do Processo de Condensação

- Construção das matrizes $[K^e]$ e $\{F^e\}$ para cada elemento, usando (3.14) e (3.15);

- Adição das matrizes expandidas (3.16).

Estes dois passos são executados simultaneamente para avaliar as construções explícitas de $[K^e]$ e $\{F^e\}$.

3.5 – Sistema Global de Equações

3.5.1 – Formulação do Sistema de Equações

Depois da condensação, a forma integral global discretizada (3.5b) é:

$$W = < \delta C_n > ([K]\{ C_n \} - \{ F \}) = 0 \qquad (3.17)$$

O problema consiste em descobrir $\{C_n\}$que faça W desaparecer não importando qual $< \delta C_n >$, sempre que satisfaça as condições de contorno sobre S_c : $c = c_S$ e $\delta c = 0$. Depois de discretizado, estas condições são:

$$\delta C_i = 0 \qquad (3.18a)$$

$$C_i = C^*_i \qquad (3.18b)$$

para todos os graus de liberdade C_i possuindo um valor específico de C^*_i.

Em resumo, o sistema algébrico:

$$[K]\{C_n\} = \{F\} \qquad (3.19)$$

pode ser resolvido por $\{C_n\}$ depois de algumas modificações necessárias para satisfazer as condições (3.18).

3.5.2 – Introdução das Condições de Contorno

As condições (3.18) podem ser introduzidas no sistema (3.19) por diferentes métodos, um dos mais utilizados consiste em:

(a) *Usar um número grande sobre os termos diagonais*

A matriz $[K]$ é condensada sem dar atenção as condições de contorno; então, cada valor específico de $C_i = C^*_i$ é introduzida como segue:

- o termo K_{ii} é substituído por $K_{ii} + \alpha$, onde α é um número muito grande com relação a todos os outros termos de K_{ij}; se α for grande o bastante, $K_{ii} + \alpha \approx \alpha$;

- todos os termos do lado direito são substituídos pelos seus valores originais diminuídos de $F_j - K_{ji}C^*_i = F^*_j$; o lado direito da equação i é - $K_{ji}C^*_i$.

Este método é muito simples de codificar e oferece, depois de solucionado, todas as variáveis desconhecidas sem perdas na precisão. A matriz de coeficientes K_{ii} se mantêm simétrica, uma vez que somente o termo diagonal é modificado.

Capítulo 4 – PROCEDIMENTO NUMÉRICO

4.1 – Introdução

Neste capítulo serão estudados os procedimentos numéricos mais utilizados na construção de códigos computacionais, para formação das matrizes elementares, e para resolução de sistemas de equações.

4.2 – Integração Numérica

4.2.1 – Introdução

No método de elementos finitos, as matrizes elementares $[k]$ e arranjos vetoriais $\{f\}$, são expressos como integrais sobre linhas, áreas e volumes (3.8b),(3.8c), definidas pela geometria do elemento real V^e:

$$[k] = \int_V{}^e [B_\delta][D][B]dV$$
$$\{f\} = \int_V{}^e \{N\} f_V dV + \int_{S_f}^* \{N\} f_S dS \qquad (4.1a)$$

No espaço do elemento de referência, estas integrais torna-se (3.13a),(3.13b):

$$[k] = \int_V{}^r [B_\delta(u)]^T [D(u)][B(u)]\det(J(u))dV^r$$
$$\{f\} = \int_V{}^r \{N(u)\} f_V \det(J(u))dV^r + \int_{S_f}^r \{N(u_s)\} f_S dS \qquad (4.1b)$$

Onde:

V^r é o volume do elemento de referência;

S^r_f é a parte do contorno do elemento de referência sobre a qual f_S é especificada;

u_S representam as coordenadas de u sobre S^r_f;

$dS = J_S dS_1 dS_2$ (ver parágrafo 3.3.2.3)

$[J]$ é a matriz Jacobiana da transformação geométrica ou mais compactadamente:

$$[k] = \int_V{}^r [k^*]dV^r$$
$$\{f\} = \int_V{}^r \{f_V^*\}dV^r + \int_{S_f}^r \{f_S^*\}dS \qquad (4.2)$$

onde:

$$[k^*] = [B_\delta(\boldsymbol{u})]^T[D(\boldsymbol{u})][B(\boldsymbol{u})]\det(J(\boldsymbol{u}));$$

$$\{f^*_V\} = \{N(\boldsymbol{u})\}f_V \det(J));$$

$$\{f^*_S\} = \{N(\boldsymbol{u}_S)\}f_S.$$

Os termos $[k^*]$, $\{f^*_V\}$ e $\{f^*_S\}$, são polinômios ou frações razoavelmente complicadas. Abaixo, uma lista com algumas das mais usuais fórmulas de monômios:

- Duas dimensões:

- Elemento de Referência Quadrilátero

$$\int_{-1}^{1}\int_{-1}^{1} u^i v^j du dv = \begin{cases} 0 & se\ i\ ou\ j\ for\ ímpar \\ \dfrac{4}{(i+1)(j+1)} & se\ i\ e\ j\ for\ par \end{cases} \tag{4.3a}$$

- Elemento de Referência Triangular

$$\int_0^1\int_0^{1-u} u^i v^j dv du = \frac{i!\,j!}{(i+j+2)!} \tag{4.3b}$$

- Três Dimensões:

- Elemento de Referência Hexaédrico

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} u^i v^j p^k du dv dp = \begin{cases} 0 & se\ i, j\ ou\ k\ for\ ímpar \\ \dfrac{8}{(i+1)(j+1)(k+1)} & se\ i,\ j\ e\ k\ for\ par \end{cases} \tag{4.3c}$$

- Elemento de Referência Tetraédrico

$$\int_0^1\int_0^{1-u}\int_0^{1-u-v} u^i v^j p^k dp dv du = \frac{i!\,j!k!}{(i+j+k+3)!} \tag{4.3d}$$

Para os casos onde a integração numérica não é possível, aplicam-se fórmulas de integração numérica que possuem a seguinte forma genérica:

$$[k] = \sum_{i=1}^{r} w_i [k^*(u_i)]$$
$$\{f\} = \sum_{i=1}^{r} w_i \{f_V^*(u_i)\} \qquad (4.4)$$

Onde u_i são as coordenadas dos r pontos de integração; w_i são os coeficientes ponderados (pesos) correspondentes a cada ponto de integração.

Em geral, (4.4) fornece um valor aproximado das integrais, todavia, para muitas classes de problemas, os erros de integração compensam o erro de discretização geométrica, resultando assim, em melhores resultados globais.

4.2.2 – Integração Numérica em Duas Dimensões

(a) *Métodos Bidirecionais*

Se houver r_1 pontos na direção u e r_2 pontos na direção v, o método de Gauss, integra o produto de um polinômio $(2r_1-1)$ em u por um polinômio $(2r_2-1)$ em v, exatamente. O método usa $r = r_1 \, . \, r_2$ pontos; ele integra todos os monômios:

$$u^i v^j, \text{ tal que } 0 \le i \le 2r_1-1$$
$$0 \le j \le 2r_2-1$$

(b) *Métodos Diretos*

$$\iint_V r \; y(u,v)dudv = \sum_{i=1}^{r} w_i y(u_i, v_i) \qquad (4.5)$$

Em particular, é possível construir fórmulas muito similares às fórmulas de Gauss, que irão integrar exatamente todos os monômios de ordem m:

$$u^i v^j, \text{ tal que } i+j \le m$$

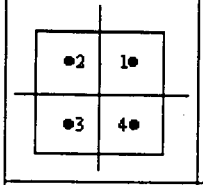
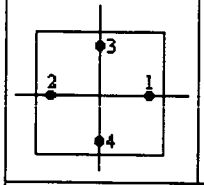
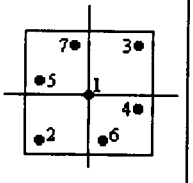
Tais métodos são frequentemente mais efetivos do que os métodos bidirecionais. Para elementos quadriláteros, as fórmulas bidirecionais possuem menos influência geométrica do que os métodos diretos; por outro lado, para os elementos triangulares, os métodos diretos possuem menos influência geométrica do que os métodos bidirecionais.

4.2.2.1 – Elementos Quadriláteros

O método a ser utilizado será o método direto:

$$\int_{-1}^{1}\int_{-1}^{1} y(u,v)dudv \approx \sum_{i=1}^{r} w_i\, y(u_i, v_i)$$

Na figura 4.1 estão os pontos de integração do elemento quadrilátero:

	Ordem m	Nº de pontos r	Coordenadas u_i	v_i	Pesos w_i
	3	4	$\pm\frac{1}{\sqrt{3}}$	$\pm\frac{1}{\sqrt{3}}$	1
	3	4	$\pm\sqrt{2/3}$ 0	0 $\pm\sqrt{2/3}$	1
	5	7	0 -r +r +s -s +t -t	0 -r +r -t +t -s +s	8/7 100/168 20/48

$$r = \sqrt{7/15}$$

$$s = \sqrt{\frac{7+\sqrt{24}}{15}}$$

$$t = \sqrt{\frac{7-\sqrt{24}}{15}}$$

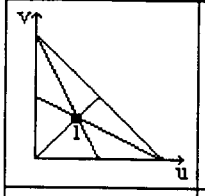
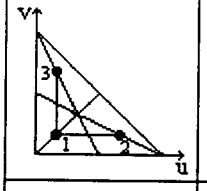
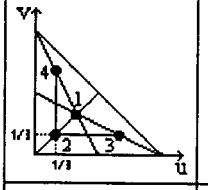
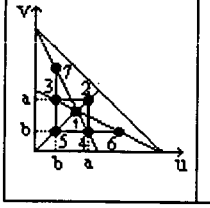
Figura 4.1 - Pontos de integração de um quadrilátero

4.2.2.2 – Elementos Triangulares

O método a ser utilizado será o método direto:

$$\int_{0}^{1}\int_{0}^{1-u} y(u,v)dvdu \approx \sum_{i=1}^{r} w_i\, y(u_i, v_i)$$

Na figura 4.2 estão os pontos de integração do elemento triangular:

	Ordem m	Nº de pontos r	Coordenadas u_i v_i	Pesos w_i
	1	1	$\frac{1}{3}$ $\frac{1}{3}$	$\frac{1}{2}$
	2	3	1/6 1/6 2/3 1/6 1/6 2/3	$\frac{1}{6}$
	3	4	1/3 1/3 1/5 1/5 3/5 1/5 1/5 3/5	-27/96 25/96
	5	7	1/3 1/3 a a 1-2a 1-2a a 1-2a b b 1-2b b b 1-2b	9/80 A B

$a = \frac{6 + \sqrt{15}}{21}$

$b = \frac{4}{7} - a$

$A = \frac{155 + \sqrt{15}}{2400}$

$B = \frac{31}{240} - A$

Figura 4.2 - Pontos de integração de um triângulo

4.2.3 – Integração Numérica em Três Dimensões

4.2.3.1 – Elementos Hexaédricos

O método a ser utilizado será o método direto:

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} y(u,v,p)dudvdp = \sum_{i=1}^{r} w_i y(u_i, v_i, p_i)$$

Na figura 4.3 estão os pontos de integração do elemento hexaédrico:

Nº de pontos r	Coordenadas u_i v_i p_i	Pesos w_i
8	a a a -a a a a -a a a a -a -a -a a -a a -a a -a -a -a -a -a	1

$a = \frac{1}{\sqrt{3}}$

Figura 4.3 - Pontos de integração de um hexaédro

4.2.3.2 – Elementos Tetraédricos

O método a ser utilizado será o método direto:

$$\int_0^1 \int_0^{1-u} \int_0^{1-u-v} y(u,v,p)dpdvdu = \sum_{i=1}^{r} w_i y(u_i, v_i, p_i)$$

Na figura 4.4 estão os pontos de integração do elemento tetraédrico:

Ordem m	Nº de pontos r	Coordenadas u_i v_i p_i	Pesos w_i	
1	1	$\frac{1}{4}$ $\frac{1}{4}$ $\frac{1}{4}$	$\frac{1}{6}$	
2	4	a a a a a b a b a b a a	$\frac{1}{24}$	$a=\frac{5-\sqrt{5}}{20}$ $b=\frac{5+\sqrt{5}}{20}$
5	7	a a a b b b b b c b c b c b b	-2/15 $\frac{3}{40}$	a = 1/4 b = 1/6 c = 1/2

Figura 4.4 - Pontos de integração de um tetraédro

4.2.4 – Escolha do Número de Pontos de Integração

A escolha do número de pontos de integração depende do tipo de elemento utilizado, da matriz particular a ser construída ([*k*] ou [*m*]), e da geometria do elemento real. Em geral, uma vez que o número de operações requeridas é igual a Npi^d, onde *Npi* é o número de pontos de integração e *d* é um inteiro correspondente ao número de dimensões do elemento, é importante reduzir *Npi* a pequenos valores aceitáveis. Tem-se comprovado que aproximações obtidos pela redução da integração possibilitam uma melhor precisão dos resultados globais do que integrações exatas sobre certas circunstâncias [2]. Zienkeiwicz [2]

estudou que o número mínimo de pontos de integração é que irá integrar det(J) corretamente. Esta última condição, embora necessária, não é sempre suficiente; os usuários devem estar atentos a possíveis problemas com uma malha contendo um número muito pequeno de elementos. Para cada ponto de integração, uma ou mais equações existem entre as variáveis nodais do elemento. Portanto, o número mínimo de pontos de integração deve sempre permitir que exista um equilíbrio entre o número total de equações estabelecidas através destes pontos e o número total da variáveis desconhecidas após a aplicação das condições de contorno.

4.3 – Solução de Sistemas de Equações Lineares

4.3.1 – Introdução

A solução do sistema:

$$[K]\ \{C_n\} = \{F\} \tag{4.6}$$

é um passo muito importante na solução do método de elementos finitos. O número n de variáveis desconhecidas C_n é diretamente proporcional ao número de nós e também ao número de graus de liberdade por nós. A precisão e a área de aplicação do método é limitada somente pelo número de equações lineares simultâneas, que podem ser resolvidas pelos computadores atuais.

Os métodos de soluções são geralmente divididos em duas classes:

(a) métodos diretos, também chamados métodos de eliminação Gaussiana;

(b) métodos iterativos, o mais popular é o de Gauss-Seidel.

Com o passar dos anos, as técnicas iterativas tornaram-se mais efetivas e simples de se programar do que os métodos de eliminação direta. Contudo, pesquisas realizadas sobre soluções algorítmicas e programação de computadores, resultaram numa escolha quase-unânime do método de eliminação Gaussiana para construção de códigos computacionais, devido a sua boa precisão e velocidade de resolução dos sistemas em geral.

4.3.2 – Eliminação Gaussiana

As variações mais comuns da técnica de Eliminação Gaussiana, dividem-se em dois passos:

(a) *Triangularização da matriz de coeficientes (também conhecida como processo de redução)* Uma série sistemática de transformações lineares são aplicadas ao sistema de equações a fim de reduzi-los a um sistema triangular (Ver parágrafo 5.2.2.1 da referência [1]):

$$\begin{bmatrix} \diagdown & S \\ 0 & \diagdown \end{bmatrix} \{C_n\} = \{F'\} \tag{4.7}$$

(b)*Substituição anterior*

Para um sistema triangular superior (4.7), a última equação contêm somente uma variável desconhecida e é imediatamente resolvida. As equações anteriores possuem duas variáveis desconhecidas, uma das quais já foi calculada. Depois da substituição, descobre-se a variável restante. O processo é repetido sucessivamente para cada linha do sistema, de baixo para cima (Ver parágrafo 5.2.2.2 da referência [1]).

Capítulo 5 – VÁRIOS TIPOS DE ELEMENTOS

5.1 – Introdução

Neste capítulo será introduzido o conceito de elemento de referência, algumas propriedades da função aproximada $c(x)$, a derivada primeira de operadores diferenciais, e após serão apresentadas as funções de interpolação dos elementos mais utilizados na prática.

Das informações contidas nos capítulos anteriores e no transcorrer deste, será possível construir as funções de interpolação $\langle N(u)\rangle$ e suas derivadas com respeito a u, v para os casos em 2D e u, v e p(3D), através das relações:

$$c = \langle N\rangle \{ c_n \}$$

$$\langle N\rangle = \langle P\rangle [P_n]^{-1} \qquad (5.0)$$

onde $[P_n]$ é a matriz nodal de interpolação. Para os elementos não-isoparamétricos, as mesmas técnicas são usadas para construir as funções de transformação geométrica $\langle N^*\rangle$; Para os elementos isoparamétricos $\langle N\rangle=\langle N^*\rangle$.

5.2. – Elemento de Referência

Será introduzido a fim de simplificar as expressões analíticas de elementos com formas complexas. Um elemento V^r é definido em um espaço abstrato, com uma forma geométrica bem simples. A geometria do elemento de referência é mapeado para dentro da geometria do elemento real, através de expressões de transformação geométrica. Por exemplo, no caso de uma região triangular:

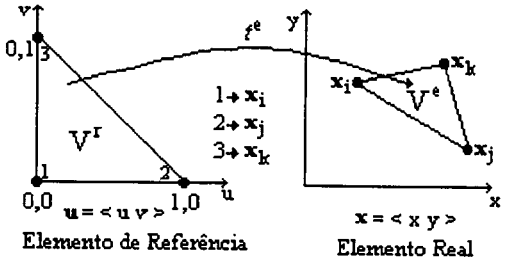


Figura 5.1 - Elemento de referência triangular

As transformações t^e definem as coordenadas de cada ponto do elemento real em termos das coordenadas abstratas u dos correspondentes pontos do elemento de referência.

$$t^e : u \rightarrow x^e = x^e(u) \tag{5.1}$$

As transformações t^e dependem da forma e da localização do elemento real. Assim, para cada elemento real existe uma transformação t^e diferente:

$$t^e : u \rightarrow x^e = x^e(u, x_i, x_j, x_k, \ldots) \tag{5.2}$$

As transformações t^e devem satisfazer certas regras, por isto para alcançar tais propósitos, cada transformação é escolhida, adotando-se as seguintes propriedades:

- ser mapeado um a um, isto é, para cada ponto do elemento referência V^r, existe um e somente um ponto no elemento real V^e;

- os nós geométricos do elemento de referência correspondem aos nós geométricos do elemento real;

- cada porção da fronteira de um elemento de referência, definido pelos nós geométricos desta fronteira, correspondem a porção da fronteira de um elemento real, definido pelos nós correspondentes.

Note-se que um elemento de referência projeta dentro de si todos os elementos reais, usando diferentes transformações t^e. Veja no exemplo da figura 5.2:

$$t : u \rightarrow x(u) = [\, N^*(u)]\{x_n\} \tag{5.3}$$

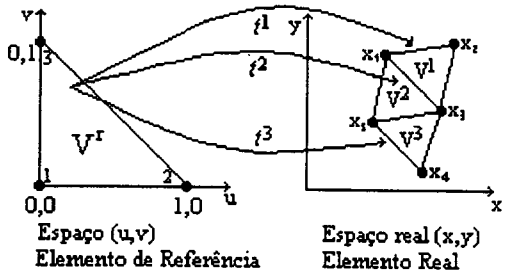


Figura 5.2 - Mapeamento do elemento real dentro do elemento de referência

Elemento 1 : $t^1\ u \rightarrow x^1 = x^1(u, x_1, x_3, x_2)$

Elemento 2 : $t^2\ u \rightarrow x^2 = x^2(u, x_1, x_5, x_3)$

Elemento 3 : $t^3\ u \rightarrow x^3 = x^3(u, x_5, x_4, x_3)$

Além disso, funções de transformação idênticas serão usadas para todas as três coordenadas.

$$x(u) = < N^*(u) > \{ x_n \}$$
$$y(u) = < N^*(u) > \{ y_n \}$$
$$z(u) = < N^*(u) > \{ z_n \}$$

Para o exemplo anterior:

$$x(u,v) = N_1^*(u,v)x_i + N_2^*(u,v)x_j + N_3^*(u,v)x_k =< N^* > \begin{Bmatrix} x_i \\ x_j \\ x_k \end{Bmatrix}$$

$$y(u,v) = N_1^*(u,v)y_i + N_2^*(u,v)y_j + N_3^*(u,v)y_k =< N^* > \begin{Bmatrix} y_i \\ y_j \\ y_k \end{Bmatrix}$$

onde (u,v) pertencem a V^r.

As funções N^*_i, normalmente escolhidas como polinômios em u, são chamadas "funções de transformação geométrica". Pode-se agora, considerar (5.3) como uma aproximação nodal do sub-domínio para funções $x(u)$ e $y(u)$. As funções N^*_i, devem ser escolhidas de tal forma que satisfaçam as propriedades descritas anteriormente.

5.2.1 – Forma Algébrica da Função Aproximada $c(x)$

Inicialmente, será escolhido uma série de n nós de interpolação de coordenadas x_i no domínio V. Estes nós não precisam coincidir com os nós geométricos. Para cada elemento V^e, será usado uma aproximação nodal para a função exata $c_{ex}(x)$:

$$c_{ex}(x) \approx (x) =< N_1(x)\ \ N_2(x)...N_{n^e}(x) > \begin{Bmatrix} c_1 \\ c_2 \\ : \\ c_{n^e} \end{Bmatrix} =< N(x) > \{c_n\} \quad (5.4)$$

Onde $\boldsymbol{x}$ pertence a V^e. c_1, c_2,..., c_n, são valores de c_{ex} , para os nós de interpolação n^e. $N(\boldsymbol{x})$ são as funções de interpolação. Agora, troca-se a aproximação sobre o elemento real pela correspondente aproximação sobre o elemento de referência:

$$c_{ex}(\boldsymbol{u}) \approx c(\boldsymbol{x}) = <N(\boldsymbol{u})> \{c_n\} \quad (5.5)$$

onde $\{c_n\}$ são as variáveis nodais do elemento, e $<N(\boldsymbol{u})>$ são as funções de interpolação do elemento de referência.

Em geral, as funções $N(\boldsymbol{x})$ são usadas somente para os elementos mais simples. As funções $N(\boldsymbol{u})$, onde $\boldsymbol{x}$ e $\boldsymbol{u}$ estão relacionadas pela transformação t, é definida por (5.3).

Na expressão (5.4), as funções $N(\boldsymbol{x})$ dependem das coordenadas nodais de cada elemento, logo, são diferentes para cada elemento. Por outro lado, na expressão (5.5), as funções $N(\boldsymbol{u})$ são independentes da geometria do elemento real V^e. Assim, uma única série de funções $N(\boldsymbol{u})$, para o mesmo elemento de referência, pode ser usada em todos os elementos.

5.2.2 – Propriedades da Função Aproximada $c(x)$

(a) *Propriedade fundamental da aproximação nodal*

Os valores da função aproximada $c(x)$, coincidem com os valores da função exata $c_{ex}(\boldsymbol{x})$ para todos os nós de interpolação:

$$c_{ex}(x_i) = c(x_i) = c_i =< N_1(x_i)\ \ N_2(x_i)... > \begin{Bmatrix} c_1 \\ c_2 \\ : \\ c_{n^e} \end{Bmatrix} \quad (5.6)$$

$$N_j(x_i) = \begin{cases} 0 & se\ \ i \neq j \\ 1 & se\ \ i = j \end{cases}$$

Similarmente, usando a aproximação sobre o elemento de referência:

$$c_{ex}(u_i) = c(u_i) = c_i =< N_1(u_i)\ N_2(u_i)... > \begin{Bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{ne} \end{Bmatrix} \qquad (5.7)$$

$$N_j(u_i) = \begin{cases} 0 & se\ i \neq j \\ 1 & se\ i = j \end{cases}$$

(b) *Continuidade dentro do elemento*

Se a função $c(x)$ é requerida para ser contínua junto com todas as suas derivadas superiores de ordem s, as funções de interpolação $N_i(x)$ de mesma qualidade podem ser usadas.

(c) *Funções de interpolação de um polinômio completo*

O erro de truncamento: $e(x)=c(x)-c_{ex}(x)$, pode ser reduzido pela diminuição no tamanho dos elementos. Para muitos problemas, também é necessário diminuir o erro sobre as derivadas das funções de aproximação. Para assegurar que o erro $c - c_{ex}$ tenda a zero com a diminuição no tamanho do elemento, é essencial que a aproximação c contenha termos constantes não-nulos, sendo assim, c será capaz de representar exatamente a função constante c_{ex} dentro do elemento. Para assegurar que o erro da derivada também tenda a zero é necessário que exista um termo em x. Assim, se $\partial c_{ex}/\partial x$ for constante, $\partial c/\partial x$ será capaz de representá-la exatamente. Em geral, se os erros sobre c_{ex} e suas derivadas de ordem superior estiverem diminuindo com o tamanho do elemento, a expressão (5.4) pode conter um polinômio completo de ordem s. Além disso, se a função c e suas derivadas, são contínuas sobre a fronteira comum com os elementos adjacentes, então os erros de truncamento para c_{ex} e suas derivadas tenderão a zero em toda parte do domínio V, incluindo as fronteiras.

Definições:

- Se somente os valores das funções são contínuos sobre a fronteira, a função é chamada de classe C^0. Quando a função e suas primeiras derivadas são contínuas, ela é de classe C^1. Em

geral, se a função e todas as suas derivadas superiores de ordem α são contínuas, ela é chamada de classe C^{α}.

- Um elemento será isoparamétrico, se as funções de transformação geométrica $N^*(\boldsymbol{u})$ forem idênticas as funções de interpolação $N(\boldsymbol{u})$. Para cada elemento, os nós geométricos são idênticos aos nós de interpolação.
- Um elemento será pseudo-paramétrico, se suas funções $N^*(\boldsymbol{u})$ e $N(\boldsymbol{u})$ forem polinômios diferentes construídos com o mesmo monômio.
- Um elemento será sub-paramétrico quando os polinômios geométricos $N^*(\boldsymbol{u})$ forem de ordem menor que as do polinômio de interpolação $N(\boldsymbol{u})$. Caso contrário, será super-paramétrico.
- O número de variáveis nodais associado com o número total de nós de interpolação de um elemento, é chamado de graus de liberdade(n_d).

5.3 – Transformação dos Operadores Diferenciais

5.3.1 – Generalidades

As equações governantes de um problema físico são escritas no domínio real, e envolvem funções desconhecidas c_{ex} e suas derivadas: $\partial c_{ex}/\partial x$, $\partial c_{ex}/\partial y$, etc. Visto que a aproximação (5.4) no espaço do elemento real é frequentemente muito complicado, é mais conveniente trabalhar no espaço do elemento de referência (5.5):

$$c_{ex} \approx c(\boldsymbol{x}) = \langle N(\boldsymbol{u}) \rangle \{c_n\} \tag{5.8}$$

junto com a transformação (5.3).

$$\begin{aligned} & t : \boldsymbol{u} \rightarrow \boldsymbol{x}(\boldsymbol{u}) = [\, N^*(u)]\{\boldsymbol{x}_n\} \\ & \boldsymbol{x} = \langle x\ y\ z \rangle \\ & \boldsymbol{u} = \langle u\ v\ p \rangle \end{aligned} \tag{5.9}$$

A transformação sendo uma a uma, obtêm-se:

$$t^{-1} : \boldsymbol{x} \rightarrow \boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x}) \tag{5.10}$$

Uma vez que a transformação inversa t^{-1} é geralmente difícil de ser construída, exceto para elementos muito simples, é mais conveniente realizar todas as operações com o elemento de referência. Para expressões contendo derivadas com relação ao espaço real (x,y,z), é necessário obter expressões equivalentes no espaço do elemento de referência (u,v,p). Tais expressões dependem da matriz Jacobiana $[J]$ da transformação.

5.3.2 – Derivada Primeira

Usando as regras de cálculo em vigor, obtêm-se a seguinte expressão:

$$\begin{Bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \\ \frac{\partial}{\partial p} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \\ \frac{\partial x}{\partial p} & \frac{\partial y}{\partial p} & \frac{\partial z}{\partial p} \end{bmatrix} \begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{Bmatrix} \tag{5.11a}$$

Ou

$$\{\partial_u\} = [J]\{\partial_x\} \tag{5.11b}$$

onde $[J]$ é a matriz Jacobiana da transformação geométrica. Uma expressão similar consiste em trocar as derivadas do espaço real pelas do espaço de referência.

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} & \frac{\partial p}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial v}{\partial y} & \frac{\partial p}{\partial y} \\ \frac{\partial x}{\partial z} & \frac{\partial y}{\partial z} & \frac{\partial z}{\partial z} \end{bmatrix} \begin{Bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \\ \frac{\partial}{\partial p} \end{Bmatrix} \tag{5.12a}$$

Assim,

$$\{\partial_x\} = [\,j\,]\{\partial_u\} \tag{5.12b}$$

Substituindo (5.12b) em (5.11b):

$$[\,j\,] = [J]^{-1} \tag{5.12c}$$

Determinação da forma explícita [j], para casos em duas e três dimensões:

* Duas dimensões:

$$[J]=\begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}; \quad [J]^{-1}=\frac{1}{\det J}\begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix} \tag{5.13}$$

* Três dimensões:

$$[J]=\begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{32} \\ J_{31} & J_{32} & J_{33} \end{bmatrix}; \quad [J]^{-1}=\frac{1}{\det(J)}\begin{bmatrix} J_{22}J_{33}-J_{32}J_{23} & J_{13}J_{32}-J_{12}J_{33} & J_{12}J_{23}-J_{13}J_{22} \\ J_{31}J_{23}-J_{21}J_{33} & J_{11}J_{33}-J_{13}J_{31} & J_{21}J_{13}-J_{23}J_{11} \\ J_{21}J_{32}-J_{31}J_{22} & J_{12}J_{31}-J_{32}J_{11} & J_{11}J_{22}-J_{12}J_{21} \end{bmatrix}$$

onde:

$$\det(J) = J_{11}(J_{22}\,J_{33} - J_{32}\,J_{23}) + J_{12}(J_{31}\,J_{23} - J_{21}\,J_{33}) + J_{13}(J_{21}\,J_{32} - J_{31}\,J_{22}) \tag{5.14}$$

Computação dos termos de $[J]$

Os vários termos de $[J]$ são obtidos diretamente pela diferenciação da expressão (5.3), reescrita da seguinte forma:

$$< x\ y\ z > = < N^*(\boldsymbol{u}) > [\{x_n\}\ \{y_n\}\ \{z_n\}] \tag{5.15}$$

$\{x_n\},\{y_n\},\{z_n\}$ são as coordenadas nodais geométricas. A matriz Jacobiana é:

$$[J]=\begin{Bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \\ \frac{\partial}{\partial p} \end{Bmatrix} < x\ y\ z > = \begin{bmatrix} < N^*_{,u} > \\ < N^*_{,v} > \\ < N^*_{,p} > \end{bmatrix} [\{x_n\}\ \{y_n\}\ \{z_n\}] \tag{5.16}$$

$$(3xn^e) \qquad (n^e x3)$$

Ela é obtida pelo produto de duas matrizes, uma contendo as derivadas das funções de transformação geométrica com respeito ao espaço do elemento de referência, e outra contendo as coordenadas reais dos nós geométricos do elemento.

Transformação de uma integral

Uma mudança de variáveis em (5.9), permite-nos mudar a integração de uma função f no domínio real V^e, em uma simples integração no espaço do elemento de referência V^r.

$$\int_{V^e} f(x)dxdydz = \int_{V^r} f(x(u))\det(J)dudvdp \qquad (5.17)$$

5.4 – Elementos Triangulares

5.4.1 – Sistema de Coordenadas

Para todos os elementos triangulares, o seguinte elemento de referência será utilizado:

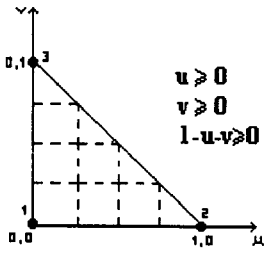
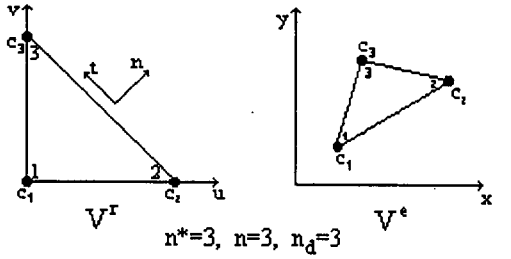


Figura 5.3 - Coordenadas do elemento de referência Triangular

As coordenadas (u,v) também podem ser interpretadas como coordenadas curvilíneas de um elemento real. Por convenção, os nós são numerados no sentido anti-horário.

5.4.2 – Elemento Linear(triângulo, três nós, C^0)



$n^*=3$, $n=3$, $n_d=3$

Figura 5.4 - Elemento Triangular Linear

Os nós geométricos e os de interpolação são idênticos.

Base polinomial: $\langle P \rangle = \langle 1 \quad u \quad v \rangle$ (5.18a)

$$[P_n]=\begin{bmatrix} < P(u_1) > \\ < P(u_2) > \\ < P(u_3) > \end{bmatrix}; \qquad [P_n]^{-1}=\begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{5.18b}$$

Substitui-se $<P>$ e $[P_n]^{-1}$ em (5.0) para obter-se N, isto vale para todos os elementos.

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.1:

Nó	{N}	{∂N/∂u}	{∂N/∂v}
1	1-u-v	-1	-1
2	u	1	0
3	v	0	1

Tabela 5.1 - Funções de forma e suas derivadas, do elemento Triangular Linear

Apresentação da matriz Jacobiana e do determinante:

$$[J]=\begin{bmatrix} x_2-x_1 & y_2-y_1 \\ x_3-x_1 & x_3-x_1 \end{bmatrix} \tag{5.18c}$$
$$\det(J)=2A=(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)$$

Onde A é a área do triângulo.

5.4.3 – Tipo Lagrangiano(continuidade C^0)

Estes elementos são obtidos pelo aumento do número de nós de interpolação sobre a fronteira e no interior dos elementos. Alguns dos nós geométricos, funções geométricas N^* e Matrizes Jacobianas $[J]$ vistas anteriormente são preservadas. Estes elementos são sub-paramétricos.

5.4.3.1 – Elemento Quadrático(triângulo, seis nós, C^0)

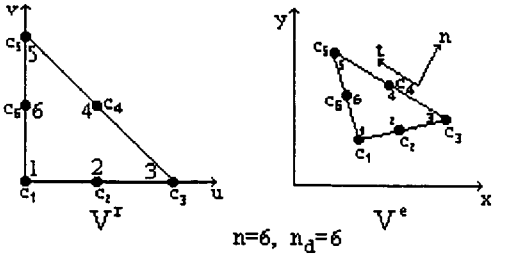


Figura 5.5 - Elemento Triangular Quadrático

Base polinomial: $\langle P \rangle = \langle 1 \quad u \quad v \quad u^2 \quad uv \quad v^2 \rangle$ (5.19a)

Coordenada dos nós: $\langle u_i \rangle = \langle 0 \quad 0;\ ½ \quad 0;\ 1 \quad 0;\ ½ \quad ½;\ 0 \quad 1;\ 0 \quad ½ \rangle$

$\langle x_i \rangle = \langle x_1 \quad y_1;\ x_2 \quad y_2;\ x_3 \quad y_3;\ x_4 \quad y_4;\ x_5 \quad y_5;\ x_6 \quad y_6 \rangle$

onde:

$$x_2 = ½\,(x_1 + x_3)$$
$$x_4 = ½\,(x_3 + x_5)$$
$$x_6 = ½\,(x_5 + x_1)$$

$$[P_n]^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -3 & 4 & -1 & 0 & 0 & 0 \\ -3 & 0 & 0 & 0 & -1 & 4 \\ 2 & -4 & 2 & 0 & 0 & 0 \\ 4 & -4 & 0 & 4 & 0 & -4 \\ 2 & 0 & 0 & 0 & 2 & -4 \end{bmatrix} \qquad (5.19b)$$

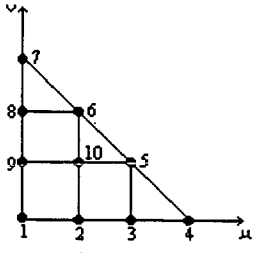
As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.2:

Nó	{N}	{∂N/∂u}	{∂N/∂v}
1	-t(1-2t)	1-4t	1-4t
2	4ut	4(t-u)	4u
3	-u(1-2u)	-1+4u	0
4	4uv	4v	4u
5	-u(1-2v)	0	-1+4v
6	4vt	-4v	4(t-v)

$t = 1 - u - v$

Tabela 5.2 - Funções de forma e suas derivadas, do elemento Triangular Quadrático

5.4.3.2 – Elemento Cúbico Completo(triângulo, dez nós, C^0)



$n=10,\ n_d=10$

Figura 5.6 - Elemento Triangular Cúbico

As funções de interpolação N, podem ser obtidas diretamente pela fórmula:

$$N(i,j,k) = \prod_{l=0}^{i-1} \frac{l - r.u}{l - i} \prod_{m=0}^{j-1} \frac{m - r.v}{m - j} \prod_{n=0}^{k-1} \frac{n - r(1-u-v)}{n - k} \qquad (5.20a)$$

Onde: i e j são os nós do elemento;

$k = r - i - j$;

r é a ordem do polinômio.

Base polinomial: $\langle P \rangle = \langle\ 1\ \ u\ \ v\ \ u^2\ \ uv\ \ v^2\ \ u^3\ \ u^2v\ \ uv^2\ \ v^3\ \rangle$ (5.21)

$$[P_n]^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -5.5 & 9 & -4.5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -5.5 & 0 & 0 & 0 & 0 & 0 & 1 & -4.5 & 9 & 0 \\ 9 & -22.5 & 18 & -4.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 18 & -22.5 & 4.5 & 0 & -4.5 & -4.5 & 0 & 4.5 & -22.5 & 27 \\ 9 & 0 & 0 & 0 & 0 & 0 & -4.5 & 18 & -22.5 & 0 \\ -4.5 & 13.5 & -13.5 & 4.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ -13.5 & 27 & -13.5 & 0 & 13.5 & 0 & 0 & 0 & 13.5 & -27 \\ -13.5 & 13.5 & 0 & 0 & 0 & 13.5 & 0 & -13.5 & 27 & -27 \\ -4.5 & 0 & 0 & 0 & 0 & 0 & 4.5 & -13.5 & 13.5 & 0 \end{bmatrix}$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.3:

Nó	c.[N]	c.{∂N/∂u}	c.{∂N/∂v}	c
1	t(-1+3t)(-2+3t)	-2+18t-27t²	-2+18t-27t²	
2	9tu(-1+3t)	9t(-1+3t-6u)+9u	-9u(-1+6t)	
3	9tu(-1+3u)	9u(1+6t-3u)-9t	-9u(-1+3u)	
4	u(-1+3u)(-2+3u)	2-18u+27u²	0	
5	9uv(-1+3u)	9v(-1+6u)	9u(-1+3u)	1/2
6	9uv(-1+3v)	9v(-1+3v)	9u(-1+6v)	
7	v(-1+3v)(-2+3v)	0	2-18v+27v²	
8	9tv(-1+3v)	-9v(-1+3v)	9v(1+6t-3v)-9t	
9	9tv(-1+3t)	-9v(-1+6t)	9t(-1+3t-6v)+9v	
10	54uvt	54v(t-u)	54u(t-v)	

$t = 1 - u - v$

Tabela 5.3 - Funções de forma e suas derivadas, do elemento Triangular Cúbico

5.5 – Elementos Quadriláteros

5.5.1 – Sistema de Coordenadas

Para todos os elementos de referência, o seguinte sistema de referência será utilizado:

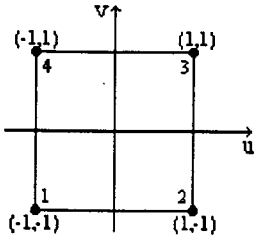


Figura 5.7 - Coordenadas do elemento de referência Quadrilátero

As coordenadas (u,v) também podem ser interpretadas como coordenadas curvilíneas de um elemento real.

Por convenção, os nós são numerados no sentido anti-horário.

5.5.2 – Elemento Bilinear(quadrilátero, quatro nós, C^0)

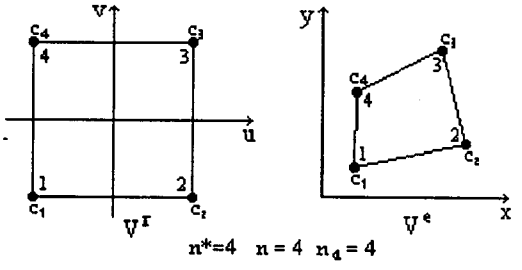


Figura 5.8 - Elemento Quadrilátero Bilinear

O elemento é iso-paramétrico.

Base polinomial: $$<P> = <\ 1\ \ u\ \ v\ \ uv\ > \tag{5.22a}$$

$$[P_n] = \begin{bmatrix} < P(u_1) > \\ < P(u_2) > \\ < P(u_3) > \\ < P(u_4) > \end{bmatrix}; \quad [P_n]^{-1} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \tag{5.22b}$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.4:

Nó	c.{N}	c.{∂N/∂u}	c.{∂N/∂v}	c
1	(1-u)(1-v)	-1+v	-1+u	
2	(1+u)(1-v)	1-v	-1-u	$\frac{1}{4}$
3	(1+u)(1+v)	1+v	1+u	
4	(1-u)(1+v)	-1-v	1-u	

Tabela 5.4 - Funções de forma e suas derivadas, do elemento Quadrilátero Bilinear

5.5.3 – Elementos Lagrangianos de Ordem Superior(C^0)

5.5.3.1 – Quadrático Incompleto(quadrilátero, oito nós, C^0)

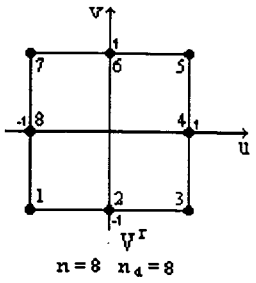


Figura 5.9 - Elemento Quadrilátero Quadrático Incompleto

Base polinomial: $\langle P \rangle = \langle\ 1 \quad u \quad v \quad u^2 \quad uv \quad v^2 \quad u^2v \quad uv^2\ \rangle$ (5.23)

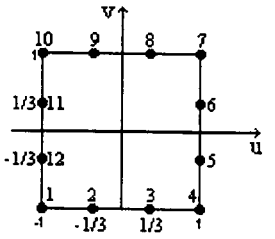
Este elemento possui continuidade da classe C^0.

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.5:

Nó	{N}	{∂N/∂u}	{∂N/∂v}
1	-(1-u)(1-v)(1+u+v)/4	(1-v)(2u+v)/4	(1-u)(u+2v)/4
2	$(1-u^2)(1-v)/2$	-(1-v)u	$-(1-u^2)/2$
3	-(1+u)(1-v)(1-u+v)/4	(1-v)(2u-v)/4	-(1+u)(u-2v)/4
4	$(1+u)(1-v^2)/2$	$(1-v^2)/2$	-(1+u)v
5	-(1+u)(1+v)(1-u-v)/4	(1+v)(2u+v)/4	(1+u)(u+2v)/4
6	$(1-u^2)(1+v)/2$	-(1+v)u	$(1-u^2)/2$
7	-(1-u)(1+v)(1+u-v)/4	(1+v)(2u-v)/4	-(1-u)(u-2v)/4
8	$(1-u)(1-v^2)/2$	$-(1-v^2)/2$	-(1-u)v

Tabela 5.5 - Funções de forma e suas derivadas, do elemento Quadrilátero Quadrático Incompleto

5.5.3.2 – Cúbico Incompleto(quadrilátero, doze nós, C^0)



$n = 12 \quad n_q = 12$

Figura 5.10- Elemento Quadrilátero Cúbico Incompleto

Base Polinomial: $\langle P \rangle = \langle \; 1 \;\; u \;\; v \;\; u^2 \;\; uv \;\; v^2 \;\; u^3 \;\; u^2v \;\; uv^2 \;\; v^3 \;\; u^3v \;\; uv^3 \; \rangle$ (5.24)

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.6:

Nó	c.{N}	c.{∂N/∂u}	c.{∂N/∂v}	c
1	$(1-u)(1-v)t$	$(1-v)(10/9+2u-3u^2-v^2)$	$(1-u)(10/9+2v-u^2-3v^2)$	
2	$(1-3u)(1-u^2)(1-v)$	$(1-v)(-3-2u+9u^2)$	$(-1+u^2)(1-3u)$	
3	$(1+3u)(1-u^2)(1-v)$	$(1-v)(3-2u-9u^2)$	$(-1+u^2)(1+3u)$	
4	$(1+u)(1-v)t$	$(1-v)(-10/9+2u+3u^2+v^2)$	$(1+u)(10/9+2v-u^2-3v^2)$	
5	$(1+u)(1-3v)(1-v^2)$	$(1-v^2)(1-3v)$	$(1+u)(-3-2v+9v^2)$	
6	$(1+u)(1+3v)(1-v^2)$	$(1-v^2)(1+3v)$	$(1+u)(3-2v-9v^2)$	9/32
7	$(1+u)(1+v)t$	$(1+v)(-10/9+2u+3u^2+v^2)$	$(1+u)(-10/9+2v+u^2+3v^2)$	
8	$(1+3u)(1-u^2)(1+v)$	$(1+v)(3-2u-9u^2)$	$(1-u^2)(1+3u)$	
9	$(1-3u)(1-u^2)(1+v)$	$(1+v)(-3-2u+9u^2)$	$(1-u^2)(1-3u)$	
10	$(1-u)(1+v)t$	$(1+v)(10/9+2u-3u^2-v^2)$	$(1-u)(-10/9+2v+u^2+3v^2)$	
11	$(1-u)(1+3v)(1-v^2)$	$(-1+v^2)(1+3v)$	$(1-u)(3-2v-9v^2)$	
12	$(1-u)(1-3v)(1-v^2)$	$(-1+v^2)(1-3v)$	$(1-u)(-3-2v+9v^2)$	

Tabela 5.6 - Funções de forma e suas derivadas, do elemento Quadrilátero Cúbico $t = -10/9 + u^2 + v^2$

5.6 – Elementos Tetraédricos(três-dimensões)

5.6.1 – Sistema de Coordenadas

Para todos os elementos tetraédricos, o seguinte elemento de referência será utilizado:

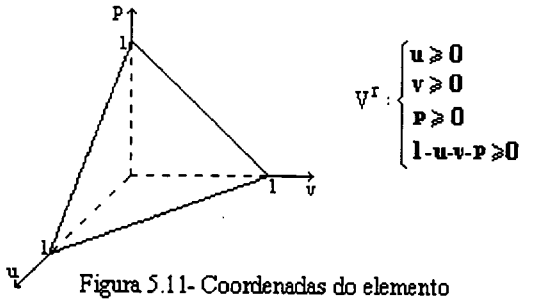


Figura 5.11- Coordenadas do elemento de referência Tetraédrico

Como nos casos anteriores, as coordenadas (u,v,p) também podem ser interpretadas como coordenadas curvilíneas de um elemento real.

5.6.2 – Elemento Linear(tetraédro, quatro nós, C^0)

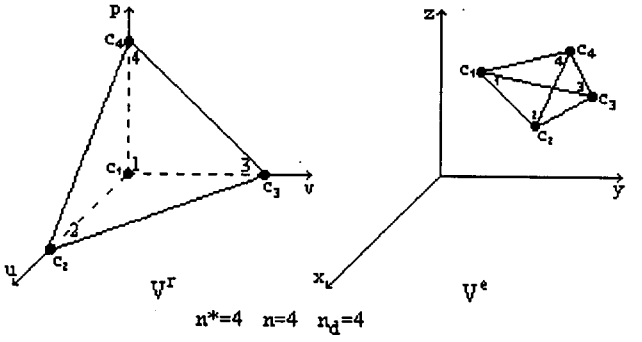


Figura 5.12 - Elemento Tetraédrico Linear

Os quatro nós geométricos são idênticos aos nós de interpolação, isto é, o elemento é isoparamétrico.

Base polinomial: $\langle P \rangle = \langle\ 1\ u\ v\ p\ \rangle$ (5.25a)

$$[P_n] = \begin{bmatrix} < P(u_1) > \\ < P(u_2) > \\ < P(u_3) > \\ < P(u_4) > \end{bmatrix} \qquad [P_n]^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} \tag{5.25b}$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.7:

Nó	{N}	{∂N/∂u}	{∂N/∂v}	{∂N/∂p}
1	1-u-v-p	-1	-1	-1
2	u	1	0	0
3	v	0	1	0
4	p	0	0	1

Tabela 5.7 - Funções de forma e suas derivadas, do elemento Tetraédrico Linear

A matriz Jacobiana é:

$$[J] = \begin{bmatrix} x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \\ x_4 - x_1 & y_4 - y_1 & z_4 - z_1 \end{bmatrix} \qquad \det(J) = 6V \tag{5.25c}$$

Onde: V é o volume do elemento real.

5.6.3 – Elementos Lagrangianos de Ordem Superior(C^0)

5.6.3.1 – Quadrático Completo(tetraédro, dez nós, C^0)

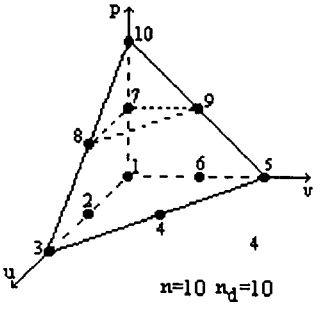


Figura5.13 - Elemento Tetraédrico Quadrático

A representação geométrica linear dos elementos anteriores é preservada. A base polinomial completa é:

$$<P> = < 1 \quad u \quad v \quad p \quad u^2 \quad uv \quad p^2 \quad vp \quad p^2 \quad up \; > \tag{5.26}$$

As funções *N* são facilmente derivadas do elemento triangular quadrático.

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.8:

Nó	{N}	{∂N/∂u}	{∂N/∂v}	{∂N/∂p}
1	-t(1-2t)	1-4t	1-4t	1-4t
2	4ut	4(t-u)	-4u	-4u
3	-u(1-2u)	-1+4u	0	0
4	4uv	4v	4u	0
5	-v(1-2v)	0	-1+4v	0
6	4vt	-4v	4(t-v)	-4v
7	4pt	-4p	-4p	4(t-p)
8	4up	4p	0	4u
9	4vp	0	4p	4v
10	-p(1-2p)	0	0	-1+4p

t = 1 - u - v - p

Tabela 5.8 - Funções de forma e suas derivadas, do elemento Tetraédrico Quadrático

5.6.3.2 – Cúbico Completo(tetraédro, vinte nós, C^0)

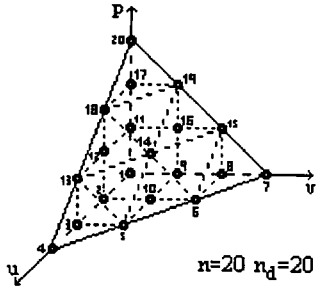


Figura 5.14 - Elemento Tetraédrico Cúbico

A base polinomial é:

$$<P> = < 1 \quad u \quad v \quad p \quad u^2 \quad uv \quad v^2 \quad vp \quad p^2 \quad up \quad u^3 \quad u^2v \quad uv^2 \quad v^3 \quad v^2p \quad vp^2 \quad p^3 \quad up^2 \quad u^2p \quad uvp \; > \tag{5.27}$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.9:

Nó	c.{N}	c.{∂N/∂u}	c.{∂N/∂v}	c.{∂N/∂p}	c
1	t(-1+3t)(-2+3t)	(-2+18t-27t²)	(-2+18t-27t²)	(-2+18t-27t²)	
2	9tu(-1+3t)	9(t(-1+3t-6u)+u)	-9u(-1+6t)	-9u(-1+6t)	
3	9tu(-1+3u)	-9(u(1+6t-3u)-t)	-9u(-1+3u)	-9u(-1+3u)	
4	u(-1+3u)(-2+3u)	(2-18u+27u²)	0	0	
5	9uv(-1+3u)	9v(-1+6u)	9u(-1+3u)	0	
6	9uv(-1+3v)	9v(-1+3v)	9u(-1+6v)	0	
7	v(-1+3v)(-2+3v)	0	(2-18v+27v²)	0	
8	9tv(-1+3v)	-9v(-1+3v)	9(v(1+6t-3v)-t)	-9v(-1+3v)	
9	9tv(-1+3t)	-9v(-1+6t)	9(t(-1+3t-6v)+v)	-9v(-1+6t)	$\frac{1}{2}$
10	54uvt	54v(t-u)	54u(t-v)	-54uv	
11	9tp(-1+3t)	-9p(-1+6t)	-9p(-1+6t)	9(t(-1+3t-6p)+p)	
12	54upt	54p(t-u)	-54up	54u(t-p)	
13	9up(-1+3u)	9p(-1+6u)	0	9u(-1+3u)	
14	27uvp/c	27vp/c	27up/c	27uv/c	
15	9vp(-1+3v)	0	9p(-1+3v)	9v(-1+3v)	
16	54vpt	-54vp	54p(t-v)	54v(t-p)	
17	9tp(-1+3p)	-9p(-1+3p)	-9p(-1+3p)	9(p(1+6t-3p)-t)	
18	9up(-1+3p)	9p(-1+3p)	0	9u(-1+6p)	
19	9vp(-1+3p)	0	9p(-1+3p)	9v(-1+6p)	
20	p(-1+3p)(-2+3p)	0	0	(2-18p+27p²)	

Tabela 5.9 - Funções de forma e suas derivadas, do elemento Tetraédrico Cúbico $t = 1 - u - v - p$

5.7 – Elementos Hexaédricos(três-dimensões)

5.7.1 – Elemento Trilinear(hexaédro, oito nós, C^0)

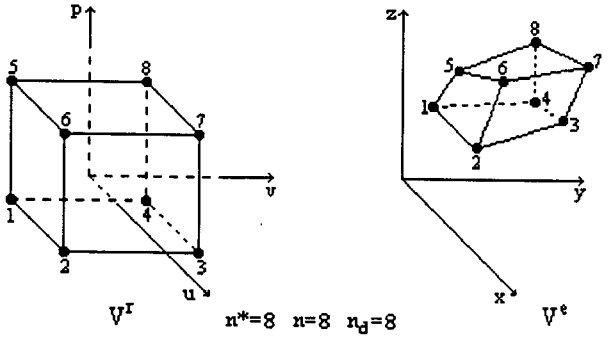


Figura 5.15 - Elemento Hexaédrico Trilinear

Este elemento é isoparamétrico, e a sua base polinomial é:

$$<P> = < \ 1 \ \ u \ \ v \ \ p \ \ uv \ \ vp \ \ up \ \ uvp \ > \qquad (5.28)$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.10:

Nó	c.{N}	c.{∂N/∂u}	c.{∂N/∂v}	c.{∂N/∂p}	c
1	(1-u)(1-v)(1-p)	-(1-v)(1-p)	-(1-u)(1-p)	-(1-u)(1-v)	
2	(1+u)(1-v)(1-p)	(1-v)(1-p)	-(1+u)(1-p)	-(1+u)(1-v)	
3	(1+u)(1+v)(1-p)	(1+v)(1-p)	(1+u)(1-p)	-(1+u)(1+v)	
4	(1-u)(1+v)(1-p)	-(1+v)(1-p)	(1-u)(1-p)	-(1-u)(1+v)	$\frac{1}{8}$
5	(1-u)(1-v)(1+p)	-(1-v)(1+p)	-(1-u)(1+p)	(1-u)(1-v)	
6	(1+u)(1-v)(1+p)	(1-v)(1+p)	-(1+u)(1+p)	(1+u)(1-v)	
7	(1+u)(1+v)(1+p)	(1+v)(1+p)	(1+u)(1+p)	(1+u)(1+v)	
8	(1-u)(1+v)(1+p)	-(1+v)(1+p)	(1-u)(1+p)	(1-u)(1+v)	

Tabela 5.10 - Funções de forma e suas derivadas, do elemento Hexaédrico Trilinear

5.7.2 – Elementos Lagrangianos de Ordem Superior(C^0)

5.7.2.1 – Quadrático Incompleto(hexaédro, vinte nós, C^0)

Este elemento é muito popular, e possue o seguinte mapeamento geométrico isoparamétrico:

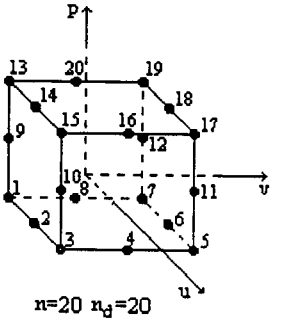


Figura 5.16 - Elemento Hexaédrico Quadrátrico Incompleto

A base polinomial é:

$$<P> = < \quad 1 \quad u \quad v \quad p \quad u^2 \quad uv \quad v^2 \quad vp \quad p^2 \quad up \quad u^2v \quad uv^2 \quad v^2p \quad vp^2 \quad up^2$$
$$u^2p \quad uvp \quad u^2vp \quad uv^2p \quad uvp^2 \quad > \qquad (5.29)$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.11:

Nó	{N}	{∂N/∂u}
1	(1-u)(1-v)(1-p)(-2-u-v-p)/8	-(1-v)(1-p)(-1-2u-v-p)/8
2	$(1-u^2)$(1-v)(1-p)/4	-u(1-v)(1-p)/2
3	(1+u)(1-v)(1-p)(-2+u-v-p)/8	(1-v)(1-p)(-1+2u-v-p)/8
4	(1+u)$(1-v^2)$(1-p)/4	$(1-v^2)$(1-p)/4
5	(1+u)(1+v)(1-p)(-2+u+v-p)/8	(1+v)(1-p)(-1+2u+v-p)/8
6	$(1-u^2)$(1+v)(1-p)/4	-u(1+v)(1-p)/2
7	(1-u)(1+v)(1-p)(-2-u+v-p)/8	-(1+v)(1-p)(-1-2u+v-p)/8
8	(1-u)$(1-v^2)$(1-p)/4	-$(1-v^2)$(1-p)/4
9	(1-u)$(1-p^2)$(1-v)/4	-(1-v)$(1-p^2)$/4
10	(1+u)$(1-p^2)$(1-v)/4	(1-v)$(1-p^2)$/4
11	(1+u)$(1-p^2)$(1+v)/4	(1+v)$(1-p^2)$/4
12	(1-u)$(1-p^2)$(1+v)/4	-(1+v)$(1-p^2)$/4
13	(1-u)(1-v)(1+p)(-2-u-v+p)/8	-(1-v)(1+p)(-1-2u-v+p)/8
14	$(1-u^2)$(1-v)(1+p)/4	-u(1-v)(1+p)/2
15	(1+u)(1-v)(1+p)(-2+u-v+p)/8	(1-v)(1+p)(-1+2u-v+p)/8
16	(1+u)$(1-v^2)$(1+p)/4	$(1-v^2)$(1+p)/4
17	(1+u)(1+v)(1+p)(-2+u+v+p)/8	(1+v)(1+p)(-1+2u+v+p)/8
18	$(1-u^2)$(1+v)(1+p)/4	-u(1+v)(1+p)/2
19	(1-u)(1+v)(1+p)(-2-u+v+p)/8	-(1+v)(1+p)(-1-2u+v+p)/8
20	(1-u)$(1-v^2)$(1+p)/4	-$(1-v^2)$(1+p)/4

Nó	{∂N/∂p}	{∂N/∂v}
1	-(1-u)(1-p)(-1-u-2v-p)/8	-(1-u)(1-v)(-1-u-v-2p)/8
2	-$(1-u^2)$(1-p)/4	-$(1-u^2)$(1-v)/4
3	-(1+u)(1-p)(-1+u-2v-p)/8	-(1+u)(1-v)(-1+u-v-2p)/8
4	-v(1+u)(1-p)/2	-(1+u)$(1-v^2)$/4
5	(1+u)(1-p)(-1+u+2v-p)/8	-(1+u)(1+v)(-1+u+v-2p)/8
6	$(1-u^2)$(1-p)/4	-$(1-u^2)$(1+v)/4
7	(1-u)(1-p)(-1-u+2v-p)/8	-(1-u)(1+v)(-1-u+v-2p)/8
8	-v(1-u)(1-p)/2	-(1-u)$(1-v^2)$/4
9	-(1-u)$(1-p^2)$/4	-p(1-u)(1-v)/2
10	-(1+u)$(1-p^2)$/4	-p(1+u)(1-v)/2
11	(1+u)$(1-p^2)$/4	-p(1+u)(1+v)/2
12	(1-u)$(1-p^2)$/4	-p(1-u)(1+v)/2
13	-(1-u)(1+p)(-1-u-2v+p)/8	(1-u)(1-v)(-1-u-v+2p)/8
14	-$(1-u^2)$(1+p)/4	$(1-u^2)$(1-v)/4
15	-(1+u)(1+p)(-1+u-2v+p)/8	(1+u)(1-v)(-1+u-v+2p)/8
16	-v(1+u)(1+p)/2	$(1-v^2)$(1+u)/4
17	(1+u)(1+p)(-1+u+2v+p)/8	(1+u)(1+v)(-1+u+v+2p)/8
18	$(1-u^2)$(1+p)/4	$(1-u^2)$(1+v)/4
19	(1-u)(1+p)(-1-u+2v+p)/8	(1-u)(1+v)(-1-u+v+2p)/8
20	-v(1-u)(1+p)/2	(1-u)$(1-v^2)$/4

Tabela 5.11 - Funções de Forma e suas derivadas, do elemento Hexaédrico Quadrático Incompleto

5.7.2.2 – Cúbico Incompleto(hexaédro, trinta e dois nós, C^0)

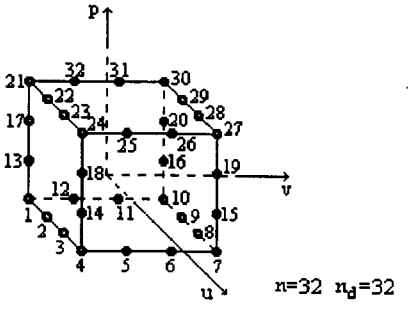


Figura 5.17 - Elemento Hexaédrico Cúbico Incompleto

A base polinomial é:

$$<P> = < \quad 1 \quad u \quad v \quad p \quad u^2 \quad uv \quad v^2 \quad vp \quad p^2 \quad up \quad u^2v \quad uv^2 \quad v^2p \quad vp^2 \quad up^2$$
$$u^2p \quad uvp \quad u^2vp \quad uv^2p \quad uvp^2 \; > \qquad (5.30)$$

As funções de Transformação Geométrica (idênticas as Funções de Interpolação), bem como suas derivadas, são dadas na Tabela 5.12:

Nó	{N}	{∂N/∂u}
1	(1-u)(1-v)(1-p)(-b+u²+v²+p²)a	(1-v)(1-p)(b-3u²-v²-p²+2u)a
2	(1-u²)(d-eu)(1-v)(1-p)c	(1-v)(1-p)(-e-(2/9)u+u²)c
3	(1-u²)(d+eu)(1-v)(1-p)c	(1-v)(1-p)(e-(2/9)u+u²)c
4	(1+u)(1-v)(1-p)(-b+u²+v²+p²)a	(1-v)(1-p)(-b+3u²+v²+p²+2u)a
5	(1+u)(1-v²)(d-ev)(1-p)c	(1-v²)(d-ev)(1-p)c
6	(1+u)(1-v²)(d+ev)(1-p)c	(1-v²)(d+ev)(1-p)c
7	(1+u)(1+v)(1-p)(-b+u²+v²+p²)a	(1+v)(1-p)(-b+3u²+v²+p²+2u)a
8	(1-u²)(d+eu)(1+v)(1-p)c	(1+v)(1-p)(e-(2/9)u-u²)c
9	(1-u²)(d-eu)(1+v)(1-p)c	(1+v)(1-p)(-e-(2/9)u+u²)c
10	(1-u)(1+v)(1-p)(-b+u²+v²+p²)a	(1+v)(1-p)(b-3u²-v²-p²+2u)a
11	(1-u)(1-v²)(d+ev)(1-p)c	-(1-v²)(d+ev)(1-p)c
12	(1-u)(1-v²)(d-ev)(1-p)c	-(1-v²)(d-ev)(1-p)c
13	(1-u)(1-v)(1-p²)(d-ep)c	-(1-v)(1-p²)(d-ep)c
14	(1+u)(1-v)(1-p²)(d-ep)c	(1-v)(1-p²)(d-ep)c
15	(1+u)(1+v)(1-p²)(d-ep)c	(1+v)(1-p²)(d-ep)c
16	(1-u)(1+v)(1-p²)(d-ep)c	-(1+v)(1-p²)(d-ep)c
17	(1-u)(1-v)(1-p²)(d+ep)c	-(1-v)(1-p²)(d+ep)c
18	(1+u)(1-v)(1-p²)(d+ep)c	(1-v)(1-p²)(d+ep)c
19	(1+u)(1+v)(1-p²)(d+ep)c	(1+v)(1-p²)(d+ep)c
20	(1-u)(1+v)(1-p²)(d+ep)c	-(1+v)(1-p²)(d+ep)c
21	(1-u)(1-v)(1+p)(-b+u²+v²+p²)a	(1-v)(1+p)(b-3u²-v²-p²+2u)a
22	(1-u²)(1-v)(1+p)(d-eu)c	(1-v)(1+p)(-e-(2/9)u+u²)c

a = 9/64
b = 19/9
c = 81/64
d = 1/9
e = 1/3

Nó	{N}	{∂N/∂u}
23	$(1-u^2)(1-v)(1+p)(d+eu)c$	$(1-v)(1+p)(e-(2/9)u-u^2)c$
24	$(1+u)(1-v)(1+p)(-b+u^2+v^2+p^2)a$	$(1-v)(1+p)(-b+3u^2+v^2+p^2+2u)a$
25	$(1+u)(1-v^2)(1+p)(d-ev)c$	$(1-v^2)(1+p)(d-ev)c$
26	$(1+u)(1-v^2)(1+p)(d+ev)c$	$(1-v^2)(1+p)(d+ev)c$
27	$(1+u)(1+v)(1+p)(-b+u^2+v^2+p^2)a$	$(1+v)(1+p)(-b+3u^2+v^2+p^2+2u)a$
28	$(1-u^2)(1+v)(1+p)(d+eu)c$	$(1+v)(1+p)(e-(2/9)u-u^2)c$
29	$(1-u^2)(1+v)(1+p)(d-eu)c$	$(1+v)(1+p)(-e-(2/9)u+u^2)c$
30	$(1-u)(1+v)(1+p)(-b+u^2+v^2+p^2)a$	$(1+v)(1+p)(b-3u^2-v^2-p^2+2u)a$
31	$(1-u)(1-v^2)(1+p)(d+ev)c$	$-(1-v^2)(1+p)(d+ev)c$
32	$(1-u)(1-v^2)(1+p)(d-ev)c$	$-(1-v^2)(1+p)(d-ev)c$

Nó	{∂N/∂p}	{∂N/∂v}
1	$(1-u)(1-p)(b-u^2-3v^2-p^2+2v)a$	$(1-u)(1-v)(b-u^2-v^2-3p^2+2p)a$
2	$-(1-u^2)(1-p)(d-ev)c$	$-(1-u^2)(1-v)(d-eu)c$
3	$-(1-u^2)(1-p)(d+ev)c$	$-(1-u^2)(1-v)(d+eu)c$
4	$(1+u)(1-p)(b-u^2-3v^2-p^2+2v)a$	$(1+u)(1-v)(b-u^2-v^2-3p^2+2p)a$
5	$(1+u)(1-p)(-e-(2/9)v+v^2)c$	$-(1+u)(1-v^2)(d-ev)c$
6	$(1+u)(1-p)(e-(2/9)v-v^2)c$	$-(1+u)(1-v^2)(d+ev)c$
7	$(1+u)(1-p)(-b+u^2+3v^2+p^2+2v)a$	$(1+u)(1+v)(b-u^2-v^2-3p^2+2p)a$
8	$(1-u^2)(1-p)(d+ev)c$	$-(1-u^2)(1+v)(d+eu)c$
9	$(1-u^2)(1-p)(d-ev)c$	$-(1-u^2)(1+v)(d-eu)c$
10	$(1-u)(1-p)(-b+u^2+3v^2+p^2+2v)a$	$(1-u)(1+v)(b-u^2-v^2-3p^2+2p)a$
11	$(1-u)(1-p)(e-(2/9)v-v^2)c$	$-(1-u)(1-v^2)(d+ev)c$
12	$(1-u)(1-p)(-e-(2/9)v+v^2)c$	$-(1-u)(1-v^2)(d-ev)c$
13	$-(1-u)(1-p^2)(d-ep)c$	$(1-u)(1-v)(-e-(2/9)p+p^2)c$
14	$-(1+u)(1-p^2)(d-ep)c$	$(1+u)(1-v)(-e-(2/9)p+p^2)c$
15	$(1+u)(1-p^2)(d-ep)c$	$(1+u)(1+v)(-e-(2/9)p+p^2)c$
16	$(1-u)(1-p^2)(d-ep)c$	$(1-u)(1+v)(-e-(2/9)p+p^2)c$
17	$-(1-u)(1-p^2)(d+ep)c$	$(1-u)(1-v)(e-(2/9)p-p^2)c$
18	$-(1+u)(1-p^2)(d+ep)c$	$(1+u)(1-v)(e-(2/9)p-p^2)c$
19	$(1+u)(1-p^2)(d+ep)c$	$(1+u)(1+v)(e-(2/9)p-p^2)c$
20	$(1-u)(1-p^2)(d+ep)c$	$(1-u)(1+v)(e-(2/9)p-p^2)c$
21	$(1-u)(1+p)(b-u^2-3v^2-p^2+2v)a$	$(1-u)(1-v)(-b+u^2+v^2+3p^2+2p)a$
22	$-(1-u^2)(1+p)(d-ev)c$	$(1-u^2)(1-v)(d-eu)c$
23	$-(1-u^2)(1+p)(d+ev)c$	$(1-u^2)(1-v)(d+eu)c$
24	$(1+u)(1+p)(b-u^2-3v^2-p^2+2v)a$	$(1+u)(1-v)(-b+u^2+v^2+3p^2+2p)a$
25	$(1+u)(1+p)(-e-(2/9)v+v^2)c$	$(1+u)(1-v^2)(d-ev)c$
26	$(1+u)(1+p)(e-(2/9)v-v^2)c$	$(1+u)(1-v^2)(d+ev)c$
27	$(1+u)(1+p)(-b+u^2+3v^2+p^2+2v)a$	$(1+u)(1+v)(-b+u^2+v^2+3p^2+2p)a$
28	$(1-u^2)(1+p)(d+ev)c$	$(1-u^2)(1+v)(d+eu)c$
29	$(1-u^2)(1+p)(d-ev)c$	$(1-u^2)(1+v)(d-eu)c$
30	$(1-u)(1+p)(-b+u^2+3v^2+p^2+2v)a$	$(1-u)(1+v)(-b+u^2+v^2+3p^2+2p)a$
31	$(1-u)(1+p)(e-(2/9)v-v^2)c$	$(1-u)(1-v^2)(d+ev)c$
32	$(1-u)(1+p)(-e-(2/9)v+v^2)c$	$(1-u)(1-v^2)(d-ev)c$

a = 9/64
b = 19/9
c = 81/64
d = 1/9
e = 1/3

Tabela 5.12 - Funções de Forma e suas derivadas, do elemento Hexaédrico Cúbico Incompleto

Capítulo 6 – DESCRIÇÃO DAS BIBLIOTECAS

6.1 – Introdução

Este capítulo tem o objetivo de descrever as subrotinas utilizadas na formação de bibliotecas contendo as funções de forma nodais (ou de interpolação) e os pontos de integração, para diferentes elementos finitos, em duas e três dimensões. Esta descrição consiste na declaração dos nomes e dos argumentos utilizados em cada subrotina, e como utilizá-las num programa de cálculo de campos, e ainda dar uma idéia geral da estrutura das subrotinas, através de um algoritmo. Exemplos de programas para testes das subrotinas, para verificar a validade das funções também serão abordados.

6.2 – Nomenclatura e Declaração dos Argumentos

Neste parágrafo será feita uma descrição da nomenclatura e declaração dos argumentos utilizados em cada subrotina.

6.2.1 – Elementos em Duas Dimensões

6.2.1.1 – Funções de Forma (ou de Interpolação)

(a) Elemento Triangular:

FNTL2(argumentos):
FN - Função de Forma Nodal
TL - elemento Triangular Linear
2 - Caso bidimensional

FNTQ2(argumentos):
FN - Função de Forma Nodal
TQ - elemento Triangular Quadrático
2 - Caso bidimensional

FNTC2(argumentos):
FN - Função de Forma Nodal
TC - elemento Triangular Cúbico
2 - Caso bidimensional

(b) Elemento Quadrilátero

FNQB2(argumentos):

FN - Função de Forma Nodal

QB - elemento Quadrilátero Bilinear

2 - Caso bidimensional

FNQQ2(argumentos):

FN - Função de Forma Nodal

QQ - elemento Quadrilátero Quadrático

2 - Caso bidimensional

FNQC2(argumentos):

FN - Função de Forma Nodal

QC - elemento Quadrilátero Cúbico

2 - Caso bidimensional

Argumentos: (u,v,xr,yr,fn,dnx,dny,detj)

- U e V: pontos de integração;
- Xr e Yr: coordenadas nodais do elemento de referência;
- Fn: valor da função de forma nodal;
- Dnx e Dny: valores das derivadas de Fn em relação a x e y;
- Detj: valor da determinante da Matriz Jacobiana.

Os argumentos descritos acima, são válidos para os seis casos mencionados anteriormente.

6.2.1.2 – Pontos de Integração

(a) Elemento Triangular

PITL2(argumentos):

PI - Pontos de Integração

TL - elemento Triangular Linear

2 - Caso bidimensional

PITQ2(argumentos):

PI - Pontos de Integração

TQ - elemento Triangular Quadrático

2 - Caso bidimensional

PITC2(argumentos):

PI - Pontos de Integração

TC - elemento Triangular Cúbico

2 - Caso bidimensional

(b) Elemento Quadrilátero

PIQB2(argumentos):

PI - Pontos de Integração

QB - elemento Quadrilátero Bilinear

2 - Caso bidimensional

PIQQ2(argumentos):

PI - Pontos de Integração

QQ - elemento Quadrilátero Quadrático

2 - Caso bidimensional

PIQC2(argumentos):

PI - Pontos de Integração

QC - elemento Quadrilátero Cúbico

2 - Caso bidimensional

Argumentos: (npi,ui,vi,wi)

- Npi: Número de Pontos de Integração
- Ui e Vi: Pontos de Integração
- Wi: peso

Os argumentos descritos acima, são válidos para os seis casos mencionados anteriormente.

6.2.2 – Elementos em Três Dimensões

6.2.2.1 – Funções de Forma (ou de Interpolação)

(a) Elemento Tetraédrico:

FNTL3(argumentos):
FN - Função de Forma Nodal
TL - elemento Tetraédrico Linear
3 - Caso tridimensional

FNTQ3(argumentos):
FN - Função de Forma Nodal
TQ - elemento Tetraédrico Quadrático
3 - Caso tridimensional

FNTC3(argumentos):
FN - Função de Forma Nodal
TC - elemento Tetraédrico Cúbico
3 - Caso tridimensional

(b) Elemento Hexaédrico:

FNHT3(argumentos):
FN - Função de Forma Nodal
HT - elemento Hexaédrico Trilinear
3 - Caso tridimensional

FNHQ3(argumentos):
FN - Função de Forma Nodal
HQ - elemento Hexaédrico Quadrático
3 - Caso tridimensional

FNHC3(argumentos):
FN - Função de Forma Nodal
HC - elemento Hexaédrico Cúbico
3 - Caso tridimensional

Argumentos: (u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)

- U, V e P: pontos de integração;
- Xr, Yr e Zr: coordenadas nodais do elemento de referência;
- Fn: valor da função de forma nodal;

- Dnx, Dny e Dnz: valores das derivadas de Fn em relação a x, y e z;
- Detj: valor da determinante da Matriz Jacobiana.

Os argumentos descritos acima, são válidos para os seis casos mencionados anteriormente.

6.2.2.2 – Pontos de Integração

(a) Elemento Tetraédrico:

PITL3(argumentos):
PI - Pontos de Integração
TL - elemento Tetraédrico Linear
3 - Caso tridimensional

PITQ3(argumentos):
PI - Pontos de Integração
TQ - elemento Tetraédrico Quadrático
3 - Caso tridimensional

PITC3(argumentos):
PI - Pontos de Integração
TC - elemento Tetraédrico Cúbico
3 - Caso tridimensional

(b) Elemento Hexaédrico:

PIHT3(argumentos):
PI - Pontos de Integração
HT - elemento Hexaédrico Trilinear
3 - Caso tridimensional

PIHQ3(argumentos):
PI - Pontos de Integração
HQ - elemento Hexaédrico Quadrático
3 - Caso tridimensional

PIHC3(argumentos):
PI - Pontos de Integração
HC - elemento Hexaédrico Cúbico
3 - Caso tridimensional

Argumentos: (npi,ui,vi,pi,wi)

- Npi: Número de Pontos de Integração
- Ui, Vi e Pi: Pontos de Integração
- Wi: peso

Os argumentos descritos acima, são válidos para os seis casos mencionados anteriormente.

6.3 – Apresentação das Bibliotecas

Inicialmente, as subrotinas foram implementadas para programas que utilizam a linguagem FORTRAN. Assim, para as bibliotecas em duas e três dimensões, as subrotinas apresentam a seguinte forma:

- para as funções de forma nodais (ou de interpolação):

SUBROUTINE FN...(ARGUMENTOS)

- para os pontos de integração:

SUBROUTINE PI.....(ARGUMENTOS)

A forma de chamada para cada subrotina é feita da seguinte maneira:

- para as funções de forma nodais (ou de interpolação):

CALL FN...(ARGUMENTOS)

- para os pontos de integração:

CALL PI.....(ARGUMENTOS)

este comando pode aparecer na estrutura do programa principal, ou dentro de um módulo (sub-programa).

Normalmente, a chamada é feita na etapa de formação do elemento, onde para cada elemento a chamada da subrotina correspondente é repetida *npi* vezes, obtendo-se assim, a matriz rigidez ou contribuição ($S_{NKTxNKT}$) de acordo com o número de nós (*nkt*) de cada elemento. Após, este processo ser repetido para todos os elementos, a matriz rigidez é condensada em um sistema matricial global (SS). Após a inserção das condições de contorno , um sistema matricial do tipo: [SS][V] = [VDR], onde [V] é a matriz das variáveis incógnitas,

e [VDR] é a matriz contendo as condições de contorno, é resolvido, obtendo-se assim as variáveis incógnitas, como por exemplo, os potenciais em cada nó.

7.3.1 – Estrutura das Subrotinas

Em geral as subrotinas contendo as funções de forma nodais (ou de interpolação) são semelhantes, isto é, possuem uma mesma estrutura modular. O que as diferencia é o módulo onde estão declaradas as funções de forma e suas derivadas, uma vez que cada elemento possue o seu próprio equacionamento.

Abaixo tem-se um algoritmo das subrotinas contendo as funções de forma nodais:

PROGRAMA PRINCIPAL
SUBROTINA FN...(ARGUMENTOS) Declaração das Variáveis 1 - Chamada da subrotina contendo as funções de forma nodais [N] e suas derivadas em relação a u e v (2D), ou u, v e p (3D). 2 - Chamada da subrotina de cálculo do determinante da matriz Jacobiana. 3 - Chamada da subrotina de cálculo das derivadas das funções de forma nodais em relação a x e y (2D), ou x, y e z (3D). Fim

Figura 6.1 - Algoritmo das subrotinas contendo as funções de forma

As subrotinas contendo os pontos de integração possuem uma estrutura mais simples, isto é, não são subdivididas em módulos, uma vez que possuem apenas dados numéricos.

6.3.2 – Teste das Subrotinas

Para verificar a validade e observar se não haviam erros, todas as subrotinas foram testadas em um programa baseado no módulo EFCS, cujo objetivo é efetuar cálculos estáticos via elementos finitos. Este software é uma das etapas de processamento que compõem o sistema EFCAD. Assim, um programa mais simples foi implementado. As alterações mais significantes foram feitas na forma de entrada de dados e, para alguns casos, a malha foi feita manualmente. Na etapa de formação do elemento, foram incluídas as subrotinas contendo as funções de forma e os pontos de integração. A saída, mostra apenas os potenciais em cada nó do elemento. As etapas principais como, condensação da matriz global, inserção das condições de contorno e resolução do sistema matricial não foram alteradas, a fim de manter a maior semelhança possível com o programa original, e assim servir como modelo para uso mais aprofundado. Para um melhor entendimento do funcionamento do programa sugere-se ver a referência [8]. Abaixo, um algoritmo do programa EFCS simplificado:

PROGRAMA PRINCIPAL
Declaração das Variáveis Inicio Chama Subrotina DATAIN para entrada de dados Chama Subrotina BAND para o calculo da Matriz Banda Chama Subrotina ZERO para Inicialização das Matrizes Chama Subrotina FORM para Formação dos Elementos e da Matriz Global SS Chama Subrotina COFRON para Inserção das Condições de Contorno (VDR) Chama subrotina RESOL para Resolução do Sistema Matricial [vv]=[vdr]/[ss] Chama subrotina SAIDA para impressão dos resultados Fim

Figura 6.2 – Algoritmo do programa EFCS simplificado

O algoritmo mostrado anteriormente é válido para duas e três dimensões, diferenciando apenas para o caso em três dimensões a inclusão das variáveis relacionadas ao eixo Z.

Exemplos dos programas em duas e três dimensões estão em ANEXO.

Com o auxílio dos programas citados anteriormente, foram realizados testes com todos os elementos em duas e três dimensões. Para os testes em 2D foi implementado um programa malhador que permite a escolha o tipo de elemento. A arquivo gerado por este malhador será o arquivo de entrada para o programa de testes em 2D. Este arquivo de entrada contêm dados sobre o tipo de elemento, número de nós de cada elemento, número total de nós, número dos nós que constituem determinado elemento, coordenadas dos nós, potenciais impostos, tipo de material, etc. Para os testes em 3D o arquivo de entrada foi feito manualmente com as mesmas características do arquivo 2D. No momento da simulação, tanto o programa de testes em 2D quanto em 3D são capazes de identificar o tipo de elemento e fornecer um arquivo de saída contendo os potenciais em cada nó e o campo elétrico de cada elemento. Em ANEXO encontram-se alguns resultados dos referidos programas. Em todos os testes realizados, os resultados obtidos foram satisfatórios.

As bibliotecas contendo as funções de forma em 2D e 3D, as bibliotecas com os pontos de integração em 2D e 3D, e todos os resultados obtidos dos programas 2D e 3D, encontram-se na Biblioteca do GRUCAD(Grupo de Concepção e Análise de Dispositivos Eletromagnéticos), ou ainda em arquivos nos computadores do referido Laboratório.

Capítulo 7 – CONCLUSÃO

7.1 – Sobre as Bibliotecas

Na resolução de problemas pelo método de elementos finitos, aplicando-se por exemplo o Método de Resíduos Ponderados de Galerkin, as funções de forma ou de interpolação exercem um papel fundamental no processo de cálculo das grandezas eletromagnéticas, como por exemplo, a distribuição do campo elétrico $\vec{E}$ e do potencial V, em aplicações envolvendo dispositivos físicos. A grande vantagem deste método está no fato de que a formulação numérica pode ser obtida diretamente das equações que definem o fenômeno físico tratado.

Contudo, é necessário um maior desenvolvimento nos estudos das funções de forma (ou de interpolação), adaptando de forma mais precisa o comportamento das grandezas dentro dos elementos. Na implementação dos programas utilizados nessa dissertação para verificação da validade das bibliotecas, foram feitas simplificações em relação ao programa original (EFCS), tal que os mesmos funcionam apenas para o caso linear, potencial escalar em 2D e 3D. Para implementação de um programa com as mesmas características do software EFCS, é necessário levar-se em consideração aspectos como a repetição geométrica de um domínio onde as fontes de campo se encontram no mesmo sentido dentro do domínio (caso de periodicidade), ou em sentidos opostos (caso de anti-periodicidade); casos que possuem partes móveis na estrutura, onde apareceria um termo relativo à velocidade; problemas que possuem uma simetria de resolução (axi-simétricos) e ainda problemas não-lineares, em que a permeabilidade μ ou a relutividade ν dos materiais ferromagnéticos são caracterizados por uma curva B(H) e portanto para cada valor de H(ou B) tem-se um valor diferente de μ ou ν. Porém, esta generalização não é o objetivo deste trabalho.

Com relação a padronização das subrotinas, principal objetivo desta dissertação, as mesmas resultaram numa estrutura simples e objetiva. Os programas implementados nessa dissertação servem como referência para correta colocação das mesmas (indicam a etapa do programa em que devem ser feitas as chamadas das subrotinas contendo as funções de forma), uma vez que estes programas apresentam subrotinas semelhantes ao módulo original. Ficará a

cargo do programador, adaptar a subrotina de entrada e de saída de resultados ao tipo de elemento, uma vez que a etapa de cálculo está indicada.

7.2 – Trabalhos Futuros

Para trabalhos futuros propõem-se uma continuação da implementação de uma biblioteca de funções de forma que atendam as exigências citadas anteriormente a fim de se manter as mesmas caracteríticas apresentadas pelo software EFCS (etapa de processamento), para casos em duas e três dimensões, lineares e não-lineares, utilizando-se Elementos Finitos e Elementos de Arestas. Além disso, o estudo requer um aprimoramento nos sistemas geradores de malhas atuais (etapa de pré-processamento), com desafios significativos no domínio 3D. Os resultados obtidos serão implementados nos sistemas FEECAD e EFCAD de Cálculo de Campos Eletromagnéticos.

Essa nova biblioteca também terá o intuito de padronizar e auxiliar o programador de softwares que utilizam o método de elementos finitos na elaboração da etapa de cálculo do programa (processador). Seria interessante efetuar implementações utilizando-se novas linguagens computacionais no ambiente Windows, como por exemplo a linguagem Fortran 90 que é uma evolução do Fortran 77, e que apresenta compatibilidade entre os sistemas UNIX e Windows.

REFERÊNCIA

[1] **Dahtt, G. and Touzot,G.,** *The Finite Element Method Displayed*, John Wiley & Sons, California,1985.

[2] **Zienkiewicz, O. C.,** *La Methods des elements finits* (translated from the English), Pluralis, France, 1976.

[3] **Gallagher, R. H.,** *Introduction aux elements finis*(translated from the English by J.L. Claudon), Pluralis, France, 1976.

[4] **Rockey, K.C., Evans,H.R., Griffiths, D.W. and Nethercot, D.A.,** *Elements finis*, (translated from the English by C. Gomes), Eyrolles, France, 1979.

[5] **Absi, E.,** *Methode de calcul numérique en elaticité*, Eyrolles, France, 1978.

[6] **Imbert, J. F.,** *Analyse des structures par éléments finis*, Cepadues Ed., France, 1979.

[7] **Strang, G. and Fix, O.J.,** *Analysis of the Finite Element Methods*, Prentice-Hall, New Jersey,1973.

[8] **Assumpção Bastos, J. P.,** *Eletromagnestismo e Cálculo de Campos*, Ed. UFSC, Brasil, 1992.

1 – PROGRAMA PARA APLICAÇÃO DAS FUNÇÕES 2D

```
C==============================09/9/97==================================C
c  OBSERVACAO: ESTE PROGRAMA ESTA TRABALHANDO APENAS COM POTENCIAL
c           ESCALAR  2D.
c           A entrada de dados e feita por um malhador regular simples. A etapa de calculo é feita
c           utilizando-se  as funções de forma de acordo com o tipo de elemento.
c
c           Este programa utiliza as seguintes subrotinas do programa EFCS:
c                           - ZERO: zerar variáveis do sistema(condição inicial)
c                           - BAND: calculo da matriz banda
c                           - SSFORM: condensar a matriz
c                           - COFRON:adicionar as condições de contorno
c                           - RESOL: resolver o sistema matricial
c
c         Subrotinas adicionadas para o cálculo via Método dos Resíduos:
c         - PITL2, PITQ2, PITC2:  pontos de integração para os diferentes elementos triangulares  2D
c         - PIQB2, PIQQ2, PIQC2:  pontos de integração para os diferentes elementos quadriláteros 2D
c         - FNTL2,FNTQ2,FNTC2:  funções de forma para os diferentes elementos triangulares 2D
c         - FNQB2,FNQQ2,FNQC2:  funções de forma para os diferentes elementos quadriláteros 2D
c         As subrotinas anteriores permitem escolhas de acordo com o tipo  de elemento.
C=======================================================================C
c NOMENCLATURA:
c  * detj - determinante do jacobiano
c  * dn() - derivada de fn em relação a u e v
c  * dnx,dny- derivada de fn em relação a x e y
c  * ee     - matriz rigidez [=S(nkt,nkt)]
c  * fn() - funcao de forma nodal
c  * fo(,)  - arranjo contendo os nos com condição de contorno
c  * iaxe   - define o tipo de eixo(0:cartesiano ou 1:axi-simetrico)
c  * icterm -termo relacionado com as fontes de aquecimento
c  * ikind  - define o tipo de potencial(0:Escalar ou 1:Vetor)
c  * irep   - numero de pares de nos periódicos da malha
c          =0> nao ha periodicidade ou =1> ha period. no domínio
c  * iterm  - valor de uma constante (iterm=0)
c  * ktri(,)-guarda o numero do meio do elemento
c  * mat()  - numero do meio em cada um dos NEL elementos
c  * mb   - matriz banda
c  * mbb  - Largura de banda
c  * mrep(,)- guarda o arranjo com os pares de nos periódicos
c  * nmat   - numero de meios dielétricos
c  * nno  - numero de nos
c  * nel  - numero de elementos
c  * nfron- numero de condições de fronteira
c  * nkt  - numero de nos de acordo com tipo de elemento
c  * nonper - =0>caso periódico ou =1> caso anti-periódico
c  * npi - número de pontos de integração
c  * perm() - permissividade dos NMAT meios
c  * ro()   - densidade superficial de carga
c  * ss()   - matriz global
c  * u,v  - pontos de integração
c  * ui,vi  - pontos de integração
```

```
c  * vvi() - potencial imposto na fronteira
c  * vdr() - vetor lado direito
c  * vv()  - potencial vetor incógnito
c  * xcor() - coordenada x do no
c  * xr,yr - coordenadas dos elementos no espaço de referencia
c  * xmu=perm()- permissividade dos NMAT meios
c  * ycor() - coordenada y do no
c  * wi    - peso
C=====================================================================C
      C------PROGRAMA PRINCIPAL
      parameter (jjfron=250,jel=10000,jrep=400,jno=10000,jfron=150)
      parameter (jnband=1000000)
      integer nno,nel,nfron,irep,nonper,ikind,iaxe,mb,mat(jno),nkt,nmat
      integer icdterm(jjfron),ktri(jel,12),iterm,fo(jjfron,jfron),ncle
      double precision xcor(jno),ycor(jno),vvi(jjfron),perm(30),ro(jno)
      double precision mrep(jrep,2),vdr(jno),ss(jnband),vv(jno)

      write(*,*)' '
      write(*,*)' '
      write(6,'(a$)')' ATENCAO!!!'
      write(6,'(a$)')' ESTE PROGRAMA ESTA OPERANDO APENAS COM'
      write(6,'(a$)')' POTENCIAL ESCALAR 2D'
      write(*,*)' '
C******-Chama DATAIN para entra de dados
      CALL DATAIN(nno,nel,nkt,nfron,ktri,xcor,ycor,vvi,fo,mat,ro,perm,
     *          nmat)

c------Chama BAND para o calculo da Matriz Banda
      CALL BAND(nel,nno,ktri,mb,nkt)

C******-Chama Subrotina ZERO para Inicialização das Matrizes
      CALL ZERO(nno,mb,vdr,vv,ss,mrep)

c-------Chama Subrotina FORM para Formação da Matriz Global SS
      CALL FORM(nel,irep,ikind,iaxe,nonper,mb,mrep,xcor,ycor,ktri,ss,
     *  nno,mat,ro,perm,nkt)

C***-Chama Subrotina COFRON para Inserção das Condições de Contorno VDR
      CALL COFRON(nfron,nno,iaxe,ikind,iterm,YCOR,MB,ICTERM,FO,VDR,
     *   VVI,SS)

C***-Chama subrotina RESOL para Resolução do Sistema Matricial
c                [vv]=[vdr]/[ss]
      CALL RESOL(nno,mb,irep,nfron,iterm,SS,VDR,FO,VVI,VV)

c------Chama subrotina SAIDA para impressão dos resultados
      CALL SAIDA(nno,nel,ikind,vv,vdr,nkt,ktri,xcor,ycor)
      STOP
      END
C=====================================================================C
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine DATAIN(nno,nel,nkt,nfron,ktri,xcor,
ycor,vvi,fo,mat,ro,perm,nmat)
character namfs*20, namfu(20)
equivalence (namfs,namfu(1))
parameter (jpont=150,JFRON=150,jel=10000)
parameter (jno=10000, jjfron=250)
integer nmat, mat(jno),nkt
integer nno,nel,nfron,fo(jjfron,jfron),ktri(jel,12),
double precision xcor(jno),ycor(jno),vvi(jjfron),
                  ro(jno),perm(30)

c---------- Leitura do arquivo de dados
     character namfu(20)
     integer ict
     equivalence (namfs,namfu(1))

     write(6,'(/a$)')'Arquivo de Entrada(.dat):'
     read(5,'(a)')namfs
     do 11 i=16,1,-1
       if(namfu(i).ne.' ')goto 22
11    continue
c     return
22    namfu(i+1)='.'
     namfu(i+2)='d'
     namfu(i+3)='a'
     namfu(i+4)='t'

     open(10,file=namfs,form='formatted')
     read(10,*)nkt,nno,nel
C---Leitura da formação da malha
     do 1 i=1,nel
1      read(10,*)(ktri(i,j),j=1,nkt),mat(i),ro(i)
C---Leitura de coordenadas dos nos
     do 2 i=1,nno
2      read(10,*)xcor(i),ycor(i)
      read(10,*)nfron
C---Leitura de condições de contorno
     do 3 i=1,nfron
      read(10,*)vvi(i)
      read(10,*)(fo(i,j),j=1,20)
3     continue
     read(10,*)nmat
C---Leitura das permissividades dos meios
     do 4 i=1,nmat
4      read(10,*)perm(i)
     return
     end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine BAND(nel,nno,ktri,mb,nkt)
     integer nel,nno,ktri(10000,12),mb,mbb
     integer ivar,ktrid,imin,imax,idif
     dimension ii(12)

     mb=0
     do 1 i=1,nel
      ivar=nkt
         do 2 k=1,ivar
          ii(k)=0
   3    if(ii(k).eq.0)ii(k)=ktri(i,k)
   2   continue
         imin=10000
         imax=0
         do 5 k=1,ivar
          if(ii(k).lt.imin)imin=ii(k)
          if(ii(k).gt.imax)imax=ii(k)
   5   continue
         idif=imax-imin
         if(mb.lt.idif)mb=idif
   1 continue
     mbb=mb+1
     write(6,'(//,a,i4)') ' Number of Elemts=',nel
     write(6,'(a,i4)')   ' Number of Nodes=',nno
     write(6,'(a,i4)')   '       Bandwidth=',mbb
     return
     end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
     subroutine ZERO(nno,mb,vdr,vv,ss,mrep)
     double precision vdr(1),ss(1),vv(1),mrep(1,1)
     integer nno,mb,mbb

     mbb=mb+1
     do 1 i=1,nno
       mrep(i,1)=0.
       mrep(i,2)=0.
       vv(i)=0.
       vdr(i)=0.
       do 1 n=1,mbb
         in=(i-1)*mbb+n
    1  ss(in)=0.
     return
     end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
  subroutine FORM(nel,irep,ikind,iaxe,nonper,mb,
 *       mrep,xcor,ycor,ktri,ss,nno,mat,ro,perm,nkt)
double precision  ss(1),xcor(1),ycor(1),mrep(1,1)
double precision perm(1),ro(1),xmu, xr(12),yr(12)
integer nel,irep,ikind,iaxe,nonper,mb,nkt
integer nno,mat(1), ktri(10000,12)
dimension ee(12,12)

    write(6,*)'  '
    write(6,*)'  Scalar Potential '
     ikind=0
    write(6,'(a$)')'   Cartesian '
     iaxe=0
     nonper=0
     irep=0
    write(6,*)' '
    write(6,'(/,a)')' Matrix Assembly ...'

    do 1 i=1,nel

c----Coordenadas dos Elementos de Referência
       do 20 j=1,nkt
         xr(j)=xcor(ktri(i,j))
         yr(j)=ycor(ktri(i,j))
20       continue
       nm=mat(i)
       xmu=perm(nm)

c---Com a permeabilidade, calcula as contribuições
         call CONTRB(nkt,xmu,xr,yr,ee)

c----Condensa as contribuições na matriz global
call SSFORM(i,nkt,ktri,ee,irep,nonper,mrep,mb,ss)

1    continue
    return
    end
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%C
subroutine CONTRB(nkt,xmu,xr,yr,ee)
integer npi,nkt
double precision xmu,xr(12),yr(12),detj,coef,
*                     dnx(12),dny(12)
double precision u,v,w,ui(12),vi(12),wi(12),fn(12)
dimension ee(12,12)

    do 4 i=1,nkt
     do 4 j=1,nkt
4       ee(i,j)=0.0

c--Chama subrotina com os ptos de Integração
     if(nkt.eq.3) call pitl2(npi,ui,vi,wi)
     if(nkt.eq.6)then
       npi=4
       call pitq2(npi,ui,vi,wi)
     endif
     if(nkt.eq.10)then
       npi=6
       call pitc2(npi,ui,vi,wi)
     endif
     if(nkt.eq.4) call piqb2(npi,ui,vi,wi)
     if(nkt.eq.8) call piqq2(npi,ui,vi,wi)
     if(nkt.eq.12)call piqc2(npi,ui,vi,wi)

c------Atribui os Pontos de Integração a u,v
     do 18 k=1,npi
        u=ui(k)
        v=vi(k)
        w=wi(k)

c------Chamada da subrotina contendo as
c          Funções de Forma Nodais
  if(nkt.eq.3)  call fntl2(u,v,xr,yr,fn,dnx,dny,detj)
  if(nkt.eq.6)  call fntq2(u,v,xr,yr,fn,dnx,dny,detj)
  if(nkt.eq.10)call fntc2(u,v,xr,yr,fn,dnx,dny,detj)
  if(nkt.eq.4)  call fnqb2(u,v,xr,yr,fn,dnx,dny,detj)
  if(nkt.eq.8)  call fnqq2(u,v,xr,yr,fn,dnx,dny,detj)
  if(nkt.eq.12)call fnqc2(u,v,xr,yr,fn,dnx,dny,detj)

       if(Abs(detj).gt.1.e-15) go to 1
       write(6,*)' Area of Triangle is Zero'

c------Calculo da matriz ee
1   do 11 l=1,nkt
      do 12 m=1,nkt
         coef=detj*xmu*w

ee(l,m)=ee(l,m)+(dnx(l)*dnx(m)+dny(l)*dny(m))*co
ef

12     continue
11     continue
18   continue
    return
    end
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%C
```

```
C%%%%%%%%%%%%%%%%%%%%%%%%C
subroutine SSFORM(i,nkt,ktri,ee,irep,nonper,
                        mrep,mb,ss)
     double precision ss(1),mrep(1,1)
     integer mb,mbb,irep,nonper,nkt,ktri(10000,12)
     dimension ee(12,12),iaux(12),mm(12)

     do j=1,nkt
       mm(j)=0
     enddo
     mbb=mb+1
     if(irep.eq.0)go to 1
c---------- Procura possível no periódico
       do 7 k0=1,nkt
          ij=ktri(i,k0)
          mj=mm(k0)
7          call Cherch(ij,mj,irep,mrep)
c----------
1       do 8 k1=1,nkt
8          iaux(k1)=ktri(i,k1)
        do  9 k2=1,nkt
9          if(mm(k2).ne.0)iaux(k2)=mm(k2)
      if(nonper.eq.0)goto 5
      do 3 m=1,nkt
        if(mm(m).eq.0)goto 3
        do 4 n=1,nkt
          if(m.eq.n)goto 4
          ee(m,n)=-ee(m,n)
          ee(n,m)=ee(m,n)
   4    continue
   3  continue
   5  do 2 m=1,nkt
        im=iaux(m)
        do 2 j=1,nkt
          ij=iaux(j)
          if(im.gt.ij)go to 2
c---------- Troca índice ij para condensar matriz
banda
          ij=ij-im+1
          ijm=(im-1)*mbb+ij
          ss(ijm)=ss(ijm)+ee(j,m)
   2  continue
      return
      end
C%%%%%%%%%%%%%%%%%%%%%%%%C
```

```
C==SUBROTINA PARA PROCURA DE
POSSIVEL NO PERIODICO==C
      Subroutine CHERCH(ij,ij1,irep,mrep)
      double precision mrep(1,1)
      integer ij,ij1,irep
      ij1=0
      do 1 i=1,irep
          if(mrep(i,2).ne.ij)go to 1
          ij1=mrep(i,1)
          return
    1 continue
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine COFRON(nfron,nno,iaxe,ikind,iterm,
*                        ycor,mb,icdterm,fo,vdr,vvi,ss)
parameter (jjfron=250,jfron=150)
double precision ss(1),vdr(1),vvi(1),ycor(1),xnorm,
*                    funcss
integer nfron,nno,iaxe,ikind,iterm,mb,mbb
integer fo(jjfron,jfron), icdterm(1)

      iterm=0
      xnorm=0.
      Mbb=mb+1
      do 10 i=1,nno
        ij=(i-1)*mbb+1
        funcss=ss(ij)
        IF(xnorm.lt.Abs(funcss)) xnorm=Abs(funcss)
   10 continue
c----- Aproximação para evitar perda dos termos
      xnorm=xnorm*1e+14
      do 1 i=1,nfron
        if(iterm.eq.0.and.icdterm(i).ne.0)goto 1
        if(iterm.eq.1.and.icdterm(i).ne.2)goto 1
        do 2 n=1,jfron
          if(fo(i,n).eq.0)goto 1
          j=fo(i,n)
          vdr(j)=vvi(i)*xnorm
if(iaxe.eq.1.and.ikind.eq.1.and.abs(ycor(j)).gt.1.e-
6)then
            vdr(j)=vdr(j)/(2.*3.141593*ycor(j))
          endif
          j1=(j-1)*mbb+1
          ss(j1)=xnorm
2       continue
    1 continue
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine RESOL(nno,mb,irep,nfron,iterm,ss,
*                          vdr,fo,vvi,vv)
parameter (jjfron=250,jfron=150)
double precision ss(1),vdr(1),vvi(1),vv(1)
double precision fat,som
Integer nno,mb,irep,nfron,iterm,nn,mbb,icont,
*        mmax,min,in,im,nno1
integer fo(jjfron,jfron)
c
      icont=0
      write(6,'(a$)') ' Matrix Solution ....'
      nn=nno-1
      mbb=mb+1
c---------------- Tridiagonaliza a matriz
      do 1 i=1,nn
        icont=icont+1
        if(icont.eq.int(nno*.2))write(6,'(a$)')'  20%'
        if(icont.eq.int(nno*.4))write(6,'(a$)')'  40%'
        if(icont.eq.int(nno*.6))write(6,'(a$)')'  60%'
        if(icont.eq.int(nno*.8))write(6,'(a$)')'  80%'
        if(icont.eq.int( nn*1.))write(6,'(a$)')' 100%'
        funci=ss((i-1)*mbb+1)
        ii=i+1
        if(IREP.ne.0.and.Abs(funci).lt..1e-5)go to 1
        mmax=i+mb
        if(mmax.gt.nno)mmax=nno
        in=(i-1)*mbb-i+1
c-- Coloca a zero todos os coef menores que 1.e-16
        do 13 m=ii,mmax
          if(abs(ss(in+m)).lt.1.e-16)ss(in+m)=0.
   13   continue
c-------- Passa as transformações
        do 2 m=ii,mmax
          fat=ss(in+m)/funci
          if(Dabs(fat).lt..1e-18)go to 2
          if(abs(vdr(i)).lt.1.e-16)goto 12
          vdr(m)=vdr(m)-vdr(i)*fat
   12     im=(m-1)*mbb-m+1
          do 5 j=m,mmax
            ss(im+j)=ss(im+j)-ss(in+j)*fat
    5     continue
    2   continue
    1 continue
c----- Com a matriz modificada, volta pela diagonal
      nno1=(nno-1)*mbb+1
    if(IREP.ne.0.and.Abs(ss(nno1)).lt..1e-12)go to 6
      vv(nno)=vdr(nno)/ss(nno1)
    6 do 7 i=1,nn
        ni=nno-i
        ni1=(ni-1)*mbb+1
        if(irep.ne.0.and.Abs(ss(ni1)).lt..1e-12)go to 7
        som=0.
        Min=i-mb+1
        if(min.lt.1)min=1
        do 8 n=min,i
          ib=nno-n+1
          som=som+dble(ss((ni-1)*mbb+ib-ni+1))*
&                          dble(vv(ib))
    8   continue
        vv(ni)=(vdr(ni)-som)/ss(ni1)
    7 continue
c--- Restitui zero nos nos dirichlet, quando eletro-
c       magnético
      if(ITERM.eq.1)return
      do 10 i=1,NFRON
        if(Abs(vvi(i)).gt.1e-30)goto 10
        do 11 j=1,jfron
          if(fo(i,j).eq.0)goto 10
          vv(fo(i,j))=0.
   11   continue
   10 continue
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine SAIDA(nno,nel,ikind,vv,vdr,nkt,ktri,
*                          xcor,ycor)
double precision vv(1),vdr(1),vvmax,vvmin,xm0,
double precision u,v,fn(12),dnx(12),dny(12),ex,ey
double precision  xr(12),yr(12),emod,somax,
*                          somay, xcor(1), ycor(1)
integer nno,ikind,ns,nkt,ktri(10000,12),nel
data numors/0/

     xmo=4.*3.1415926*1.e-7
     numors=numors+1
     vvmax=-1.e+20
     vvmin= 1.e+20
     if(ikind.eq.0)then
       do 1 i=1,nno
        if(vv(i).gt.vvmax)vvmax=vv(i)
        if(vv(i).lt.vvmin)vvmin=vv(i)
 1     continue
     else
       do 10 i=1,nno
        vdr(i)=vv(i)*xm0
         if(vdr(i).gt.vvmax)vvmax=vdr(i)
         if(vdr(i).lt.vvmin)vvmin=vdr(i)
 10     continue
     endif
     write(6,'(///a,e10.4)')' Min Pot=',vvmin
     write(6,'(a,e10.4)')' Max Pot=',vvmax
     write(6,'(/,a)')' Storing Results'

c-------Impressao dos resultados em um arquivo
      write(6,*)' '
      write(6,*)' OS RESULTADOS
ENCONTRAM-SE NO ARQUIVO SAIDA.DAT'
      write(6,*)' '
      open (11,file='saida.dat',form='formatted')
      write(11,39)
 39
format('C==========================C')
      write(11,40)
 40    format(' PROGRAMA DE CALCULO DE
CAMPOS POR ELEMENTOS FINITOS ')
      write(11,41)
 41    format(' UTILIZANDO A FUNCAO DE
FORMA NODAL PARA O ELEMENTO ')
      if(nkt.eq.3)then
        write(11,42)
 42      format(' TRIANGULAR LINEAR DE
PRIMEIRA ORDEM ')
      endif
      if(nkt.eq.6)then
        write(11,43)
 43      format(' TRIANGULAR QUADRATICO
*                 DE SEGUNDA ORDEM ')
      endif
      if(nkt.eq.10)then
        write(11,44)
 44      format(' TRIANGULAR CUBICO DE
*                 TERCEIRA ORDEM ')
      endif
      if(nkt.eq.4)then
        write(11,45)
 45     format(' QUADRILATERO BILINEAR DE
*                 PRIMEIRA ORDEM ')
      endif
      if(nkt.eq.8)then
        write(11,46)
 46    format(' QUADRILATERO QUADRATICO
*                DE SEGUNDA ORDEM ')
      endif
      if(nkt.eq.12)then
        write(11,47)
 47    format(' QUADRILATERO CUBICO DE
*                 TERCEIRA ORDEM ')
      endif
      write(11,48)
 48
format('C==========================C')
      write(11,*)' Numero de Nos=',nno
      write(11,*)' Numero de Elementos=',nel
      write(11,*)' '

c------Impressao de potencialidades nos NOS
      write(11,*)' POTENCIAL EM CADA NO '

      do 81 m=1,nno
 81     write(11,90)m,vv(m)
 90     format(' No:',i3,' Potencial=',e10.4)
       write(11,*)' '

c------Impressao dos campos nos elementos
      write(11,*)'VALOR DOS CAMPOS
*             ELETRICOS EM CADA
ELEMENTO'

c---CHAMA GRAD PARA CALCULO DOS
c    CAMPO OU GRADIENTES
     do 102 in=1,nel
        somax=0.
        Somay=0.
```

```
c----Coordenadas dos Elementos de Referencia
      do 20 j=1,nkt
       xr(j)=xcor(ktri(in,j))
       yr(j)=ycor(ktri(in,j))
20     continue

      u=0.
      v=0.

      if(nkt.eq.3) call fntl2(u,v,xr,yr,fn,dnx,dny,detj)
      if(nkt.eq.6) call fntq2(u,v,xr,yr,fn,dnx,dny,detj)
      if(nkt.eq.10)call fntc2(u,v,xr,yr,fn,dnx,dny,detj)
      if(nkt.eq.4) call fnqb2(u,v,xr,yr,fn,dnx,dny,detj)
      if(nkt.eq.8) call fnqq2(u,v,xr,yr,fn,dnx,dny,detj)
      if(nkt.eq.12)call fnqc2(u,v,xr,yr,fn,dnx,dny,detj)

      do 76 j=1,nkt
        somax=somax+(dnx(j)*vv(ktri(in,j)))
        somay=somay-(dny(j)*vv(ktri(in,j)))
76      continue

      ex=-somax
      ey=-somay
      emod=sqrt(ex*ex+ey*ey)
      if(abs(ex).lt.1e-8)ex=0.0
      write(11,95)in,ex,ey,emod
95     format(' Elem:',i3,' Ex=',e10.4,' Ey=',e10.4,' Er=',e10.4)
102  continue
    return
    end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

2 – PROGRAMA PARA APLICAÇÃO DAS FUNÇÕES 3D

```
C================================22/9/97=================================C
c  OBSERVAÇÃO: ESTE PROGRAMA ESTA TRABALHANDO APENAS COM POTENCIAL
c           ESCALAR  3D.
c        A entrada de dados é feita manualmente, por não haver malhadores específicos para todos os
c  casos aqui estudados. A etapa de calculo é feita utilizando-se  as funções de forma de acordo  com
c  o tipo de elemento.
c
c           Este programa utiliza as seguintes subrotinas do programa EFCS:
c                  - ZERO: zerar variáveis do sistema(condição inicial)
c                  - BAND: calculo da matriz banda
c                  - SSFORM: condensar a matriz
c                  - COFRON: adicionar as condições de contorno
c                  - RESOL: resolver o sistema matricial
c
c        Subrotinas adicionadas para o cálculo via Método dos Resíduos:
c        -PITL3,PITQ3,PITC3: pontos de integração para os diferentes elementos tetraédricos 3D
c        -PIHT3,PIHQ3,PIHC3: pontos de integração para os diferentes elementos hexaédricos 3D
c        -FNTL3,FNTQ3,FNTC3: funções de forma para os diferentes elementos tetraédricos 3D
c        -FNHT3,FNHQ3,FNHC3: funções de forma para os diferentes elementos hexaédricos 3D
c
c        As subrotinas anteriores permitem escolhas de acordo com o tipo  de elemento.
C=========================================================================C
c NOMENCLATURA:
c  * detj          - determinante do jacobiano
c  * dn() - derivada de fn em relação a u e v
c  * dnx,dny- derivada de fn em relação a x e y
c  * ee(,)    - matriz rigidez [=S(nkt,nkt)]
c  * fn() - função de forma nodal
c  * fo(,)  - arranjo contendo os nos com condição de contorno
c  * iaxe   - define o tipo de eixo(0:cartesiano ou 1:axi-simetrico)
c  * icterm -termo relacionado com as fontes de aquecimento
c  * ikind  - define o tipo de potencial(0:Escalar ou 1:Vetor)
c  * irep   - numero de pares de nos periódicos da malha
c        =0> não ha periodicidade ou =1> ha period. no domínio
c  * iterm  - valor de uma constante (iterm=0)
c  * ktri(,)- guarda o numero do meio do elemento
c  * mat()  - numero do meio em cada um dos NEL elementos
c  * mb   - matriz banda
c  * mbb  - Largura de banda
c  * mrep(,)- guarda o arranjo com os pares de nos periódicos
c  * nmat   - numero de meios dielétricos
c  * nno  - numero de nos
c  * nel  - numero de elementos
c  * nfron- numero de condições de fronteira
c  * nkt  - numero de nos de acordo com tipo de elemento
c  * nonper - =0>caso periódico ou =1> caso anti-periódico
c  * npi - número de pontos de integração
c  * perm() - permissividade dos NMAT meios
c  * ro()   - densidade superficial de carga
c  * ss()   - matriz global
c  * tp - numero que identifica o tipo de elemento 3D
c  * u,v, p  - pontos de integração
```

```
c  * ui,vi, pi  - pontos de integração
c  * vvi()  - potencial imposto na fronteira
c  * vdr()  - vetor lado direito
c  * vv()   - potencial vetor incógnito
c  * xcor() - coordenada x do nó
c  * ycor() - coordenada y  do nó
c  * zcor() - coordenada z  do nó
c  * xr,yr  - coordenadas dos elementos no espaço de referencia
c  * xmu=perm()- permissividade dos NMAT meios
c  * wi    -  peso
C=================================================================C
C------PROGRAMA PRINCIPAL
      parameter (jjfron=250,jel=10000,jrep=400,jno=10000,jfron=150)
      parameter (jnband=1000000)
      integer nno,nel,nfron,irep,nonper,ikind,iaxe,mb,mat(jno)
      integer icdterm(jjfron),ktri(jel,32),iterm,fo(jjfron,jfron)
      integer tp,nkt,nmat,ncle
      double precision xcor(jno),ycor(jno),zcor(jno),vvi(jjfron)
      double precision mrep(jrep,2),vdr(jno),ss(jnband),vv(jno)
      double precision perm(30),ro(jno)

      write(*,*)' '
      write(*,*)' '
      write(6,'(a$)')' ATENCAO!!!'
      write(6,'(a$)')' ESTE PROGRAMA ESTA OPERANDO APENAS COM'
      write(6,'(a$)')' POTENCIAL ESCALAR EM 3D'
      write(*,*)' '

C------Chama DATAIN para entra de dados
      CALL DATAIN(nno,nel,nkt,tp,nfron,ktri,xcor,ycor,zcor,vvi,fo,mat,ro,perm,nmat)

c------Chama BAND para o calculo da Matriz Banda
      CALL BAND(nel,nno,ktri,mb,nkt)

C------Chama Subrotina ZERO para Inicialização das Matrizes
      CALL ZERO(nno,mb,vdr,vv,ss,mrep)

C------Chama Subrotina FORM para Formação da Matriz Global SS
      CALL FORM(nel,irep,ikind,iaxe,nonper,mb,mrep,xcor,ycor,zcor,ktri,ss,nno,mat,ro,perm,nkt,tp)

C******-Chama Subrotina COFRON para Inserção das Condições de Contorno VDR
      CALL COFRON(nfron,nno,iaxe,ikind,iterm,YCOR,MB,ICTERM,FO,VDR,VVI,SS)

C******-Chama subrotina RESOL para Resolução do Sistema Matricial
C                          [vv]=[vdr]/[ss]
      CALL RESOL(nno,mb,irep,nfron,iterm,SS,VDR,FO,VVI,VV)

c------Chama subrotina SAIDA para impressão dos resultados
      CALL SAIDA(nno,nel,ikind,vv,vdr,nkt,tp,ktri,xcor,ycor,zcor)
      STOP
      END
C=================================================================C
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
      subroutine DATAIN(nno,nel,nkt,tp,nfron,ktri,xcor,
     *                  ycor,zcor,vvi,fo,mat,ro,perm,nmat)
      character namfs*20,namfu(20)
      equivalence (namfs,namfu(1))
      parameter (jpont=150,jfron=150,jel=10000
      parameter (jno=10000,jjfron=250)
      integer nmat,tp
      integer nno,nel,nfron,fo(jjfron,jfron),ktri(jel,32),nkt
      double precision xcor(jno),ycor(jno),zcor(jno),
     *                 vvi(jjfron)
      double precision ro(jno),perm(30), mat(jno),
c------------ Leitura do arquivo de dados
      character namfu(20)
      integer ict
      equivalence (namfs,namfu(1))

      write(6,'(/a$)')'Arquivo de Entrada(.dat):'
      read(5,'(a)')namfs
      do 11 i=16,1,-1
        if(namfu(i).ne.' ')goto 22
11    continue
c     return
22    namfu(i+1)='.'
      namfu(i+2)='d'
      namfu(i+3)='a'
      namfu(i+4)='t'

      open(10,file=namfs,form='formatted')
      read(10,*)tp,nkt,nno,nel

C---Leitura da formação da malha
       do 1 i=1,nel
1       read(10,*)(ktri(i,j),j=1,nkt),mat(i),ro(i)

C---Leitura de coordenadas dos nos
      do 2 i=1,nno
2       read(10,*)xcor(i),ycor(i),zcor(i)
       read(10,*)nfron
C---Leitura de condições de contorno
      do 3 i=1,nfron
       read(10,*)vvi(i)
       read(10,*)(fo(i,j),j=1,20)
3     continue
      read(10,*)nmat
C---Leitura das permissividades dos meios
      do 4 i=1,nmat
4       read(10,*)perm(i)
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
      subroutine BAND(nel,nno,ktri,mb,nkt)
      integer nel,nno,ktri(10000,32),mb,mbb
      integer ivar,ktrid,imin,imax,idif
      dimension ii(32)

      mb=0
      do 1 i=1,nel
       ivar=nkt
          do 2 k=1,ivar
           ii(k)=0
    3    if(ii(k).eq.0)ii(k)=ktri(i,k)
    2   continue
          imin=10000
          imax=0
          do 5 k=1,ivar
           if(ii(k).lt.imin)imin=ii(k)
           if(ii(k).gt.imax)imax=ii(k)
    5   continue
          idif=imax-imin
          if(mb.lt.idif)mb=idif
    1 continue
      mbb=mb+1
      write(6,'(//,a,i4)') ' Number  of Elemts=',nel
      write(6,'(a,i4)')    ' Number  of Nodes=',nno
      write(6,'(a,i4)')    '        Bandwidth=',mbb
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
      subroutine ZERO(nno,mb,vdr,vv,ss,mrep)
      double precision vdr(1),ss(1),vv(1),mrep(1,1)
      integer nno,mb,mbb

      mbb=mb+1
      do 1 i=1,nno
        mrep(i,1)=0.
        mrep(i,2)=0.
        vv(i)=0.
        vdr(i)=0.
        do 1 n=1,mbb
          in=(i-1)*mbb+n
     1  ss(in)=0.
       return
       end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
Subroutine FORM(nel,irep,ikind,iaxe,nonper,mb,
* mrep,xcor,xcor,zcor,ktri,ss,nno,mat,ro,perm,nkt,tp)
double precision  xcor(1),ycor(1),zcor(1),xr(32),
*                 yr(32),zr(32)
double precision  ss(1),mrep(1,1),perm(1),ro(1),xmu
integer nel,irep,ikind,iaxe,nonper,mb,ktri(10000,32)
integer nno,mat(1),nkt,tp
dimension ee(32,32)

      write(6,*)'  '
      write(6,*)'  Scalar Potential '
       ikind=0
      write(6,'(a$)')'   Cartesian '
       iaxe=0
       nonper=0
       irep=0
      write(6,*)' '
      write(6,'(/,a)')' Matrix Assembly ...'

      do 1 i=1,nel

c----Coordenadas dos Elementos de Referência
        do 20 j=1,nkt
          xr(j)=xcor(ktri(i,j))
          yr(j)=ycor(ktri(i,j))
          zr(j)=zcor(ktri(i,j))
20        continue
        nm=mat(i)
        xmu=perm(nm)

c----Com a permeabilidade, calcula as contribuições
          call CONTRB(nkt,tp,xmu,xr,yr,zr,ee)

c----Condensa as contribuições na matriz global
   call SSFORM(i,nkt,ktri,ee,irep,nonper,mrep,mb,ss)

1     continue
      return
      end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
C%%%%%%%%%%%%%%%%%%%%%%%%%%%C
subroutine CONTRB(nkt,tp,xmu,xr,yr,ee)
integer npi,nkt,tp
double precision xr(32),yr(32),zr(32),dnx(32),coef
double precision xmu,detj, dny(32),dnz(32),fn(32)
double precision u,v,p,w,ui(14),vi(14),pi(14),wi(14)
dimension ee(32,32)

      do 4 i=1,nkt
       do 4 j=1,nkt
4        ee(i,j)=0.0
```

```
c----Chama subrotina com os Pontos de
Integração
  if(nkt.eq.4) call pitl3(npi,ui,vi,pi,wi)
  if(nkt.eq.10)call pitq3(npi,ui,vi,pi,wi)
  if(nkt.eq.20.and.tp.eq.3)call pitc3(npi,ui,vi,pi,wi)
  if(nkt.eq.8)call pih3(npi,ui,vi,pi,wi)
  if(nkt.eq.20.and.tp.eq.5)call pih3(npi,ui,vi,pi,wi)
  if(nkt.eq.32)call pih3(npi,ui,vi,pi,wi)

C------Atribui os Pontos de Integração a u,v,p
     do 18 k=1,npi
        u=ui(k)
        v=vi(k)
        p=pi(k)
        w=wi(k)

C------Chamada da subrotina contendo as Funções
de Forma Nodais
   if(nkt.eq.4) call fntl3(u,v,p,xr,yr,zr,fn,dnx,dny,
                           dnz,detj)
   if(nkt.eq.10)call fntq3(u,v,p,xr,yr,zr,fn,dnx,dny,
    &                      dnz,detj)
   if(nkt.eq.20.and.tp.eq.3)call fntc3(u,v,p,xr,yr,zr,
    &                      fn,dnx,dny,dnz,detj)
   if(nkt.eq.8) call fnht3(u,v,p,xr,yr,zr,fn,dnx,dny,
    &                      dnz,detj)
   if(nkt.eq.20.and.tp.eq.5)call fnhq3(u,v,p,xr,yr,zr,
    &                      fn,dnx,dny,dnz,detj)
   if(nkt.eq.32)call fnhc3(u,v,p,xr,yr,zr,fn,dnx,dny,
    &                      dnz,detj)

        if(Abs(detj).gt.1.e-15) go to 1
        write(6,*)' Area of Triangle is Zero'

c------Calculo da matriz ee
1       do 11 l=1,nkt
        do 12 m=1,nkt
          coef=detj*xmu*w

ee(l,m)=ee(l,m)+(dnx(l)*dnx(m)+dny(l)*dny(m)+
&                dnz(l)*dnz(m))*coef
12       continue
11      continue
18    continue
      return
      end
C%%%%%%%%%%%%%%%%%%%%%%%%%%%C
```

```
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%C
Subroutine SSFORM(i,nkt,ktri,ee,irep,nonper,
                        mrep,mb,ss)
     double precision ss(1),mrep(1,1)
     integer mb,mbb,irep,nonper,nkt,ktri(10000,32)
     dimension ee(32,32),iaux(32),mm(32)

     do j=1,nkt
      mm(j)=0
     enddo
     mbb=mb+1
     if(irep.eq.0)go to 1
c---------- Procura possível no periódico
       do 7 k0=1,nkt
          ij=ktri(i,k0)
          mj=mm(k0)
7          call Cherch(ij,mj,irep,mrep)
c----------
1       do 8 k1=1,nkt
8           iaux(k1)=ktri(i,k1)
        do  9 k2=1,nkt
9           if(mm(k2).ne.0)iaux(k2)=mm(k2)
     if(nonper.eq.0)goto 5
     do 3 m=1,nkt
       if(mm(m).eq.0)goto 3
       do 4 n=1,nkt
         if(m.eq.n)goto 4
         ee(m,n)=-ee(m,n)
         ee(n,m)=ee(m,n)
   4    continue
   3 continue
   5 do 2 m=1,nkt
       im=iaux(m)
       do 2 j=1,nkt
         ij=iaux(j)
         if(im.gt.ij)go to 2
c---------- Troca índice ij para condensar matriz
banda
         ij=ij-im+1
         ijm=(im-1)*mbb+ij
         ss(ijm)=ss(ijm)+ee(j,m)
   2 continue
     return
     end
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%C
```

```
C==SUBROTINA PARA PROCURA DE
POSSIVEL NO PERIODICO==C
     subroutine CHERCH(ij,ij1,irep,mrep)
     double precision mrep(1,1)
     integer ij,ij1,irep

     ij1=0
     do 1 i=1,irep
          if(mrep(i,2).ne.ij)go to 1
          ij1=mrep(i,1)
          return
    1 continue
     return
     end
c:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine COFRON(nfron,nno,iaxe,ikind,iterm,
*              ycor,mb,icdterm,fo,vdr,vvi,ss)
parameter (jjfron=250,jfron=150)
double precision ss(1),vdr(1),vvi(1),ycor(1),xnorm,
*                    funcss
integer nfron,nno,iaxe,ikind,iterm,mb,mbb,icdterm(1)
integer fo(jjfron,jfron)

     iterm=0
     xnorm=0.
     Mbb=mb+1
     do 10 i=1,nno
       ij=(i-1)*mbb+1
       funcss=ss(ij)
       IF(xnorm.lt.Abs(funcss)) xnorm=Abs(funcss)
   10 continue
c----- Aproximação para evitar perda dos termos
     xnorm=xnorm*1e+14
     do 1 i=1,nfron
       if(iterm.eq.0.and.icdterm(i).ne.0)goto 1
       if(iterm.eq.1.and.icdterm(i).ne.2)goto 1
       do 2 n=1,jfron
         if(fo(i,n).eq.0)goto 1
         j=fo(i,n)
         vdr(j)=vvi(i)*xnorm

if(iaxe.eq.1.and.ikind.eq.1.and.abs(ycor(j)).gt.1.e-
6)then
           vdr(j)=vdr(j)/(2.*3.141593*ycor(j))
         endif
         j1=(j-1)*mbb+1
         ss(j1)=xnorm
2        continue
    1 continue
     return
     end
c:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

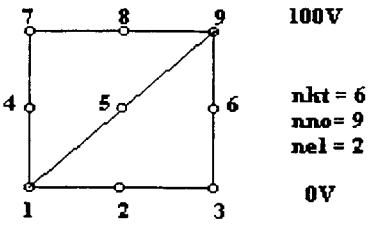
```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine RESOL(nno,mb,irep,nfron,iterm,ss,
*                          vdr,fo,vvi,vv)
parameter (jjfron=250,jfron=150)
double precision ss(1),vdr(1),vvi(1),vv(1)
integer nno,mb,irep,nfron,iterm,nn,mbb,icont,
*         mmax,min,in,im,nno1
double precision fat,som
integer fo(jjfron,jfron)
c
     icont=0
     write(6,'(a$)') ' Matrix Solution ....'
     nn=nno-1
     mbb=mb+1
c---------------- Tridiagonaliza a matriz
     do 1 i=1,nn
       icont=icont+1
       if(icont.eq.int(nno*.2))write(6,'(a$)')'  20%'
       if(icont.eq.int(nno*.4))write(6,'(a$)')'  40%'
       if(icont.eq.int(nno*.6))write(6,'(a$)')'  60%'
       if(icont.eq.int(nno*.8))write(6,'(a$)')'  80%'
       if(icont.eq.int( nn*1.))write(6,'(a$)')' 100%'
       funci=ss((i-1)*mbb+1)
       ii=i+1
       if(IREP.ne.0.and.Abs(funci).lt..1e-5)go to 1
       mmax=i+mb
       if(mmax.gt.nno)mmax=nno
       in=(i-1)*mbb-i+1
c-- Coloca a zero todos os coef menores que 1.e-16
       do 13 m=ii,mmax
        if(abs(ss(in+m)).lt.1.e-16)ss(in+m)=0.
  13   continue
c------- Passa as transformações
       do 2 m=ii,mmax
         fat=ss(in+m)/funci
         if(Dabs(fat).lt..1e-18)go to 2
         if(abs(vdr(i)).lt.1.e-16)goto 12
         vdr(m)=vdr(m)-vdr(i)*fat
  12      im=(m-1)*mbb-m+1
         do 5 j=m,mmax
          ss(im+j)=ss(im+j)-ss(in+j)*fat
   5      continue
   2   continue
   1 continue
c----- Com a matriz modificada, volta pela diagonal
    nno1=(nno-1)*mbb+1
    if(IREP.ne.0.and.Abs(ss(nno1)).lt..1e-12)go to 6
      vv(nno)=vdr(nno)/ss(nno1)
   6 do 7 i=1,nn
       ni=nno-i
       ni1=(ni-1)*mbb+1
       if(irep.ne.0.and.Abs(ss(ni1)).lt..1e-12)go to 7
       som=0.
       Min=i-mb+1
       if(min.lt.1)min=1
       do 8 n=min,i
         ib=nno-n+1
         som=som+dble(ss((ni-1)*mbb+ib-ni+1))*
&                          dble(vv(ib))
   8   continue
       vv(ni)=(vdr(ni)-som)/ss(ni1)
   7 continue
c------ Restitui zero nos nos dirichlet, quando
eletromagnético
     if(ITERM.eq.1)return
     do 10 i=1,NFRON
       if(Abs(vvi(i)).gt.1e-30)goto 10
       do 11 j=1,jfron
         if(fo(i,j).eq.0)goto 10
         vv(fo(i,j))=0.
  11     continue
  10  continue
     return
     end
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

```
c::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
subroutine SAIDA(nno,nel,ikind,vv,vdr,nkt,tp,ktri,
*                xcor,ycor,zcor)
double precision vv(1),vdr(1),vvmax,vvmin,xm0
*               xcor(1),ycor(1),zcor(1),emod
double precision u,v,p,fn(32),dnx(32),dny(32),
*               dnz(32),somax,somay,somaz
double precision  xr(32),yr(32),zr(32),ex,ey,ez
integer nno,ikind,ns,nkt,ktri(10000,32),nel,tp
data numors/0/

    XM0=4.*3.1415926*1.e-7
    Numors=numors+1
    Vvmax=-1.e+20
    Vvmin= 1.e+20
    if(ikind.eq.0)then
      do 1 i=1,nno
       if(vv(i).gt.vvmax)vvmax=vv(i)
       if(vv(i).lt.vvmin)vvmin=vv(i)
 1    continue
    else
      do 10 i=1,nno
       vdr(i)=vv(i)*xm0
       if(vdr(i).gt.vvmax)vvmax=vdr(i)
       if(vdr(i).lt.vvmin)vvmin=vdr(i)
 10    continue
    endif
    write(6,'(///a,e10.4)')' Min Pot=',vvmin
    write(6,'(a,e10.4)')' Max Pot=',vvmax
    write(6,'(/,a)')' Storing Results'

c-------Impressão dos resultados em um arquivo
      write(6,*)' '
      write(6,*)' OS RESULTADOS
ENCONTRAM-SE NO ARQUIVO SAIDA.DAT'
      write(6,*)' '
      open (11,file='saida.dat',form='formatted')
      write(11,39)
 39
format('C============================C')
      write(11,40)
 40    format(' PROGRAMA DE CALCULO DE
*       CAMPOS POR ELEMENTOS FINITOS ')
      write(11,41)
 41    format(' UTILIZANDO A FUNCAO DE
*       FORMA NODAL PARA O ELEMENTO ')
      if(nkt.eq.4)then
        write(11,42)
 42      format(' TETRAEDRICO LINEAR DE
*                PRIMEIRA ORDEM ')
        endif
      if(nkt.eq.10)then
        write(11,43)
 43      format(' TETRAEDRICO QUADRATICO
*                DE SEGUNDA ORDEM ')
      endif
      if(nkt.eq.20.and.tp.eq.3)then
        write(11,44)
 44      format(' TETRAEDRICO CUBICO DE
*                TERCEIRA ORDEM ')
      endif
      if(nkt.eq.8)then
        write(11,45)
 45      format(' HEXAEDRICO TRILINEAR DE
*                PRIMEIRA ORDEM ')
      endif
      if(nkt.eq.20.and.tp.eq.5)then
        write(11,46)
 46      format(' HEXAEDRICO QUADRATICO
*                DE SEGUNDA ORDEM ')
      endif
      if(nkt.eq.32)then
        write(11,47)
 47      format(' HEXAEDRICO CUBICO DE
*                TERCEIRA ORDEM ')
      endif
      write(11,48)
 48
format('C============================C')
      write(11,*)' Numero de Nos=',nno
      write(11,*)' Numero de Elementos=',nel
      write(11,*)' '

C------Impressão de potencialidades nos NÓS
      write(11,*)' POTENCIAL EM CADA NO '

do 81 m=1,nno
  81    write(11,90)m,vv(m)
  90    format(' No:',i3,' Potencial=',e10.4)
        write(11,*)' '

C------Impressão dos campos nos elementos
      write(11,*)'VALOR DOS CAMPOS
*           ELETRICOS EM CADA ELEMENTO'

C---CHAMA GRAD  PARA CALCULO DOS
CAMPO OU GRADIENTES
    do 102 in=1,nel
       somax=0.
       somay=0.
       somaz=0.
```

```
c----Coordenadas dos Elementos de Referencia
      do 20 j=1,nkt
       xr(j)=xcor(ktri(in,j))
       yr(j)=ycor(ktri(in,j))
       zr(j)=zcor(ktri(in,j))
20     continue

       u=0.
       v=0.
       p=0.

      if(nkt.eq.4) call fntl3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)
      if(nkt.eq.10)call fntq3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)
      if(nkt.eq.20.and.tp.eq.3)call fntc3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)
      if(nkt.eq.8) call fnht3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)
      if(nkt.eq.20.and.tp.eq.5)call fnhq3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)
      if(nkt.eq.32)call fnhc3(u,v,p,xr,yr,zr,fn,dnx,dny,dnz,detj)

       do 76 j=1,nkt
        somax=somax+(dnx(j)*vv(ktri(in,j)))
        somay=somay-(dny(j)*vv(ktri(in,j)))
        somaz=somaz-(dnz(j)*vv(ktri(in,j)))
76      continue

       ex=-somax
       ey=-somay
       ez=-somaz
       emod=sqrt(ex*ex+ey*ey+ez*ez)
       if(abs(ex).lt.1e-8)ex=0.0
       if(abs(ey).lt.1e-8)ey=0.0
       write(11,95)in,ex,ey,ez,emod
95     format(' Elem:',i3,' Ex=',e10.4,' Ey=',e10.4,' Ez=',e10.4,
   *        ' Er=',e10.4,)
102  continue
   return
   end
c:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::c
```

3 – RESULTADOS OBTIDOS DOS PROGRAMAS EM 2D E 3D

Triangular Quadrático 2D



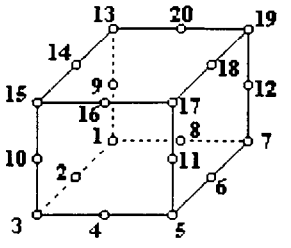
Arquivo de Entrada:

```
 6
 9
 2
 1 2 3 6 9 5 1  .0000E+00
 9 8 7 4 1 5 1  .0000E+00
  .0000  .0000
  .5000  .0000
 1.0000  .0000
  .0000  .5000
  .5000  .5000
 1.0000  .5000
  .0000 1.0000
  .5000 1.0000
 1.0000 1.0000
 2
 100.00
  7 8 9 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0
   .00
  1 2 3 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0
  1
.1000E+01
```

Arquivo de Saída:

```
C============================C
PROGRAMA DE CÁLCULO DE CAMPOS
POR ELEMENTOS FINITOS UTILIZANDO A
FUNÇÃO DE FORMA NODAL PARA O
ELEMENTO TRIANGULAR QUADRÁTICO
DE SEGUNDA ORDEM
C============================C
 Número de Nós = 9
 Número de Elementos = 2
 POTENCIAL EM CADA NÓ
Nó:  1 Potencial = .0000E+00
Nó:  2 Potencial = .0000E+00
Nó:  3 Potencial = .0000E+00
Nó:  4 Potencial = .5000E+02
Nó:  5 Potencial = .5000E+02
Nó:  6 Potencial = .5000E+02
Nó:  7 Potencial = .1000E+03
Nó:  8 Potencial = .1000E+03
Nó:  9 Potencial = .1000E+03

VALOR DOS CAMPOS ELÉTRICOS EM
CADA ELEMENTO
Elem:  1  Ex = .0000E+00  Ey = .1000E+03
          Er = .1000E+03
Elem:  2  Ex = .0000E+00  Ey = .1000E+03
          Er = .1000E+03
```

HEXAÉDRICO QUADRÁTICO



tp = 5
nkt = 20
nno= 20
nel = 1

100V - Nós: 13 até 20
0V - Nós: 1 até 8

ARQUIVO DE ENTRADA:

```
  5
 20
 20
  1
 1    2    3    4    5    6    7    8
 9   10   11   12   13   14   15   16
17   18   19   20    1    .0000E+00
    .0000    .0000    .0000
    .5000    .0000    .0000
   1.0000    .0000    .0000
   1.0000    .5000    .0000
   1.0000   1.0000    .0000
    .5000   1.0000    .0000
    .0000   1.0000    .0000
    .0000    .5000    .0000
    .0000    .0000    .5000
   1.0000    .0000    .5000
   1.0000   1.0000    .5000
    .0000   1.0000    .5000
    .0000    .0000   1.0000
    .5000    .0000   1.0000
   1.0000    .0000   1.0000
   1.0000    .5000   1.0000
   1.0000   1.0000   1.0000
    .5000   1.0000   1.0000
    .0000   1.0000   1.0000
    .0000    .5000   1.0000
  2
 100.00
 13   14   15   16   17   18   19   20
  0    0    0    0    0    0    0    0
 0    0    0    0
.00
  1    2    3    4    5    6    7    8
  0    0    0    0    0    0    0    0
  0    0    0    0
  1
.1000E+01
```

ARQUIVO DE SAIDA:

```
C============================================================C
PROGRAMA DE CALCULO DE CAMPOS POR ELEMENTOS FINITOS UTILIZANDO
A FUNÇÃO DE FORMA NODAL PARA O ELEMENTO HEXAÉDRICO QUADRÁTICO
DE SEGUNDA ORDEM
C============================================================C
  Número de Nós= 20
  Número de Elementos= 1

  POTENCIAL EM CADA NÓ
 No:  1 Potencial= .0000E+00
 No:  2 Potencial= .0000E+00
 No:  3 Potencial= .0000E+00
 No:  4 Potencial= .0000E+00
 No:  5 Potencial= .0000E+00
 No:  6 Potencial= .0000E+00
 No:  7 Potencial= .0000E+00
 No:  8 Potencial= .0000E+00
 No:  9 Potencial= .5000E+02
 No: 10 Potencial= .5000E+02
 No: 11 Potencial= .5000E+02
 No: 12 Potencial= .5000E+02
 No: 13 Potencial= .1000E+03
 No: 14 Potencial= .1000E+03
 No: 15 Potencial= .1000E+03
 No: 16 Potencial= .1000E+03
 No: 17 Potencial= .1000E+03
 No: 18 Potencial= .1000E+03
 No: 19 Potencial= .1000E+03
 No: 20 Potencial= .1000E+03

VALOR DOS CAMPOS ELÉTRICOS EM CADA ELEMENTO
 Elem:  1
  Ex=-.2891E-05  Ey= .7188E-05
  Ez= .1000E+03  Er= .1000E+03
```