Rodrigo Alves Augusto

Arquitetura Orientada por Objetos para o MEF de Alta Ordem com Aplicações em Mecânica Estrutural

122/2012

CAMPINAS
2012

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA

Rodrigo Alves Augusto

Arquitetura Orientada por Objetos para o MEF de Alta Ordem com Aplicações em Mecânica Estrutural

Orientador: Prof. Dr. Marco Lúcio Bittencourt

Tese de Doutorado apresentada à Faculdade de Engenharia Mecânica da Universidade Estadual de Campinas, para a obtenção do título de Doutor em Engenharia Mecânica, na Área de Mecânica dos Sólidos e Projeto Mecânico

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA TESE DEFENDIDA PELO ALUNO RODRIGO ALVES AUGUSTO, E ORIENTADO PELO PROF. DR MARCO LÚCIO BITTENCOURT

.....
ASSINATURA DO ORIENTADOR

CAMPINAS
2012

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Au45a

Augusto, Rodrigo Alves
Arquitetura Orientada por Objetos para o MEF de Alta Ordem com Aplicações em Mecânica Estrutural \ Rodrigo Alves Augusto. – Campinas, SP: [s.n.], 2012.

Orientador: Marco Lúcio Bittencourt.
Tese de Doutorado - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Método dos Elementos Finitos. 2. Deformações (Mecânica). 3. Programação Não-Linear
4. Programação Orientada a Objetos (Computação). I. Bittencourt, Marco Lúcio, 1972-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título

Título em Inglês:	Object-Oriented Architecture for High-Order FEM with Applications on Structural Mechanics
Palavras-chave em Inglês:	Finite Element Method, Deformations (Mechanical), Nonlinear programming, Object-Oriented Programming (Computer)
Área de concentração:	Mecânica dos Sólidos e Projeto Mecânico
Titulação:	Doutor em Engenharia Mecânica
Banca Examinadora:	Carlos Alberto Cimini Junior, Auteliano Antunes dos Santos Junior, Bruno Souza Carmo, Pablo Andrés Muñoz Rojas
Data da defesa:	29/10/2012
Programa de Pós Graduação:	Engenharia Mecânica

UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA MECÂNICA

COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

DEPARTAMENTO DE PROJETO MECÂNICO

TESE DE DOUTORADO

Arquitetura Orientada por Objetos para o MEF de Alta Ordem com Aplicações em Mecânica Estrutural

Autor: Rodrigo Alves Augusto
Orientador: Prof. Dr. Marco Lúcio Bittencourt

A Banca Examinadora composta pelos membros abaixo aprovou esta Tese:

Prof. Dr. Marco Lúcio Bittencourt, Presidente
DPM/FEM/UNICAMP

Prof. Dr. Carlos Alberto Cimini Junior
DEES/UFMG

Prof. Dr. Auteliano Antunes dos Santos Junior.
DPM/FEM/UNICAMP

Prof. Dr. Bruno Souza Carmo
DME/EPUSP/USP

Prof. Dr. Pablo Andrés Muñoz Rojas
DEM/UDESC/JOINVILLE

Campinas, 29 de Outubro de 2012.

Dedicatória

Dedico este trabalho à minha família e a todos que me apoiaram.

Agradecimentos

Este trabalho é a realização de um antigo sonho, trabalhar com pesquisa de alta tecnologia. Meus sinceros agradecimentos ao Prof. Dr. Marco Lúcio Bittencourt pela oportunidade e tempo dedicado à minha orientação.

Aos meus pais Antonio e Elisabeth, meu irmão Rafael e ao meu queridos avós João Alves (*in memorian*), Orlanda Alves e Antonio Augusto (*in memorian*) pelo incentivo e apoio em todos os momentos difíceis.

À minha esposa Tyeme por todo carinho, compreensão, incentivo e amor, principalmente nos meus momentos de irritação.

Aos amigos da UNICAMP, em especial, Fabiano Marquezzi, Edilson Borges, Carlos Eduardo Leite Pereira, e Eduardo Carvalho, Rodrigo Gerardin, Pedro H. Baptistella, Fabiano Bargos, Jaime Izuka e Guilherme Allegre, Daniel Leonardo Martins, Felipe Furlan pelo espírito colaborador no processo de aprimoramento dos estudos em Engenharia Mecânica.

Aos amigos de todos os momentos, Lourenço Lopretti, Tamara Lopretti, Fábio Moraes Garcia, Daniel Godoy, Marcelo Jannuzzi e Renato Antunes Campos por demonstrarem paciência e serenidade comigo nos momentos mais difíceis.

Aos colegas, professores e funcionários da FEM que, direta ou indiretamente, contribuíram para a realização desse trabalho.

Aos amigos da General Motors do Brasil, Nicolau Botelho, Júlio Hirose, Leandro Abreu, Leonardo Leite, Valter Barragan, André Bergel e Edson L. Duque que me impulsionaram no meu desenvolvimento pessoal e profissional.

À FAPESP, DPM, FEM e UNICAMP pelo suporte financeiro e institucional destinados a este projeto.

Todos os problemas solucionáveis são triviais, todos problemas não-triviais são insolucionáveis.

Santayana

Resumo

Nesta tese é apresentada uma arquitetura orientada por objetos para Método dos Elementos Finitos de Alta Ordem (MEF-AO) para solucionar problemas de grandes deformações com material hiperelástico em mecânica estrutural. O programa hp^2fem foi construído em linguagem de programação MatLab, sendo totalmente modular e facilmente estensível para qualquer tipo de problema. A formulação do problema de grandes deformações foi realizada com auxílio de operadores cinemáticos, de tensões e equações constitutivas. A solução do sistema de equação não-linear for feita através do Método de Newton-Raphson. Apresentam-se resultados para a simulação de problema de grandes deformações usando material neo-Hookeano compressível em problemas 2D e 3D e métodos de alta ordem. Também, problemas de contato e otimização de forma são tratados aqui. Conclui-se que a arquitetura funciona muito bem e que o uso do MEF-AO traz grandes benefícios quanto à taxa de convergência.

Palavras-chave: Método dos Elementos Finitos, Métodos de Alta Ordem, Grandes Deformações, Programação Não-Linear, Programação Orientada por Objetos.

Abstract

This thesis presents an object oriented architecture for the High-Order Finite Element Method (HO-FEM) to solve problems of large deformations in structural mechanics. The software hp^2fem was implemented using the Matlab programming language and is fully modular and easily extensible to any problem. The formulation of large deformation considered the kinematic, stress-strain operators and constitutive equations. The solution of nonlinear system equations is performed by the Newton-Raphson method. Results are presented for the simulation of large deformation problem using neo-Hookean compressible material in 2D and 3D problems and the high order methods. Also, contact problems and shape optimization are performed here. It is concluded that the architecture works very well and that the use of the HO-FEM brings great benefits in terms of convergence rate.

Keywords: Finite Element Method, High Order Methods, Large Deformations, Nonlinear Programming, Object Oriented Programming.

Lista de Ilustrações

Lista de Tabelas

SUMÁRIO

1 Introdução

Atualmente, a busca por projetos de engenharia mais econômicos e confiáveis é a principal missão de muitas empresas em setores como automobilístico, construção civil e metal-mecânica. Dessa forma, antigas práticas de basear projetos quase unicamente na experiência da equipe de trabalho envolvida e na construção de protótipos para testes foram abandonadas. É neste cenário de projetos de alta qualidade com menor custo que a simulação computacional se torna uma ferramenta de projeto importante. Um dos procedimentos mais utilizado no momento é o Método dos Elementos Finitos (MEF). Este é um método numérico usado na solução de sistemas de equações diferenciais que descrevem uma ampla variedade de problemas de engenharia. O MEF permite que projetos sejam desenvolvidos, aperfeiçoados e validados em um ambiente de protótipos virtuais. Isso exclui a necessidade de construção de inúmeros protótipos e testes físicos, tornando assim o projeto mais econômico, seguro e controlável.

O MEF surgiu por volta dos anos 40. Desde então, tem sido, estudado e aperfeiçoado devido à crescente procura por soluções mais precisas para modelos mais complexos. É um consenso na literatura atual que o MEF de Alta Ordem (MEF-AO) é uma das ferramentas numéricas mais adequadas para lidar com problemas reais de crescente complexidade e múltipla escala. Observam-se, na sua maioria, aplicações do MEF-AO a problemas transientes de Mecânica de Fluidos, Transferência de Calor e Eletromagnetismo e muito poucos trabalhos aplicados em problemas não-lineares de Mecânica dos Sólidos.

Neste trabalho, pretende-se aplicar o MEF-AO em problemas de grandes deformações utilizando material neo-Hookeano. A lei constitutiva para o material não é mais linear, pois a cinemática que descreve o mecanismo deformação é não-linear, o que gera sistemas de equações não-lineares, cujas soluções não são triviais. No entanto, utilizando o método de Newton-Raphson, é possível aproximar esses sistemas não-lineares em sistema de equações lineares.

Problemas estruturais mais gerais somente podem ser analisados através da formulação de fenômenos não-lineares associados ao comportamento do material e às interações da estrutura. Como exemplo, tem-se o problema de contato, o qual está presente em aplicações reais como juntas mecânicas, rolamentos mecânicos, impactos de automóveis, juntas biomecânicas e muitas

outras que exibam interações mecânicas entre dois ou mais corpos.

O problema de contato é definido como dois ou mais corpos sob carregamento e condições de contorno que estarão em contato quando o espaço (*gap*) entre esses corpos é fechado. Na presença do atrito, o movimento de deslizamento dos corpos um sobre o outro é resistido por forças de fricção (MIJAR; ARORA, 2000a).

Do ponto de vista da formulação, o problema de contato apresenta dificuldades para a formulação do modelo e obtenção de solução. A superfície real de contato não é conhecida *a priori* e muito menos as tensões superficiais, sejam elas tangenciais e/ou normais. Assim, não se pode definir as condições de contorno na região de contato de forma usual. Uma condição de contorno que se aplica à região de contato corresponde a não interpenetração dos sólidos e pode ser formulada em forma de condições de desigualdade (SERPA, 1996).

Quando atrito é considerado, surgem dificuldades de formulação e complexidade do sistema de solução associado ao problema (WRIGGERS, 2006). O problema é não-linear, o contato e as leis de atrito não são contínuas e diferenciáveis, além de possuírem múltiplas soluções, o que requer uma formulação matemática rigorosa para análise e otimização não-linear.

É nesse contexto que se torna extremamente difícil criar um um software padrão de elementos finitos que consiga resolver os problemas de contato mais gerais, incluindo o efeito de atrito, com algoritmos robustos. Esse tem sido o principal desafio da comunidade científica, ou seja, desenvolver um método eficiente e robusto para o contato mecânico computacional (LAURSEN, 2002).

Existem dois tipos de formulação matemática para solução de problemas de contato: a primeira denominada de *Inequações Variacionais* e a outra *Equações Variacionais* (MIJAR; ARORA, 2000a).

Na abordagem usando inequações variacionais utilizada nos trabalhos de Panagiotopoulos (1985), Kikuschi e Oden (1988), Refaat e Meguid (1994, 1995, 1998), o problema de contato é formulado via *Princípio do Trabalhos Virtuais* (PTV) na forma de inequação. A superfície de contato não é conhecida *a priori*, sendo, assim, determinada durante o processo de solução do problema de mínimo. Pode-se empregar um Método de Penalidades ou Lagrangiano Aumentado para regulari-

zar a condição de contato e um método numérico para solução do sistema de equações não-lineares. Assim, é feito um estimador para o campo de deslocamento, e o funcional da Inequação Variacional é minimizado sujeito às condições de contorno e contato para obter o campo de deslocamento atual. O campo de deslocamento inicial e o atualizado são usados para checar se a Inequação Variacional é satisfeita ou não. O processo é iterativo até que se atinja o critério de convergência adotado. A convergência numérica é o maior problema desta formulação, uma vez que a área de contato é modificada a cada iteração.

Já a abordagem usando equações variacionais utiliza o PTV na forma de equação. Assume-se que a superfície de contato é conhecida. Forças de contato e deslocamentos são tratados como variáveis primárias. No processo de solução numérica, a carga é incrementada ao longo das iterações. A abordagem da Equação Variacional é utilizada por softwares de elementos finitos tais como ANSYS e ABAQUS. A utilização do método do Lagrangiano Aumentado, nesta abordagem, tem se mostrado preferível quando comparado ao Método das Penalidades (MIJAR; ARORA, 2000a). Existe a vantagem do melhor condicionamento numérico e no sucesso em resolver grandes sistemas com problemas de contato. Porém o rigor matemático não é tão forte como da abordagem de Inequação Variacional. Isso faz com que a Equação Variacional apresente dificuldades de formulação do problema, uma vez que certas condições como a superfície de contato são impostas ao modelo com o intuito de simplificá-lo, podendo gerar equívocos ou erros na obtenção da solução final.

Com relação ao processo de solução do problema de contato pode-se fazer uso de técnicas de otimização como o *Método do Lagrangiano Aumentado* (MLA) ou *Método das Penalidades* (MP). O Método das Penalidades possui os inconvenientes do mal condicionamento quando o parâmetro de penalidade torna-se muito grande e de a solução depender fortemente do número de passos de carregamento. Isto, ocorre devido a baixa precisão em determinar um valor de força de contato nos nós em contato. Estas dificuldades e exemplos são relatados por Mijar e Arora (2000a, 2004a), Laursen (2002), Simo e Laursen (1992), Bittencourt e Creus (1998).

O Método do Lagrangiano Aumentado possui a vantagem de ter o parâmetro de penalidade atualizado automaticamente no algoritmo sem a necessidade de interferência do usuário no valor final. É essa característica que torna o método bastante atrativo no ponto de vista computacional (MIJAR; ARORA, 2004a, 2004b).

A representação nó-segmento é a forma discretização mais utilizada para resolver o problema de contato e amplamente difundida nos pacotes comerciais de MEF. O inconveniente desse tipo de discretização surge em problemas que envolvem grandes deformações. Para esse tipo de problema, a convergência não é satisfatória como mostrado nos trabalhos de Puso e Laursen (2004a), Tur, Fuenmayor e Wriggers (2009). Assim, como visto em Puso e Laursen (2004a), Tur, Fuenmayor e Wriggers (2009), Fischer e Wriggers (2005), tem-se aplicado o chamado *Método Mortar* para discretizar o problema de contato com grandes deformações. O Método Mortar tem o benefício adicional de permitir a adoção de elementos finitos de alta ordem obtendo-se uma alta taxa de convergência e uma discretização suave dos contornos de um dado corpo.

Já a otimização de forma é uma alternativa ao processo de tentativa e erro, e tem apresentado sucesso. A otimização consiste em uma abordagem sistemática de modelagem e busca da melhor solução possível, direcionada por critérios objetivamente definidos. Em outras palavras, essa metodologia busca a sistematização da atividade de projeto. O conhecimento e a experiência acumulados são aplicados na definição de *critérios de performance* e *variáveis de projeto*, definindo de maneira única e conveniente uma solução para o problema. Ao introduzir uma abordagem genérica, a metodologia de otimização permite tratar problemas originais ou não de maneira bastante semelhante.

A aplicação desses conceitos no projeto de estruturas iniciou-se nos trabalhos de Schmit (1981), Vanderplaats (1982), Belegundu e Arora (1985a, 1985b), Ding (1986), Haftka e Grandhi (1986), Choi e Haug (1983), Choi e Seong (1986), Haug, Choi e Komkov (1986), Seong e Choi (1987), Haftka e Adelman (1989) para problemas lineares. E desde então, nos últimos trinta anos vem sofrendo grande desenvolvimento.

A formulação e implementação desses conceitos na análise de sensibilidade e otimização estrutural de parâmetros e forma em problemas lineares foi realizado pelo grupo nos trabalhos de Silva (1997), Silva e Bittencourt (1999c, 1997, 1999b, 1999a, 2000).

A partir da década de 90, verifica-se um interesse crescente nos problemas estruturais não-lineares como visto em Haftka e Adelman (1989), Choi e Santos (1987), Fancello (1993), Fancello, Haslinger e Feijóo (1995), Bestle e Eberhard (2003), Herskovits et al. (2000), Li et al. (2003a, 2003b), Kim, Choi e Chen (2000), Kovács, Szabó e Szota (2001), Kim, Park e Choi (2001), Al-Dojayli e Meguid (2002), Evgrafov e Patriksson (2003). Isto ocorre devido ao amadurecimento

dos recursos de análise (formulação, software e hardware), permitindo a modelagem mais precisa de uma série de fenômenos, utilização cada vez maior de novos materiais (como polímeros e compósitos) e a necessidade de estender os desenvolvimentos da análise de sensibilidade e otimização para problemas mais complexos. Dentre estes, destacam-se a formulação elastoplática e hiperelástica dos materiais e o problema de contato.

Ferreira (2002) desenvolveu as expressões para análise de sensiblidade em problemas de dano contínuo com aplicação em fadiga. O conceito de derivada topológica em problemas de elasticidade linear 2D e 3D para otimização de topologia pode é abordado em Driemeier (2002). Já na tese de Silva (2003) desenvolveu-se expressões para análise de sensibilidade em problemas de hiperelasticidade e grandes deformações aplicadas para otimização de forma de problemas tridimensionais .

Considerando estes tipos de problemas em mecânica estrutural, são entendidos os aspectos computacionais e as dificuldades de implementação de uma arquitetura de software que permita resolver uma ampla variedade de problemas de mecânica estrutural. Assim, a arquitetura proposta em MatLab pretende ser multidisciplinar e de fácil tradução para um linguagem mais robusta, como o C++, que permite a solução de problemas maiores.

O MatLab foi utilizado por ser um ambiente de programação bastante útil e presente em vários centros de ensino, pesquisa e engenharia. Programar sobre esse ambiente é simples, pois possui uma vasta coleção de bibliotecas ou "toolboxes" que permite que o desenvolvedor de códigos se preocupe somente com a arquitetura do seu programa e não com detalhes menores como cuidar de indexação de arrays ou conversão entre tipos de variáveis.

1.1 Objetivo

O principal objetivo deste trabalho é estudar e implementar os conceitos fundamentais para propor e construir uma arquitetura orientada por objetos que utiliza o MEF-AO para solucionar principalmente problemas estruturais não-lineares. Para tal, propõe-se construir um programa de elemento finitos de alta-ordem em MatLab orientado por objetos utilizando os conceitos do MEF-AO em modelos mecânicos estruturais.

Desta forma, esta tese é uma continuação de outros trabalhos de Vazquez (2004), Vazquez (2008), Bargos (2009) e Furlan (2011) coordenados pelo grupo do Prof Dr. Marco Lúcio Bittencourt e fornece a base para implementações de outros problemas com características não-lineares.

1.2 Contribuições do Trabalho

Como contribuição resultante do desenvolvimento desta tese de doutorado podem ser listados os seguintes itens:

- Desenvolvimento de uma arquitetura orientada por objetos para o MEF-AO;
- Implementação das funções de forma de alta-ordem a partir da tensorização das funções 1D em ambiente MATLAB usando programação orientada por objetos;
- Aplicação das funções de alta ordem em problemas de mecânica estrutural não-linear;
- Estudo e implementação de problema de contato 2D em ambiente de programação MatLab orientado por objetos para malhas estruturadas e não-estruturadas;
- Estudo e comparação de métodos de parametrização utilizados na otimização de forma: método das camadas unitárias e método dos deslocamentos fictícios;
- Implementação de scripts que permitam que o programa GiD (CIMME, 2008) gere arquivos de entrada para o hp^2fem para solução de problemas estruturais.

1.3 Organização do Texto

Esta tese possui a seguinte organização:

Capítulo 2 é dedicado a revisão bibliográfica dos conceitos que são abordados nesta tese.

Capítulo 3 aborda os elementos essenciais do MEF-AO, tais como geração das funções de forma, integração numérica e diferenciação das funções de forma.

Capítulo 4 descreve os elementos de Mecânica do Contínuo para deformação finita. Apresentam-se a formulação Lagrangiana total e a sua forma matricial usada na construção do programa de elementos finitos.

Capítulo 5 introduz os conceitos de solução do problema de contato com e sem atrito.

Capítulo 6 mostra uma introdução à otimização de forma utilizando duas forma de parametrizar o domínio: o método das camadas unitárias e o método dos deslocamento fictícios.

Capítulo 7 fornece uma visão da arquitetura do software de elementos finitos de alta ordem, denominado de hp^2fem, desenvolvido ao longo deste trabalho. Esse programa é baseada na linguagem MatLab e faz uso da tecnologia de orientação por objetos. Através da linguagem de modelagem visual de software UML, apresentam-se os diagramas de classes para facilitar o desenvolvimento e gerenciamento do código. Os detalhes de desenvolvimento estão presentes neste capítulo.

Capítulo 8 apresenta os estudos de casos para validar o algoritmo de solução de problemas de grandes deformações, contato e otimização de forma. Para isso, são utilizados dois programas de referência, FLagSHyP (desenvolvido por Bonet e Wood (2008)) e ANSYS. Aplica-se o método de elementos finitos de alta ordem, para verificar a convergência espectral para problemas de grandes deformações.

Capítulo 9 apresenta as conclusões do trabalho e as perspectivas futuras para outros trabalhos.

2 Revisão Bibliográfica

2.1 MEF-AO

Nas últimas décadas, cresceu a aplicação das versões p/hp do MEF em elasticidade linear, problemas de transferência de calor e dinâmica dos fluidos. Esse aumento se deve à alta precisão, excelente taxa convergência e possibilidade de tensorização das funções de forma, características do MEF-AO.

Os primeiros trabalhos sobre o MEF datam da década de 1940 como, por exemplo, Hrennikoff (1941), Courant (1943), que visavam aplicações em matemática. Em Hrennikoff (1941) foi resolvido um problema de elasticidade plana dividindo-se o domínio em pequenos pedaços nos quais a rigidez era aproximada por elementos de barras, vigas e molas. Foi no trabalho de Courant (1943) que se utilizou pela primeira vez o termo aproximação polinomial para solucionar um problema de Dirichlet em uma malha de triângulos. Na década de 1950, o MEF foi introduzido na comunidade de engenharia com o trabalho de Turner et al. (1956), no qual foram utilizados os procedimentos de aproximação por partes das equações diferenciais em problemas de elasticidade linear e estratégia de montagem essencial para o MEF. Entretanto, o método tornou-se mais conhecido na engenharia a partir dos anos de 1960 e 1970 com os trabalhos de Oden (1969), Zienkiewicz e Cheung (1967), Oden (1972), que aplicaram o MEF sobre o *princípio variacional*, utilizando as funções de interpolação (também conhecidas como funções de forma) para aproximar a solução de problemas de mecânica dos fluidos e mecânica estrutural.

Nas décadas de 1980 e 1990, o MEF teve o ápice de seu desenvolvimento. E o método foi divido em três variantes principais: versão-h, versão-p e versão-hp. Na *versão-h* do método, a convergência da solução numérica é obtida a partir da redução do tamanho médio (h) do elemento da malha de discretização, mantendo o grau das funções de interpolação. Como exemplo de trabalhos que utilizam o MEF na sua versão h, tem-se Bathe (1996), Hughes (2000), Johnson (1990), Zienkiewicz e Taylor (1989). Já, na *versão-p*, a convergência é obtida aumentando-se a ordem p das funções de interpolação e mantendo-se fixo o tamanho do elemento da malha de discretização. Os principais trabalhos sobre a *versão-p* do MEF são Babuska, Szabó e Katz (1981), Babuska et al. (1991), Babuska e Suri (1987), Karniadakis e Sherwin (1999), Solín, Segeth e el (2004). A versão

mista, denominada de *versão-hp*, foi também desenvolvida e aumenta-se a taxa de convergência utilizando em conjunto redução do tamanho médio da malha de elementos finitos e aumentando a ordem das funções de interpolação, como apresentado em Babuska (1988), Babuska e Guo (1988), Babuska e Suri (1990), Szabó e Babuska (1991).

Pode-se afirmar que a versão-p tem se desenvolvido muito nos últimos 20 anos é comumente denominada *Método dos Elementos Finitos de Alta Ordem* (MEF-AO). O MEF-AO se desenvolveu muito na área de Mecânica dos Fluidos, como mostrado nos trabalhos reunidos no livro de Karniadakis e Sherwin (2005). Mais tarde, o método passou a ser aplicado em problemas de elasticidade não linear, mais especificamente de elastoplasticidade, como em Jeremic e Xenophontos (1999). Neste trabalho, estudaram-se os benefícios em obter uma melhor representação da zona de cisalhamento e distribuição da deformação total para um material elastoplástico, quando o material (areia) apresenta um alto gradiente de deformação, porém contínuo em uma região bem específica.

Yosibash et al. (2007) mostraram os benefícios do MEF-AO em problemas axisimétricos envolvendo de grandes deformações e carregamentos seguidores, isto é, carregamentos que são dependentes dos vetores normais e tangenciais do elemento que se alteram no decorrer do processo de deformação finita. Neste trabalho, emprega-se material elástico compressível e compara-se a solução numérica obtida contra a solução exata. Demonstra-se também a eficiência da *versão-p* do MEF em comparação com a *versão-h*.

Apresentaram-se em Heisserer et al. (2008), as propriedades livre de restrições da *versão-p* para o deslocamento quando aplicados ao material neo-Hookeano quasi-incompressível sob deformação finita. Comparam-se a soluções semi-analítica e numérica com o objetivo de verificar a robustez do MEF-AO com respeito à restrição volumétrica. Também é derivada uma solução analítica para o caso compressível e comparada com a solução numérica.

Dong e Yosibash (2009) estudaram a taxa de convergência e os efeitos de programação paralela na solução de problemas estáticos e dinâmicos de sólidos submetidos a deformações infinitesimais e finitas (usando material neo-Hookeano compressível). Nota-se que a convergência exponencial e o paralelismo permitem a solução de problemas com grandes quantidades de graus-de-liberdade com tempo computacional menor.

Como dito anteriormente, o MEF-AO, além da característica de apresentar uma convergência exponencial, tem o benefício adicional de representar melhor geometrias complexas. Sherwin e Peiró (2002) mostraram uma técnica de geração de malha automática capaz de produzir malhas não-estruturadas e não-conformes, usando elementos de alta ordem 3D sobre uma geometria CAD. Como exemplo, é construída uma malha de tetraedros e prismas sobre a representação CAD de uma artéria de coração humano.

Outra linha de pesquisa do MEF-AO é formada por trabalhos que investigam o aspecto matemático e computacional do método. Em Bittencourt (2005) apresentam-se as funções de formas modais e nodais para triângulos e tetraedros tensorizadas a partir das funções unidimensionais expressas em coordenadas baricêntricas. As funções nodais apresentadas seguem a família dos polinômios de Lagrange que formam as funções de forma padrão da *versão-h* encontradas em Hughes (2000). Já as funções modais utilizam os polinômios de Jacobi. Bittencourt, Vazquez e Vazquez (2007) mostraram a construção das funções de forma via polinômios de Jacobi que resulta em um aumento de esparsidade das matrizes de massa e rigidez. Porém, essa esparsidade é dependente da escolha dos pesos utilizados pelos polinômios de Jacobi. Já em Bittencourt e Nogueira Jr. (2007), é verificada a eficiência das funções de forma tensorizáveis unidimensionais, apresentadas nos trabalhos antecedentes, obtendo operadores (matrizes de massa e rigidez) bem condicionados para a obtenção de solução em problemas não-lineares usando malha de tetraedros.

Na tese de Vazquez (2008), desenvolveu-se as funções de interpolação e regras de integração tensorizáveis para o MEF-AO a partir das funções unidimensionais, considerando sistemas de referências locais e elementos não-distorcidos. Já a dissertação de Bargos (2009), aplicou as funções encontradas em Vazquez (2008) para o caso de elementos distorcidos descritos no sistema de referência global.

Em Bittencourt e Vazquez (2009) desenvolveu-se um método de gerar matrizes de massa e rigidez para problemas de Poison 2D e 3D a partir de funções de forma unidimensionais de alta-ordem. Dependendo da escolha da regra de integração e ponto de colocação, o método permite a obtenção de matrizes esparsa.

No trabalho de Furlan (2011), apresentou-se algoritmos locais de integração explícitos e implícitos aplicados ao MEF-AO, baseados na decomposição por autovetores das matrizes de massa

e rigidez para problemas elastodinâmicos lineares. O procedimento de solução é realizado para cada elemento da malha e os resultados são suavizados no contorno dos elementos usando a aproximação por mínimos quadrados. O autor considera os métodos de diferença central e Newmark para o desenvolvimento dos procedimentos de solução elemento por elemento. No algoritmo local explícito, observou-se que as soluções convergem para as soluções globais obtidas com a matriz de massa consistente. Já o algoritmo local implícito necessitou de subiterações para alcançar convergência. Exemplos bi e tridimensionais de elasticidade linear e não linear são apresentados. Os resultados mostraram precisão apropriada para problemas com solução analítica.

2.2 Problema de Contato

Os primeiros estudos sobre problemas de contato datam do século XIX. Hertz (1881) investigou o contato sem atrito entre dois corpos elásticos onde apresenta-se uma solução analítica. Foi nos trabalhos de Signorini (1933) e Signorini (1959) que estabeleceu uma formulação para a definição do problema de valor de contorno e as condições de contorno associadas, para tratar o contato sem atrito entre um corpo deformável e um anteparo rígido. Esse problema é conhecido como *Problema de Contato de Signorini*.

Entre os anos de 1970 a 2000, o problema de contato foi exaustivamente estudado tanto do ponto de vista de modelo matemático como no âmbito computacional. Mijar e Arora (2000a) dividiram o problema de contato em duas correntes baseadas na forma matemática como o problema de contato é abordado. A primeira denominada de *inequação variacional*, que utiliza o princípio dos trabalhos virtuais do problema de contato na forma de inequação e soluciona o problema de contato através de um problema de mínimo. A área de contato é determinada durante o processo de solução. Na segunda abordagem, conhecida como *equação variacional*, a área de contato era conhecida (ou pelo menos estimada) e o princípio dos trabalhos virtuais foi escrito na forma de equação.

A formulação matemática do problema de contato (com e sem atrito) via inequação variacional foi discutida com todo seu rigor matemático pela monografia de Kikuschi e Oden (1988). Provas matemáticas e soluções numéricas foram discutidas através de exemplos e comparada com soluções analíticas. Neste trabalho usaram-se elementos quadrados de 9 nós para solucionar um

problema de Hertz. O problema de contato foi resolvido de forma incremental utilizando o método da penalidades. Dessa maneira, a superfície de contato e as forças de contato foram estimadas pelos parâmetros de penalidade.

Em Refaat e Meguid (1994), o problema de contato com atrito também foi formulado matematicamente por uma inequação variacional e é solucionado por um procedimento numérico que soluciona o problema em duas fases. A primeira fase soluciona o problema com respeito à condição de tensões tangenciais e define-se a área de contato. Já na segunda fase, conhecendo-se a área de contato, o problema é resolvido em termos das tensões normais. O procedimento utiliza o método da programação quadrática em conjunto com os multiplicadores de Lagrange para solucionar o problema de contato com atrito além de dispensar o uso de parâmetro de penalidades definidos pelo o usuário.

Refaat e Meguid (1998) apresentaram um algoritmo que permite resolver de forma direta um problema de contato na forma de inequação variacional. O algoritmo utiliza técnica de programação quadrática e evita a utilização de métodos de regularização. Uma estimativa inicial do deslocamento foi realizada e a inequação variacional foi minimizada sujeita às condições de contorno. Assim, foi obtido um campo de deslocamento atualizado o qual foi comparado à estimativa inicial. Se a convergência não é obtida, atualiza-se a estimativa inicial com o campo de deslocamento obtido na iteração corrente e o problema é solucionado novamente até que a convergência seja alcançada.

Outra forma de resolver a inequação variacional, mostrada em Heege e Alart (1996), é adicionando-se pseudos-potenciais, não diferenciáveis, que representam as leis de atrito e as condições de contato ao potencial elástico do sistema. O funcional resultante contém o conjunto convexo para a condição de atrito. Este conjunto convexo depende das forças de contato normal que, por sua vez, dependem do campo de deslocamentos. Tanto o campo de deslocamento como as forças de contato normal são variáveis desconhecidas. Dessa forma, o problema de contato com atrito é resolvido por um algoritmo que envolve o método de Newton-Raphson (para solucionar a equação de equilíbrio) e Lagrangiano aumentado para ajustar as forças de contato. As superfícies de contato, são descritas, neste trabalho, através de curvas parametrizadas (funções de Bézier).

Para problemas práticos de engenharia, a inequação variacional não é a abordagem adequada para a criação de algoritmos de solução de problema de contato. Prefere-se a abordagem via equa-

ções variacionais que permite a criação de algoritmos de fácil incorporação aos códigos de elementos finitos já existentes.

Em Bathe e Chaughary (1985) foi abordado o problema de contato com atrito na forma de equação variacional. Elementos finitos planos e axisimétricos foram utilizados para solucionar o problema de contato submetido à deformações finitas via método dos multiplicadores de Lagrange em conjunto com o método de Newton-Raphson. A discretização nó-segmento foi utilizada para criar as quantidades matriciais referentes ao contato e o carregamento foi aplicado em incrementos.

Já Giannakopoulos (1989) utilizou o método da penalidades para impor as condições de contato e adotou o método do mapeamento de retorno (return-mapping) para atualizar as condições de atrito em problemas de contato 2D envolvendo grandes deformações.

Um algoritmo para solucionar problemas de contato com atrito 2D com grandes deformações e plasticidade usando discretização nó-segmento foi apresentado por Simo e Laursen (1992). Este algoritmo utilizou o método do Lagrangiano aumentado e o mapeamento de retorno para ajustar as forças de contato normal e tangencial. A equação de equilíbrio do sistema é solucionado pelo método de Newton-Raphson. Neste trabalho foram comparados o método das penalidades e o Lagrangiano aumentado, sendo este último mais eficiente e robusto que o primeiro. Foi mostrado também que a escolha dos parâmetros de penalidade pode produzir mal condicionamento do sistema de equações.

Laursen e Simo (1993) apresentaram uma forma alternativa para escrever o método do Lagrangiano aumentado que permite tornar o problema de contato com atrito um problema simétrico. Este algoritmo é testado em um problema de extrusão de alumínio, utilizando um coeficiente de atrito baixo, $\mu = 0,1$. Para este caso, os resultados obtidos para operadores simétricos e não-simétricos são os mesmos.

A simetrização das matrizes de contato com atrito é objetivo do trabalho de Ju e Rowlands (1999). Aplicando a discretização nó-superfície, cria-se um procedimento para construir a matriz tangente simétrica para o problema de contato 3D. Demonstra-se que os benefícios computacionais, tais como tempo de solução menor e menos espaço de armazenamento, são alcançados pelo algoritmo proposto.

O trabalho de Bittencourt e Creus (1998) propõe um algoritmo para solucionar o contato com atrito em problemas tridimensional envolvendo grande deformação e não-linearidade do material. Para o caso de múltiplos corpos em contato, é criado um elemento de contato que mantém a sua forma plana (mesmo que a face original não tenha essa forma plana). Este elemento de contato é localizado no plano médio formado pelos nós do elemento alvo. Um procedimento de busca de contato é aplicado para cada nó contator contra os nós do elemento alvo. Para evitar interpenetrações inesperadas e não-simétricas, o algoritmo altera as definições de corpo alvo e contator a cada iteração.

Nos trabalhos de Mijar e Arora (2004a, 2004b) são apresentados uma excelente revisão bibliográfica sobre problema de contato e propõe um algoritmo, denominado de *ALM2*, que soluciona o problema de contato com atrito em duas fases utilizando o método do Lagrangiano aumentado. Na primeira fase soluciona o contato normal e na segunda o contato tangencial. O algoritmo é testado apenas para um problema de uma barra em contato com uma superfície rígida, sujeito à pequenos deslizamentos. O *ALM2* atualiza de forma automatizada os parâmetros de penalidade a cada iteração do Lagrangiano aumentado.

Os trabalhos até agora apresentados utilizam a discretização chamada de nó-segmento (ou nó-superfície, em caso de problemas 3D). Porém, em problemas de contato envolvendo grandes deslizamentos, a convergência nem sempre é obtida, como mostrado em Yang (2009). Dessa forma, a partir dos anos 2000, o problema de contato passou a ser discretizado por um método chamado de *Método Mortar*. Este método, apresentado pelos autores Bernardi, Maday e Patera (1994), Wohlmuth (2000b, 2000a), foi originalmente apresentado como um método de decomposição de domínios, usado para acoplar malhas não-conformes e/ou realizar integração numérica sobre malhas com graus não-uniformes. No método mortar, a condição de continuidade na interface é realizada na forma de uma integral ao contrário da condição nodal normalmente empregada no MEF (YANG, 2009).

Basicamente, o método mortar pode ser tratado como um problema de mínimo, sendo as variáveis desconhecidas definidas no lado da interface entre os subdomínio denominado de não-mortar. É nesse lado, que pode ser um segmento (no caso de discretização 2D) ou uma face (para discretização 3D) que realiza-se a integração numérica (BERNARDI; MADAY; PATERA, 1994). Neste mesmo trabalho, emprega-se multiplicadores de Lagrange para gerar o espaço mortar para satisfa-

zer a condição *inf-sup* (definida em Bathe (1996)).

Por permitir o acoplamento entre malhas não-conformes, o método mortar tem sido cada vez mais empregado em problema de contato. Hild (1997) estuda a convergência matemática do método mortar aplicado ao problema de contato com atrito. Já, em McDevitt e Laursen (2000) o método mortar é acoplado à lei de Coulomb para solucionar o problema de contato com atrito de forma numérica. Para isso, utiliza o conceito de superfície intermediária de contato onde é realizado o processo de integração numérica (usando a quadratura de Lobatto) das parcelas referentes ao contato. Prefere-se utilizar a regularização do contato usando o método da penalidades ao método dos multiplicadores de Lagrange. Essa escolha reduz o número de variáveis desconhecidas, apesar do método das penalidades ser muito propício a gerar sistemas de equações mal-condicionados. Apenas problemas bidimensionais (problema de Signorini e de Hertz) são resolvidos. A aplicação do método mortar ao problema de contato tridimensional é feita em Puso e Laursen (2004a), para o caso sem atrito, e em Puso e Laursen (2004b) incluindo o atrito. Em ambos os trabalhos emprega-se cinemática finita e malha tridimensional curvada para provar que o método mortar é mais eficiente e robusto que a discretização nó-segmento (ou nó-superfície). Utilizam-se pontos de integração de Gauss-Radau e para satisfazer a conservação do momento angular, reduz-se a ordem de integração para as integrais do contato.

A combinação do método mortar com polinômios de alta ordem é realizada em Fischer e Wriggers (2006). Os autores enfatizam que o uso de elementos quadráticos em discretização nó-segmento produz inconsistência na transmissão de tensões de contato, pois nessa discretização assume-se que a tensão de contato é constante ao redor do nó contator. Aliando o método mortar com polinômios quadráticos é possível incluir uma aproximação quadrática do vetor de tensão, ou seja, a distribuição de tensão na superfície de contato deixa de ser linear, produzindo resultados melhores em problemas de contato envolvendo grandes deformações. Utiliza-se método da penalidade para regularizar o contato e uma forma geométrica de atualização das forças de atrito denominada de cone de movimento (moving-cone) apresentada de forma detalhada em Wriggers (2006).

Mais recentemente, o trabalho de Konyukhov, Gene e Wall (2009) aplicou o refinamento p em conjunto com o método mortar usando uma técnica de minimização de problemas chamada de Primal-Dual Active Set Strategy (PDASS) em problema de contato sem atrito com grandes deformações.

2.3 Otimização de Forma e Análise de Sensibilidade

Segundo Silva (2003), a teoria matemática aplicada à otimização é oriunda do século XVII, no qual Euler estabeleceu o cálculo diferencial e as condições suficientes e necessárias para estabelecer um problema de mínimo (ou máximo). Foi nessa época que Lagrange expandiu essas condições para estabelecer um problema de minimização usando restrições de igualdade. Somente no século XX, Karush, Kuhn e Tucker tornaram possível a solução de um problema de mínimo sujeito a restrições de desigualdade e, dessa forma, obteve-se assim o enunciado geral do problema de otimização.

De acordo com Arora (2004), um problema de otimização estrutural é constituído por cinco elementos essenciais:

1. definição do problema;
2. coleta de informação;
3. identificação/definição das variáveis de projeto;
4. função objetivo (ou função custo);
5. identificação das restrições.

A *função objetivo*, de acordo com Christensen e Klarbring (2009), é uma função escalar diferenciável cujo valor numérico deve ser o melhor valor encontrado para o projeto em questão, levando-se em consideração as restrições que o limitam. Essa função possui como parâmetros de entrada as *variáveis de projeto* que são medidas que definem, por exemplo, a seção de uma barra, o módulo de elasticidade de um material, ou as coordenadas dos nós de uma malha de elementos finitos (ARORA, 2004).

Com relação à otimização de forma, o trabalho de Zienkiewicz e Campbell (1973) foi o primeiro a apresentar uma formulação numérica para esse tipo de otimização estrutural. Aqui, aplicou-se o MEF na solução do problema tendo como variável de projeto as coordenadas dos nós

dos elementos do contorno. Aplicou-se o método de diferenciação direta das equações discretas para a análise de sensibilidade e a programação linear sequencial para solução do problema de mínimo.

A formulação contínua da análise de sensibilidade foi desenvolvida durante a década de 1980, tendo como os trabalhos de Cea (1981), Zolésio (1981), Choi e Haug (1983). Nesses trabalhos aplicam-se o conceito de derivada material para escrever as expressões de análise de sensibilidade de funcionais de performance em relação à variação do domínio do problema estrutural na forma de integrais sobre seu contorno. Estabelece-se, também, o conceito de *campo de velocidades de projeto*, que possui um importante papel no desenvolvimento de códigos computacionais para otimização de forma. O campo de velocidades de projeto consiste na derivada da forma do domínio do problema estrutural com relação às variáveis de projeto. Por exemplo, como colocado em Korelc e Kristanié (2005), a variação das coordenadas nodais da malha de elementos finitos de um pórtico com respeito à variável de projeto que define o espaçamento do seu vão-livre forma um campo de velocidades de projeto.

A análise de sensibilidade é um procedimento crítico na obtenção da forma ótima durante o processo de otimização. Em seu trabalho, Haug, Choi e Komkov (1986) calcularam analiticamente as derivadas de performances estruturais sem a necessidade de discretização e a definição de um tamanho de passo (necessário no método de diferenças finitas). Entretanto, realizar a análise de sensibilidade na forma contínua necessita de grande conhecimento teórico-matemático para construir as expressões de sensibilidade (SILVA, 2003).

Outros dois passos cruciais da otimização de forma são a necessidade de descrever a geometria a ser otimizada na forma parametrizada e geração automática de malhas. A cada alteração de forma, uma nova malha deve ser gerada ou criam-se regras para deformar a malha inicial (SILVA, 2003). Em Bennett e Botkin (1985) foi apresentada uma abordagem integrada de técnicas e ferramentas computacionais para construção de uma ferramenta automatizada de projeto. Os autores parametrizaram o contorno do corpo em termos de funções de forma e pontos de controle. Ainda, de acordo com os autores, é vital a utilização do campo de velocidades na atualização da malha de elementos finitos. Isso evita o comportamento oscilatório do funcional de tensão e o grande número de iterações exigidos para a convergência do problema.

Os trabalhos de Ding (1986), Yang e Choi (1985) demonstram a superioridade de representação parametrizada do contorno via *B-splines* para o cálculo da sensibilidade e a necessidade de combinação entre geração automática de malha e análise adaptável em problemas lineares de componentes individuais. Segundo Silva (2003), a combinação dessas duas técnicas é proibitiva em se tratando de problemas não-lineares. Logo, o custo computacional é alto e a sequência de iterações pode se tornar instável quando a malha é regerada para todo novo projeto. Essa instabilidade se deve ao fato das expressões de sensibilidade não levar em consideração a não-linearidade do processo de regeração da malha.

Com relação à construção de algoritmos eficientes para análise de sensibilidade usando o MEF, Choi e Seong (1986) desenvolveram um procedimento de reescrever as integrais de contorno como integrais de domínio. Dessa forma, a sensibilidade da estrutura é obtida através da soma das parcelas de cada integral de domínio. Porém, esse método tem um alto custo computacional, pois o campo de velocidades de projeto deve ser avaliado no interior do domínio, onde estão a grande concentração de elementos e cuja definição não é unívoca (SILVA, 2003).

Assim, surgiu a necessidade de desenvolver uma alternativa para diminuir o custo computacional. Seong e Choi (1987) construíram o chamado *método de camada unitária* que permite construir um campo de velocidades de projeto não-nulo numa camada adjacente ao contorno e nulo no restante do domínio. Assim, as expressões de análise de sensibilidade precisam ser avaliadas somente nessa camada. Entretanto, o método possui as desvantagens de ser muito sensível a distorção de malha durante o processo de mudança de forma e ser restrito a geometria simples.

Outro método utilizado para atualizar o projeto durante o processo de otimização é o chamado *método dos deslocamento fictícios*. Neste método, discutido em Yao e Choi (1989), Silva (2003), Cada campo de velocidades de projeto é determinado através da solução do problema linear fictício (composto por um material elástico linear de módulo de elasticidade unitário e coeficiente de Poisson nulo). Cada variável de forma dá origem a um campo de velocidades não-nulo em uma parcela do contorno e a outro campo que será nulo no resto do contorno associado. Utiliza-se também a mesma malha da análise de resposta para o campo de deslocamento. A principal característica deste método é originar campos que possibilitam atualizar a malha com maior flexibilidade e ao mesmo tempo manter a malha original para passos maiores de perturbações (SILVA; BITTENCOURT, 2007).

Zhang, Beckers e Fleury (2000) criaram uma metodologia que identifica automaticamente as variáveis de projeto que são independentes para otimização de forma. A formulação é simples e reduz a todos os subproblemas de otimização em uma forma compacta sem restrições de igualdade. Além disso, é estabelecido uma análise de sensibilidade semi-analítica. A eficiência da implementação é determinada pela conexão entre o campo de velocidade que descreve a malha de elementos finitos na deformação. Elementos finitos adaptativos baseados em estimador de erro são usados para evitar uma grande distorção da malha durante o processo de otimização.

Em Choi e Chang (1994), apresentou-se um estudo bastante abrangente sobre os aspectos teóricos e exigências práticas que devem ser considerados no cálculo correto do campo de velocidades de projeto. Um dos pontos enfatizados nesse trabalho é que o cálculo da análise de sensibilidade a mudança de forma não pode ser desacoplado da malha em que foram realizados. É crítico, segundo os autores, é combinar a parametrização do modelo feito por programas CAD a um programa de geração automática de malha. Assim, fica claro que o campo de velocidades é necessário para a atualização da malha de elementos finitos. Também, são explicadas as diferenças em se computar a variação dos funcionais de performance (tais como tensão, deslocamento ou deformação) utilizando a derivada material do campo de deslocamento ou resolvendo-se um problema adjunto. O primeiro método é denominado de *método direto* e o segundo de *método adjunto*.

Silva (1997) apresentou os conceitos de otimização estrutural e análise de sensibilidade em problemas de elasticidade linear. Implementou-se o algoritmo de programação quadrática recursiva de pontos interiores de otimização desenvolvido por Herskovits (1986), Herskovits e Coelho (1989), o qual apresenta um alta taxa de convergência, apesar de ser necessário definir um ponto inicial viável. Também, utiliza-se a formulação contínua de sensibilidade e o método adjunto para escrever as expressões de sensibilidade à mudança de forma, para geometrias parametrizadas em NURBS (PIEGL; TILLER, 1997). Outro ponto importante do trabalho é a definição dos funcionais de performance estrutural desacoplados da malha de elementos finitos.

Schleupen, Maute e Ramm (2000) mostraram que na otimização de forma, em todo o processo de procura do mínimo (ou máximo), com derivada fixa, a topologia da malha deve se manter fixa. Em geral, se a malha de elementos finitos é refinada durante o processo de otimização, a convergência pode ser melhorada através do controle do erro da sensibilidade. Para isto, são utilizados indicadores de erros locais, por exemplo, norma do erro do deslocamento ou norma do erro da

tensão aplicados a um elemento.

Silva (2003) estendeu a sua dissertação de mestrado (SILVA, 1997) para problemas envolvendo não-linearidade material. Compara-se a solução numérica obtida usando o método da camada unitária e o método do contorno fictício. O autor conclui que o método da camada unitária apesar de possuir a vantagem de apresentar um custo computacional baixo, pode, em alguns casos, gerar degradação da malha. Essa desvantagem é proporcionada pela aplicação de pertubações somente em uma única camada de elementos do contorno do corpo. Já, o método de deslocamento fictício é mais caro computacionalmente, porém não se adotando um método iterativo de solução de sistemas lineares com precisão baixa, o custo computacional se reduz sem impacto significativo nos resultados de análise de sensibilidade. O maior desafio encontrado pelo autor foi acoplar o seu código de análise de sensibilidade à um gerador de malha comercial (para isso usou-se o GiD (CIMME, 2008) que permitisse a geração de automática de malha (quando a medida de distorção de elementos pedia alteração da malha) e descrição da geometria de forma parametrizada (nesse caso utilizou-se NURBS para descrição dos contornos).

Muñoz-Rojas, Fonseca e Creus (2004) desenvolveram um método alternativo ao os métodos da camada unitária e dos deslocamentos fictícios. Este método consiste no uso de suavizações Laplacianas simplificadas (relocação nodal) para regularizar a malha durante o processo de otimização em problemas de grandes deformações.

A aplicação dos fundamentos matemáticos e computacionais da análise de sensibilidade e otimização de forma em problema de contato linear sem atrito foram discutidos em Fancello (1993). Devido à não-diferenciabilidade da equação de estado para esse tipo de problema, apenas funcionais que não dependem do campo de deslocamento (tais como energia de deformação total e energia recíproca) foram empregados. O trabalho emprega a formulação contínua de sensibilidade e uma caracterização da geometria independente da malha de elementos finitos, tornando-se assim o problema de otimização independente de uma discretização particular do domínio.

Em Kim (1999) empregou-se o método direto para o cálculo da derivada material do campo de deslocamento para o problema de contato com atrito envolvendo grandes deformações, não-linearidade material (material elasto-plástico) e problema de impacto. Para a análise de resposta, aplicou-se o método conhecido como *MeshFree*. Nesse método, as respostas são obtidas com a

mesma precisão, se comparadas ao MEF de baixa ordem, usando um menor número de graus-de-liberdade em situações de grandes distorções de malha. Os resultados foram estendidos para aplicação em problemas de conformação mecânica nos trabalhos de Kim, Choi e Chen (2001) (problemas 2D) e Kim, Kyung e Choi (2002) (problemas 3D). Porém, em geral, as matrizes são mais densamente provoadas e o custo de integração pode ser bem mais caro.

Desenvolveu-se em Herskovits et al. (2000) um algoritmo para otimização de problema de contato sem atrito, tendo como função objetivo o volume (ou massa) do corpo sujeito sujeito a um determinado valor de tensão de von Mises. O autor divide o problema de otimização em dois níveis. No primeiro nível a função objetivo é minimizada com respeito a variável de projeto. Já, o segundo nível, resolve-se o problema de contato através de minimização da sua função de estado tendo como restrição de desigualdade a condição de não penetração dos corpos, obtendo-se, assim, o campo de deslocamento. Utiliza-se, como procedimento de otimização o método de pontos interiores de Herskovits (HERSKOVITS, 1986; HERSKOVITS; COELHO, 1989).

Stupkiewicz et al. (2002) usaram um pacote de algebra simbólica (AceGen) para realizar a diferenciação direta e, assim, desenvolver as equações de sensibilidade em um problema de contato envolvendo plasticidade finita bidimensional. Devido aos efeitos de grandes deformações, o autor utiliza a discretização nó-segmento em conjunto ao método das penalidades para modelar e resolver o problema de contato.

Páczelt e Szabó (2002) mostraram os benefícios na utilização da versão hp do MEF para solução numérica do problema de otimização entre dois corpos cilíndricos em contato. Assumem-se pequenas deformações e linearidade material em problemas bidimensionais. A malha é sempre ajustada, usando um procedimento de recolocação nodal, para evitar aparecimentos de singularidades na interface do contato.

Choi e Kim (2005b) reuniram todos os trabalhos desenvolvidos pelo seu grupo da Universidade de Iowa com relação ao estudo de otimização de forma em problemas de contato. São mostrados resultados envolvendo elasto-plasticidade plana e problemas de impacto. Esse livro forneceu as bases iniciais para o desenvolvimento deste presente trabalho.

Também em 2005, Páczelt e Mróz (2005) estudaram a forma ótima do contato levando em

consideração os efeitos do desgaste provocado pelo atrito. Para isso, considerou-se a minimização de três funcionais denominados de taxa de desgaste volumétrico generalizado, potência de dissipação de atrito generalizado e potência de dissipação do desgaste generalizado. Todos estes funcionais são dependentes da pressão de contato normal e da variação do deslocamento tangencial e produzem diferentes áreas ótimas de contato. Utiliza-se a versão p do MEF para obter melhor convergência na zona de contato. O autor mostra que para $p = 8$ obtém-se a convergência desejada. Mais tarde, Páczelt e Mróz (2007) estenderam o trabalho anterior, estudando agora os feitos do deslizamento tangencial durante o desgaste. O processo de desgaste assume uma dependência não-linear com relação à tração de atrito e à velocidade relativa de deslizamento. Quando aplica-se um desgaste uniforme sobre a região de contato durante o deslizamento, o estado estacionário é obtido. Demonstra-se que no regime estacionário os parâmetros usados na formulação do desgaste são dependentes um dos outros.

Acharjee e Zabaras (2006) aplicaram a análise de sensibilidade contínua na obtenção da forma ótima em projetos de ferramentaria, onde a situação de contato mecânico tridimensional é bastante presente. Aplica-se o procedimento de diferenciação direta para obter as expressões de sensibilidade. Utilizou-se o algoritmo proposto por Laursen e Simo (1993) para resolver o problemas de contato. As superfícies de contato foram parametrizadas usando curvas de *Bézier*.

Recentemente, o estudo de problemas de otimização de forma em contato estão mais voltados para aplicação e uso de softwares comerciais. Em Vondrák e Kozubek (2009) foram utilizadas técnicas de computação paralela e decomposição de domínio para determinar a forma ótima de contato tridimensional sem atrito. A dissertação de Mirisola (2009) estudou a otimização da superfície de contato do olhal de uma biela usando o programa comercial ANSYS.

3 Construção de Funções de Forma para o Método dos Elementos Finitos de Alta Ordem

O MEF é um método numérico utilizado na solução de sistemas de equações diferenciais parciais. Basicamente, o MEF padrão particiona o domínio de interesse em diversos subdomínios representados por elementos, cuja solução pode ser aproximada por funções polinomiais lineares dentro deste subdomínio. Para melhorar a precisão da solução, muitas vezes o domínio é refinado, ou seja, o tamanho médio dos elementos é reduzido. Já, o MEF-AO tem como característica aplicar polinômios de alta ordem para aproximar a solução e melhorar a precisão e a taxa de convergência aumentando a ordem das funções.

Este capítulo tem como objetivo apresentar uma breve descrição e levantar os principias conceitos do MEF-AO. Isto facilitará a discussão sobre a aplicação do método em problemas de não-linearidade geométrica. Os trabalhos de Karniadakis e Sherwin (2005), Vazquez (2008), Bargos (2009), Vos (2011) são as principais fontes empregadas.

3.1 Formulação de Galerkin

Para a solução de um problema de valor de contorno, o MEF utiliza a formulação variacional ou a forma fraca do problema e a solução aproximada das equações variacionais.

Por exemplo, considere a seguinte equação diferencial unidimensional definida em um domínio fechado $\Omega_{[a,b]}$

$$\frac{d^2u(x)}{dx^2} = f(x) \tag{3.1}$$

com condições de contorno apropriadas, por exemplo,

$$u(a) = q \text{ (condição de Dirichlet) e} \tag{3.2a}$$

$$-\frac{du(b)}{dx} = h \text{ (condição de Neumann).} \tag{3.2b}$$

No método de Galerkin[1], um caso particular do *método dos resíduos ponderados* (KARNIADAKIS; SHERWIN, 2005), a forma fraca relativa à (3.1) pode ser obtida pré-multiplicando os dois lados da equação por uma função peso $v \in \mathcal{V}$ (sendo $\mathcal{V}$ o espaço de funções teste que possuem derivadas quadrado-integráveis e cinematicamente admissíveis, $v(b) = 0$ onde $u(b) = \bar{u}$).

Integra-se o resultado sobre o domínio Ω, para encontrar $u \in \mathcal{U}$ (espaço de soluções admissíveis) tal que

$$\int_\Omega \frac{dv}{dx}\frac{du}{dx}dx = \int_\Omega vf\,dx + \left[v\frac{du}{dx}\right]_a^b, \qquad \forall v \in \mathcal{V}. \tag{3.3}$$

A equação (3.3) é também chamada de equação variacional e a sua solução é conhecida como solução fraca ou generalizada (HUGHES, 2000).

Pode-se escrever o produto interno da equação (3.3) na forma bilinear $a(u, v)$ discutida em Rektorys (1980), ou seja,

$$a(v, u) = (v, f) \qquad \forall v \in \mathcal{V}. \tag{3.4}$$

O próximo passo no método de Galerkin consiste em criar uma aproximação finita de $\mathcal{U}$ e $\mathcal{V}$. Estes espaços de funções são definidas por $\mathcal{U}^h$ e $\mathcal{V}^h$, respectivamente. O superescrito h se refere à *discretização* ou *malha*, do domínio Ω, parametrizado por um tamanho característico h. Pode-se dizer que $\mathcal{U}^h$ e $\mathcal{V}^h$ são subespaços de $\mathcal{U}$ e $\mathcal{V}$, respectivamente.

A solução aproximada é representada por uma combinação linear das funções de *base* (também denominadas de *forma* ou *interpolação*), isto é,

$$u^h = \sum_{i=1}^{n} N_i \hat{u}_i, \tag{3.5}$$

sendo N_i as funções de forma.

[1] O método dos resíduos ponderados é anterior à formulação de Galerkin. No método dos resíduos ponderados a solução é aproximada através de uma combinação linear de funções de uma base. Maiores detalhes sobre o método é obtido em Hughes (2000)

Adotando a mesma discretização para a função v, o problema discretizado a ser solucionado para u_i é dado por

$$\sum_{i=1}^{n} a(N_j, N_i)u_i = (N_j, f), \qquad \forall i, j \in \mathcal{N}. \tag{3.6}$$

Adotando a notação matricial, obtém-se

$$\mathbf{A}\mathbf{u} = \mathbf{f}, \tag{3.7}$$

sendo $\mathbf{u}$ o vetor de coeficientes u_i, $\mathbf{A}$ a matriz do sistema cuja a forma é

$$\mathbf{A}[j, i] = a(N_j, N_i), \tag{3.8}$$

e o vetor $\mathbf{f}$ é dado por

$$\mathbf{f}[j] = (N_j, f). \tag{3.9}$$

3.2 Funções de Forma

A discretização em elementos finitos começa pela decomposição do domínio Ω em uma malha $|\boldsymbol{\Gamma}|$ formada por elementos de forma regulares como por exemplo quadriláteros e/ou triângulos tal que

$$\Omega = \bigcup_{\gamma \in \boldsymbol{\Gamma}} \Omega_\gamma \tag{3.10}$$

Cada elemento do domínio Ω_γ é imagem de um elemento do *espaço de referência* Ω_{ref}. Por exemplo, para um elemento quadrado qualquer, existe uma relação biunívoca relacionando as coordenadas físicas (x_1, x_2) do elemento Ω_γ à coordenada dos sistema de referência (ξ_1, ξ_2). Essa

relação, denominada de mapeamento, para esse exemplo, é dada por

$$x_1 = map_1^{\gamma}(\xi_1, \xi_2), \tag{3.11}$$

$$x_2 = map_2^{\gamma}(\xi_1, \xi_2) \tag{3.12}$$

e poder ser visualizada na Figura 3.1.

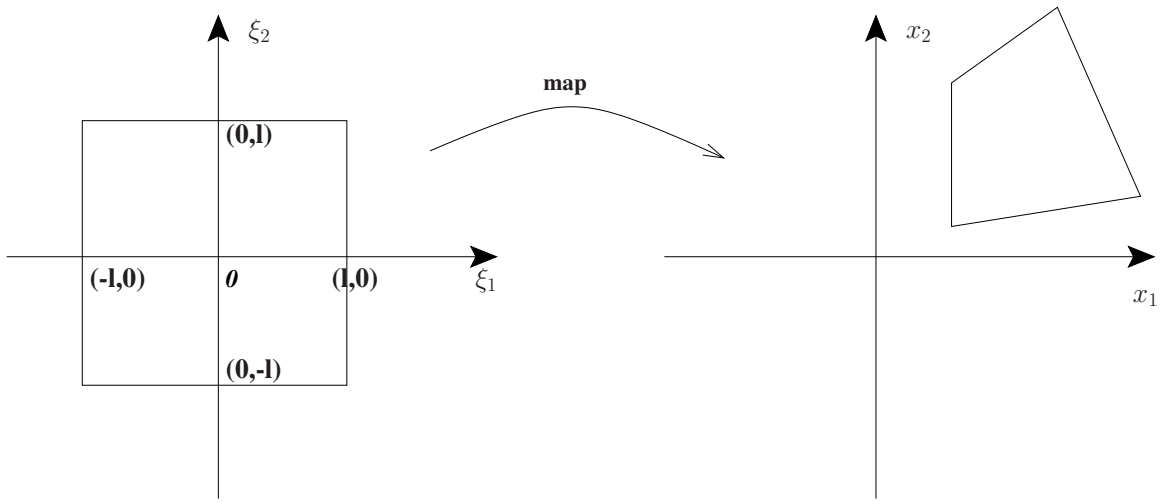


Figura 3.1: Relação entre o espaço de referência e o espaço de domínio para um elemento quadrilátero.

Analogamente, pode-se estender o mapeamento para a as coordenada x_3, ou seja,

$$x_1 = map_1^{\gamma}(\xi_1, \xi_2, \xi_3); \tag{3.13}$$

$$x_2 = map_2^{\gamma}(\xi_1, \xi_2, \xi_3); \tag{3.14}$$

$$x_3 = map_3^{\gamma}(\xi_1, \xi_2, \xi_3). \tag{3.15}$$

3.2.1 Funções de Forma 1D

Um dos principais aspectos ao se estudar o método de elementos finitos de alta ordem é estudar a técnica conhecida como *expansão da base* ou *tensorização* que consiste na construção de funções de forma 2D e/ou 3D a partir do produto tensorial das funções 1D como visto em Vazquez

(2008).

Uma das possibilidades é escrever as funções de forma unidimensionais na base nodal em termos dos polinômios de Lagrange[2]. Supondo um elemento unidimensional definido $\Omega = \{\xi_1 | -1 \leq \xi_1 \leq 1\}$ e dado um conjunto de $P+1$ pontos nodais nesse domínio, os polinômios de Lagrange podem ser escritos na forma de quociente de produtos da seguinte maneira

$$h_p^P(\xi_1) = \frac{\Pi_{q=0,q\neq p}^P(\xi_1 - \xi_{1_q})}{\Pi_{q=0,q\neq p}^P(\xi_{1_p} - \xi_{1_q})}, \tag{3.16}$$

sendo $h_p^0(\xi_1) = 1$

Costuma-se empregar a seguinte notação para definir as funções de forma de vértices e internas (VOS, 2011)

$$\phi_p(\xi_1) = \begin{cases} \frac{1}{2}(1-\xi_1)\mathcal{L}_p^{P-1}(\xi_1), & p = 0, \\ \frac{1}{2}(1+\xi_1)\mathcal{L}_p^{P-1}(\xi_1), & p = P, \\ \frac{1}{4}(1-\xi_1)(1+\xi_1)\mathcal{L}_p^{P-2}(\xi_1), & 0 < p < P. \end{cases} \tag{3.17}$$

sendo

$$\mathcal{L}_p^{P-1}(\xi_1) = -\frac{\Pi_{q=1,q\neq p}^P(\xi_1 - \xi_{1_q})}{\Pi_{q=0,q\neq p}^P(\xi_{1_p} - \xi_{1_q})}, \tag{3.18}$$

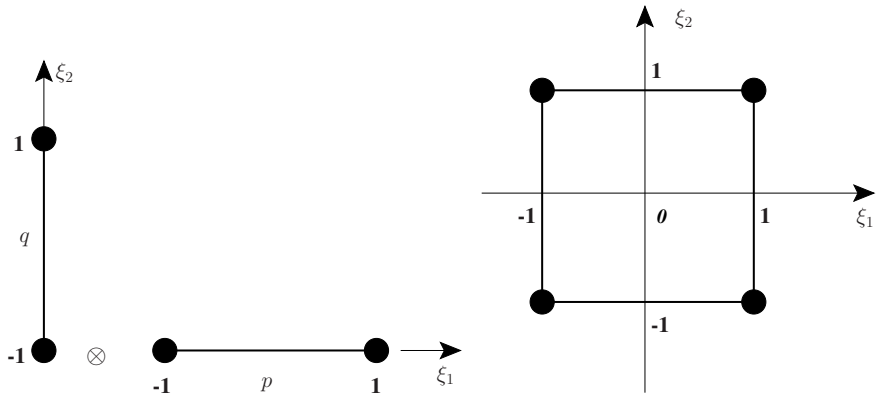
$$\mathcal{L}_p^{P-2}(\xi_1) = -4\frac{\Pi_{q=1,q\neq p}^{P-1}(\xi_1 - \xi_{1_q})}{\Pi_{q=0,q\neq p}^P(\xi_{1_p} - \xi_{1_q})}, \tag{3.19}$$

[2]Existem outras possibilidades de escrever as funções de forma. Por exemplo, em Karniadakis e Sherwin (2005) são utilizados polinômios de Jacobi para descrever as funções de forma. O polinômio de Jacobi é muito utilizado para representar funções de alta-ordem por serem ortogonais.

3.2.2 Funções de Forma 2D e 3D

As funções 2D e 3D estudas neste trabalho consistem nas funções de forma para os elementos quadrado e hexaedro. Dessa maneira, não serão tratados aqui, a construção das funções de forma para os elementos triângulo e tetraedro (VAZQUEZ, 2008; BARGOS, 2009).

O elemento quadrilateral definido em Ω_{ref} é dado por um quadrado unitário $\mathcal{Q}^2 = \{(\xi_1, \xi_2) \in [-1, 1] \otimes [-1, 1]\}$. Assim, as funções de forma para este elemento serão dados pelo produto tensorial das funções unidimensionais nas direções (ξ_1, ξ_2) através dos índices p e q mostrados na Figura 3.2



(a) Procedimento de tensorização. (b) Elemento quadrado definido em Ω_{ref}.

Figura 3.2: Construção das funções de forma para quadrados através da tensorização das funções unidimensionais (VAZQUEZ, 2008).

Para o elemento quadrado, a forma geral das funções de forma que representam o resultado da tensorição das funções 1D é dado por

$$N_n(\xi_1, \xi_2) = \phi_p(\xi_1)\phi_q(\xi_2), \;\; 0 \leq p, q \leq P \tag{3.20}$$

e a solução aproximada $u(\xi_1, \xi_2)$ é obtida pela seguinte expressão

$$u(\xi_1, \xi_2) = \sum_{n \in \mathcal{N}} N_n(\xi_1, \xi_2) u_n = \sum_{p=0}^{P} \sum_{q=0}^{P} \phi_p(\xi_1)\phi_q(\xi_2) u_{pq}. \tag{3.21}$$

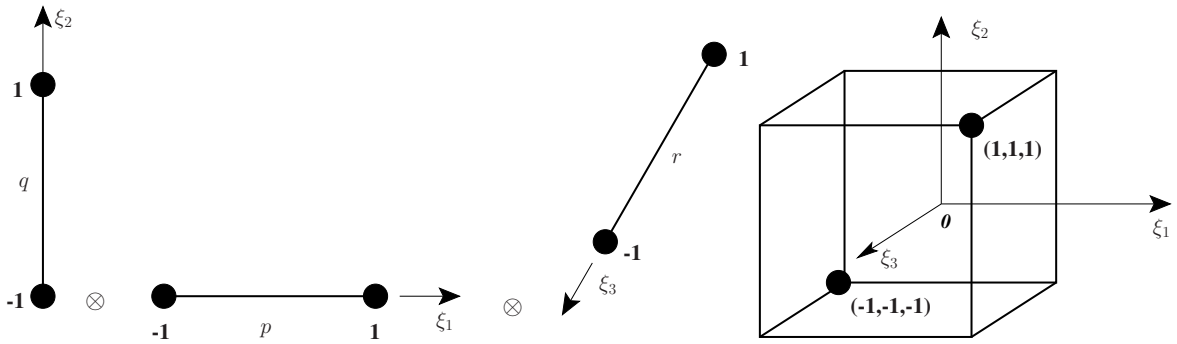
Já para o hexaedro unitário, dado por $\mathcal{Q}^3 = \{(\xi_1, \xi_2, \xi_3) \in [-1, 1] \otimes [-1, 1] \otimes [[-1, 1]\}$, as suas funções de forma são dadas por

$$N_n(\xi_1, \xi_2, \xi_3) = \phi_p(\xi_1)\phi_q(\xi_2)\phi_r(\xi_3), \;\; 0 \leq p, q, r \leq P. \tag{3.22}$$

De forma análoga ao quadrado, a solução aproximada é

$$u(\xi_1, \xi_2, \xi_3) = \sum_{n \in \mathcal{N}} N_n(\xi_1, \xi_2, \xi_3) u_n = \sum_{p=0}^{P} \sum_{q=0}^{P} \sum_{r=0}^{P} \phi_p(\xi_1)\phi_q(\xi_2)\phi_r(\xi_3) u_{pqr}. \tag{3.23}$$

A Figura 3.3 ilustra o processo de tensorização usando os índices p, q e r.



(a) Procedimento de tensorização.

(b) Elemento hexaedro definido em Ω_{ref}.

Figura 3.3: Construção das funções de forma para hexaedros através da tensorização das funções unidimensionais (VAZQUEZ, 2008).

O uso de produto tensorial para outros elementos tais como triângulos e tetraedros podem ser vistas nos trabalhos de Karniadakis e Sherwin (2005) e Vazquez (2008).

3.3 Integração Numérica

A integração numérica é um procedimento numérico intrínseco ao MEF, sendo a quadratura de Gauss a técnica de integração mais utilizada por sua alta precisão em integrar funções sendo o integrando, $u(\xi)$ suave o suficiente.

O conceito fundamental da quadratura de Gauss consiste em aproximar uma integral por uma soma finita. Para uma função unidimensional $u(\xi)$, definida em $\mathcal{Q}^1 = \{\xi \in [-1, 1]\}$, a integral numérica usando o quadratura de Gauss para $Q - 1$ pontos é representada por

$$\int_{-1}^{1} u(\xi) d\xi \approx \sum_{i=0}^{Q-1} w_i u(\xi_i), \tag{3.24}$$

sendo w_i e ξ, respectivamente, o peso e a coordenada do ponto de integração para Q pontos de integração dentro do segmento.

Como mostrado nos trabalhos Karniadakis e Sherwin (2005), Bittencourt (2005), Vazquez (2008) e Bargos (2009), a inclusão ou não dos nós finais do segmento pode definir regras de integração diferentes como Gauss-Legendre, Gauss-Radau-Legendre, Gauss-Lobatto-Legendre.

Para um quadrado unitário $\mathcal{Q}^2$ e um hexaedro unitário $\mathcal{Q}^3$ a integração numérica sobre esses domínios são definidos da seguinte forma

$$\int_{-1}^{1}\int_{-1}^{1} u(\xi_1, \xi_2) d\xi_1 d\xi_2 \approx \sum_{i=0}^{Q_1-1}\sum_{j=0}^{Q_2-1} w_i w_j u(\xi_{1_i}, \xi_{2_j}), \tag{3.25}$$

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} u(\xi_1, \xi_2, \xi_3) d\xi_1 d\xi_2 d\xi_3 \approx \sum_{i=0}^{Q_1-1} w_i \sum_{j=0}^{Q_2-1} w_j \sum_{k=0}^{Q_3-1} w_i w_j w_k u(\xi_{1_i}, \xi_{2_j}, \xi_{3_k}), \tag{3.26}$$

respectivamente.

Porém, nem sempre o elemento unitário definido em Ω_{ref} coincide com a forma do elemento

definido no espaço em Ω_γ. Dessa forma, a solução interpolada para um elemento quadrado arbitrário deve ser definida como

$$u(x_1, x_2) = \sum_{n \in \mathcal{N}} N_n(map_1^{-1}(x_1, x_2), (map_2^{-1}(x_1, x_2))u_n, \tag{3.27}$$

sendo map_i^-1 um operador de transformação de coordenada de (x_1, x_2) para (ξ_1, ξ_2). De maneira semelhante, o conceito se estende para um elemento hexaedro arbitrário definido em Ω_γ como

$$u(x_1, x_2, x_3) = \sum_{n \in \mathcal{N}} N_n(map_1^{-1}(x_1, x_2, x_3), (map_2^{-1}(x_1, x_2, x_3), (map_3^{-1}(x_1, x_2, x_3))u_n. \tag{3.28}$$

Assim, para realizar a integração numérica de uma função tridimensional descrita em um hexaedro arbitrário Ω_γ deve se aplicar uma mudança de coordenada de tal forma que

$$\int_{\Omega_\gamma} u(x_1, x_2, xi_3) dx_1 dx_2 dx_3 = \int_{\Omega_{ref}} u(\xi_1, \xi_2, \xi_3) \| det(J) \| d\xi_1 d\xi_2 d\xi_3, \tag{3.29}$$

sendo J a *matriz do Jacobiano* que define a transformação de coordenada. A expressão geral é dada por

$$det(J) = \begin{vmatrix} \frac{\partial x_1}{\partial \xi_1} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \\ \frac{\partial x_1}{\partial \xi_2} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \\ \frac{\partial x_1}{\partial \xi_3} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \end{vmatrix}. \tag{3.30}$$

A expressão para um quadrado arbitrário pode ser obtida omitindo-se o termos dependentes de ξ_3 e x_3.

3.4 Cálculo das Derivadas

Sabendo que em muitos problemas de mecânica estrutural o MEF faz uso de derivadas na definição de operadores é necessário, dessa maneira, calcular o valor dessa derivada nos pontos de quadratura para resolver a integral considerada. Emprega-se a *diferenciação por colocação* conforme Vos (2011).

Considerando uma função arbitrária unidimensional $u(\xi)$, definida em $\mathcal{Q}^1 = \{\xi_1 \in [-1, 1]\}$, esta função pode ser aproximada por

$$u(\xi_1) \approx u^h(\xi_1) = \sum_{j=0}^{Q-1} h_j(\xi_1) u(\xi_{1_j}), \tag{3.31}$$

sendo Q o número de pontos de integração, h_j são os polinômios de Lagrange dados pela seguinte equação (3.16). Assim, a derivada de $u(\xi)$ é calculada por

$$\frac{du(\xi_i)}{d\xi} \approx \frac{du^h(\xi_i)}{d\xi} = \sum_{j=0}^{Q-1} d_{ij} u(\xi_j), \tag{3.32}$$

sendo d_{ij} o elemento da matriz de diferenciação dada por

$$d_{ij} = \frac{dh_j(\xi_i)}{d\xi}. \tag{3.33}$$

Assim, com essa técnica pode-se calcular o valor da derivada da função no ponto de integração baseado nos valores da função calculada no mesmo ponto.

Por exemplo, para um elemento quadrado unitário $\mathcal{Q}^2$, a derivada parcial com respeito a ξ_1 pode ser resolvida como

$$\frac{\partial u(\xi_{1_r}, \xi_{2_s})}{\partial \xi_1} \approx \frac{\partial u^h(\xi_{1_r}, \xi_{2_s})}{\partial \xi_1} = \sum_{i=0}^{Q-1} \sum_{j=0}^{Q-1} \frac{dh_i(\xi_{1_r})}{d\xi_1} h_j(\xi_{2_s}) u(\xi_{1_i}, \xi_{2_j}). \tag{3.34}$$

Devido à propriedade de colocação para a representação de Lagrange, $h_j(\xi_s) = \delta_{js}$, pode-se simplificar a equação (3.34) para a seguinte forma (VOS, 2011)

$$\frac{\partial u^h(\xi_{1_r}, \xi_{2_s})}{\partial \xi_1} = \sum_{i=0}^{Q-1} \sum_{j=0}^{Q-1} d_{ri} \delta_{js} u(\xi_{1_i}, \xi_{2_s}) = \sum_{i=0}^{Q-1} d_{ri} u(\xi_{1_i}, \xi_{2_j}). \tag{3.35}$$

sendo d_{ri} o coeficiente da matriz de derivadas dada por (3.33). A derivada parcial com relação ξ_2 é expressa por

$$\frac{\partial u^h(\xi_{1_r}, \xi_{2_s})}{\partial \xi_2} = \sum_{i=0}^{Q-1} d_{si} u(\xi_{1_r}, \xi_{2_i}). \tag{3.36}$$

O conceito é facilmente estendido para um elemento hexaedro qualquer, bastando acrescentar as coordenadas ξ_3 e x_3. Assim a derivada parcial com relação a ξ_3 á calculada como

$$\frac{\partial u^h(\xi_{1_r}, \xi_{2_s}, \xi_{3_t})}{\partial \xi_3} = \sum_{i=0}^{Q-1} d_{ti} u(\xi_{1_r}, \xi_{2_s}, \xi_{3_i}). \tag{3.37}$$

O próximo passo é apresentar a diferenciação com relação as coordenadas (x_1, x_2, x_3). essa diferenciação é feita utilizando a inversa da matriz do Jacobiano, ou seja,

$$\begin{vmatrix} \frac{\partial u}{\partial x_1} \\ \frac{\partial u}{\partial x_2} \\ \frac{\partial u}{\partial x_3} \end{vmatrix} = \begin{bmatrix} \frac{\partial x_1}{\partial \xi_1} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \\ \frac{\partial x_1}{\partial \xi_2} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \\ \frac{\partial x_1}{\partial \xi_3} & \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_1} \end{bmatrix}^{-1} \begin{vmatrix} \frac{\partial u}{\partial \xi_1} \\ \frac{\partial u}{\partial \xi_2} \\ \frac{\partial u}{\partial \xi_3} \end{vmatrix}. \tag{3.38}$$

Se o elemento possuir lados curvos, ou o elemento for muito distorcido, o Jacobiano será próximo de zero ou até mesmo negativo, e a integração não será exata.

Os conceitos do MEF-AO apresentados anteriormente trazem uma série de dificuldades computacionais. Cabe mencionar a geração de pontos de integração e colocação, assim como das funções de interpolação para elementos 2D e 3D. Além disso, as matrizes dos elementos são mais densas e com maior números de condição acarretando um grande esforço de armazenamento e

computação. Esses aspectos são cruciais no desenvolvimento de uma arquitetura de software para o MEF-AO.

Assim, esse capítulo apresentou todos os elementos primordiais na formulação do MEF-AO que serão empregados na discretização do problema de grandes deformações e na construção de um programa de MEF-AO orientado por objetos.

4 Problemas de Grandes Deformações

Na descrição linear do movimento de um corpo sólido, assume-se que os deslocamentos e as deformações são infinitesimais e a resposta do material é linear. Nesse contexto, não existem diferenças entre as configurações deformada e não deformada. No entanto, em problemas de não linearidade geométrica, os efeitos da mudança da geometria são considerados, pois tais mudanças muitas vezes são grandes. Assim, torna-se imprescindível realizar a distinção entre as geometrias não-deformada e deformada. Consequentemente, são feitas distinções entre as várias medidas de deformação e tensão.

Para o caso de grandes deformações, determinar a resposta do material do corpo com relação à sua deformação é de extrema importância. Dessa forma, estabelece-se o equacionamento matemático que representa o comportamento idealizado do material em questão. Esse equacionamento matemático é denominado de *lei constitutiva*. O enfoque, aqui, será dado às equações constitutivas que correlacionam as componentes de tensão às componentes de deformação dentro do regime não-linear. Será utilizado o material neo-Hookeano compressível, considerado pela literatura como o mais simples dos materiais hiperelásticos.

Este capítulo apresenta os conceitos de Mecânica do Contínuo não linear para que em seguida desenvolvam-se as formas matriciais a serem implementadas em um código de elementos finitos de alta ordem. Os trabalhos Reddy (2006), Bathe (1996), Belytschko, Liu e Moran (2000), Bonet e Wood (2008), Bhatti (2006), Holzapfel (2000), Ogden (1997), BAŞAR e Weichert (2000) são excelentes referências sobre o assunto e encontram-se outros tipos de materiais como os materiais de Ogden e Mooney-Rivlin, os quais não serão tratados nesse presente trabalho.

4.1 Cinemática do Contínuo

Considera-se um cenário no qual um corpo deformável é submetido a esforços externos. Tais esforços fazem com que a geometria inicial seja alterada, ocorrendo, assim, uma deformação. Segundo Reddy (2006), se o carregamento aplicado for dependente do tempo, a deformação do corpo será uma função do tempo. No entanto, se o carregamento for aplicado vagarosamente (em

pequenas quantidades a cada instante), a deformação será dependente apenas desse incremento de carregamento. Em ambas as considerações, as forças internas e externas se equilibram ao final do processo de carregamento do corpo.

Para entender melhor o processo de deformação do corpo, a Mecânica do Contínuo define os conceitos de *configuração* e *movimento*. Também, é preciso estabelecer as nomenclaturas e notações que serão utilizadas nesse trabalho. Dessa forma, seja o corpo deformável, representado na Figura 4.1, submetido a um processo de deformação. No tempo $t = 0$, tem-se a *configuração inicial* representada por $\mathcal{C}_0$. Supondo que o processo de deformação se encerre em $t = 1$, $\mathcal{C}_1$ representará a *configuração deformada*. Na configuração inicial, o ponto P ocupa a posição $\mathbf{X}(X_1, X_2, X_3)$, sendo (X_1, X_2, X_3) as *coordenadas materiais*. Depois de aplicado um carregamento, o corpo deforma-se e o ponto P passa a ocupar a nova posição $\mathbf{x}(x_1, x_2, x_3)$, sendo (x_1, x_2, x_3) as *coordenadas espaciais* do ponto P na configuração deformada.

O movimento pode ser expresso por um mapeamento ϕ entre as configurações inicial e deformada, isto é,

$$\mathbf{x} = \phi(\mathbf{X}, t) \tag{4.1}$$

Na descrição *Lagrangiana*, o movimento do corpo é dado em termos da configuração de referência. Frequentemente, adota-se a configuração inicial como configuração de referência. Nesta descrição, a coordenada corrente é expressa em função das coordenadas materiais $\mathbf{X} = (X_1, X_2, X_3)$ e do tempo t, ou seja,

$$\mathbf{x} = \mathbf{x}(\mathbf{X}, t). \tag{4.2}$$

As quantidades medidas sobre o corpo são descritas, também, em termos das coordenadas materiais e do tempo t. Por exemplo, para a densidade do corpo denota-se

$$\rho = \rho(\mathbf{X}, t). \tag{4.3}$$

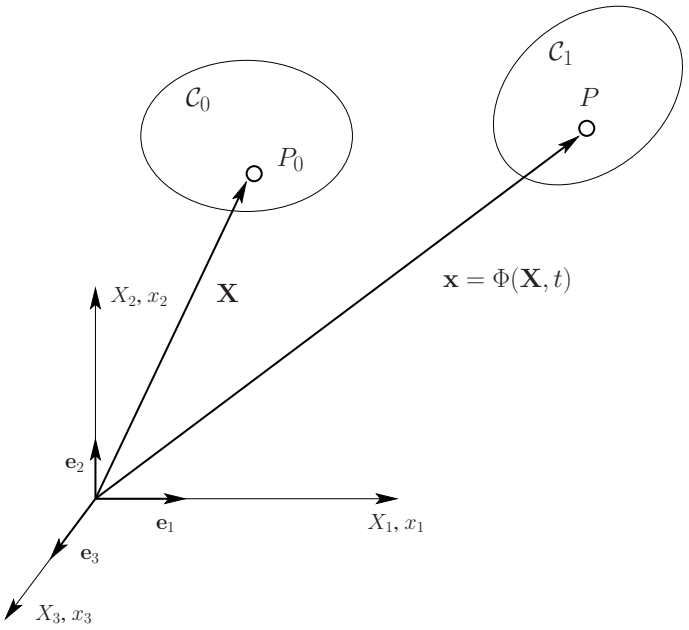


Figura 4.1: Descrições material e espacial e o vetor de deslocamentos $\mathbf{u}$ (BONET; WOOD, 2008).

4.1.1 Tensor Gradiente de Deformação

Considere dois pontos P e Q de um corpo, como mostrado na Figura 4.2, na configuração inicial $\mathcal{C}_0$. Os vetores posição de P e Q são denotados por $\mathbf{X}_P$ e $\mathbf{X}_Q$, respectivamente. A posição relativa entre P e Q em $\mathcal{C}_0$ é dada por

$$d\mathbf{X} = \mathbf{X}_Q - \mathbf{X}_P. \tag{4.4}$$

Após o processo de deformação, os pontos P e Q são denotados, respectivamente, por p e q na configuração deformada $\mathcal{C}$. O vetor posição relativa na descrição $\mathcal{C}$ é representado da seguinte forma

$$d\mathbf{x} = \mathbf{x}_q - \mathbf{x}_p. \tag{4.5}$$

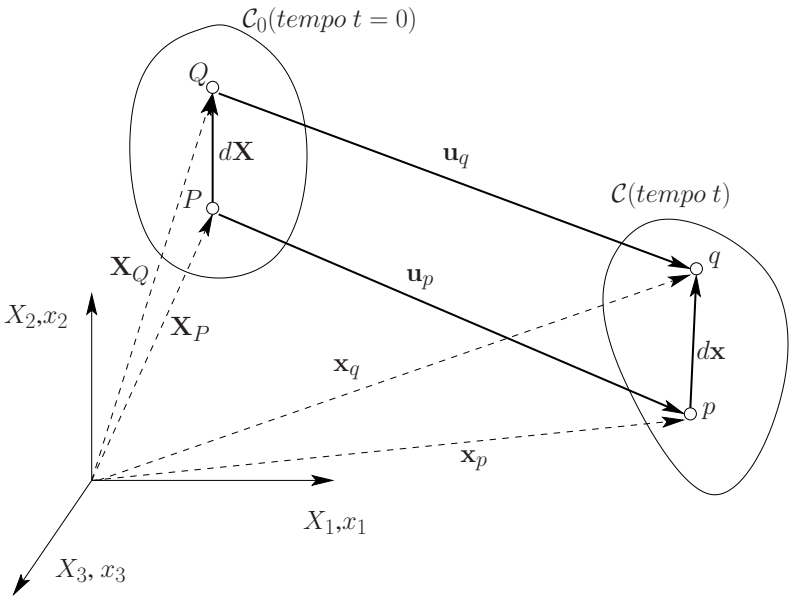


Figura 4.2: Deformação de um elemento $\overline{PQ}$ em um meio contínuo (REDDY, 2006).

Os vetores deslocamentos, $\mathbf{u}$, dos pontos P e Q são dados, respectivamente, por

$$\mathbf{u}_P = \mathbf{x}_p - \mathbf{X}_P, \tag{4.6a}$$

$$\mathbf{u}_Q = \mathbf{x}_q - \mathbf{X}_Q. \tag{4.6b}$$

Na análise de deformação finita, uma das quantidades mais importantes a serem calculadas é o *tensor gradiente de deformação*, $\mathbf{F}$. Este tensor de segunda ordem fornece a relação entre as quantidades calculadas antes e depois do processo de deformação. Assim, um segmento material $d\mathbf{X}$ se relaciona com o segmento espacial $d\mathbf{x}$ como,

$$d\mathbf{x} = \mathbf{F}\ d\mathbf{X}, \tag{4.7}$$

sendo o gradiente de deformação $\mathbf{F}$ dado por

$$\mathbf{F} = \left(\frac{\partial \mathbf{x}}{\partial \mathbf{X}}\right)^T \equiv (\boldsymbol{\nabla}_0 \mathbf{x})^T, \tag{4.8}$$

e $\boldsymbol{\nabla}_0$ o operador gradiente com respeito ao vetor de coordenadas materiais $\mathbf{X}$. Pode-se, também,

escrever a relação

$$d\mathbf{X} = \mathbf{F}^{-1}\, d\mathbf{x}. \tag{4.9}$$

O tensor inverso do gradiente de deformação $\mathbf{F}^{-1}$ é dado por

$$\mathbf{F}^{-1} = \frac{\partial \mathbf{X}}{\partial \mathbf{x}} \equiv \boldsymbol{\nabla}\mathbf{X}, \tag{4.10}$$

e $\boldsymbol{\nabla}$ o operador gradiente com respeito ao vetor de coordenadas espaciais $\mathbf{x}$.

O determinante do gradiente de deformação é denominado *Jacobiano do movimento* segundo (REDDY, 2006) e será denotado por J. O tensor $\mathbf{F}$ não é simétrico.

4.1.2 Tensor Gradiente de Deslocamentos

O gradiente de deslocamento é obtido pela diferença entre os elementos espacial $d\mathbf{x}$ e material $d\mathbf{X}$, ou seja,

$$\begin{aligned} d\mathbf{u} &= d\mathbf{x} - d\mathbf{X} = \mathbf{F}d\mathbf{X} - d\mathbf{X}; \\ &= (\mathbf{F} - \mathbf{I})d\mathbf{X} = \boldsymbol{\nabla}_0\mathbf{u}\, d\mathbf{X}, \end{aligned} \tag{4.11}$$

sendo $\mathbf{I}$ o *tensor identidade de segunda ordem* e $\boldsymbol{\nabla}_0\mathbf{u}$ o *tensor gradiente de deslocamento na configuração material* dado por

$$\boldsymbol{\nabla}_0\mathbf{u} = \mathbf{F} - \mathbf{I} \tag{4.12}$$

4.1.3 Tensores de Deformação

Como apresentado na Figura 4.2, a diferença entre dois pontos P e Q na configuração inicial é dada por $d\mathbf{X}$. Já na configuração deformada essa diferença (lembrado que nesta configuração os pontos são denotados por p e q) é representada por $d\mathbf{x}$. Deseja-se, então, medir a variação da

distância $d\mathbf{X}$ entre os pontos P e Q quando o corpo se deforma e os pontos movem-se para as novas posições p e q, respectivamente.

O quadrado das distâncias entre os pontos materiais P e Q e os pontos espaciais p e q são dadas, respectivamente, por

$$(dS)^2 = d\mathbf{X} \cdot d\mathbf{X}, \tag{4.13}$$

$$\begin{aligned}(ds)^2 &= d\mathbf{x} \cdot d\mathbf{x} = (\mathbf{F}d\mathbf{X}) \cdot (\mathbf{F}d\mathbf{X}) = d\mathbf{X} \cdot (\mathbf{F}^T\mathbf{F}) \cdot d\mathbf{X} \\ &\equiv d\mathbf{X} \cdot \mathbf{C}d\mathbf{X},\end{aligned} \tag{4.14}$$

sendo $\mathbf{C}$ um tensor simétrico, denominado *tensor de deformação à direita de Cauchy*, e definido como

$$\mathbf{C} = \mathbf{F}^T\mathbf{F}. \tag{4.15}$$

A mudança do quadrado das distâncias entre os pontos P e Q na configuração material e p e q na configuração espacial, calculada tomando-se como referência a configuração material, fornece a medida de deformação denominada de *tensor de deformação de Green-Lagrange*, isto é,

$$\begin{aligned}(ds)^2 - (dS)^2 &= (d\mathbf{X} \cdot \mathbf{C}d\mathbf{X}) - (d\mathbf{X} \cdot d\mathbf{X}) \\ &= d\mathbf{X} \cdot (\mathbf{C} - \mathbf{I})d\mathbf{X} = 2d\mathbf{X} \cdot \mathbf{E}d\mathbf{X},\end{aligned} \tag{4.16}$$

sendo $\mathbf{E}$ tensor simétrico de deformação de Green-Lagrange dado por

$$\begin{aligned}\mathbf{E} &= \frac{1}{2}(\mathbf{C} - \mathbf{I}) = \frac{1}{2}\left(\mathbf{F}^T\mathbf{F} - \mathbf{I}\right) \\ &= \frac{1}{2}\left[(\boldsymbol{\nabla}_0\mathbf{u})^T + (\boldsymbol{\nabla}_0\mathbf{u}) + (\boldsymbol{\nabla}_0\mathbf{u})^T(\boldsymbol{\nabla}_0\mathbf{u})\right].\end{aligned} \tag{4.17}$$

É evidente que quando o corpo não se deforma, tem-se $\mathbf{E} = \mathbf{0}$. Isso resulta do fato que não há mudança entre as distâncias dos pontos nas configurações material e espacial, ou seja, o gradiente de deformação é $\mathbf{F} = \mathbf{I}$.

Entretanto, no caso de deformações infinitesimais não é feita distinção entre as configurações material e espacial. Neste caso, tem-se o *tensor de deformação infinitesimal* $\mathbf{E}^*$, o qual é representado, de acordo com Spencer (2004), por

$$\mathbf{E}^* = \frac{1}{2}[(\boldsymbol{\nabla}_0 \mathbf{u}) + (\boldsymbol{\nabla}_0 \mathbf{u})^T] = \frac{1}{2}\left(\mathbf{F} + \mathbf{F}^T\right) - \mathbf{I}. \tag{4.18}$$

Fica evidente que $\mathbf{E}^*$ é simétrico e calculado omitindo-se os termos não-lineares do gradiente de deslocamento.

4.2 Medidas de Tensão

Considere o corpo mostrado na Figura 4.3, em sua configuração deformada, para um elemento de área espacial da em torno de $\mathbf{x}$ do ponto q com normal $\mathbf{n}$ e um diferencial de força $d\mathbf{f}(\mathbf{n})$. Neste cenário, o vetor tensão é dado por

$$\mathbf{t}(\mathbf{n}) = \frac{d\mathbf{f}}{da}. \tag{4.19}$$

O *tensor de tensão de Cauchy* $\boldsymbol{\sigma}$ é definido como um tensor simétrico dado por

$$d\mathbf{f} = \mathbf{t}da = \boldsymbol{\sigma}(\mathbf{n}da) \rightarrow \mathbf{t} = \sigma\mathbf{n}. \tag{4.20}$$

Nos trabalhos de Bonet e Wood (2008), Ogden (1997), Holzapfel (2000), Belytschko, Liu e Moran (2000), prova-se que o tensor $\boldsymbol{\sigma}$ é simétrico através das equações de equilíbrio em termos de momentos. O tensor de tensão de Cauchy $\boldsymbol{\sigma}$ é espacial.

Se o diferencial de força $d\mathbf{f}$ for representado como o produto da tensão pelo diferencial de área não-deformada dA, é necessário definir uma nova medida de tensão, denominada *primeiro tensor de tensão de Piola-Kirchhoff* $\mathbf{P}$ ou também conhecida como *tensão nominal* (BELYTSCHKO; LIU; MORAN, 2000). O tensor $\mathbf{P}$ é definido pela seguinte expressão

$$d\mathbf{f} = \mathbf{P}(\mathbf{N}dA), \tag{4.21}$$

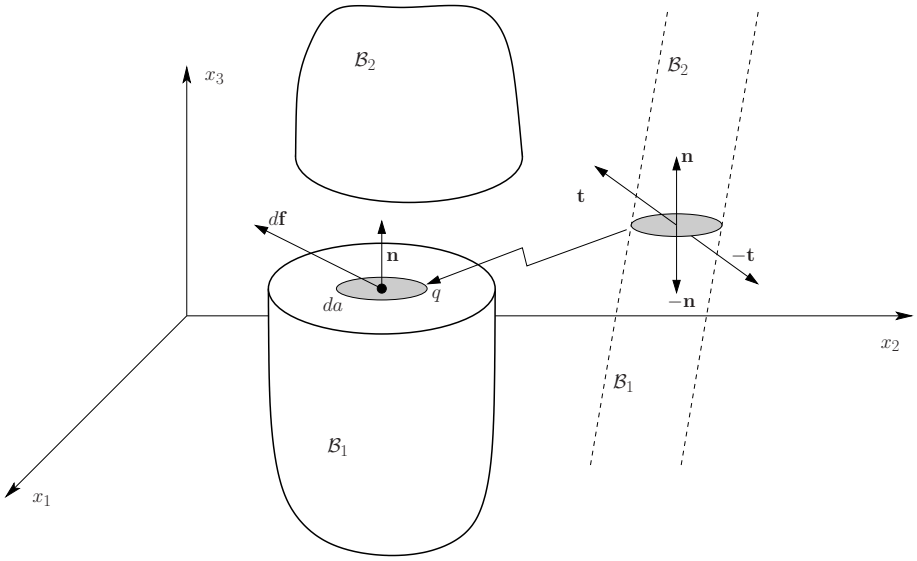


Figura 4.3: Representação da força de superfície (BONET; WOOD, 2008).

sendo $\mathbf{N}$ o vetor normal na descrição material.

O primeiro tensor de tensão de Piola-Kirchhoff não é simétrico. Por essa razão, prefere-se usar uma segunda forma de representação do tensor tensão na configuração não-deformada.

A forma simétrica de representação do tensor de tensão na configuração material é denominada *segundo tensor de tensão de Piola-Kirchhoff* e denotado por $\mathbf{S}$. Utilizando-se de uma forma análoga à apresentada na equação (4.9), pode-se transformar o vetor de força $d\mathbf{f}$ na configuração deformada para uma representação material $d\mathbf{F}$, ou seja,

$$d\mathbf{F} = \mathbf{F}^{-1} d\mathbf{f} = \mathbf{F}^{-1}(\mathbf{P}\mathbf{N}dA) = (\mathbf{P}\mathbf{F}^{-T})\mathbf{N}dA = \mathbf{S}\mathbf{N}dA. \tag{4.22}$$

4.3 Equações Constitutivas

Os materiais denominados hiperelásticos homogêneos pressupõem a existência de um funcional de energia de deformação que depende somente do gradiente de deformação, isto é, $\Psi(\mathbf{F})$.

Sabendo-se que o material hiperelástico deriva do material elástico, pode-se escrever as seguintes funções de resposta para o primeiro tensor de tensão de Piola-Kirchhoff e ao tensor de tensão de Cauchy

$$\mathbf{P} = \mathfrak{G}(\mathbf{F}) = \frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}, \tag{4.23}$$

$$\boldsymbol{\sigma} = \mathfrak{g}(\mathbf{F}) = J^{-1}\mathbf{P}\mathbf{F}^{-T} = J^{-1}\frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}\mathbf{F}^{-T} = J^{-1}\mathbf{F}\left(\frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}\right)^{T}, \tag{4.24}$$

sendo $\mathfrak{g}$ e $\mathfrak{G}$ as funções de resposta do material associadas ao tensor de tensão de Cauchy e ao primeiro tensor de tensão de Piola-Kirchhoff, respectivamente.

As equações (4.23) e (4.24) são conhecidas como *equações constitutivas*, as quais, estabelecem um modelo que aproxima o comportamento do material devido a uma ação de deformação. Um material considerado perfeitamente elástico é definido como um material cuja dissipação interna de energia é zero (HOLZAPFEL, 2000). Assim, tem-se

$$\mathcal{D}_{int} = \mathbf{P} : \dot{\mathbf{F}} - \dot{\Psi} = \left(\mathbf{P} - \frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}\right) : \dot{\mathbf{F}} = 0. \tag{4.25}$$

Como $\mathbf{F}$ e, consequentemente, $\dot{\mathbf{F}}$ são arbitrários, a expressão $\left(\mathbf{P} - \frac{\partial \Psi(\mathbf{F})}{\partial \mathbf{F}}\right)$ deve ser zero, recuperando a equação (4.23).

Em Holzapfel (2000), mostram-se as considerações e deduções matemáticas que são utilizadas para se obter as formas equivalentes dos funcionais de energia de deformação. Assim, pode-se escrever o funcional como

$$\Psi(\mathbf{F}) = \Psi(\mathbf{C}) = \Psi(\mathbf{E}). \tag{4.26}$$

4.3.1 Formas reduzidas das equações constitutivas

Aqui serão definidas as formas reduzidas das equações constitutivas para o material hiperelástico, considerando deformação finita.

Derivando-se o funcional $\Psi(\mathbf{F}) = \Psi(\mathbf{C})$ com relação ao tempo, obtém-se

$$\begin{aligned}\dot{\Psi} &= \text{tr}\left[\left(\frac{\partial\Psi(\mathbf{F})}{\partial\mathbf{F}}\right)^T\dot{\mathbf{F}}\right] = \text{tr}\left[\left(\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}\right)\dot{\mathbf{C}}\right] \\ &= \text{tr}\left[\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}\left(\dot{\mathbf{F}^T}\mathbf{F} + \mathbf{F}^T\dot{\mathbf{F}}\right)\right] = 2\text{tr}\left(\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}\mathbf{F}^T\dot{\mathbf{F}}\right).\end{aligned} \tag{4.27}$$

A equação (4.27) é válida para qualquer tensor $\dot{\mathbf{F}}$. Sabendo que $\mathbf{C}$ é um tensor de segunda ordem simétrico, o gradiente de $\Psi(\mathbf{C})$, usado na equação (4.27), também será simétrico. Da equação (4.27), conclui-se que

$$\left(\frac{\partial\Psi(\mathbf{F})}{\partial\mathbf{F}}\right)^T = 2\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}\mathbf{F}^T. \tag{4.28}$$

Substituindo (4.28) em (4.24), tem-se a equação constitutiva reduzida para o tensor de Cauchy

$$\boldsymbol{\sigma} = J^{-1}\mathbf{F}\left(\frac{\partial\Psi(\mathbf{F})}{\partial\mathbf{F}}\right)^T = 2J^{-1}\mathbf{F}\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}\mathbf{F}^T, \tag{4.29}$$

lembrando que $J = \det\mathbf{F}$.

As expressões para o primeiro tensor de tensão de Piola-Kirchhoff ($\mathbf{P}$ é não simétrico) e para o segundo tensor de tensão de Piola-Kirchhoff ($\mathbf{S}$ é simétrico) são, respectivamente,

$$\mathbf{P} = 2\mathbf{F}\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}, \tag{4.30a}$$

$$\mathbf{S} = 2\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}} = \frac{\partial\Psi(\mathbf{E})}{\partial\mathbf{E}}, \tag{4.30b}$$

notando que $\mathbf{E} = \frac{1}{2}(\mathbf{C} - \mathbf{I})$.

A função resposta do material da equação (4.23) é dada por

$$\mathfrak{G} = 2\frac{\partial\Psi(\mathbf{C})}{\partial\mathbf{C}}. \tag{4.31}$$

4.3.2 Materiais Hiperelásticos Isotrópicos

Um material é considerado *isotrópico* se a sua relação tensão-deformação, obtida experimentalmente, é a a mesma em todas as direções. Sabendo disso, deve-se representar esse fenômeno físico em um modelo matemático que permita a realização de simulação computacional com grande precisão. Logo, nessa seção apresentam-se os conceitos principais para caracterizar o modelo matemático de um material *hiperelástico isotrópico*. Maiores detalhes podem ser obtidos nos trabalhos de Ogden (1997), Holzapfel (2000), Bhatti (2006), BAŞAR e Weichert (2000), SOUZA NETO, PeriĆ e Owen (2009).

Considere um ponto material $\mathbf{X}$ de um corpo elástico contínuo que ocupa uma região Ω_0 no instante de tempo $t = 0$. Um movimento Φ mapeia o ponto $\mathbf{X} \in \Omega_0$ para $\mathbf{x} = \Phi(\mathbf{X}, t) \in \Omega$ em um instante t.

Se um movimento de corpo rígido for imposto na configuração de referência, isso fará que o corpo, que antes ocupava a região Ω_0, seja transladado por um vetor $\mathbf{c}$ e rotacionado por um tensor ortogonal $\mathbf{Q}$ de acordo com

$$\mathbf{X}^* = \mathbf{c} + \mathbf{Q}\mathbf{X}. \tag{4.32}$$

Assim, o corpo move-se da região Ω_0 para a nova região Ω_0^* (nova configuração de referência), e o ponto material $\mathbf{X}$ para uma nova localização material $\mathbf{X}^* \in \Omega_0^*$.

Para um movimento diferente, tal que $\mathbf{x} = \Phi^*(\mathbf{X}^*, t)$ move Ω_0^* para a configuração Ω, tem-se

$$\mathbf{x} = \Phi(\mathbf{X}, t) = \Phi^*(\mathbf{X}, t). \tag{4.33}$$

Usando a regra da cadeia e a relação dada pela equação (4.32), o gradiente de deformação $\mathbf{F}$ pode ser escrito como

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}^*}\mathbf{Q} = \mathbf{F}^*\mathbf{Q}. \tag{4.34}$$

sendo $\mathbf{F}^*$ o gradiente de deformação relativo à região Ω_0^*.

Usando a equação (4.34) escreve-se $\mathbf{F}^*$ como

$$\mathbf{F}^* = \mathbf{F}\mathbf{Q}^T. \tag{4.35}$$

Quando o material considerado é *isotrópico* com relação à sua configuração de referência Ω_0, os valores da energia de deformação $\Psi(\mathbf{F})$ e $\Psi(\mathbf{F}^*)$ são iguais para todos os tensores ortogonais $\mathbf{Q}$. Pode-se assim escrever o funcional de energia de deformação da seguinte maneira

$$\Psi(\mathbf{F}) = \Psi(\mathbf{F}^*) = \Psi(\mathbf{F}\mathbf{Q}^T). \tag{4.36}$$

Conclui-se a partir da equação (4.36), que para qualquer movimento imposto a um corpo elástico em uma configuração de referência translada e/ou rotacionada, se a energia de deformação no tempo t for a mesma, o material será isotrópico.

De acordo com Holzapfel (2000), o funcional de energia também pode ser expresso em função do tensor à direita de Cauchy-Green, isto é,

$$\Psi(\mathbf{F}) = \Psi(\mathbf{C}). \tag{4.37}$$

Se o material for hiperelástico isotrópico, pode-se afirmar que

$$\Psi(\mathbf{C}) = \Psi(\mathbf{C}^*) = \Psi(\mathbf{F}^{*T}\mathbf{F}^*) = \Psi(\mathbf{Q}\mathbf{F}^T\mathbf{F}\mathbf{Q}^T). \tag{4.38}$$

Sabendo que $\mathbf{C} = \mathbf{F}^T\mathbf{F}$, pode-se reescrever a equação (4.38) como

$$\Psi(\mathbf{C}) = \Psi(\mathbf{Q}\mathbf{C}\mathbf{Q}^{\mathbf{T}}). \tag{4.39}$$

Mostra-se em Holzapfel (2000) que para um material hiperelástico isotrópico, tem-se que

$\Psi(\mathbf{C}) = \Psi(\mathbf{b})$, ou seja, pode-se escrever o funcional de energia de deformação em termos do tensor à esquerda de Cauchy-Green.

4.3.3 Equações constitutivas em termos dos invariantes

De acordo com Spencer (2004), os três invariantes principais de $\mathbf{C}$ são

$$I_1 = \text{tr}\mathbf{C}, \tag{4.40a}$$

$$I_2 = \frac{1}{2}[(\text{tr}\mathbf{C})^2 - \text{tr}\mathbf{C}^2], \tag{4.40b}$$

$$I_3 = \det \mathbf{C} = \det \mathbf{F}. \tag{4.40c}$$

Para se determinar a equação constitutiva para o material hiperelástico isotrópico em termos dos invariantes de deformação, diferencia-se $\Psi(\mathbf{C}) = \Psi(I_1, I_2, I_3)$ com respeito ao tensor $\mathbf{C}$. Assume-se que $\Psi(\mathbf{C})$ possui derivadas contínuas com respeito aos invariantes I_1, I_2 e I_3. Usando a regra da cadeia, obtém-se

$$\frac{\partial \Psi(\mathbf{C})}{\partial \mathbf{C}} = \frac{\partial \Psi}{\partial I_1}\frac{\partial I_1}{\partial \mathbf{C}} + \frac{\partial \Psi}{\partial I_2}\frac{\partial I_2}{\partial \mathbf{C}} + \frac{\partial \Psi}{\partial I_3}\frac{\partial I_3}{\partial \mathbf{C}}. \tag{4.41}$$

As derivadas dos invariantes com relação ao tensor $\mathbf{C}$ são dadas por

$$\frac{\partial I_1}{\partial \mathbf{C}} = \frac{\partial \text{tr}\mathbf{C}}{\partial \mathbf{C}} = \frac{\partial(\mathbf{I} : \mathbf{C})}{\partial \mathbf{C}} = \mathbf{I}, \tag{4.42a}$$

$$\frac{\partial I_2}{\partial \mathbf{C}} = \frac{1}{2}\left(2\text{tr}\mathbf{C}\ -\mathbf{I} - \frac{\partial \text{tr}(\mathbf{C}^2)}{\partial \mathbf{C}}\right) = I_1\mathbf{I} - \mathbf{C}, \tag{4.42b}$$

$$\frac{\partial I_3}{\partial \mathbf{C}} = I_3\mathbf{C}^{-1}. \tag{4.42c}$$

Substituindo as equações (4.41) e (4.42) em (4.30b), obtém-se a forma mais geral da relação tensão em termos dos três primeiros invariantes, o qual caracteriza o material hiperelástico

isotrópico sob deformação finita

$$\mathbf{S} = 2\frac{\partial \Psi(\mathbf{C})}{\partial \mathbf{C}} = 2\left[\left(\frac{\partial \Psi}{\partial I_1} + I_1\frac{\partial \Psi}{\partial I_2}\right)\mathbf{I} - \frac{\partial \Psi}{\partial I_2}\mathbf{C} + I_3\frac{\partial \Psi}{\partial I_3}\mathbf{C}^{-1}\right]. \tag{4.43}$$

Pré-multiplicando-se ou pós-multiplicando-se a equação (4.43) por $\mathbf{C}$ o resultado é o mesmo, o que é uma consequência da isotropia do material.

4.3.4 Tensor de Elasticidade

Na descrição material, o tensor elástico é dado por

$$\mathbb{D} = 2\frac{\partial \mathbf{S}}{\partial \mathbf{C}} \qquad \text{ou} \qquad D_{ijkl} = 2\frac{\partial S_{ij}}{\partial C_{kl}}, \tag{4.44}$$

sendo $i, j, k, l = 1, 2, 3$ os índices do tensor de quarta ordem. O tensor $\mathbb{D}$ caracteriza o gradiente da função $\mathbf{S}$ e relaciona o trabalho-conjugado entre os tensores de deformação e tensão como mostrado em Holzapfel (2000). Este tensor mede a mudança que a deformação provoca sobre a tensão.

Lembrando que $\mathbf{E} = (\mathbf{C} - \mathbf{I})/2$, a equação (4.44) pode ser reescrita

$$\mathbb{D} = \frac{\partial \mathbf{S}}{\partial \mathbf{E}} \qquad \text{ou} \qquad D_{ijkl} = \frac{\partial S_{ij}}{\partial E_{kl}}. \tag{4.45}$$

Assumindo a existência de um funcional de energia Ψ, então $\mathbf{S}$ pode ser derivado de Ψ de acordo com $\mathbf{S} = 2\partial\Psi/\partial\mathbf{C}$. O tensor elástico é também definido como

$$\mathbb{D} = 4\frac{\partial^2 \Psi}{\partial \mathbf{C}\partial \mathbf{C}} \qquad \text{ou} \qquad D_{ijkl} = 4\frac{\partial^2 \Psi}{\partial C_{ij}\partial C_{kl}}, \tag{4.46}$$

Logo, pode-se definir a equação que representa, de forma mais geral, o tensor material para

um material hiperelástico qualquer em função apenas dos seus três invariantes principais, isto é,

$$\begin{aligned}\mathbb{D} &= 2\frac{\partial \mathbf{S}}{\partial \mathbf{C}} = 4\frac{\partial^2 \Psi(I_1, I_2, I_3)}{\partial \mathbf{C} \partial \mathbf{C}} \\ &= c_1 \mathbf{I} \otimes \mathbf{I} + c_2(\mathbf{I} \otimes \mathbf{C} + \mathbf{C} \otimes \mathbf{I}) + c_3(\mathbf{I} \otimes \mathbf{C}^{-1} + \mathbf{C}^{-1} \otimes \mathbf{I}) \\ &+ c_4(\mathbf{C} \otimes \mathbf{C}) + c_5(\mathbf{C} \otimes \mathbf{C}^{-1} + \mathbf{C}^{-1} \otimes \mathbf{C}) \\ &+ c_6(\mathbf{C}^{-1} \otimes \mathbf{C}^{-1}) + c_7(\mathbf{C}^{-1} \odot \mathbf{C}^{-1}) + c_8 \mathbb{I},\end{aligned} \tag{4.47}$$

sendo os tensores de quarta ordem $\mathbf{C}^{-1} \odot \mathbf{C}^{-1}$ e $\mathbb{I}$ definidos por

$$(\mathbf{C}^{-1} \odot \mathbf{C}^{-1})_{ijkl} = \frac{1}{2}(C^{-1}_{ik} C_{jl} + C^{-1}_{il} C_{jk}), \tag{4.48}$$

$$\mathbb{I} = \mathbf{I} \otimes \mathbf{I} = \delta_{ij}\delta_{ik}, \tag{4.49}$$

Já, os coeficientes $c_1, .., c_8$ são dados por

$$\left.\begin{aligned}
c_1 &= 4\left(\frac{\partial^2 \Psi}{\partial I_1 \partial I_1} + 2I_1 \frac{\partial^2 \Psi}{\partial I_1 \partial I_2} + \frac{\partial \Psi}{\partial I_2} + I_1^2 \frac{\partial^2 \Psi}{\partial I_2 \partial I_2}\right), \\
c_2 &= -4\left(\frac{\partial^2 \Psi}{\partial I_1 \partial I_2} + I_1 \frac{\partial^2 \Psi}{\partial I_2 \partial I_2}\right), \\
c_3 &= 4\left(I_3 \frac{\partial^2 \Psi}{\partial I_1 \partial I_3} + I_1 I_3 \frac{\partial^2 \Psi}{\partial I_2 \partial I_3}\right), \\
c_4 &= 4\frac{\partial^2 \Psi}{\partial I_2 \partial I_2}, \\
c_5 &= -4I_3 \frac{\partial \Psi}{\partial^2 I_2 \partial I_2 \partial I_3}, \\
c_6 &= 4\left(I_3 \frac{\partial \Psi}{\partial I_3} + I_3^2 \frac{\partial^2 \Psi}{\partial I_3 \partial I_3}\right), \\
c_7 &= -4I_3 \frac{\partial \Psi}{\partial I_3}, \\
c_8 &= -4\frac{\partial \Psi}{\partial I_2}.
\end{aligned}\right\} \tag{4.50}$$

Para um material hiperelástico isotrópico, o tensor $\mathbb{D}$ satisfaz as seguintes condições de sime-

tria

$$D_{ijkl} = D_{klij} = D_{jikl} = D_{ijlk}. \tag{4.51}$$

A seguir, apresentam-se as simplificações realizadas para o caso particular de um material hiperelástico isotrópico conhecido como *neo-Hookeano*.

4.3.5 Material neo-Hookeano Compressível

São muitos os funcionais de energia de deformação usados para descrever propriedades elásticas de materiais compressíveis e incompressíveis como visto em Holzapfel (2000), BAŞAR e Weichert (2000). Para um material hiperelástico isotrópico compressível, tem-se o caso particular conhecido como material *neo-Hookeano compressível*. Possui as características encontradas em um material linear elástico (BONET; WOOD, 2008) e o seu funcional de energia é definido da seguinte forma

$$\Psi = \frac{\lambda}{2}(\ln \det \mathbf{F})^2 - \mu \ln \det \mathbf{F} + \frac{1}{2}\mu \left(\mathrm{tr}\mathbf{C} - 3\right) \tag{4.52}$$

sendo λ e μ os parâmetros do material, conhecidos como *constantes de Lamé*, e dados por

$$\lambda = \frac{\nu E}{(1+\nu)(1-2\nu)}, \tag{4.53a}$$

$$\mu = \frac{E}{2(1+\nu)}. \tag{4.53b}$$

Percebe-se que para esse material, $I_2 = 0$. Logo, pode-se simplificar as equações (4.43) e (4.47) e escrever o segundo tensor de tensão de Piola-Kirchhoff como

$$\mathbf{S} = \mu(\mathbf{I} - \mathbf{C}^{-1}) + \lambda \left[\ln(\det\ \mathbf{F})\right] \mathbf{C}^{-1}, \tag{4.54}$$

O tensor de elasticidade na descrição material é dado por

$$\mathbb{D} = \lambda(\mathbf{C}^{-1} \otimes \mathbf{C}^{-1}) + (\mu - \lambda \ln(\det \mathbf{F}))(\mathbf{C}^{-1} \odot \mathbf{C}^{-1}) \tag{4.55}$$

ou em notação indicial

$$D_{ijkl} = \lambda C_{ij}^{-1} C_{kl}^{-1} + (\mu - \lambda \ln(\det \mathbf{F}))(C_{ik}^{-1} C_{jl}^{-1} + C_{il}^{-1} C_{kj}^{-1}). \tag{4.56}$$

4.4 Princípio do Trabalho Virtual

O *Princípio do Trabalho Virtual* (PTV) assume que a soma dos trabalhos virtuais externos e dos esforços internos, provenientes da deformação, seja igual a zero para qualquer ação virtual, isto é

$$\delta W = \delta W_{\text{int}} - \delta W_{\text{ext}} = 0. \tag{4.57}$$

No PTV, as grandezas cinemáticas são escritas em termos de deslocamentos virtuais $\delta\mathbf{u}$. Assim, é necessário definir as medidas de deformação virtual, as quais serão escritas em função de $\delta\mathbf{u}$.

Como primeira medida de deformação virtual, tem-se o tensor gradiente de deformação virtual dado por

$$\delta\mathbf{F} = \boldsymbol{\nabla}_0 \delta\mathbf{u} \qquad \text{ou} \qquad \delta F_{ij} = \frac{\partial \delta u_i}{\partial X_j}. \tag{4.58}$$

O inverso do tensor gradiente de deformação virtual é representado por (HOLZAPFEL, 2000)

$$\delta\mathbf{F}^{-1} = -\mathbf{F}^{-1}(\boldsymbol{\nabla}_0 \delta\mathbf{u}\mathbf{F}^{-1}) = -\mathbf{F}^{-1}\boldsymbol{\nabla}\delta\mathbf{u} \qquad \text{ou} \qquad \delta F_{ij}^{-1} = -F_{ik}^{-1}\frac{\partial \delta u_k}{\partial x_j}. \tag{4.59}$$

Nota-se que a relação

$$\boldsymbol{\nabla}\delta\mathbf{u} = \boldsymbol{\nabla}_0\delta\mathbf{u}\mathbf{F}^{-1} \tag{4.60}$$

é uma transformação válida como apresentada em (BONET; WOOD, 2008; HOLZAPFEL, 2000).

Com base nas equações (4.58) e usando a definição do tensor de Green-Lagrange dado por (4.17), pode-se estabelecer o tensor de deformação virtual de Green-Lagrange $\delta\mathbf{E}$

$$\begin{aligned}\delta\mathbf{E} &= \frac{1}{2}\left[\delta(\mathbf{F}^T)\mathbf{F} + \mathbf{F}\delta\mathbf{F}\right] \\ &= \frac{1}{2}\left[(\mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u})^T + \mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u}\right] \\ &= \frac{1}{2}\left[(\boldsymbol{\nabla}_0\delta\mathbf{u})^T\mathbf{F} + \mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u}\right].\end{aligned} \tag{4.61}$$

4.4.1 Princípio do Trabalho Virtual na Descrição Material

Denota-se o domínio e a superfície do corpo contínuo na configuração espacial $\mathcal{C}_1$, respectivamente, por Ω e $\partial\Omega$ e por Ω_0 e $\partial\Omega_0$ na descrição material $\mathcal{C}_0$.

Logo, o trabalho virtual realizado pelas forças internas, δW_{int}, é dado pelo produto interno entre o segundo tensor de tensão de Piola-Kirchhoff $\mathbf{S}$ e a deformação virtual de Green-Lagrange $\delta\mathbf{E}$

$$\delta W_{\text{int}} = \int_{\Omega_0} \mathbf{S} : \delta\mathbf{E} dV. \tag{4.62}$$

sendo dV o diferencial de volume na configuração inicial. Observa-se que $\mathbf{S}$ e $\mathbf{E}$ são pares de tensores conjugados energeticamente conforme definido em (BONET; WOOD, 2008).

A parcela referente ao trabalho virtual externo é dada por

$$\delta W_{\text{ext}} = \int_{\Omega_0} \mathbf{f}\cdot\delta\mathbf{u}\, dV + \int_{\partial\Omega_0} \mathbf{t}\cdot\delta\mathbf{u}\, dA \tag{4.63}$$

sendo $\mathbf{f}$ o vetor de força de corpo, $\mathbf{t}$ é o vetor de força de superfície e dA é o diferencial de área. Todas as quantidades são calculadas na configuração inicial $\mathcal{C}_0$.

Finalmente, de acordo com Bonet e Wood (2008), pode-se estabelecer a equação para o PTV na descrição material como

$$\delta W = \int_{\Omega_0} \mathbf{S} : \delta\mathbf{E} dV - \int_{\Omega_0} \mathbf{f} \cdot \delta\mathbf{u}\, dV - \int_{\partial\Omega_0} \mathbf{t} \cdot \delta\mathbf{u}\, dA = 0. \tag{4.64}$$

4.5 Carregamento Dependente da Deformação

Carregamentos concentrados na direção dos eixos (X_1, X_2, X_3) são forças classificadas como não dependente da deformação, pois, a direção de aplicação da carga, mesmo após o processo de deformação, permanece inalterada. Esse é uma hipótese válida para problemas de pequenas deformações. Porém, se uma pressão uniforme p for aplicada sobre um elemento de área da, o vetor de força de tração $\mathbf{t}$ representado por $\mathbf{t} = p\mathbf{n}$ será dependente da direção normal. Assim, na hipótese de grandes deformações, é necessário analisar o trabalho virtual realizado por esse carregamento δW^p_{ext}.

Na configuração deformada $\mathcal{C}$, δW^p_{ext} é representado na forma vetorial por

$$\delta W^p_{\text{ext}} = \int_{\partial\Omega} p\ \ \mathbf{n} \cdot (\delta\mathbf{u})\, da \tag{4.65}$$

sendo $\mathbf{n}$ o vetor normal à superfície carregada na configuração $\mathcal{C}$.

Percebe-se pela equação (4.65) que a magnitude do elemento de área e a orientação do vetor normal são dependentes da deformação. Dessa forma, qualquer modificação na geometria afeta diretamente a condição de equilíbrio, o que resultará no aparecimento de uma rigidez associada à pressão aplicada. Esse fato torna-se mais claro ao ser realizado o procedimento de linearização sobre δW^p_{ext}, como será visto mais adiante.

Para um problema 3D, o vetor normal a uma superfície possui a seguinte representação

$$\mathbf{n} = \frac{\frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2}}{\left\| \frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2} \right\|}. \tag{4.66}$$

A magnitude da área é fornecida por

$$da = \left\| \frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2} \right\| d\xi_1 d\xi_2, \tag{4.67}$$

sendo $\frac{\partial \mathbf{x}}{\partial \xi_1}$ e $\frac{\partial \mathbf{x}}{\partial \xi_2}$ os vetores tangenciais da superfície parametrizada.

Substituindo as equações (4.66) e (4.67) na equação (4.65), obtém-se o trabalho virtual realizado pela pressão na forma parametrizada

$$\delta W^p_{\text{ext}} = \int_{\partial \Omega_{\xi_1 \xi_2}} p \, \delta \mathbf{u} \cdot \left(\frac{\partial \mathbf{x}}{\partial \xi_1} \times \frac{\partial \mathbf{x}}{\partial \xi_2} \right) \, d\xi_1 d\xi_2 \tag{4.68}$$

No caso bidimensional, o vetor normal passa a ser representado por

$$\mathbf{n} = \frac{\frac{\partial \mathbf{x}}{\partial \xi} \times [0; 0; 1]}{\left\| \frac{\partial \mathbf{x}}{\partial \xi} \times [0; 0; 1] \right\|} \tag{4.69}$$

sendo

$$\frac{\partial \mathbf{x}}{\partial \xi} = \begin{Bmatrix} dx_1/d\xi \\ dx_2/d\xi \end{Bmatrix}. \tag{4.70}$$

Dessa forma, reescreve-se o vetor normal como

$$\mathbf{n} = \begin{Bmatrix} \dfrac{dx_2/d\xi}{J_c} \\ -\dfrac{dx_1/d\xi}{J_c} \end{Bmatrix} \tag{4.71}$$

sendo

$$J_c = \sqrt{\left(\frac{dx_2}{d\xi}\right)^2 + \left(\frac{dx_1}{d\xi}\right)^2}. \tag{4.72}$$

A magnitude da área é fornecida por

$$da = J_c d\xi. \tag{4.73}$$

Finalmente, o trabalho virtual externo realizado por um carregamento de pressão para um problema 2D é dado por

$$\delta W^p_{\text{ext}} = \int_{\partial\Omega} p\, \delta\mathbf{u} \cdot \begin{Bmatrix} dx_2/d\xi \\ -dx_1/d\xi \end{Bmatrix} d\xi. \tag{4.74}$$

4.6 Procedimento de Linearização

Na Mecânica do Contínuo, problemas não-lineares representados pela equação (4.64) devem ser linearizados e a solução é obtida através de processos iterativos, sendo o método de Newton-Raphson o procedimento numérico mais utilizado.

O método consiste em expressar a equação (4.64) em uma série de Taylor truncada. Por exemplo, seja

$$\mathcal{F}(\mathbf{u}) = \mathbf{0} \tag{4.75}$$

um sistema de equações não-lineares e $\mathbf{u}$ o conjunto de variáveis primárias a serem determinadas.

Considerando que $\mathbf{u}_0$ é uma estimativa inicial para a solução e $\Delta\mathbf{u}$ é um incremento, pode-se empregar $\mathbf{u} = \mathbf{u}_0 + \Delta\mathbf{u}$ como uma solução aproximada de (4.75). Para representar $\mathcal{F}$ em uma série de Taylor, deve-se antes definir uma função auxiliar $\mathcal{K}$ de um único parâmetro escalar fictício

ϵ dada por

$$\mathcal{K}(\epsilon) = \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}). \tag{4.76}$$

Expandindo $\mathcal{K}$ em uma série de Taylor em torno de $\epsilon = 0$, tem-se

$$\mathcal{K}(\epsilon) = \mathcal{K}(0) + \epsilon \left.\frac{d\mathcal{K}}{d\epsilon}\right|_{\epsilon=0} + \left.\frac{\epsilon^2}{2}\frac{d^2\mathcal{K}}{d\epsilon^2}\right|_{\epsilon=0} + ... \tag{4.77}$$

Substituindo $\mathcal{K}$ por sua definição dada em (4.76) na série Taylor anterior, obtém-se

$$\mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}) = \mathcal{F}(\mathbf{u}_0) + \epsilon \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}) + \left.\frac{\epsilon^2}{2}\frac{d^2}{d\epsilon^2}\right|_{\epsilon=0} \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}) + ... \tag{4.78}$$

Truncando esta série de Taylor no segundo termo, consegue-se a mudança ou incremento na função não-linear $\mathcal{F}(\mathbf{u})$ como

$$\mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}) - \mathcal{F}(\mathbf{u}_0) \approx \epsilon \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}). \tag{4.79}$$

Como ϵ é um parâmetro fictício, usado somente para gerar a série de Taylor, pode-se eliminá-lo fazendo $\epsilon = 1$. Dessa maneira, a aproximação linear para o incremento de $\mathcal{F}(\mathbf{u})$ é

$$\mathcal{F}(\mathbf{u}_0 + \Delta\mathbf{u}) - \mathcal{F}(\mathbf{u}_0) \approx \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}), \tag{4.80}$$

sendo o termo do lado direito a derivada direcional de $\mathcal{F}(\mathbf{u})$ calculada em $\mathbf{u}_0$ na direção de $\Delta\mathbf{u}$, a qual é escrita como

$$D\mathcal{F}(\mathbf{u}_0)[\Delta\mathbf{u}] = \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \mathcal{F}(\mathbf{u}_0 + \epsilon\Delta\mathbf{u}). \tag{4.81}$$

Finalmente, o valor de $\mathcal{F}(\mathbf{u}_0 + \Delta\mathbf{u})$ pode ser aproximado linearmente como

$$\mathcal{F}(\mathbf{u}_0 + \Delta\mathbf{u}) \approx \mathcal{F}(\mathbf{u}_0) + D\mathcal{F}(\mathbf{u}_0)[\Delta\mathbf{u}]. \tag{4.82}$$

Substituindo a equação (4.82) no sistema de equações não-lineares (4.75) e fazendo $\mathbf{u} = \mathbf{u}_0 + \Delta\mathbf{u}$, tem-se

$$\mathcal{F}(\mathbf{u}) = \mathcal{F}(\mathbf{u}_0 + \Delta\mathbf{u}) \approx \mathcal{F}(\mathbf{u}_0) + D\mathcal{F}(\mathbf{u}_0)[\Delta\mathbf{u}] = \mathbf{0}, \tag{4.83}$$

o que torna o sistema de equações não-lineares original em um sistema de equações lineares com respeito a $\Delta\mathbf{u}$.

Assumindo que (4.83) pode ser resolvida para $\Delta\mathbf{u}$, o procedimento de Newton-Raphson é estabelecido como

$$D\mathcal{F}(\mathbf{u}_k)[\Delta\mathbf{u}] = -\mathcal{F}(\mathbf{u}_k); \qquad \mathbf{u}_{k+1} = \mathbf{u}_k + \Delta\mathbf{u}. \tag{4.84}$$

Agora, deseja-se aplicar o conceito de linearização sobre o PTV definido pela equação (4.57). Se for aplicado a regra da cadeia para a derivada direcional, a forma linearizada de (4.57) é

$$\delta W + D\delta W[\Delta\mathbf{u}] = 0, \tag{4.85}$$

sendo

$$D\delta W[\Delta\mathbf{u}] = D\delta W_{\text{int}}[\Delta\mathbf{u}] - D\delta W_{\text{ext}}[\Delta\mathbf{u}] \tag{4.86}$$

a linearização dos componentes do trabalho virtual interno e externo. Em um primeiro momento, o carregamento de superfície considerado será do tipo não dependente da deformação. Dessa maneira, $D\delta W_{\text{ext}}[\Delta\mathbf{u}] = 0$, para ambas as descrições. Apenas na seção 4.6.2, será tratado o procedimento de linearização do trabalho virtual externo para o carregamento dependente de deformação.

4.6.1 Linearização do Princípio do Trabalho Virtual na Descrição Material: Lagrangiano Total

Aplicando a regra da cadeia para a derivada direcional e a definição de tensor material elástico, a derivada direcional do trabalho virtual interno na descrição material é dada segundo Holzapfel (2000) por

$$\begin{aligned} D\delta W_{\text{int}}[\Delta\mathbf{u}] &= \int_{\Omega_0} D(\mathbf{S} : \delta\mathbf{E})[\Delta\mathbf{u}]dV \\ &= \int_{\Omega_0} \delta\mathbf{E} : D\mathbf{S}[\Delta\mathbf{u}]dV + \int_{\Omega_0} \mathbf{S} : D\delta\mathbf{E}[\Delta\mathbf{u}]dV, \end{aligned} \tag{4.87}$$

sendo $\delta\mathbf{E}$ o tensor de deformação virtual de Green-Lagrange dado pela equação (4.61); $D\mathbf{S}[\Delta\mathbf{u}]$ e $D\delta\mathbf{E}[\Delta\mathbf{u}]$ as derivadas direcionais do segundo tensor de Piola-Kirchhoff e do tensor de deformação virtual de Green-Lagrange, respectivamente.

Começando pela linearização do segundo tensor de tensão de Piola-Kirchhoff (lembrando que esse tensor é simétrico), tem-se

$$D\mathbf{S}[\Delta\mathbf{u}] = \frac{\partial\mathbf{S}}{\partial\mathbf{E}} : D\mathbf{E}[\Delta\mathbf{u}] = \mathbb{D} : D\mathbf{E}[\Delta\mathbf{u}], \tag{4.88}$$

sendo $D\mathbf{E}[\Delta\mathbf{u}]$ a linearização do tensor de deformação de Green-Lagrange na direção $\Delta\mathbf{u}$. Essa grandeza é definida por

$$D\mathbf{E}[\Delta\mathbf{u}] = \frac{1}{2}\left[(\boldsymbol{\nabla}_0\Delta\mathbf{u})^T\mathbf{F} + \mathbf{F}^T\boldsymbol{\nabla}_0\Delta\mathbf{u}\right]. \tag{4.89}$$

Resta a representação de $D\delta\mathbf{E}[\Delta\mathbf{u}]$ para completar a linearização de $DW_{\text{int}}[\Delta\mathbf{u}]$ na configuração material, a qual é dada por

$$D\delta\mathbf{E}[\Delta\mathbf{u}] = \frac{1}{2}\left[(\boldsymbol{\nabla}_0\delta\mathbf{u})^T\boldsymbol{\nabla}_0\Delta\mathbf{u} + (\boldsymbol{\nabla}_0\Delta\mathbf{u})^T\boldsymbol{\nabla}_0\delta\mathbf{u}\right]. \tag{4.90}$$

Logo, a equação (4.87) é reescrita como

$$
\begin{aligned}
D W_{\text{int}}[\Delta \mathbf{u}] = & \int_{\Omega_0} \frac{1}{2}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T \mathbf{F} + \mathbf{F}^T \boldsymbol{\nabla}_0 \delta \mathbf{u}\right] : \mathbb{D} : \frac{1}{2}\left[(\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T \mathbf{F} + \mathbf{F}^T \boldsymbol{\nabla}_0 \Delta \mathbf{u}\right] dV \\
& + \int_{\Omega_0} \mathbf{S} : \frac{1}{2}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T \boldsymbol{\nabla}_0 \Delta \mathbf{u} + (\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T \boldsymbol{\nabla}_0 \delta \mathbf{u}\right] dV.
\end{aligned}
\tag{4.91}
$$

Devido à simetria dos tensores $\mathbf{S}$ e $\mathbb{D}$, as seguintes propriedades podem ser aplicadas (BHATTI, 2006)

$$
\begin{aligned}
\mathbf{S} : \frac{1}{2}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T \boldsymbol{\nabla}_0 \Delta \mathbf{u} + (\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T \boldsymbol{\nabla}_0 \delta \mathbf{u}\right] = & \frac{1}{2}\text{tr}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T (\boldsymbol{\nabla}_0 \Delta \mathbf{u})\mathbf{S}\right] \\
& + \frac{1}{2}\text{tr}\left[(\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T (\boldsymbol{\nabla}_0 \delta \mathbf{u})\mathbf{S}\right] \\
& \equiv \boldsymbol{\nabla}_0 \delta \mathbf{u} : (\boldsymbol{\nabla}_0 \Delta \mathbf{u})\mathbf{S} \\
& \equiv \mathbf{S} : [(\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T \boldsymbol{\nabla}_0 \delta \mathbf{u}],
\end{aligned}
\tag{4.92}
$$

$$
\begin{aligned}
\frac{1}{2}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T \mathbf{F} + \mathbf{F}^T \boldsymbol{\nabla}_0 \delta \mathbf{u}\right] : \mathbb{D} : \frac{1}{2}\left[(\boldsymbol{\nabla}_0 \delta \mathbf{u})^T \mathbf{F} + \mathbf{F}^T \boldsymbol{\nabla}_0 \delta \mathbf{u}\right] = & \\
\mathbf{F}^T \boldsymbol{\nabla}_0 \delta \mathbf{u} : \mathbb{D} : \mathbf{F}^T \boldsymbol{\nabla}_0 \Delta \mathbf{u}. &
\end{aligned}
\tag{4.93}
$$

Dessa forma, a derivada direcional do trabalho virtual interno na descrição material é

$$
D W_{\text{int}}[\Delta \mathbf{u}] = \int_{\Omega_0} \mathbf{S} : [(\boldsymbol{\nabla}_0 \Delta \mathbf{u})^T \boldsymbol{\nabla}_0 \delta \mathbf{u}] dV + \int_{\Omega_0} \left[\mathbf{F}^T \boldsymbol{\nabla}_0 \delta \mathbf{u} : \mathbb{D} : \mathbf{F}^T \boldsymbol{\nabla}_0 \Delta \mathbf{u}\right] dV. \tag{4.94}
$$

A formulação que resulta na equação (4.94) é também conhecida na literatura como *Lagrangiano Total*.

4.6.2 Linearização do Princípio do Trabalho Virtual para Carregamento Dependente de Deformação

Com relação ao carregamento dependente da deformação, tratada nesta tese por uma pressão, apresentado na seção 4.5,o trabalho virtual é dado pelas equações (4.68) e (4.74), respectivamente, para problemas bi e tridimensionais.

Iniciando pela linearização do caso tridimensional (os detalhes podem ser encontrados em (BHATTI, 2006)), a derivada direcional da equação (4.68) é

$$\begin{aligned} DW^p_{\text{ext}}[\Delta\mathbf{u}] &= \int_{\partial\Omega_{\xi_1\xi_2}} p\delta\mathbf{u}\cdot D\left(\frac{\partial\mathbf{x}}{\partial\xi_1}\times\frac{\partial\mathbf{x}}{\partial\xi_2}\right)[\Delta\mathbf{u}]\,d\xi_1 d\xi_2 \\ &= \int_{\partial\Omega_{\xi_1\xi_2}} p\delta\mathbf{u}\cdot\left[D\left(\frac{\partial\mathbf{x}}{\partial\xi_1}\right)[\Delta\mathbf{u}]\times\frac{\partial\mathbf{x}}{\partial\xi_2}+\frac{\partial\mathbf{x}}{\partial\xi_1}\times D\left(\frac{\partial\mathbf{x}}{\partial\xi_2}\right)[\Delta\mathbf{u}]\right]d\xi_1\,d\xi_2 \\ &= \int_{\partial\Omega_{\xi_1\xi_2}} p\left[\left(\frac{\partial\Delta\mathbf{u}}{\partial\xi_1}\right)^T\frac{\partial\mathbf{x}}{\partial\xi_2}\times\delta\mathbf{u}-\left(\frac{\partial\Delta\mathbf{u}}{\partial\xi_2}\right)^T\frac{\partial\mathbf{x}}{\partial\xi_1}\times\delta\mathbf{u}\right]d\xi_1\,d\xi_2. \end{aligned} \tag{4.95}$$

O produto tensorial irá gerar um sistema de equações não-simétrico.

Para o caso bidimensional, a linearização é dada por

$$\begin{aligned} \delta W^p_{\text{ext}} &= \int_{\partial\Omega_\xi} p\,\delta\mathbf{u}\cdot D\begin{Bmatrix} dx_2/d\xi \\ -dx_1/d\xi\end{Bmatrix}[\Delta\mathbf{u}]d\xi \\ &= \int_{\partial\Omega_\xi} p\,\delta\mathbf{u}\cdot\begin{Bmatrix} d\Delta u_2/d\xi \\ -d\Delta u_1/d\xi\end{Bmatrix}d\xi. \end{aligned} \tag{4.96}$$

sendo Δu_1 e Δu_2 os incrementos nas direções x_1 e x_2, respectivamente.

4.7 Discretização em Elementos Finitos e Procedimento de Solução

Estabelecidas as equações de equilíbrio linearizadas para a descrição material, pode-se descrever as quantidades envolvidas na forma discretizada fazendo uso de notação matriz-vetor, também conhecida como notação de Voigt. O foco estará no problema tridimensional com três graus de liberdade por nó (u_1, u_2, u_3). As formas bidimensionais podem ser encontradas nos trabalhos de Reddy (2006), Bathe (1996).

Na seção 4.7.1, aplica-se o método dos elementos finitos para discretizar as quantidades cinemáticas que formarão a base para as formas matriciais presentes na formulação Lagrangiana total (seção 4.7.2). Estabelecidas as formas discretizadas, o procedimento de solução será descrito na seção 4.7.4.

4.7.1 Cinemática Discretizada

A discretização usando elementos finitos isoparamétricos para interpolar a geometria inicial em termos das coordenadas nodais iniciais (ou materiais) $\mathbf{X}_a$ é

$$\mathbf{X} = \sum_{a=1}^{n} N_a(\xi_1, \xi_2, \xi_3)\mathbf{X}_a, \tag{4.97}$$

sendo $N_i(\xi_1, \xi_2, \xi_3)$ as funções de forma e n denota o número de nós ou modos do elemento. Após o processo de deformação, as coordenadas estão em termos da descrição espacial. Pode-se também escrever a expressão (4.97) na representação matricial

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} = \begin{bmatrix} N_1 & 0 & 0 & \cdots & N_n & 0 & 0 \\ 0 & N_1 & 0 & \cdots & 0 & N_n & 0 \\ 0 & 0 & N_1 & \cdots & 0 & 0 & N_n \end{bmatrix} \begin{pmatrix} X_1^1 \\ X_2^1 \\ X_3^1 \\ \vdots \\ X_1^n \\ X_2^n \\ X_3^n \end{pmatrix} \Rightarrow \mathbf{X} = \mathbf{N}^T\mathbf{X}^n. \tag{4.98}$$

A coordenada espacial interpolada do elemento é expressa como

$$\mathbf{x} = \sum_{a=1}^{n} N_a(\xi_1, \xi_2, \xi_3)\mathbf{x}_a \qquad \text{ou} \qquad \mathbf{x} = \mathbf{N}^T\mathbf{x}^n. \tag{4.99}$$

Por sua vez, o campo de deslocamento é interpolado por

$$\mathbf{u} = \sum_{a=1}^{n} N_a(\xi_1, \xi_2, \xi_3)\mathbf{u}_a \qquad \text{ou} \qquad \mathbf{u} = \mathbf{N}^T\mathbf{u}^n. \tag{4.100}$$

Assim, de forma análoga, pode-se escrever o deslocamento virtual $\delta\mathbf{u}$ e o incremento de deslocamento $\Delta\mathbf{u}$ nas formas interpoladas, respectivamente, da seguinte maneira

$$\delta\mathbf{u} = \sum_{a=1}^{n} N_a(\xi_1, \xi_2, \xi_3)\delta\mathbf{u}_a \qquad \text{ou} \qquad \delta\mathbf{u} = \mathbf{N}^T\delta\mathbf{u}^n \tag{4.101a}$$

$$\Delta\mathbf{u} = \sum_{a=1}^{n} N_a(\xi_1, \xi_2, \xi_3)\Delta\mathbf{u}_a \qquad \text{ou} \qquad \Delta\mathbf{u} = \mathbf{N}^T\Delta\mathbf{u}^n \tag{4.101b}$$

É possível, também, representar o gradiente do incremento do deslocamento e o gradiente de deslocamento virtual em notação matricial. No caso da descrição material, simbolizam-se as quantidades $\boldsymbol{\nabla}_0\Delta\mathbf{u}$ e $\boldsymbol{\nabla}_0\delta\mathbf{u}$ nas suas respectivas formas matriciais $\Delta\mathbf{H}$ e $\delta\mathbf{H}$, respectivamente, como

$$\begin{pmatrix} \frac{\partial \Delta u_1}{\partial X_1} \\ \frac{\partial \Delta u_1}{\partial X_2} \\ \frac{\partial \Delta u_1}{\partial X_3} \\ \frac{\partial \Delta u_2}{\partial X_1} \\ \frac{\partial \Delta u_2}{\partial X_2} \\ \frac{\partial \Delta u_2}{\partial X_3} \\ \frac{\partial \Delta u_3}{\partial X_1} \\ \frac{\partial \Delta u_3}{\partial X_2} \\ \frac{\partial \Delta u_3}{\partial X_3} \end{pmatrix} = \begin{bmatrix} \frac{\partial N_1}{\partial X_1} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_1^n} & 0 & 0 \\ \frac{\partial N_1}{\partial X_2} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_2^n} & 0 & 0 \\ \frac{\partial N_1}{\partial X_3} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_3^n} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial X_1} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_1^n} & 0 \\ 0 & \frac{\partial N_1}{\partial X_2} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_2^n} & 0 \\ 0 & \frac{\partial N_1}{\partial X_3} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_3^n} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial X_1} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_1^n} \\ 0 & 0 & \frac{\partial N_1}{\partial X_2} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_2^n} \\ 0 & 0 & \frac{\partial N_1}{\partial X_3} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_3^n} \end{bmatrix} \begin{pmatrix} \Delta u_1^1 \\ \Delta u_2^1 \\ \Delta u_3^1 \\ \vdots \\ \Delta u_1^n \\ \Delta u_2^n \\ \Delta u_3^n \end{pmatrix} \Rightarrow \Delta\mathbf{H} = \mathbf{B}_m^T\Delta\mathbf{u},$$

(4.102)

$$\begin{pmatrix} \frac{\partial \delta u_1}{\partial X_1} \\ \frac{\partial \delta u_1}{\partial X_2} \\ \frac{\partial \delta u_1}{\partial X_3} \\ \frac{\partial \delta u_2}{\partial X_1} \\ \frac{\partial \delta u_2}{\partial X_2} \\ \frac{\partial \delta u_2}{\partial X_3} \\ \frac{\partial \delta u_3}{\partial X_1} \\ \frac{\partial \delta u_3}{\partial X_2} \\ \frac{\partial \delta u_3}{\partial X_3} \end{pmatrix} = \begin{bmatrix} \frac{\partial N_1}{\partial X_1} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_1^n} & 0 & 0 \\ \frac{\partial N_1}{\partial X_2} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_2^n} & 0 & 0 \\ \frac{\partial N_1}{\partial X_3} & 0 & 0 & \cdots & \frac{\partial N_n}{\partial X_3^n} & 0 & 0 \\ 0 & \frac{\partial N_1}{\partial X_1} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_1^n} & 0 \\ 0 & \frac{\partial N_1}{\partial X_2} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_2^n} & 0 \\ 0 & \frac{\partial N_1}{\partial X_3} & 0 & \cdots & 0 & \frac{\partial N_n}{\partial X_3^n} & 0 \\ 0 & 0 & \frac{\partial N_1}{\partial X_1} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_1^n} \\ 0 & 0 & \frac{\partial N_1}{\partial X_2} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_2^n} \\ 0 & 0 & \frac{\partial N_1}{\partial X_3} & \cdots & 0 & 0 & \frac{\partial N_n}{\partial X_3^n} \end{bmatrix} \begin{pmatrix} \delta u_1^1 \\ \delta u_2^1 \\ \delta u_3^1 \\ \vdots \\ \delta u_1^n \\ \delta u_2^n \\ \delta u_3^n \end{pmatrix} \Rightarrow \delta \mathbf{H} = \mathbf{B}_m^T \delta \mathbf{u}. \tag{4.103}$$

O gradiente de deformação é interpolado sobre o elemento fazendo a diferenciação da equação (4.99) com respeito à coordenada inicial. A forma discretizada do gradiente de deformação é

$$\mathbf{F} = \sum_{a=1}^{n} \mathbf{x}_i \otimes \frac{\partial N_a}{\partial \mathbf{X}} \tag{4.104}$$

sendo

$$\frac{\partial N_a}{\partial \mathbf{X}} = \left(\frac{\partial \mathbf{X}}{\partial \boldsymbol{\xi}} \right)^{-T} \frac{\partial N_a}{\partial \boldsymbol{\xi}} \tag{4.105a}$$

$$\frac{\partial \mathbf{X}}{\partial \boldsymbol{\xi}} = \sum_{a=1}^{n} \mathbf{X}_i \otimes \frac{\partial N_a}{\partial \boldsymbol{\xi}}. \tag{4.105b}$$

A expressão (4.105b) é a representação da matriz Jacobiana. Na formulação Lagrangiana total, esta matriz não é alterada durante o procedimento de solução de Newton-Raphson, por isso, pode ser calculada somente na primeira iteração de Newton-Raphson.

Na formulação Lagrangiana atualizada, a matriz Jacobiana é dada pela seguinte expressão

$$\frac{\partial \mathbf{x}}{\partial \boldsymbol{\xi}} = \sum_{a=1}^{n} \mathbf{x}_i \otimes \frac{\partial N_a}{\partial \boldsymbol{\xi}}. \tag{4.106}$$

Entretanto, na descrição espacial, a matriz do Jacobiano é alterada a cada iteração de Newton-Raphson.

Devido à importância que as equações (4.104), (4.105b) e (4.106) possuem no MEF, escreve-se, a seguir, as suas respectivas notações indiciais por

$$F_{ij} = \sum_{a=1}^{n} x_{a,i} \frac{\partial N_a}{\partial X_j} \qquad i,j = 1,2,3; \tag{4.107}$$

$$\frac{\partial X_i}{\partial \xi_\alpha} = \sum_{a=1}^{n} X_{a,i} \frac{\partial N_a}{\partial \xi_\alpha} \qquad i,\alpha = 1,2,3. \tag{4.108}$$

$$\frac{\partial x_i}{\partial \xi_\alpha} = \sum_{a=1}^{n} x_{a,i} \frac{\partial N_a}{\partial \xi_\alpha} \qquad i,\alpha = 1,2,3. \tag{4.109}$$

A partir da equação (4.104) e (4.107) é possível obter a forma discreta do tensor à direita de Cauchy-Green como

$$\mathbf{C} = \mathbf{F}^T\mathbf{F} = \sum_{a,b} (\mathbf{x}_a \cdot \mathbf{x}_b) \frac{\partial N_a}{\partial \mathbf{X}} \otimes \frac{\partial N_b}{\partial \mathbf{X}} \qquad \text{ou} \qquad C_{ij} = \sum_{k=1}^{3} F_{ki}F_{kj}. \tag{4.110a}$$

4.7.2 Lagrangiano Total

Na formulação Lagrangiana total, a equação de equilíbrio linearizada, quando as forças de superfície são independentes da deformação, é dada por

$$\begin{aligned}\int_{\Omega_0} \left[\mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u} : \mathbb{D} : \mathbf{F}^T\boldsymbol{\nabla}_0\Delta\mathbf{u}\right] dV + \int_{\Omega_0} \mathbf{S} : [(\boldsymbol{\nabla}_0\Delta\mathbf{u})^T\boldsymbol{\nabla}_0\delta\mathbf{u}]dV \\ = -\int_{\Omega_0} \mathbf{S} : \left[\mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u}\right] dV + \int_{\Omega_0} \mathbf{f}\cdot\delta\mathbf{u}\, dV \\ + \int_{\partial\Omega_0} \mathbf{t}\cdot\delta\mathbf{u}\, dA.\end{aligned} \tag{4.111}$$

Como mostrado em Bhatti (2006), os termos desta equação podem ser arranjados na notação matriz-vetor da seguinte maneira

$$\int_{\Omega_0} \left[\mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u} : \mathbb{D} : \mathbf{F}^T\boldsymbol{\nabla}_0\Delta\mathbf{u}\right] dV \equiv \int_{\Omega_0^e} \delta\mathbf{H}^T\tilde{\mathbf{F}}\mathbf{D}\tilde{\mathbf{F}}\Delta\mathbf{H}dV^e, \tag{4.112a}$$

$$\int_{\Omega_0} \mathbf{S} : [(\boldsymbol{\nabla}_0\Delta\mathbf{u})^T\boldsymbol{\nabla}_0\delta\mathbf{u}]dV \equiv \int_{\Omega_0^e} \delta\mathbf{H}^T\tilde{\mathbf{S}}\Delta\mathbf{H}dV^e, \tag{4.112b}$$

$$\int_{\Omega_0} \mathbf{S} : \left[\mathbf{F}^T\boldsymbol{\nabla}_0\delta\mathbf{u}\right] dV \equiv \int_{\Omega_0^e} \delta\mathbf{H}^T\tilde{\mathbf{F}}\overline{\mathbf{S}}dV^e, \tag{4.112c}$$

$$\int_{\Omega_0} \mathbf{f}\cdot\delta\mathbf{u}\, dV \equiv \int_{\Omega_0^e} \mathbf{f}\cdot(\mathbf{N}\delta\mathbf{u}^n)dV^e, \tag{4.112d}$$

$$\int_{\partial\Omega_0} \mathbf{t}\cdot\delta\mathbf{u}\, dA \equiv \int_{\partial\Omega_0^e} \mathbf{t}\cdot(\mathbf{N}\delta\mathbf{u}^n)dA^e, \tag{4.112e}$$

sendo $\tilde{\mathbf{F}}$ o gradiente de deformação escrito na forma matricial compacta 6×9, isto é,

$$\tilde{\mathbf{F}} = \begin{bmatrix} \frac{\partial x_1}{\partial X_1} & 0 & 0 & \frac{\partial x_2}{\partial X_1} & 0 & 0 & \frac{\partial x_3}{\partial X_1} & 0 & 0 \\ 0 & \frac{\partial x_1}{\partial X_2} & 0 & 0 & \frac{\partial x_2}{\partial X_2} & 0 & 0 & \frac{\partial x_3}{\partial X_2} & 0 \\ 0 & 0 & \frac{\partial x_1}{\partial X_3} & 0 & 0 & \frac{\partial x_2}{\partial X_3} & 0 & 0 & \frac{\partial x_3}{\partial X_3} \\ \frac{\partial x_1}{\partial X_2} & \frac{\partial x_1}{\partial X_1} & 0 & \frac{\partial x_2}{\partial X_2} & \frac{\partial x_2}{\partial X_1} & 0 & \frac{\partial x_3}{\partial X_2} & \frac{\partial x_3}{\partial X_1} & 0 \\ 0 & \frac{\partial x_1}{\partial X_3} & \frac{\partial x_1}{\partial X_2} & 0 & \frac{\partial x_2}{\partial X_3} & \frac{\partial x_2}{\partial X_2} & 0 & \frac{\partial x_3}{\partial X_3} & \frac{\partial x_3}{\partial X_2} \\ \frac{\partial x_1}{\partial X_3} & 0 & \frac{\partial x_1}{\partial X_1} & \frac{\partial x_2}{\partial X_3} & 0 & \frac{\partial x_2}{\partial X_1} & \frac{\partial x_3}{\partial X_3} & 0 & \frac{\partial x_3}{\partial X_1} \end{bmatrix}. \tag{4.113}$$

A notação $\mathbf{D}$ é a representação matricial compacta do tensor elástico de quarta ordem $\mathbb{D}$. Para obter a matriz constitutiva 6×6, os coeficientes do tensor (calculados com a equação (4.56)) são agrupados da seguinte forma

$$\mathbf{D} = \begin{bmatrix} D_{1111} & D_{1122} & D_{1133} & \frac{1}{2}(D_{1112} + D_{1121}) & \frac{1}{2}(D_{1123} + D_{1132}) & \frac{1}{2}(D_{1113} + D_{1131}) \\ D_{2211} & D_{2222} & D_{2233} & \frac{1}{2}(D_{2212} + D_{2221}) & \frac{1}{2}(D_{2223} + D_{2232}) & \frac{1}{2}(D_{2213} + D_{2231}) \\ D_{3311} & D_{3322} & D_{3333} & \frac{1}{2}(D_{3312} + D_{3321}) & \frac{1}{2}(D_{3323} + D_{3332}) & \frac{1}{2}(D_{3313} + D_{3331}) \\ D_{1211} & D_{1222} & D_{1233} & \frac{1}{2}(D_{1212} + D_{1221}) & \frac{1}{2}(D_{1223} + D_{1232}) & \frac{1}{2}(D_{1213} + D_{1231}) \\ D_{2311} & D_{2322} & D_{2333} & \frac{1}{2}(D_{2312} + D_{2321}) & \frac{1}{2}(D_{2323} + D_{2332}) & \frac{1}{2}(D_{2313} + D_{2331}) \\ D_{3111} & D_{3122} & D_{3133} & \frac{1}{2}(D_{3112} + D_{3121}) & \frac{1}{2}(D_{3123} + D_{3132}) & \frac{1}{2}(D_{3113} + D_{3131}) \end{bmatrix}. \tag{4.114}$$

O segundo tensor de tensão de Piola-Kirchhoff, então, precisa ser escrito como uma matriz compacta 9×9 da seguinte maneira

$$\tilde{\mathbf{S}} = \begin{bmatrix} S_{11} & S_{12} & S_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ S_{21} & S_{22} & S_{23} & 0 & 0 & 0 & 0 & 0 & 0 \\ S_{31} & S_{32} & S_{33} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{11} & S_{12} & S_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{21} & S_{22} & S_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{31} & S_{32} & S_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & S_{11} & S_{12} & S_{13} \\ 0 & 0 & 0 & 0 & 0 & 0 & S_{21} & S_{22} & S_{23} \\ 0 & 0 & 0 & 0 & 0 & 0 & S_{31} & S_{32} & S_{33} \end{bmatrix}. \tag{4.115}$$

$\overline{\mathbf{S}}$ representa a forma vetorial compacta do segundo tensor de tensão de Piola-Kirchhoff, ou seja,

$$\overline{\mathbf{S}} = \{S_{11}, S_{22}, S_{33}, S_{12}, S_{23}, S_{13}\}^T. \tag{4.116}$$

Logo, o PTV linearizado toma a seguinte forma

$$(\delta\mathbf{u}^n)\int_{\Omega_0^e}\left[\mathbf{B}_m\overline{\mathbf{S}}\mathbf{B}_m^T+\mathbf{B}\tilde{\mathbf{F}}^T\mathbf{D}\tilde{\mathbf{F}}\mathbf{B}_m^T\right]dV^e\Delta\mathbf{u}^n=-(\delta\mathbf{u}^n)\int_{\Omega_0^e}\mathbf{B}_m\tilde{\mathbf{F}}^T\overline{\mathbf{S}}dV^e \\ +(\delta\mathbf{u}^n)\int_{\Omega_0^e}\mathbf{N}\mathbf{f}dV^e+(\delta\mathbf{u}^n)\int_{\partial\Omega_0^e}\mathbf{N}\mathbf{t}dA^e. \tag{4.117}$$

Como $\delta\mathbf{u}^n$ é um deslocamento virtual, pode-se omití-lo da equação (4.117).

A representação clássica do modelo de elementos finitos para o PTV linearizado na formulação Lagrangiana total é

$$(\mathbf{K}_c+\mathbf{K}_s)\Delta\mathbf{u}^n=-\mathbf{R}_i+\mathbf{R}_f+\mathbf{R}_t \tag{4.118}$$

sendo $\mathbf{K}_c$ a matriz de rigidez constitutiva (BONET; WOOD, 2008) ou matriz rigidez corrente (BHATTI, 2006); $\mathbf{K}_s$ é a matriz de rigidez de tensão (ou geométrica); $\mathbf{R}_i$ é o vetor de esforço nodal equivalente devido à tensão na configuração corrente; $\mathbf{R}_f$ é o vetor de carregamento nodal equivalente devido à força de corpo; $\mathbf{R}_t$ é o vetor de força nodal equivalente produzido pelas forças de superfície. Essas quantidades estão resumidas a seguir, respectivamente, como

$$\mathbf{K}_c=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\mathbf{B}_m\tilde{\mathbf{F}}^T\mathbf{D}\tilde{\mathbf{F}}\mathbf{B}_m^T Jd\xi_1d\xi_2d\xi_3, \tag{4.119a}$$

$$\mathbf{K}_s=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\mathbf{B}_m\overline{\mathbf{S}}\mathbf{B}_m^T Jd\xi_1d\xi_2d\xi_3, \tag{4.119b}$$

$$\mathbf{R}_i=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\mathbf{B}_m\tilde{\mathbf{F}}^T\overline{\mathbf{S}}Jd\xi_1d\xi_2d\xi_3, \tag{4.119c}$$

$$\mathbf{R}_f=\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\mathbf{N}\mathbf{f}Jd\xi_1d\xi_2d\xi_3, \tag{4.119d}$$

$$\mathbf{R}_t=\int_{-1}^{1}\int_{-1}^{1}\mathbf{N}\mathbf{t}Jd\xi_1d\xi_2. \tag{4.119e}$$

A matriz de tangente do elemento $\mathbf{K}$ é a soma das matrizes $\mathbf{K}_c$ e $\mathbf{K}_s$, ou seja,

$$\mathbf{K}=\mathbf{K}_c+\mathbf{K}_s. \tag{4.120}$$

Já a força desbalanceada $\mathbf{R}$ é a soma das parcelas

$$\mathbf{R} = -\mathbf{R}_i + \mathbf{R}_f + \mathbf{R}_t. \tag{4.121}$$

Logo, a equação (4.118) pode ser escrita na seguinte forma reduzida

$$\mathbf{K}\Delta\mathbf{u} = \mathbf{R}. \tag{4.122}$$

4.7.3 Carregamento Dependente da Deformação

Com base nos trabalhos de Bhatti (2006), Bonet e Wood (2008), a matriz de rigidez proveniente da linearização do carregamento de pressão em uma superfície é dada por

$$[\mathbf{K}_t]_{ij} = \int_{\partial\Omega_{\xi_1\xi_2}} p(\mathbf{a}_{ij} - \mathbf{c}_{ij}) d\xi_1 d\xi_2, \tag{4.123}$$

sendo

$$\mathbf{a}_{ij} = N_i \frac{\partial N_j}{\partial \xi_1} \begin{bmatrix} 0 & \frac{\partial x_3}{\partial \xi_2} & -\frac{\partial x_2}{\partial \xi_2} \\ -\frac{\partial x_3}{\partial \xi_2} & 0 & \frac{\partial x_1}{\partial \xi_2} \\ \frac{\partial x_2}{\partial \xi_2} & -\frac{\partial x_1}{\partial \xi_2} & 0 \end{bmatrix}; \qquad \mathbf{c}_{ij} = N_i \frac{\partial N_j}{\partial \xi_2} \begin{bmatrix} 0 & \frac{\partial x_3}{\partial \xi_1} & -\frac{\partial x_2}{\partial \xi_1} \\ -\frac{\partial x_3}{\partial \xi_1} & 0 & \frac{\partial x_1}{\partial \xi_1} \\ \frac{\partial x_2}{\partial \xi_1} & -\frac{\partial x_1}{\partial \xi_1} & 0 \end{bmatrix}. \tag{4.124}$$

Já, no caso de um problema bidimensional, tem-se a seguinte matriz de rigidez de acordo com (MOK et al., 1999)

$$\mathbf{K}_t = \int_{\partial\Omega_\xi} p \begin{bmatrix} 0 & N_1 \frac{\partial N_1}{\partial \xi} & \cdots & 0 & N_1 \frac{\partial N_n}{\partial \xi_1} \\ -N_1 \frac{\partial N_1}{\partial \xi} & 0 & \cdots & -N_1 \frac{\partial N_n}{\partial \xi} & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & N_n \frac{\partial N_1}{\partial \xi} & \cdots & 0 & N_1 \frac{\partial N_n}{\partial \xi} \\ -N_n \frac{\partial N_1}{\partial \xi} & 0 & \cdots & -N_n \frac{\partial N_n}{\partial \xi} & 0 \end{bmatrix} d\xi \tag{4.125}$$

As matrizes calculadas em (4.123) e (4.125) são não simétricas e devem ser adicionadas a

matriz tangente do elemento, tornando-a também não-simétrica.

4.7.4 Procedimento de Solução por Newton-Raphson

O procedimento de Newton-Raphson é o esquema mais utilizado na literatura para solucionar problemas não-lineares usando elementos finitos. O método consiste em um processo iterativo para determinar a solução do sistema de equação linearizado dado por (4.122) em incrementos de carregamento sujeito a um critério de convergência.

De maneira simples, o carregamento pode ser aplicado em uma série de incrementos como

$$\mathbf{R}_e = \mathbf{R}_f + \mathbf{R}_t = \sum_{j=1}^{l} \Delta \mathbf{R}_{e_j} \tag{4.126}$$

sendo l o número total de incrementos e $\Delta \mathbf{R}_{e_j}$ o incremento do carregamento. A adoção de incrementos de carregamento, permite que se atinja a convergência de maneira mais fácil. No caso de material hiperelástico, a maneira como esses incrementos são aplicados não interfere na solução final. Porém, no caso de materiais não hiperelásticos essa conclusão não é válida (BONET; WOOD, 2008).

Sendo o método de Newton-Raphson um processo iterativo, um critério de convergência adequado dever utilizado para que o resultado obtido seja satisfatório dentro do nível de tolerância estipulado. Se o valor da tolerância for muito grande, pode-se ter uma solução imprecisa. No entanto, a escolha de uma valor muito pequeno para a tolerância, acarreta em um alto esforço computacional.

Como mostrado em Bathe (1996), existem três critérios de convergência que podem ser aplicados no procedimento de Newton-Raphson na solução de um problema de grandes deformações. O primeiro critério é baseado na norma L^2 do deslocamento. Esse critério relaciona a norma L^2 do incremento de deslocamento para a $k-$ésima iteração de Newton-Raphson e a norma L^2 do vetor de deslocamento para o tempo $t + \Delta t$, ou seja,

$$\frac{\|\Delta \mathbf{u}_{k+1}\|}{\|^{t+\Delta t}\mathbf{u}_{k+1}\|} \leq \epsilon_d. \tag{4.127}$$

Esse critério precisa de um valor ϵ_d bem próximo de zero para obter uma solução precisa. Mas mesmo assim, deve ser utilizado com bastante cautela, pois mesmo satisfazendo a equação (4.127), o valor da norma da força desbalanceada pode ser grande.

Já o segundo critério de convergência é baseado na medida da norma L^2 do vetor de força desbalanceada. Dessa forma, o critério exige que relação entre a norma do vetor de força desbalanceada e o vetor de incremento de carregamento externo fique dentro de uma tolerância, isto é,

$$\frac{\|^{t+\Delta t}\mathbf{R}_e - {}^{t+\Delta t}\mathbf{R}_i\|}{\|^{t+\Delta t}\mathbf{R}_e - {}^{t}\mathbf{R}_i\|} \leq \epsilon_f. \tag{4.128}$$

No entanto, satisfazer esse critério de convergência com um valor de tolerância muito próximo de zero é uma tarefa difícil para alguns problemas hiperelásticos. Em alguns casos, mesmo o incremento de deslocamento sendo muito pequeno, tem-se ainda um alto valor para o critério da força desbalanceada.

Dessa forma, surge um terceiro critério que calcula a norma L^2 do trabalho realizado pela força desbalanceada sobre o incremento de deslocamento durante cada iteração de Newton-Raphson com a norma L^2 do trabalho realizado pela força desbalanceada sobre o incremento do deslocamento na primeira iteração. Essa relação é dada pela seguinte expressão

$$\frac{\|\Delta\mathbf{u}_{k+1} \cdot ({}^{t+\Delta t}\mathbf{R}_e - {}^{t+\Delta t}\mathbf{R}_{i_k})\|}{\|\Delta\mathbf{u}_1 \cdot ({}^{t+\Delta t}\mathbf{R}_e - {}^{t}\mathbf{R}_i)\|} \leq \epsilon_e. \tag{4.129}$$

O procedimento de Newton-Raphson está esquematizado no Algoritmo 4.1.

Algoritmo 4.1 Algoritmo de Newton-Raphson.

Entrada geometria, propriedades do material e parâmetros de solução
Inicializar $\mathbf{R}_i = \mathbf{0}$, $\mathbf{x} = \mathbf{X}$, $\mathbf{R}_e = \mathbf{0}$
for j = 1 : número de passos de carregamento **do**
  Calcular $\Delta\mathbf{R}_e$
  Calcular $\mathbf{R}_e = \mathbf{R}_e + \Delta\mathbf{R}_e$
  Calcular $\mathbf{K}$
  while Critério de convergência $\leq \epsilon$ **do**
    Resolver $\mathbf{K}\Delta\mathbf{u} = -\mathbf{R}$
    Atualizar $\mathbf{u} = \mathbf{u} + \Delta\mathbf{u}$
    Atualizar $\mathbf{K}$, $\mathbf{R}_i$, $\mathbf{x} = \mathbf{x} + \Delta\mathbf{u}$
    Calcular critério de convergência
  end while
end for

5 Contato Bidimensional em Pequenas Deformações

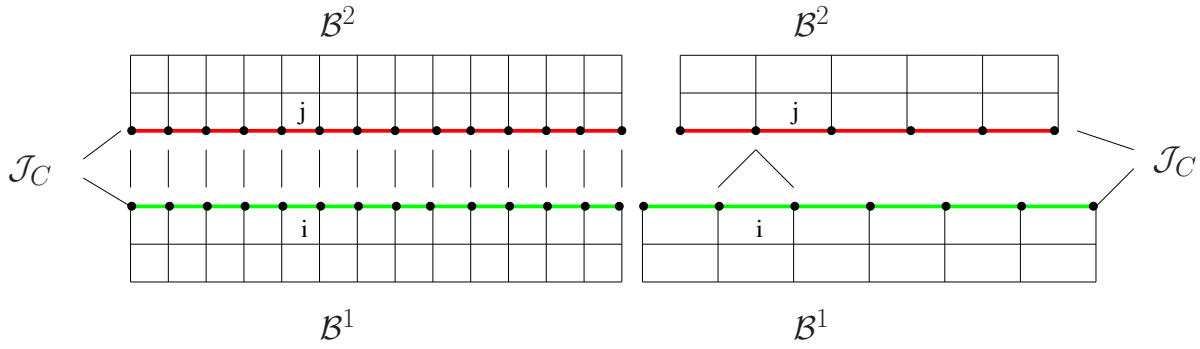
De forma geral, o problema de contato é considerado como um problema de não-linearidade cinemática segundo Bhatti (2006). Neste capítulo, o problema de contato será abordado segundo a cinemática infinitesimal (WRIGGERS, 2006), ou seja, os efeitos de grandes rotações e/ou grandes deformações provenientes da parte não-linear do tensor de deformação de Green-Lagrange serão desprezados. Assim, para esse caso é possível incorporar as restrições na base nodal.

Em geral, as malhas usadas para problemas de contato com pequenas deformações possuem a característica de ser uma relação nó-nó única (biunívoca) para um dado par de contato $\mathcal{J}_C$, ou seja, os nós do elemento j do contorno do corpo $\mathcal{B}^2$ (contator) coincidem com os nós do elemento i do contorno do corpo $\mathcal{B}^1$ (alvo) como mostrado na Figura 5.1(a). Essas malhas são denominadas de casadas ("matching") e são usadas na discretização usando elementos isoparamétricos, desde que o grau dos elementos da superfície de contato seja o mesmo. Entretanto, os geradores automáticos de malhas muitas vezes produzem malhas não-casadas ("non-matching") como mostrado na Figura 5.1(b). Para este tipo de malha, percebe-se que, para um certo nó do elemento j do corpo $\mathcal{B}^2$, não existirá uma relação biunívoca com um dos nós do elemento i do corpo $\mathcal{B}^1$. Dessa forma, devem ser utilizadas discretizações tais como nó-segmento descrito em Wriggers (2006), Ju, Stonet e Rowlands (1995) e Bittencourt e Creus (1998) ou via métodos Mortar e de Nitsche mostrados em Wriggers (2006). Nesse trabalho, apenas a discretização nó-segmento será utilizada para malhas não-casadas.

Exemplos teóricos serão apresentados e discutidos ao longo do texto para verificar algumas dificuldades encontradas na solução do problema de contato, tais como tipo de discretização, método de regularização das restrições de contato e critério de parada da solução via método de Newton-Raphson.

5.1 Cinemática do Contato

Sejam dois corpos $\mathcal{B}^1$ (alvo) e $\mathcal{B}^2$ (contator) cujos pontos são descritos pelas coordenadas materiais $\mathbf{X}^1$ e $\mathbf{X}^2$, respectivamente, que se aproximam durante o processo de deformação e en-



(a) Malha ajustada ("matching meshes").

(b) Malha não-ajustada ("non-matching").

Figura 5.1: Malhas para par de contato $\mathcal{J}_C$.

tram em contato em partes de seus contornos indicado por Γ_C. Esta situação é ilustrada na Figura 5.2. Inicialmente, os dois pontos ocupam posições distintas no espaço e após o processo de deformação, passam a ocupar a mesma posição espacial na configuração corrente (deformada), ou seja, $\varphi(\mathbf{X}^1) = \varphi(\mathbf{X}^2)$. A seguir apresentam-se as relações cinemáticas do contato para as direções normal e tangencial, as quais são necessárias para formular as três condições de contato possíveis: contato normal; contato com adesão e contato com deslizamento.

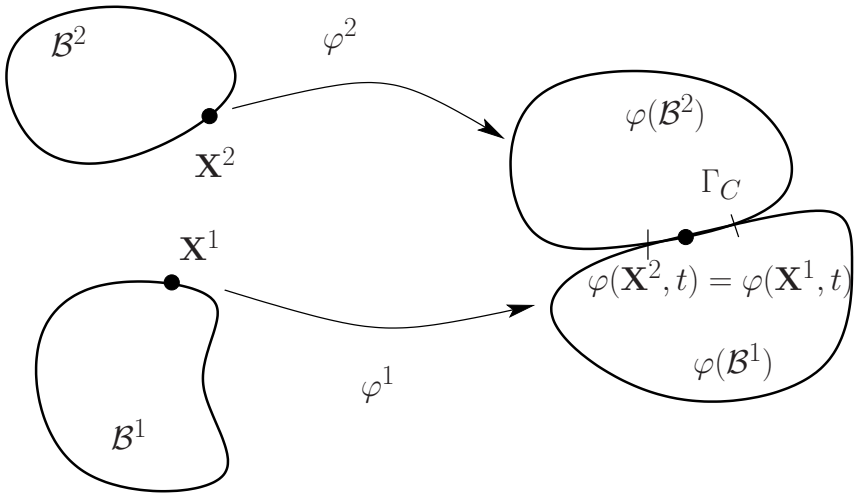


Figura 5.2: Deformação de dois corpos em contato (WRIGGERS, 2006).

5.1.1 Contato Normal

Para exemplificar a situação de contato normal, considere os dois corpos da Figura 5.3 que estão entrando em contato devido a um processo de deformação. Sejam $\mathbf{X}^1$ e $\mathbf{X}^2$as coordenadas materiais de dois pontos arbitrários dos corpos $\mathcal{B}^1$ e $\mathcal{B}^2$, respectivamente. Durante o processo de deformação, os corpos entram em contato e as coordenadas espaciais dos pontos dos corpos são dadas por

$$\mathbf{x}^\alpha = \mathbf{X}^\alpha + \mathbf{u}^\alpha, \tag{5.1}$$

sendo α o índice que indica o corpo (nesse caso $\alpha = 1, 2$), $\mathbf{u}^\alpha$ o campo de deslocamento do corpo $\mathcal{B}^\alpha$ e $\mathbf{x}^\alpha$ as coordenadas espaciais na configuração corrente $\varphi(\mathcal{B}^\alpha)$ dos corpos $\mathcal{B}^\alpha$.

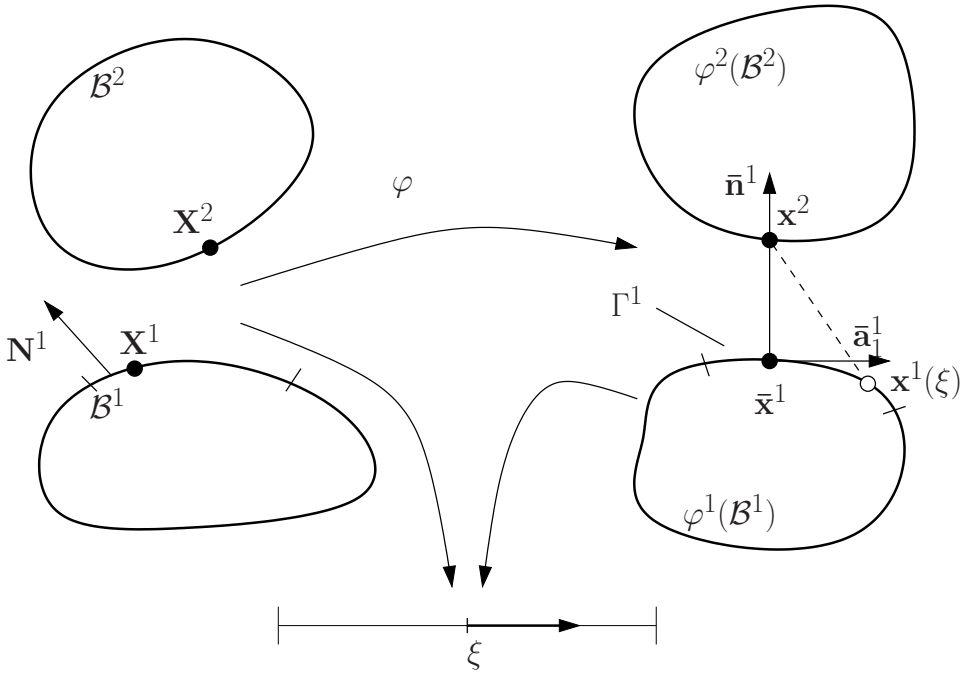


Figura 5.3: Configuração deformada dos corpos $\mathcal{B}^\alpha$ (WRIGGERS, 2006).

A condição de não-penetração entre os corpos é dada por

$$(\mathbf{x}^2 - \mathbf{x}^1) \cdot \mathbf{n}^1 \geq 0, \tag{5.2}$$

sendo $\mathbf{n}^1$ o vetor normal associado ao corpo $\mathcal{B}^1$

Assume-se que o contorno Γ_C^1 no para o problema de contato 2D é parametrizado pela coordenada convectiva ξ, de tal forma que as coordenadas espaciais dos pontos em Γ_C^1 são expressas por

$$\mathbf{x}^1 = \mathbf{x}^1(\xi) \tag{5.3}$$

Assume-se ainda que o contorno de contato constitui-se em uma região convexa localmente. Dessa maneira pode-se relacionar cada ponto $\mathbf{x}^2$ de Γ_C^2 a um ponto $\bar{\mathbf{x}}^1 = \mathbf{x}^1(\xi)$ através do problema de mínima distância

$$\hat{d}^1(\xi) = ||\mathbf{x}^2 - \bar{\mathbf{x}}^1|| = \min_{\mathbf{x}^1 \subseteq \Gamma_C^1} ||\mathbf{x}^2 - \mathbf{x}^1(\xi)||. \tag{5.4}$$

A solução de (5.4), é encontrada a partir da condição

$$\frac{d}{d\xi}\hat{d}^1(\xi) = \frac{\mathbf{x}^2 - \mathbf{x}^1(\xi)}{||\mathbf{x}^2 - \mathbf{x}^1(\xi)||} \cdot \mathbf{x}^1_{,1}(\xi) = 0, \tag{5.5}$$

sendo $\mathbf{x}^1_{,1}(\xi)$ o vetor tangente $\mathbf{a}_1^1$. Assim, o primeiro termo da equação anterior tem a mesma direção do vetor normal $\mathbf{n}^1$. Portanto, $\mathbf{n}^1 \cdot \mathbf{a}_1^1 = 0$, provando a ortogonalidade da solução.

O ponto de contato na superfície alvo, $\bar{\mathbf{x}}^1$, é calculado usando a solução ótima $\bar{\xi}$ de terminado a partir de (5.5). Usa-se o símbolo da barra sobre as quantidades que foram calculadas com o resultado do problema de mínimo dado pela equação (5.5), isto é, $\bar{\mathbf{a}}_1^1$ e $\bar{\mathbf{n}}^1$ são calculadas com o valor de $\bar{\xi}$.

Dessa forma, com o valor da coordenada convectiva $\bar{\xi}$ e do vetor tangente, $\bar{\mathbf{a}}_1^1$, para o elemento alvo, pode-se calcular o vetor normal $\bar{\mathbf{n}}^1$ da seguinte forma

$$\bar{\mathbf{n}}^1 = \frac{\mathbf{e}_3 \times \bar{\mathbf{a}}_1^1}{||\mathbf{e}_3 \times \bar{\mathbf{a}}_1^1||}, \tag{5.6}$$

sendo $\mathbf{e}_3 = [0, 0, 1]^T$ o vetor unitário na direção perpendicular ao plano xy.

Com esses resultados, a condição de não-penetração ou gap, que é dada pela equação

$$g_n = (\mathbf{x}^2 - \bar{\mathbf{x}}^1) \cdot \bar{\mathbf{n}}^1 \geq 0. \tag{5.7}$$

Convenciona-se que quando $g_n \leq 0$ representa a penetração, indicando que os corpos estão inter-penetrando um ao outro. Caso contrário, tem-se a condição de gap indicando que os corpos não estão em contato (BHATTI, 2006; ANSYS INC., 2008). Para pequenas deformações, pode-se reescrever a equação (5.7) como

$$g_n = (\mathbf{u}^2 - \mathbf{u}^1) \cdot \mathbf{n}^1 + g_0 \geq 0, \tag{5.8}$$

sendo g_0 a distância inicial entre os pontos dos corpos em contato definido por

$$g_0 = (\mathbf{X}^2 - \mathbf{X}^1) \cdot \mathbf{n}^1. \tag{5.9}$$

A equação (5.8) será utilizada na discretização usando elementos isoparamétricos, enquanto a equação (5.7) será empregada na discretização nó-segmento.

5.1.2 Contato Tangencial

Para o contato na presença de atrito, deve-se distinguir dois possíveis casos. No primeiro caso denominado *adesão*, os pontos em contato não se movimentam na direção tangencial devido ao atrito. No segundo caso, denominado *deslizamento*, existe um movimento relativo na direção tangencial entre os pontos em contato. Assim, deve-se criar um equacionamento de modo a relacionar o movimento e a deformação que os corpos em contato sofrem na interface, o que será feito a seguir para as duas condições de movimento tangencial na interface.

Condição de adesão

A condição de adesão pode ser considerada como uma condição derivada do problema de projeção, dado pela equação (5.5). Pela própria definição de condição de adesão que diz que os nós

em contato não possuem deslocamentos tangenciais relativos, pode-se escrever a seguinte equação que define a condição de gap tangencial para problemas bidimensionais,

$$g_t = (\mathbf{x}^2 - \bar{\mathbf{x}}^1) \cdot \bar{\mathbf{a}}^1 = 0. \tag{5.10}$$

Condição de deslizamento

O movimento na direção tangencial entre dois corpos em contato se deve segundo Wriggers (2006) à mudança do ponto $\mathbf{x}^2$ com relação a sua projeção $\bar{\mathbf{x}}^1$. Isso significa que o ponto dado pela coordenada convectiva $\bar{\xi}$ se moverá na superfície de contato de $\mathcal{B}^1$. Assim, existe uma velocidade relativa entre os nós em contato entre os instantes T_0 e T que deve ser determinada. Primeiro, calcula-se o diferencial do deslizamento tangencial em termos do corpo $\mathcal{B}^1$ como

$$dg_t = d\xi = \dot{\bar{\xi}} dT. \tag{5.11}$$

Logo, o deslizamento total é dado por

$$g_t = \int_{T_0}^{T} |\dot{\bar{\xi}}| dT. \tag{5.12}$$

Para o cálculo da equação (5.12) é necessário conhecer o valor de $\dot{\bar{\xi}}$, o qual é calculado pela relação (WRIGGERS, 2006)

$$\dot{\bar{\xi}} = \frac{1}{\bar{\mathbf{x}}^1_{,1} \cdot \bar{\mathbf{x}}^1_{,1} - g_n \bar{\mathbf{x}}^1_{,11}} [(\dot{\mathbf{x}}^2 - \dot{\bar{\mathbf{x}}}^1) \cdot \bar{\mathbf{x}}^1_{,1} + g_n \bar{\mathbf{n}}^1 \cdot \dot{\bar{\mathbf{x}}}^1_{,1}]. \tag{5.13}$$

No caso de contato envolvendo pequenas deformações, a equação (5.13) pode ser reescrita como

$$\dot{\bar{\xi}} = (\dot{\mathbf{u}}^2 - \dot{\bar{\mathbf{u}}}^1) \cdot \bar{\mathbf{x}}^1_{,1} = (\dot{\mathbf{u}}^2 - \dot{\bar{\mathbf{u}}}^1) \cdot \bar{\mathbf{a}}^1_1. \tag{5.14}$$

Usando-se estes resultados, define-se uma outra importante relação cinemática. Essa relação consiste na função velocidade relativa tangencial, a qual é definida como uma derivada no sentido

de Lie, ou seja,

$$\mathcal{L}_v g_t = \dot{g}_t = \dot{\bar{\xi}}, \tag{5.15}$$

A integral da equação (5.12) pode ser reescrita passando o limite de integração no domínio do tempo para o domínio das coordenadas convectivas calculadas nos instantes T_0 e T, isto é,

$$g_t = \int_{\xi_0}^{\xi} \sqrt{\bar{\mathbf{x}}^1_{,1} \cdot \bar{\mathbf{x}}^1_{,1}} d\xi. \tag{5.16}$$

O limite superior da integral (5.16) é dependente da deformação. Entretanto, pode-se transformá-la em uma integral com limite fixo, usando-se a a formulação via elementos isoparamétricos.

Variação dos Gaps Normal e Tangencial

Para construção da discretização em elementos finitos e posterior solução via método iterativo, é importante conhecer as variações dos gaps normal e tangencial. A variação do gap normal para contato 2D é dada por

$$\delta g_n = \delta[\mathbf{x}^2 - \mathbf{x}^1(\bar{\xi})] \cdot \mathbf{n}^1(\bar{\xi}). \tag{5.17}$$

Usando a regra da cadeia, a variação sobre todos os termos dependente da deformação leva a seguinte equação

$$\delta g_n = [\boldsymbol{\eta}^2 - \bar{\boldsymbol{\eta}}^1 - \bar{\mathbf{x}}^1_{,\alpha} \delta\xi] \cdot \bar{\mathbf{n}}^1 + [\mathbf{x}^2 - \bar{\mathbf{x}}^1] \cdot \delta\bar{\mathbf{n}}^1, \tag{5.18}$$

sendo $\boldsymbol{\eta}^\alpha = \delta\mathbf{x}^\alpha$ o vetor de função teste. Simplificado a equação (5.18) para incluir apenas efeitos de pequenas deformações, chega-se a seguinte equação

$$\delta g_n = [\boldsymbol{\eta}^2 - \bar{\boldsymbol{\eta}}^1] \cdot \bar{\mathbf{n}}^1. \tag{5.19}$$

Já a variação do gap tangencial é mais complicada e a dedução completa pode ser encontrada em Wriggers (2006). Como feito para o caso da derivada no tempo para o deslocamento tangencial

(veja equação (5.15)), a variação do gap tangencial é dada por

$$\delta g_t = \delta\bar{\xi}, \tag{5.20}$$

sendo $\delta\bar{\xi}$ dado por

$$\delta\bar{\xi} = \frac{1}{\bar{\mathbf{x}}^1_{,1}\cdot\bar{\mathbf{x}}^1_{,1} - g_n\bar{\mathbf{x}}^1_{,11}}[(\boldsymbol{\eta}^2 - \bar{\boldsymbol{\eta}}^1)\cdot\bar{\mathbf{x}}^1_{,1} + g_n\bar{\mathbf{n}}^1\cdot\bar{\boldsymbol{\eta}}^1_{,1}]. \tag{5.21}$$

Para pequenas deformações a equação (5.21) se reduz à seguinte expressão

$$\delta\bar{\xi} = (\boldsymbol{\eta}^2 - \bar{\boldsymbol{\eta}}^1)\cdot\bar{\mathbf{x}}^1_{,1} = (\boldsymbol{\eta}^2 - \bar{\boldsymbol{\eta}}^1)\cdot\bar{\mathbf{a}}^1_1. \tag{5.22}$$

Na próxima seção, mostra-se o processo de determinação dos pontos em contato, ou seja, conhecer em que pontos é aplicada a cinemática de contato.

5.1.3 Busca pelo Contato

Em um problema de contato, destacam-se dois procedimentos de extrema importância:

1. busca por nós em contato, que normalmente é um procedimento computacionalmente caro, pois demanda muito tempo de computação na solução de um problema de mínima distância. Esse problema consiste em determinar a menor distância ortogonal entre dois elementos em contato;
2. aplicação correta das condições de contato normal e tangencial.

Nessa seção apresentam-se detalhes sobre o primeiro item, ficando para a seção 5.2 os esclarecimentos sobre o segundo. De maneira geral, a busca pelo contato consiste em retornar o elemento alvo que estará em contato com um dado nó contator. Esse procedimento é usado na discretização nó-segmento (seção 5.3.2) e em contato com grandes deformações. Para a discretização usando elementos isoparamétricos em pequenas deformações (seção 5.3.1), não existe a necessidade da busca

pelo contato uma vez que os pares de contato $\mathcal{J}_C$ já são fornecidos pelo gerador de malha e não ocorrem grandes deslizamentos entre os nós em contato. Baseado na Figura 5.4, o procedimento de busca pode ser dividido em duas fases:

1. identificação dos possíveis elementos alvos, isto é, encontrando o elemento alvo para um dado nó contator, $\mathbf{x}^2$, onde a distância ortogonal entre o nó contator e o elemento alvo seja o mínimo, de acordo com a equação (5.4). Na Figura 5.4 o elemento alvo selecionado possui a incidência $\mathbf{x}_i^1$-$\mathbf{x}_{i+1}^1$;

2. encontrar a coordenada convectiva ξ da projeção resolvendo por método iterativo a equação (5.5).

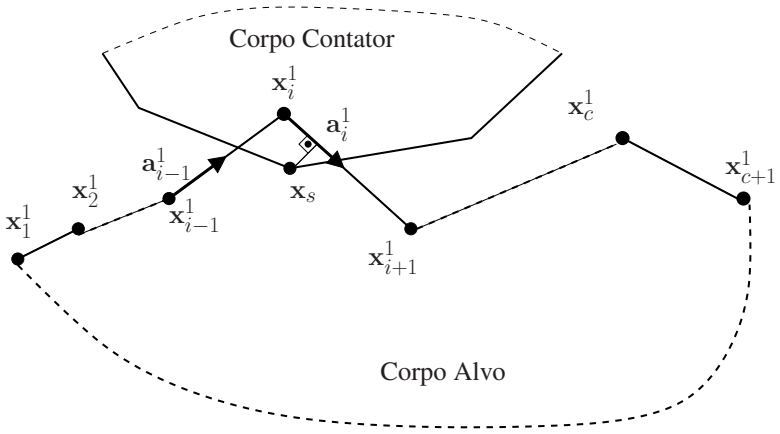


Figura 5.4: Busca pelo contato para o caso bidimensional (WRIGGERS, 2006).

O processo de busca do contato está exemplificado no Algoritmo 5.1.

As Listagens 5.1 e 5.2 mostram a rotina MatLab usada para encontrar a coordenada convectiva $\bar{\xi}$ e o mapeamento que deve ser feito no elemento alvo para calcular $\bar{\mathbf{x}}^1$, respectivamente.

Algoritmo 5.1 Determinação do elemento alvo.

requer: pares de contato $\mathcal{J}_C$ definidos pelo usuário
  for n_c= 1 : número de nós contatores **do**
    Retorna as coordenadas $\mathbf{x}^2$ do nó contator n_c
    for el_t = 1 : número de elementos alvos **do**
      Retorna as coordenadas ($\mathbf{x}^1$) e o formato do elemento el_t
      Retorna o mapeamento para o elemento el_t
      Usar método iterativo para resolver a equação (5.5) para xi
      Calcular $\bar{\mathbf{x}}^1$ com o valor de $\bar{\xi}$
      Calcular a função gap/interpenetração dado pela equação (5.7)
      Criar um vetor de dados $g_n(el_t)$ para armazenar o valor da equação (5.7)
    end for
    Retornar o menor valor armazenado no vetor de dados g_n e o índice correspondente (que indica o elemento alvo)
  end for

Listagem 5.1: Rotina para cálculo da coordenada convectiva ξ.

```
function [x0, xt, yt] = GetContactPoint2D(tcoord,ccoord,tp)
%tcoord  = coordenada do nó contator projetado no alvo
%ccoord  = coordenada do elemento contator
%tp      = tipo do elemento alvo.
xc = ccoord(1); yc = ccoord(2); %coordenada do nó contator
x0 = fzero(@dfa,0);             %resolve o problema de mínimo
fhandle = @dist;
    %Problema de minimização
    function dd = dist(a)
        %rotina que devolve o mapeamento do elemento alvo
        [xt,yt,dxa,dya,nmap]=LineMap2D(a, tcoord, tp);
        %função objetivo : distância ortogonal entre o
        %ponto contator e sua projeção no
        %elemento alvo
        dd = (xt-xc)*dxa+(yt-yc)*dya; %função objetivo
    end
end
```

Listagem 5.2: Rotina para cálculo da coordenada do ponto de contato na face do elemento alvo.

```
function [x,y,dxa,dya,nmap] = LineMap2D(a, tcoord, tp)
%tp = 1 -> linha
%tp = 2 -> quadrático
%a = coordenada convectiva ($\xi$)
%coordenada dos nós do elemento alvo
switch(tp)
    case 1
        %mapeamento linear
        nmap  = [0.5*(1-a), 0.5*(1+a)];
        dnmap = [-0.5, 0.5];
    case 2
        %mapeamento quadrático
        nmap  = [0.5*a*(a-1), 0.5*a*(1+a), 1-a^2];
        dnmap = [-0.5+a, 0.5+a, -2*a];
end
%interpolação geométrica do nó projetado
x = nmap*tcoord(:,1); y = nmap*tcoord(:,2);
dxa = dnmap*tcoord(:,1); dya = dnmap*tcoord(:,2);
```

5.2 Regularização da Interface de Contato via Método das Penalidades

5.2.1 Equações Constitutivas para Interface do Contato Normal

A formulação a qual trata o contato normal (também conhecida como contato sem atrito) como uma condição de restrição unilateral é em geral usada apenas para tratar as restrições geométricas de maneira correta, ou seja, analisa se existe contato ou não entre os corpos durante o processo de deformação. Neste caso, o atrito não interfere e a equação constitutiva para o contato normal é dado pela a força de contato normal proveniente da reação à penetração na superfície de contato.

A condição matemática para que não ocorra a interpenetração entre os corpos em contato é dada pela equações (5.7) ou (5.8) para o caso de pequenas deformações. O contato ocorre quando $g_n = 0$ e, neste caso, o vetor de tensão é dado por

$$\mathbf{t}^1 = \sigma \bar{\mathbf{n}}^1 = t_n \mathbf{n}^1 + t_t^1 \bar{\mathbf{a}}_1^1. \tag{5.23}$$

Na condição de contato, a componente de tensão normal t_n não pode ser zero. O vetor tensão atua em ambos os corpos porém com sentidos opostos devido a condição de equilíbrio, conforme mostrada na Figura 5.5.

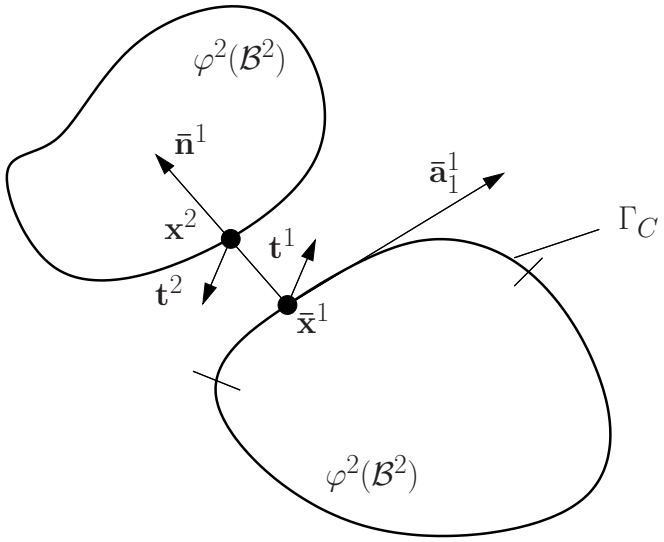


Figura 5.5: Tensão na interface de contato (LAURSEN; SIMO, 1993).

Para que se caracterize o problema de contato as seguintes condições devem ser satisfeitas: $g_n = 0$ e $t_n < 0$ (força de compressão). Caso exista um gap entre os corpos, então tem-se $g_n > 0$ e $t_n = 0$. Isto leva ao seguinte conjunto de condições para o contato unilateral

$$g_n \geq 0, \tag{5.24a}$$

$$t_n \leq 0, \tag{5.24b}$$

$$t_n \, g_n = 0. \tag{5.24c}$$

$$\tag{5.24d}$$

As condições dadas por (5.24) e ilustradas na Figura 5.6 são conhecidas como condições de HERTZ-SIGNORINI-MOREAU. Essas condições estabelecem a base para tratar o problema de contato sem atrito dentro de um contexto de otimização (WRIGGERS, 2006).

A região 1 da Figura 5.6 indica o quadrante onde ocorre o contato. Deve-se permitir uma certa tolerância à penetração entre os corpos para que o problema de contato seja solucionado usando o *Método das Penalidades*, conforme será visto na seção 5.2.4 em conjunto com o *Método de Newton-Raphson* (REDDY, 2006). As linhas escuras representam as condições dadas por (5.24).

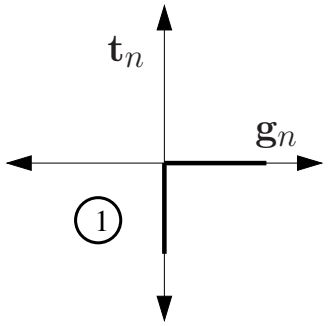


Figura 5.6: Força de contato versus função gap (MIJAR; ARORA, 2004a).

5.2.2 Equações Constitutivas para Interface do Contato Tangencial

Para a direção tangencial, utiliza-se a lei de Coulomb modificada, sendo as equações constitutivas para força de atrito formuladas via teoria da elasto-plasticidade. Usa-se o método do *mapeamento de retorno* para integrar a equação constitutiva de atrito e determinar suas componentes. Essa formulação foi utilizada em conjunto com o MEF por Laursen e Simo (1993) e Giannakopoulos (1989). A formulação a seguir representa o caso 2D do contato com atrito, assim, algumas grandezas vetoriais foram omitidas para facilitar o desenvolvimento. Para o caso 3D, Bittencourt e Creus (1998) desenvolve as formas vetoriais.

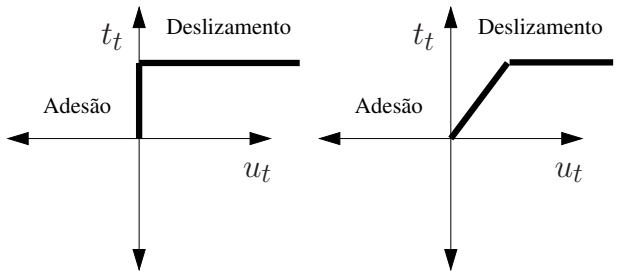
A lei de Coulomb para contato 2D fornece a seguinte relação para o atrito

$$f = |t_t| - \mu|t_n| < 0 \qquad \text{Condição de adesão;} \tag{5.25}$$

$$f = |t_t| - \mu|t_n| = 0 \qquad \text{Condição de deslizamento,} \tag{5.26}$$

sendo μ o coeficiente de atrito.

Baseado nas equações (5.25) e (5.26) e na Figura 5.7(a), pode-se perceber quais são as duas razões pelos quais (WRIGGERS, 2006; GIANNAKOPOULOS, 1989; BITTENCOURT; CREUS, 1998; LAURSEN, 2002) utilizaram a formulação elasto-plástica para descrever a relação constitutiva para o atrito na interface de contato. São elas: 1) a descontinuidade da lei de Coulomb na passagem da situação de adesão para deslizamento como mostrado na Figura 5.7(a) que causam dificuldades computacionais e 2) situações experimentais que comprovam esses efeitos.



(a) Lei de Coulomb. (b) Lei modificada de Coulomb.

Figura 5.7: Leis de atrito de Coulomb (MIJAR; ARORA, 2000b)

Logo, pode-se dividir o gap tangencial em duas parcelas. A primeira com característica reversível (comportamento elástico) denominado de g_t^e que representa adesão. A segunda parte com característica irrevesível (comportamento plástico) chamado de g_t^s. Dessa maneira pode-se escrever a expressão que representa o gap tangencial como,

$$g_t = g_t^e - g_t^s. \tag{5.27}$$

Para $f < 0$, a força de contato tangencial é modelada por uma relação linear, isotrópica e elástica, isto é,

$$t_t = k_t g_t^e, \tag{5.28}$$

sendo k_t uma constante elástica definida por um parâmetro de penalidade tangencial usada na solução do problema de contato usando o *Método das Penalidades*.

Quando $f = 0$, existe um deslizamento que deve ser tratado como uma deformação irreversível, já que o trabalho de deformação neste caso é dependente do caminho. Assim, é necessário escrever a equação (5.27) como uma taxa de deformação, ou seja,

$$\dot{g}_t = \dot{g}_t^e - \dot{g}_t^s. \tag{5.29}$$

A taxa de deformação da parcela g_t^s é determinada usando a relação não-associativa da plasticidade (BITTENCOURT; CREUS, 1998)

$$\dot{g}_t^s = \dot{\lambda}\frac{\partial f^*}{\partial t_t}, \tag{5.30}$$

sendo $\dot{\lambda}$ um escalar a ser determinado e

$$f^* = |t_t|. \tag{5.31}$$

Para o deslizamento, a taxa da força tangencial pode ser calculada como

$$\dot{t}_t = k_t(\dot{g}_t - \dot{g}_t^s). \tag{5.32}$$

Para completar a consistência da formulação deve-se considerar que

$$\dot{f} = \frac{\partial f}{\partial t_t}t_t = 0. \tag{5.33}$$

O procedimento de atualização e integração das taxas escritas anteriormente são feitas com o uso do mapeamento de retorno que será descrito a seguir.

5.2.3 Mapeamento de Retorno

O procedimento de mapeamento de retorno aqui descrito foi baseado nos trabalhos de Wriggers (2006) e Bittencourt e Creus (1998) e os passos serão detalhados a seguir.

A integração da equação (5.15) para o caso 2D fornece a seguinte expressão dentro de um

passo de carregamento ΔT_{N+1}

$$\Delta g_{t_{N+1}} = (\bar{\xi}_{N+1} - \bar{\xi_N}). \tag{5.34}$$

O deslizamento total ou gap tangencial $g_{t_{N+1}} = g_{t_N} + \Delta g_{t_{N+1}}$ deve ser decomposto, como explicado na seção anterior, em uma parcela elástica e outra plástica (veja equação (5.27)). Assim, pode-se calcular a função teste para a equação (5.28) e também a função f (equações (5.25) e (5.26)) que fornece as condições de deslizamento ou adesão da seguinte forma

$$t^{trial}_{t_{N+1}} = k_t(g_{t_{N+1}} - g^s_{t_N}), \tag{5.35}$$

$$f^{trial}_{s_{N+1}} = |t^{trial}_{t_{N+1}}| - \mu t_{n_{N+1}}. \tag{5.36}$$

Para $f^{trial}_{s_{N+1}} \leq 0$, tem-se a condição de adesão e $t_t = t^{trial}_{t_{N+1}}$. Caso contrário, tem-se a condição de deslizamento e deve-se então corrigir o valor da parte plástica do deslizamento tangencial g^s_t

$$g^s_{t_{n+1}} = g^s_{t_n} + \lambda \, \text{sign}(t^{trial}_{t_{N+1}}). \tag{5.37}$$

sendo $\lambda = \frac{1}{k_t}(|t^{trial}_{t_{N+1}}| - \mu t_{n_{N+1}})$ para o caso de utilizar a lei de Coulomb.

Logo, para o caso de deslizamento o valor final para a força de contato tangencial é dado por

$$t_{t_{N+1}} = \mu t_{n_{N+1}} \, \text{sign}(t^{trial}_{t_{N+1}}). \tag{5.38}$$

Esta atualização final completa a integração sobre a interface de contato. O processo de return mapping usado neste trabalho pode ser resumido no Algoritmo 5.2.

Algumas considerações devem ser feitas com relação ao método de return mapping como visto em (MIJAR; ARORA, 2005). De fato, o método é muito dependente do passo de carregamento utilizado, isto é, precisa muitas vezes de um passo pequeno para que se obtenha o resultado desejado. Os valores de penalidade tangencial e normal também influenciam no resultado como será discutido na seção 8.3. Valores elevados de penalidades podem causar um mal condicionamento da

Algoritmo 5.2 Procedimento de return mapping para integração local da lei de atrito.

requer: $t_{n_{N+1}}, g_{t_{N+1}}, g^s_{t_N}$
  Calcular: $t^{trial}_{t_{N+1}} = k_t(g_{t_{N+1}} - g^s_{t_N})$
  Calcular: $f^{trial}_{s_{N+1}} = |t^{trial}_{t_{N+1}}| - \mu t_{n_{N+1}}$
  if $f^{trial}_{s_{N+1}} \leq 0$ **then**
    $t_{t_{N+1}} = t^{trial}_{t_{N+1}}$
  else
    Calcular: $\lambda = \frac{1}{k_t}(|t^{trial}_{t_{N+1}}| - \mu t_{n_{N+1}})$
    Atualizar o deslizamento plástico: $g^s_{t_{n+1}} = g^s_{t_n} + \lambda \text{ sign}(t^{trial}_{t_{N+1}})$
    Calcular nova força de fricção: $t_{t_{N+1}} = k_t(g_{t_{N+1}} - g^s_{t_{N+1}})$
  end if

matriz de rigidez resultando em dificuldades na solução numérica. Porém, o return mapping é um procedimento numérico que permite tratar a interface de atrito de uma forma simples e de rápida implementação em códigos de elementos finitos.

A próxima seção aborda o *Método das Penalidades* para descrição do problema de contato na forma variacional para posterior construção da discretização em Elementos Finitos dos elementos de contato 2D.

5.2.4 Método das Penalidades

Neste trabalho foi utilizado o *Método das Penalidades* para regularizar as equações constitutivas na interface do contato. Segundo Mijar e Arora (2000b), Mijar e Arora (2004a) e Laursen e Simo (1993) o Método das Penalidades possui as seguintes vantagens: ser de fácil implementação e ter um número reduzido de incógnitas a serem resolvidas se comparados com outros métodos, tais como *Método do Lagrangiano Aumentado*[1] e *Método dos Multiplicadores de Lagrange*.

[1] O Método do Lagrangiano Aumentado também possui o mesmo número de incógnitas que o Método das Penalidades. Porém é necessário um processo iterativo para incrementar o fator penalizador que é alterado automaticamente a cada iteração de contato do Lagrangiano Aumentado (MIJAR; ARORA, 2004a).

O funcional de energia que representa a parcela do contato 2D é dado por

$$\Pi_C = \frac{1}{2}\int_{\Gamma_C} (t_n g_n + t_t g_t) dA. \tag{5.39}$$

A equação (5.39) deve ser adicionada ao funcional de energia total do sistema para um corpo submetido a uma deformação elástica.

A variação da equação (5.39) fornece

$$C_C = \int_{\Gamma_C} (t_n \delta g_n + t_t \delta g_t) dA. \tag{5.40}$$

Usando o *Método das Penalidades* para definir a força de contato normal e tangencial, têm-se

$$t_n = k_n g_n, \tag{5.41}$$

$$t_t = k_t g_t, \tag{5.42}$$

sendo k_n e k_t os parâmetros de penalidades para o contato normal e tangencial que devem ser valores muito maiores que o módulo de elasticidade usado como propriedade do material. A parcela g_t da equação (5.42) deve ser atualizada via return mapping como discutido na seção 5.2.3 e no Algoritmo 5.2.

A linearização da equação (5.40) para contato com pequenas deformações é dada por

$$C_C^{lin} = \int_{\Gamma_C} (k_n \Delta g_n \delta g_n + \Delta t_t \delta g_t) dA \tag{5.43}$$

O próximo passo consiste em criar os elementos de contato, o que será feito usando elementos isoparamétricos (seção 5.3.1) e discretização nó-segmento (seção 5.3.2).

5.3 Discretização em Elementos Finitos

As discretizações aqui estudas e implementadas no programa hp^2fem são específicas para pequenas deformações, isto é, grandes deslizamentos não são permitidos devido as hipóteses cinemáticas adotadas. A discretização usando elementos isoparamétricos só é possível em malhas casadas, devido à relação de unicidade entre os nós. A formulação é simples e permite que os vetores normais e tangenciais sejam definidos automaticamente pela topologia do elemento. Sendo assim, possível calcular os gaps normal e tangencial de forma direta via interpolação das suas quantidades. A formulação nó-segmento é mais robusta e permite trabalhar com malhas não casadas. Porém, sua implementação não é trivial, pois a incidência do elemento de contato formado entre o nó contator e o elemento alvo não é conhecida a priori (veja seção 5.1.3). Dessa forma, essa seção tratará da construção dos elementos de contato usando as duas discretizações.

5.3.1 Elementos Isoparamétricos

Aqui a discretização da superfície de contato é dada pela interpolação usando elementos isoparamétricos. O elemento de contato gerado não permite grandes deslizamentos entre os nós e necessita de uma malha casada (veja Figura 5.1(a)). Não é necessário o procedimento de busca pelo contato para determinar o valor da projeção do nó contator sobre o elemento alvo, devido à relação nó-a-nó obrigatória. Por essa razão, omite-se a notação da barra sobre as quantidades que serão calculadas a seguir. A função gap definida pela equação (5.8) para pequenas deformações, a sua variação e a sua linearização são discretizadas usando interpolação isoparamétrica, ou seja,

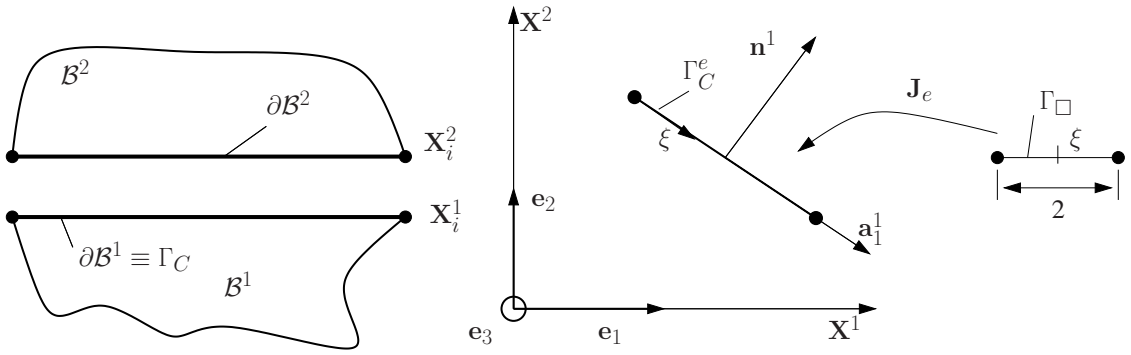
$$g_n = \sum_i N_i(\xi)(\mathbf{u}_i^2 - \mathbf{u}_i^1) \cdot \mathbf{n}^1 + g_{0_i}, \tag{5.44a}$$

$$\delta g_n = \sum_i N_i(\xi)(\boldsymbol{\eta}^2 - \boldsymbol{\eta}^1) \cdot \mathbf{n}^1, \tag{5.44b}$$

$$\Delta g_n = \sum_i N_i(\xi)(\Delta\mathbf{u}_i^2 - \Delta\mathbf{u}_i^1) \cdot \mathbf{n}^1, \tag{5.44c}$$

sendo i o número do nó do elemento isoparamétrico.

Baseado na Figura 5.8, pode-se determinar o vetor normal $\mathbf{n}^1$ e o vetor tangencial $\mathbf{a}_1^1$ com relação às funções de forma para o elemento isoparamétrico utilizado para discretizar a superfície de contato. Por exemplo, para um elemento linear de dois nós as funções de forma são



(a) Dois elementos isoparamétricos em contato.

(b) Mapeamento isoparamétrico.

Figura 5.8: Elemento de contato isoparamétrico bidimensional (WRIGGERS, 2006).

$$N_1 = \frac{1}{2}(\xi - 1), \tag{5.45a}$$

$$N_2 = \frac{1}{2}(\xi + 1), \tag{5.45b}$$

cujas derivadas locais são dadas por

$$\frac{dN_1}{d\xi} = -\frac{1}{2}, \tag{5.46a}$$

$$\frac{dN_2}{d\xi} = \frac{1}{2}. \tag{5.46b}$$

O vetor tangente unitário pode ser escrito como,

$$\mathbf{a}_1^1 = \frac{1}{Je}\begin{bmatrix} dx/d\xi \\ dy/d\xi \end{bmatrix} = \frac{1}{Je}\begin{bmatrix} \sum_i x_i \, dN_i/d\xi \\ \sum_i y_i \, dN_i/d\xi \end{bmatrix}, \tag{5.47}$$

sendo x_i e y_i as componentes do vetor posição $\mathbf{X}_i$ para o nó i nas direções x e y, respecitvamentes.

J_e é o jacobiano dado por

$$J_e = \sqrt{\left(\frac{dx}{d\xi}\right)^2 + \left(\frac{dy}{d\xi}\right)^2}. \tag{5.48}$$

A partir do vetor tangencial dado pela equação (5.47), pode-se construir o vetor normal como

$$\mathbf{n}^1 = \frac{\mathbf{e}_3 \times \mathbf{a}_1^1}{||\mathbf{e}_3 \times \mathbf{a}_1^1||} = \frac{\mathbf{N}^1}{||\mathbf{N}^1||}, \tag{5.49}$$

sendo $\mathbf{e}_3$ a direção perpendicular ao plano.

Assim pode-se escrever as equações (5.44b) e (5.44c) como

$$g_n = \sum_i [N_i(\xi)\hat{\mathbf{N}}(\xi)/||\mathbf{N}^1||]\mathbf{u}_i, \tag{5.50a}$$

$$\delta g_n = \sum_i \boldsymbol{\eta}_i^T [N_i(\xi)\hat{\mathbf{N}}(\xi)/||\mathbf{N}^1||], \tag{5.50b}$$

$$\Delta g_n = \sum_i [N_i(\xi)\hat{\mathbf{N}}(\xi)/||\mathbf{N}^1||]\Delta\mathbf{u}_i, \tag{5.50c}$$

sendo

$$\hat{\mathbf{N}} = \begin{bmatrix} \mathbf{N}^1 \\ -\mathbf{N}^1 \end{bmatrix} \tag{5.51}$$

e

$$\mathbf{u}_i = \begin{bmatrix} \mathbf{u}_i^2 \\ \mathbf{u}_i^1 \end{bmatrix}, \qquad \Delta\mathbf{u}_i = \begin{bmatrix} \Delta\mathbf{u}_i^2 \\ \Delta\mathbf{u}_i^1 \end{bmatrix} \qquad \boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_i^2 \\ \boldsymbol{\eta}_i^1 \end{bmatrix}. \tag{5.52}$$

Para o caso particular onde o corpo alvo é um anteparo rígido, pode-se reescrever a equação (5.51) como

$$\hat{\mathbf{N}} = \begin{bmatrix} \mathbf{N}^1 \\ \mathbf{0} \end{bmatrix}. \tag{5.53}$$

O gap tangencial, a variação do gap tangencial e a sua linearização também são expressos usando funções de interpolação como no caso das parcelas referentes a componente normal, isto é,

$$g_t = \sum_i N_i(\xi)(\mathbf{u}_i^2 - \mathbf{u}_i^1) \cdot \mathbf{a}_1^1, \tag{5.54a}$$

$$\delta g_t = \sum_i N_i(\xi)(\boldsymbol{\eta}_i^2 - \boldsymbol{\eta}_i^1) \cdot \mathbf{a}_1^1, \tag{5.54b}$$

$$\Delta g_t = \sum_i N_i(\xi)(\Delta\mathbf{u}_i^2 - \Delta\mathbf{u}_i^1) \cdot \mathbf{a}_1^1. \tag{5.54c}$$

Em forma compacta,

$$g_t = \sum_i [N_i(\xi)\hat{\mathbf{T}}_1^1(\xi)/||\mathbf{a}_1^1||]\mathbf{u}_i, \tag{5.55a}$$

$$\delta g_t = \sum_i \boldsymbol{\eta}_i^T [N_i(\xi)\hat{\mathbf{T}}_1^1(\xi)/||\mathbf{a}_1^1||], \tag{5.55b}$$

$$\Delta g_t = \sum_i [N_i(\xi)\hat{\mathbf{T}}_1^1(\xi)/||\mathbf{a}_1^1||]\Delta\mathbf{u}_i, \tag{5.55c}$$

sendo,

$$\hat{\mathbf{T}}_1^1 = \begin{bmatrix} \mathbf{a}_1^1 \\ -\mathbf{a}_1^1 \end{bmatrix}. \tag{5.56}$$

Para o caso particular onde o corpo alvo é rígido, tem-se

$$\hat{\mathbf{T}}_1^1 = \begin{bmatrix} \mathbf{a}_1^1 \\ \mathbf{0} \end{bmatrix}. \tag{5.57}$$

Com auxílio da equação (5.40), escreve-se cada um dos termos na forma matricial. Primeiro, expressa-se a força de contato normal $\mathbf{F}_n$ como

$$\int_{\Gamma_C^e} k_n g_n \delta g_n d\Gamma \approx \bigcup_{C=1}^{n_c} \sum_{i=1}^{m} \boldsymbol{\eta}_{Ci}^T \mathbf{F}_i^n, \tag{5.58}$$

com

$$\mathbf{F}_i^n = \int_{-1}^{+1} k_n g_n(\xi) N_i(\xi) \hat{\mathbf{N}}(\xi) d\xi. \tag{5.59}$$

Na equação (5.59), g_n é dado por (5.50a).

A linearização da equação (5.58) resulta na construção da matriz de rigidez de contato com o seguinte formato

$$\mathbf{K}_{ik}^n = \int_{\Gamma_C^e} k_n \Delta g_n \delta g_n d\Gamma \approx \bigcup_{C=1}^{n_c} \sum_{i=1}^{m} \sum_{k=1}^{m} \boldsymbol{\eta}_{Ci}^T \mathbf{C}_{ik} \Delta \mathbf{u}_{Ck}, \tag{5.60}$$

sendo

$$\mathbf{C}_{ik}^n = \int_{-1}^{+1} k_n N_i(\xi) N_k(\xi) \hat{\mathbf{N}} \hat{\mathbf{N}}^T \frac{1}{||\mathbf{N}^1||} d\xi \tag{5.61}$$

A parcela referente ao contato tangencial, pode ser aproximada usando o método das penalidades como

$$\int_{\Gamma_C^e} t_t \delta g_t d\Gamma \approx \bigcup_{C=1}^{n_c} \sum_{i=1}^{m} \boldsymbol{\eta}_{Ci}^T \mathbf{F}_i^t, \tag{5.62}$$

A partir daí, $\mathbf{F}_i^t$ deve ser escrita para cada uma das condições de atrito. Para o caso de adesão, tem-se a seguinte forma discretizada

$$\mathbf{F}_i^{st} = \int_{-1}^{+1} k_t g_t(\xi) N_i(\xi) \hat{\mathbf{T}}_1^1(\xi) d\xi, \tag{5.63}$$

sendo g_t dado pela equação (5.55a).

A matriz de rigidez proveniente da linearização da equação (5.63) para o caso de desliza-

mento é dada por

$$\mathbf{K}_{ik}^{st} = \int_{\Gamma_C^e} k_t \Delta g_t \delta g_t d\Gamma \approx \bigcup_{C=1}^{n_c} \sum_{i=1}^{m} \sum_{k=1}^{m} \boldsymbol{\eta}_{Ci}^T \mathbf{C}_{ik}^{st} \Delta \mathbf{u}_{Ck}, \tag{5.64}$$

na qual

$$\mathbf{C}_{ik}^{st} = \int_{-1}^{+1} k_t N_i(\xi) N_k(\xi) \hat{\mathbf{T}}_1^1 (\hat{\mathbf{T}}_1^1)^T \frac{1}{||\mathbf{N}^1||} d\xi \tag{5.65}$$

Já para o caso de deslizamento, deve-se aplicar os procedimentos de return mapping dado na seção 5.2.3 e realizar os procedimentos de linearizações das quantidades como descritas em (WRIGGERS, 2006). Assim, os resultados finais para o vetor força de contato tangencial e matriz de rigidez tangencial devido ao efeito de deslizamento são, respectivamente,

$$\mathbf{F}_i^{slip} = \int_{-1}^{+1} t_{t_{N+1}} N_i(\xi) \hat{\mathbf{T}}_1^1(\xi) \text{sign}(t_{t_{N+1}}^{trial}) d\xi, \tag{5.66}$$

$$\mathbf{K}_{ik}^{slip} = \int_{\Gamma_C^e} t_t \delta g_t d\Gamma \approx \bigcup_{C=1}^{n_c} \sum_{i=1}^{m} \sum_{k=1}^{m} \boldsymbol{\eta}_{Ci}^T \mathbf{C}_{ik}^{slip} \Delta \mathbf{u}_{Ck}, \tag{5.67}$$

na qual

$$\mathbf{C}_{ik}^{slip} = \int_{-1}^{+1} t_{t_{N+1}} \text{sign}(t_{t_{N+1}}^{trial}) N_i(\xi) N_k(\xi) \hat{\mathbf{T}}_1^1 (\hat{\mathbf{T}}_1^1)^T \frac{1}{||\mathbf{N}^1||} d\xi \tag{5.68}$$

sendo $t_{t_{N+1}}$ o valor atualizado da força de contato tangencial pelo método do return mapping. Nota-se que a matriz de rigidez do contato tangencial para o caso de deslizamento não é simétrica. A integração numérica é feita usando dois pontos de Gauss para o caso analisado. Porém, devido aos valores dos parâmetros de penalidade ou o efeito do coeficiente de atrito pode ocorrer o fenômeno denominado de travamento, onde a solução oscila entre dois valores de convergência. Para evitar o travamento, pode-se utilizar a subintegração, ou seja, usar 1 ponto de Gauss para realizar a integração numérica. Esse procedimento é recomendado por Belytschko, Liu e Moran (2000) como utilizado em elementos quadrado de 4 nós sujeitos a grande deformação. O mesmo fenômeno

pode ocorrer na discretização nó-segmento que será abordada a seguir. O código para discretização usando a formulação isoparamétrica foi adicionada ao programa hp^2fem.

5.3.2 Nó-Segmento

A discretização de elementos finitos usando nó-segmento pode ser encontrada nos trabalhos de Bathe e Chaughary (1985), Ju, Stonet e Rowlands (1995), Ju e Rowlands (1999), Bittencourt e Creus (1998), a qual é muito utilizada em códigos comerciais de elementos finitos tais como (ANSYS INC., 2008; ALTAIR ENGINEERING, INC, 2008). Para essa discretização é necessário realizar a busca pelo contato e a cinemática como descritos nas seções 5.1.3 e 5.1, respectivamente.

Detalhes sobre a discretização podem ser obtidas em Wriggers (2006). O interesse aqui está em construir as matrizes de rigidez e os vetores de força de contato para serem adicionadas ao código hp^2fem. Consideram-se dois corpos em contato. O corpo 1, ou alvo, é discretizado por uma malha $\mathbf{\Omega}_1^h$. Sua superfície é discretizada por um elemento alvo com dois nós, como mostrado na Figura 5.9, em contato com um nó contator do corpo 2 discretizado por uma malha $\mathbf{\Omega}_2^h$. As

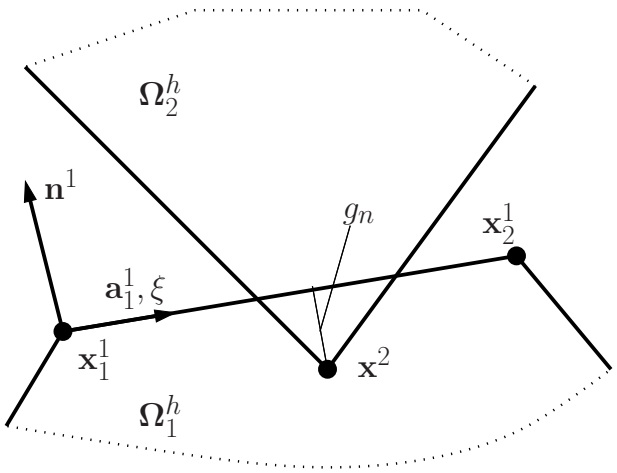


Figura 5.9: Discretização nó-segmento.

funções de interpolação para o elemento alvo são as mesmas fornecidas por (5.45). O vetor normal e tangencial são calculados pelas equações (5.49) e (5.47). A única diferença é que essas quantidades são calculadas usando o valor $\bar{\xi}$ encontrado via solução do problema de mínimo dado na expressão

(5.5).

Logo, pode-se escrever o gap normal como

$$g_n = (\mathbf{x}^2 - \bar{\mathbf{x}}_1^1) \cdot \bar{\mathbf{n}}^1 = [\mathbf{x}^2 - (N_1(\bar{\xi})\mathbf{x}_1^1 + N_2(\bar{\xi})\mathbf{x}_2^1)] \cdot \bar{\mathbf{n}}^1. \tag{5.69}$$

Já o gap tangencial para a condição de adesão é dado pela diferença entre as coordenadas convectivas entre dois passos de carregamentos durante o procedimento de mapeamento de retorno, ou seja,

$$g_t^{st} = \int_{\xi_0}^{\bar{\xi}} |J_e| d\bar{\xi} = (\bar{\xi} - \bar{\xi}_0) J_e, \tag{5.70}$$

sendo J_e o jacobiano do elemento alvo que no caso do elemento segmento representa o seu comprimento; $\bar{\xi}$ é a coordenada convectiva do passo de carregamento T; e $\bar{\xi}_0$ é a coordenada convectiva para T_0. Assim, todas as outras quantidades cinemáticas para o problema de contato são reescritas da seguinte forma

$$\delta g_n = [\boldsymbol{\eta}^2 - N_1(\bar{\xi})\boldsymbol{\eta}_1^1 - N_2(\bar{\xi})\boldsymbol{\eta}_2^1] \cdot \bar{\mathbf{n}}^1, \tag{5.71a}$$

$$\Delta g_n = [\Delta\mathbf{u}^2 - N_1(\bar{\xi})\Delta\mathbf{u}_1^1 - N_2(\bar{\xi})\Delta\mathbf{u}_2^1] \cdot \bar{\mathbf{n}}^1, \tag{5.71b}$$

$$\delta g_t^{st} = [\boldsymbol{\eta}^2 - N_1(\bar{\xi})\boldsymbol{\eta}_1^1 - N_2(\bar{\xi})\boldsymbol{\eta}_2^1] \cdot \bar{\mathbf{a}}_1^1, \tag{5.71c}$$

$$\Delta g_t^{st} = [\Delta\mathbf{u}^2 - N_1(\bar{\xi})\Delta\mathbf{u}_1^1 - N_2(\bar{\xi})\Delta\mathbf{u}_2^1] \cdot \bar{\mathbf{a}}_1^1, \tag{5.71d}$$

$$\delta g_t^{slip} = [\boldsymbol{\eta}^2 - N_1(\bar{\xi})\boldsymbol{\eta}_1^1 - N_2(\bar{\xi})\boldsymbol{\eta}_2^1] \cdot \bar{\mathbf{a}}_1^1, \tag{5.71e}$$

Para a condição de deslizamento, a linearização deve ser feita sobre a força de contato tangencial que é atualizada pelo método de return mapping. Assim, tem-se,

$$\Delta t_{t_{N+1}} = \mu k_n \Delta g_{n_{N+1}} \text{sign}(t_{t_{N+1}}^{trial}) \begin{bmatrix} \bar{\mathbf{n}}^1 \\ -N_1(\bar{\xi})\bar{\mathbf{n}}^1 \\ -N_2(\bar{\xi})\bar{\mathbf{n}}^1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}^2 \\ \Delta\mathbf{u}_1^1 \\ \Delta\mathbf{u}_2^1 \end{bmatrix}. \tag{5.72}$$

Para facilitar a representação matricial das quantidades referentes à força e rigidez de contato,

definem-se os seguintes vetores auxiliares:

$$\boldsymbol{\eta}_C = \begin{bmatrix} \boldsymbol{\eta}^2 & \boldsymbol{\eta}_1^1 & \boldsymbol{\eta}_2^1 \end{bmatrix}, \tag{5.73}$$

$$\mathbf{N}_C = \begin{Bmatrix} \bar{\mathbf{n}}^1 \\ -N_1(\bar{\xi})\bar{\mathbf{n}}^1 \\ -N_2(\bar{\xi})\bar{\mathbf{n}}^1 \end{Bmatrix}, \tag{5.74}$$

e

$$\mathbf{T}_C = \begin{Bmatrix} \bar{\mathbf{a}}_1^1 \\ -N_1(\bar{\xi})\bar{\mathbf{a}}_1^1 \\ -N_2(\bar{\xi})\bar{\mathbf{a}}_1^1 \end{Bmatrix}. \tag{5.75}$$

A força de contato normal e a rigidez podem ser escritas, respectivamente, como

$$\mathbf{F}_C^n = k_n g_n \mathbf{N}_C, \tag{5.76}$$

e

$$\mathbf{K}_C^n = k_n \mathbf{N}_C \mathbf{N}_C^T, \tag{5.77}$$

Para a condição de adesão, tem-se

$$\mathbf{F}_C^{st} = k_t g_t^{st} \mathbf{T}_C, \tag{5.78}$$

e

$$\mathbf{K}_C^{st} = k_t \mathbf{T}_C \mathbf{T}_C^T. \tag{5.79}$$

Já na condição de deslizamento, as parcelas são das por

$$\mathbf{F}_C^{slip} = \mu k_n g_n \mathbf{T}_C, \tag{5.80}$$

e

$$\mathbf{K}_C^{slip} = \mu k_n \text{sign}(t_{t_{N+1}}^{trial}) \mathbf{T}_C \mathbf{N}_C^T. \tag{5.81}$$

A seguir será apresentado o algoritmo utilizado para resolver o problema de contato sem atrito e com atrito.

5.4 Solução de Problema de Contato

Sendo o problema de contato um problema não-linear, é necessário aplicar um método iterativo, como o método de Newton-Raphson, para resolver a seguinte equação para ${}^{T+\Delta T}\Delta\mathbf{u}^{(i+1)}$

$$({}^{T+\Delta T}\mathbf{K}^{(i)} + {}^{T+\Delta T}\mathbf{K}_C^{(i)})\, {}^{T+\Delta T}\Delta\mathbf{u}^{(i+1)} = {}^{T+\Delta T}\mathbf{P}^{(i)} - {}^{T+\Delta T}\mathbf{F}_{int}^{(i)} + {}^{T+\Delta T}\mathbf{F}_C^{(i)}, \tag{5.82}$$

sendo

$(T + \Delta T)$ = passo de carregamento;

(i) = contador de iterações de Newton-Raphson;

$\mathbf{K}$ = matriz de rigidez dos elementos Ω_e^h;

$\mathbf{K}_C$ = matriz de rigidez dos elementos de contato;

$\Delta\mathbf{u}$ = vetor de incrementos dos deslocamentos;

$\mathbf{P}$ = vetor de carregamentos externos;

$\mathbf{F}_{int}$ = vetor de força interna dos elementos Ω_e^h proveniente do processo de linearização;

$\mathbf{F}_C$ = vetor de forças de contato.

O Algoritmo 5.3 mostra os passos utilizados para solucionar o problema de contato usando o *Método das Penalidades* que será usado na próxima seção para resolver os estudos de casos.

Algoritmo 5.3 Problema de Contato com atrito usando o *Método das Penalidades*.

Inicializar: $\mathbf{u} = \mathbf{0}$
Decidir valor do passo de carregamento $\Delta\mathbf{P} = \mathbf{P}/$(número de passos de carregamento)
for n = 1 : número de passos de carregamento **do**
  ${}^{T+\Delta T}\mathbf{P} = {}^{T}\mathbf{P} + \Delta\mathbf{P}$
  Atualizar/inicializar grandezas cinemáticas de contato (g_t^e,g_t^s,t_n)
  Inicializar contator de Newton-Raphson $i = 0$
  Inicializar critério de parada $conv = 1$
  while $conv >$ precisão ou $i <$ número máximo de iterações **do**
    $i = i + 1$
    if $i = 1$ **then**
      Considerar todos os nós em estado de adesão
    else
      Atualizar as forças de atrito usando o Algoritmo 5.2
    end if
    Resolva a equação de Newton-Raphson (5.82)
    Atualizar vetor de deslocamentos $\mathbf{u}^{(i)} = \mathbf{u}^{(i-1)} + \Delta\mathbf{u}^{i}$
    Atualizar vetores de $\mathbf{F}_{int}^{i}$ e $\mathbf{F}_{C}^{(i)}$.
    Checar a convergência $R = ||\mathbf{F}_{int}^{i} + \mathbf{F}_{C}^{(i)}||/(1 + ||{}^{T+\Delta T}\mathbf{P}||) <$ tolerância
  end while
end for

6 Otimização de Forma em Problemas Estruturais

Um ambiente de otimização de forma requer uma conectividade entre diversas ferramentas tais como análise de resposta, análise de sensibilidade, geração de malhas, controle de geometria (parametrização e atualização) e algoritmo de otimização. Essa conectividade se dá através do campo de velocidade de projeto proveniente da análise de sensibilidade à mudança de forma, conforme descrito e discutido em Silva (2003), Choi e Kim (2005a, 2005b). As expressões de sensibilidade não manipulam as variáveis de forma diretamente, apenas os campos de velocidades induzidos por elas. Assim, a descrição geométrica deve fornecer uma regra geral e reutilizável para obtenção desses campos.

Na otimização de forma, deve-se garantir que o contorno da malha sempre coincida com o contorno da geometria Silva (2003).

O algoritmo de otimização estudado é extremamente dependente da hipótese de continuidade e suavidade dos funcionais do problema. Em cada iteração, os valores das variáveis de projeto são alteradas. Deve-se, então, sempre checar a consistência da malha e se necessário gerar uma nova malha a partir da geometria atualizada. Essa é a razão para que exista uma integração consistente entre o algoritmo de otimização e um software de controle de geometria que possibilite a geração automática de malha.

Assim, o objetivo deste capítulo é discutir os elementos essenciais do processo de otimização de forma para em seguida no capítulo seguinte seja discutido a integração do programa de otimização escrito com um programa de parametrização de geometria e geração automática de malha.

6.1 O Problema de Otimização

6.1.1 Modelo Matemático

A formulação de um problema de otimização consiste na passagem de um modelo físico para um modelo matemático muito bem definido, que possa garantir a precisão dos resultados. Esse modelo matemático é constituído de três itens;

Função Objetivo : é o parâmetro que será otimizado, em outras palavras, é o que deseja ser minimizado. Num problema de mecânica do contínuo, a função objetivo é dado por um funcional de performance estrutural. A função objetivo é dependente das variáveis de projeto. Nesse trabalho a função objetivo será indicada pela letra f;

Variáveis de Projeto : é um conjunto de variáveis que descrevem o sistema. As variáveis de projeto são mutáveis durante o processo de otimização, isto é, elas podem mudar de valores até que a função objetivo atinja o valor ótimo, satisfazendo os critérios de restrições. Nesse trabalho as variáveis de projeto será indicada pela letra $\mathbf{P}$;

Restrições : podem ser representadas por funções (lineares e não-lineares) ou por valores escalares. São as restrições que indicam os limites e os critérios aceitáveis do projeto. As restrições podem ser de igualdade $\mathbf{h} = \mathbf{0}$ e/ou de desigualdade $\mathbf{g} \leq \mathbf{0}$. Em problemas de mecânica estrutural as restrições são dadas por funcionais de performance estruturais como visto em Silva (1997), Choi e Kim (2005a), Christensen e Klarbring (2009).

Assim, de acordo com Arora (2004) podemos representar um problema de otimização na forma matemática como,

$$\begin{aligned} &\min\ f(\mathbf{P}) \\ &\text{sujeita à} \\ &\qquad g_i(\mathbf{P}) \leq 0 \quad i = 1, 2, ..., n \\ &\qquad h_j(\mathbf{P}) = 0 \quad j = 1, 2, ..., p \end{aligned} \tag{6.1}$$

sendo $\mathbf{P} \in \Re^n$, $\mathbf{f} : \Re^n \rightarrow \Re$, $g_i : \Re^n \rightarrow \Re$ e $h_j : \Re^n \rightarrow \Re \; \forall i, j \in \aleph$. Assumindo que o problema tenha solução $\mathbf{P}^* \in \Re$, as restrições nesse ponto podem ser tanto satisfeitas na igualdade (restrições ativas) quanto na desigualdade (restrições inativas). Em um problema de otimização, deve-se ter sempre $p < n$ para se ter um problema determinado. No caso de $p = n$, a única solução possível é aquela obtida do sistema de n equações por n incógnitas. Para o caso onde $p > n$, o problema é sobredeterminado, tendo-se equações redundantes ou inexistir solução.

6.2 Parametrização do Contorno

A parametrização do contorno da geometria é um processo fundamental para a realização da otimização de forma. É através da parametrização que o modelo físico é convertido para um modelo matemático. Dessa forma, o objeto, agora, passa a ser descrito por curvas, superfícies ou volumes, que são entidades definidas por funções. A mais importante característica da otimização de forma é a conexão direta entre a geometria parametrizada e o modelo de elementos finitos, mais precisamente, com o campo de velocidades que será abordado neste capítulo.

A parametrização do contorno de estruturas pode ser realizada usando métodos tais como:

- coordenadas nodais do contorno;
- segmentos de retas e arcos de circunferências;
- funções polinomiais;
- funções B-spline.

No primeiro método, as coordenadas nodais dos elementos são usadas como variáveis de projeto durante a otimização de forma e alterada diretamente no processo de otimização. Apesar de ser o método mais simples de parametrização, de acordo com Choi e Kim (2005b), alguns inconvenientes podem aparecer, tais como,

- como as coordenadas dos nós do modelo são usadas como variáveis de projeto, para modelos elaborados, o problema de otimização tende a se tornar enorme. Isto resulta no crescimento

significativo dos custos computacionais, podendo ocasionar dificuldades para a solução do problema de otimização;

- a primeira derivada não é contínua entre os nós do contorno, o que resulta em projetos impraticáveis e inaceitáveis;
- não se pode garantir a precisão numérica do projeto, devido a possível distorção da malha de elementos finitos durante o processo de otimização. Uma alternativa para esse problema é usar as coordenadas dos pontos geométricos como variável de projeto e usar um mapeamento isoparamétrico para gerar a malha.

As funções polinomiais produzem curvas que passam por todos os pontos de um modelo. Quanto maior a ordem do polinômio, maior será a precisão da curva e melhor a parametrização. Porém, a oscilação do contorno também será maior (CHOI; KIM, 2005b). Para maiores informações, Rogers e Adams (1990) é uma boa referência sobre o assunto.

Para eliminar o problema de oscilação do contorno, emprega-se *B-splines* que são funções compostas de segmentos de polinômios de baixa ordem. Estes segmentos são combinados para maximizar a suavização do contorno. A base B-spline permite que a ordem seja alterada sem aumentar o número de vértices do polígono.

A generalização das B-splines resulta na base NURBS (*Non Uniform Rational B-Spline*). A base NURBS é utilizada intensivamente na indústria mecânica, tornando-se o padrão para representação de modelos e transferência de arquivos de geometria entre plataformas CAD e CAE. Como visto em Piegl e Tiller (1997), o sucesso da base NURBS deve-se aos seguintes fatores:

1. provê uma base matemática concisa para representação de qualquer entidade geométrica;
2. seu algoritmo é numericamente estável e de fácil interpretação;
3. a geometria é invariante para transformações lineares, tais como translação, rotação, projeções paralela e ortogonal.

A seguir, introduz-se os conceitos de geração de curvas e superfícies NURBS usadas na parametrização do contorno de geometrias submetidas ao processo de otimização de forma.

Curva NURBS

Uma curva NURBS de grau p é definida por Piegl e Tiller (1997),

$$\mathbf{X}(r) = \frac{\sum_{i=0}^{n} N_{i,p}(r)\beta_i \, \mathbf{X}^i}{\sum_{i=0}^{n} N_{i,p}(r)\beta_i} \quad a \leq r \leq b, \tag{6.2}$$

sendo $\mathbf{X}^i$ as coordenadas dos pontos de controle, β_i são os pesos de cada ponto de controle e $N_{i,p}(r)$ é a função B-spline de grau p (ordem $p+1$) definida da forma mais geral em um vetor de nodos (knots) não-periódico e não-uniforme dado por

$$\mathbf{r} = \{\underbrace{a, ..., a}_{p+1}, r_{p+1}, ..., r_{m-p-1}, \underbrace{b, ..., b}_{p+1}\} \tag{6.3}$$

e a e b são os nodos para os limites inferior e superior respectivamente e m é o índice do nodo. A relação entre o número de nodos ($m+1$), o grau p da função $N_{i,p}(r)$ e o número de pontos de controle ($n+1$) é dado por $m = n + p + 1$ (PIEGL; TILLER, 1997). Considerando $a = 0$, $b = 1$ e $\beta_i > 0$ para todo i, tem-se a base B-spline racional

$$R_{i,p}(r) = \frac{N_{i,p}(r)\beta_i}{\sum_{j=0}^{n} N_{j,p}(r)\beta_j}. \tag{6.4}$$

Reescrevendo a equação (6.2) usando a equação (6.4), chega-se a seguinte expressão para a curva NURBS

$$\mathbf{X}(r) = \sum_{i=0}^{n} R_{i,p}(r)\mathbf{X}^i. \tag{6.5}$$

É importante saber que os pesos e as coordenadas físicas de um ponto pertencem a um espaço

de coordenadas de quatro dimensões (ROGERS; ADAMS, 1990).

A base não-racional B-spline, $N_{i,p}(r)$, pode ser obtida pela fórmula recursiva de Cox-deBoor-Mansfield, apresentada em (ROGERS; ADAMS, 1990; PIEGL; TILLER, 1997), como

$$N_{i,0}(r) = \begin{cases} 1 & \text{se } r_i \leq r < r_{i+1}, \\ 0 & \text{outros casos,} \end{cases}$$
$$N_{i,p}(r) = \frac{r - r_i}{r_{i+p} - r_i} N_{i,p-1}(r) + \frac{r_{i+p+1} - r}{r_{i+p+1} - r_{i+1}} N_{i+1,p-1}(r), \quad p = 2, ..., p+1, \tag{6.6}$$

sendo r_i os elementos do vetor de nodos com dimensão $m + 1$.

O vetor de nodos deve ser uma série monotonicamente crescente de números reais (PIEGL; TILLER, 1997). Existem três tipos de vetores de nodos aplicáveis em uma base B-spline:

uniforme (periódico) : são vetores que possuem valores igualmente espaçados e distribuídos entre um valor mínimo e máximo dados. Na prática, o valor mínimo é zero. Se forem normalizados estarão no intervalo $[0, 1]$ e terão incrementos de um valor decimal fixado. São usados para gerar curvas fechadas (SILVA, 2003). Como exemplo de vetores de nodos uniformes, tem-se

$[0\ \ 1\ \ 2\ \ 3\ \ 4]$ com incremento de uma unidade,

$[-0,2\ \ -0,1\ \ 0\ \ 0,1\ \ 0,2]$ com incremento de 0,1 unidades,

$[0\ \ 0,25\ \ 0,50\ \ 0,75\ \ 1,0]$ normalizado com incremento de 0,25 unidades;

uniforme aberto : são vetores que tem multiplicidade dos valores dos nodos no final da série igual ao valor da ordem $p + 1$ da função de base B-Spline. Vetores uniformes abertos são gerados pela seguinte regra (ROGERS; ADAMS, 1990)

$$\begin{cases} r_i = 0, & 1 \leq i \leq p+1, \\ r_i = i - p + 1, & p+2 \leq i \leq n+1 \\ r_i = n - p + 1, & q+2 \leq i \leq n+p+2. \end{cases} \tag{6.7}$$

São exemplos de vetores de nodos uniforme aberto não normalizados para $n + 1 = 5$ pontos

de controle,

$$p = 1: \ [0 \ 0 \ 1 \ 2 \ 3 \ 4 \ 4],$$

$$p = 2: \ [0 \ 0 \ 0 \ 1 \ 2 \ 3 \ 3 \ 3],$$

$$p = 3: \ [0 \ 0 \ 0 \ 0 \ 1 \ 2 \ 2 \ 2 \ 2].$$

Como vetores normalizados,

$$p = 1: \ [0 \ 0 \ 0,25 \ 0,50 \ 0,75 \ 1 \ 1],$$

$$p = 2: \ [0 \ 0 \ 0 \ 1/3 \ 2/3 \ 1 \ 1 \ 1],$$

$$p = 3: \ [0 \ 0 \ 0 \ 0 \ 1/2 \ 1 \ 1 \ 1 \ 1];$$

não-uniformes (não-periódicos) : vetores não-uniformes podem ter valores com espaçamento desigual e/ou valores internos múltiplos (ROGERS; ADAMS, 1990). O número de nodos é $n + p + 1$. Têm-se os seguintes vetores de nodos como exemplos para $n + 1 = 4$ pontos de controle

$$p = 1: \ [0 \ 0,28 \ 0,5 \ 0,72 \ 1],$$

$$p = 2: \ [0 \ 1 \ 2 \ 2 \ 3 \ 4],$$

$$p = 4: \ [0 \ 0 \ 0 \ 1 \ 1 \ 2 \ 2 \ 2].$$

Geralmente, os vetores de nodos são normalizados para o intervalo $[0, 1]$ (SILVA, 2003). Como dito anteriormente, uma curva NURBS é uma curva B-spline gerada sobre um vetor de nodos não-uniforme e não-periódico. Dessa maneira, pode-se afirmar que uma curva NURBS é a forma mais completa e geral de uma de B-spline.

Superfície NURBS

De acordo com Piegl e Tiller (1997), uma superfície NURBS de grau p na direção de r e grau h na direção de s, construída sobre um vetor de nodos não-periódico e funções B-spline não-

racionais para $n+1$ pontos de controle na direção de r e $t+1$ pontos de controle na direção de s, é definida pela seguinte expressão

$$\mathbf{X}(r,s) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{t} N_{i,p}(r)M_{j,h}(s)\beta_{i,j}\mathbf{X}^{ij}}{\sum_{i=0}^{n}\sum_{j=0}^{t} N_{i,p}(r)M_{j,h}(s)\beta_{i,j}} \;\; 0 \leq r,s \leq 1. \tag{6.8}$$

$\mathbf{X}^{ij}$ são os pontos de controle (vértices) que definem uma rede tridimensional, $\beta_{i,j}$ são os pesos dos pontos de controle, e $N_{i,p}(r)$ e $M_{i,h}(s)$ as funções B-spline não-racionais dada pela fórmula recursiva (6.6) em cada direção paramétrica.

Usando funções racionais contínuas por partes, pode-se escrever

$$R_{i,j}(r,s) = \frac{N_{i,p}(r)M_{j,h}(s)w_{i,j}}{\sum_{k=0}^{n}\sum_{l=0}^{t} N_{k,p}(r)M_{l,h}(s)\beta_{k,l}}. \tag{6.9}$$

Conseqüentemente, a equação (6.8) pode ser reescrita como,

$$\mathbf{X}(r,v) = \sum_{i=0}^{n}\sum_{j=0}^{t} R_{i,j}(r,v)\mathbf{X}^{ij}. \tag{6.10}$$

Percebe-se que as fronteiras de uma superfície NURBS são dadas por curvas NURBS. Isto faz com que, em um processo de otimização de forma, ao escolher-se como variáveis de projetos determinados pontos de controle de uma curva NURBS, indiretamente, estão sendo selecionados os pontos de controle equivalentes na superfície NURBS.

6.2.1 Campos de Velocidades

Definindo uma variável de projeto com uma coordenada l ($l = 1,2,3$) de um determinado ponto de controle escolhido de uma entidade NURBS, isto é, $d_i = X_l^{ij}$, a sensibilidade da curva ou superfície NURBS a uma variação de d_i é

$$\frac{\partial \mathbf{X}}{\partial d_i} = \frac{\partial \mathbf{X}}{\partial X_l^{ij}}. \tag{6.11}$$

Segundo Silva (2003) para o exemplo de uma superfície NURBS, tem-se

$$\frac{\partial \mathbf{X}}{\partial X_1^{ij}} = \left\{ \begin{array}{c} S_{i,j}(r,s) \\ 0 \\ 0 \end{array} \right\}, \quad \frac{\partial \mathbf{X}}{\partial X_2^{ij}} = \left\{ \begin{array}{c} 0 \\ S_{i,j}(r,s) \\ 0 \end{array} \right\}, \quad \frac{\partial \mathbf{X}}{\partial X_3^{ij}} = \left\{ \begin{array}{c} 0 \\ 0 \\ S_{i,j}(r,s) \end{array} \right\}. \tag{6.12}$$

com $X^{ij} = (X_1^{ij}, X_2^{ij}, X_3^{ij})$.

Se a variável é o peso ij, i.e., $d_i = \beta_{ij}$, então,

$$\frac{\partial \mathbf{X}}{\partial d_i} = \frac{\partial \mathbf{X}}{\partial \beta_{ij}} = \frac{1}{\beta_{ij}} \left[\mathbf{X}^{ij} - \mathbf{X}\left(r,s\right) \right] S_{ij}\left(r,s\right). \tag{6.13}$$

Logo, o campo de velocidade no contorno do modelo discreto deve ser obtido avaliando-se (6.11) e (6.13) nas coordenadas paramétricas (r,s) de cada nó da superfície.

Tendo como base (6.10) e (6.11), nota-se que parametrizações baseadas em pontos de controle dão origem a campos de velocidades que, quando utilizados na atualização de coordenadas nodais, garantem que nós sobre o contorno $\partial\mathcal{B}$ permanecerão sempre sobre esse contorno, uma vez que nós e geometria serão atualizados exatamente com a mesma expressão, independentemente do tamanho da perturbação.

Considere, por exemplo, a parametrização de uma coordenada do ponto de controle $\mathbf{X}^{mn}$. Logo,

$$\mathbf{X}\left(r,s\right) + \delta d \mathbf{V}\left(r,s\right) = \left\{ \begin{array}{ll} \sum_{i=0}^{n} \sum_{j=0}^{t} \breve{\mathbf{X}}^{ij} S_{ij}\left(r,s\right), & \\ & \\ \breve{\mathbf{X}}^{ij} = \mathbf{X}^{ij}, & i \neq m,\ j \neq n, \\ \breve{\mathbf{X}}^{ij} = \mathbf{X}^{ij} + \delta d, & i = m,\ j = n. \end{array} \right.$$

O lado esquerdo da expressão anterior representa o algoritmo de atualização das coordenadas dos nós sobre o contorno, ou seja, a cada posição nodal soma-se a respectiva velocidade multiplicada pela perturbação. O lado direito da mesma expressão indica a regra de atualização da representação geométrica. A grandeza δd é proveniente do algoritmo de minimização.

No caso da variável estar associada a um ponto de controle de uma curva NURBS que é a fronteira entre duas superfícies NURBS, qualquer modificação se propaga para ambas as superfícies. Tais modificações são descritas por campos de velocidades parciais $\mathbf{V}_1$ and $\mathbf{V}_2$ em cada uma das superfícies. O campo de velocidades resultante é a união $\mathbf{V} = \mathbf{V}_1 \cup \mathbf{V}_2$. Da mesma forma, se a variável está associada a um ponto que é um vértice de n_{surf} superfícies, o campo de velocidades resultante é a união de campos parciais, ou seja, $\mathbf{V} = \mathbf{V}_1 \cup \mathbf{V}_2 \cup \ldots \cup \mathbf{V}_{n_{surf}}$.

Pode se obter de forma análogas as expressões para curvas NURBS.

Recuperação de Coordenadas Paramétricas

Como desenvolvido muito bem em Silva (2003), as expressões (6.11) e (6.13) mostram que o campo de velocidades de variáveis relacionadas a características de curvas e superfícies NURBS exigem o cálculo das funções de base nos nós localizados sobre o contorno. Para isso, torna-se necessário conhecer as coordenadas paramétricas (r, s) correspondentes a cada nó sobre o contorno $\partial\mathcal{B}$. Tais informações não são geralmente fornecidas pelos geradores de malha, sendo necessário dispor de recursos para determiná-las. Além disso, mesmo que os parâmetros sejam conhecidos, a aplicação da transformação de domínio altera seus valores, os quais precisam ser atualizados antes da avaliação dos campos de velocidades.

Para geometrias 2D, uma abordagem inicial seria a aproximação linear, baseada na aproximação dos parâmetros r a partir das distâncias entre os nós sobre o contorno. Considere a distância entre dois nós consecutivos do contorno

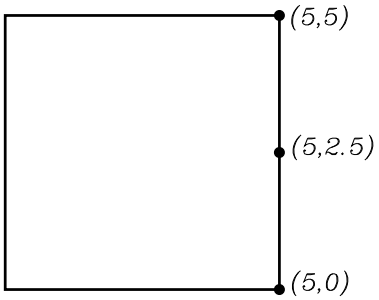
$$\delta_i = \|\mathbf{X}_i - \mathbf{X}_{i-1}\|,$$

sendo $\mathbf{X}_i$ as coordenadas do nó i. A aproximação $\tilde{r}_i$ do parâmetro r_i correspondente seria então,

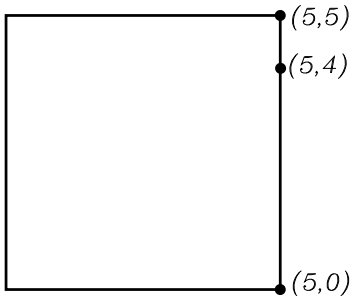
$$\tilde{r}_i = \left(\sum_{k=0}^{i} \delta_k\right) / \left(\sum_{k=0}^{Nc} \delta_k\right), \qquad \tilde{r}_0 = 0,0, \tag{6.14}$$

sendo Nc o número total de nós na região considerada do contorno.

Esta solução, mostra-se bastante imprecisa, ocasionando erros sistemáticos de análise de sen-



(a) Ponto de controle central em posição simétrica.



(b) Ponto de controle central deslocado.

Figura 6.1: Exemplos para teste de cálculo de curvas NURBS (SILVA, 2003).

sibilidade, mesmo em casos simples. Por exemplo, considere o domínio quadrado mostrado na Figura 6.1(a), no qual um dos lados é descrito por uma B-spline quadrática, estando as coordenadas dos 3 pontos de controle indicadas na mesma figura. Aplicando (6.7), o vetor de nodos é

$$\mathbf{r} = \left[\begin{array}{cccccc} 0 & 0 & 0 & 1 & 1 & 1 \end{array} \right].$$

As funções de base são obtidas de(6.6)

$$\begin{array}{lllll} N_{0,0} = 0, & N_{1,0} = 0, & N_{2,0} = 1, & N_{3,0} = 0, & N_{4,0} = 0, \\ N_{0,1} = 0, & N_{1,1} = 1 - r, & N_{2,1} = r, & N_{3,1} = 0, & \\ N_{0,2} = (1-r)^2, & N_{1,2} = 2r\,(1-r), & N_{2,2} = r^2. & & \end{array}$$

Aplicando a última linha em(6.5), tem-se a seguinte expressão para as coordenadas dos pontos sobre o lado do domínio indicado na Figura 6.1(a)

$$\mathbf{x}\,(r) = \left\{ \begin{array}{c} 5,0 \\ 0,0 \end{array} \right\} (1-r)^2 + \left\{ \begin{array}{c} 5,0 \\ 2,5 \end{array} \right\} 2r\,(1-r) + \left\{ \begin{array}{c} 5,0 \\ 5,0 \end{array} \right\} r^2 = \left\{ \begin{array}{c} 5 \\ 5r \end{array} \right\}, \qquad 0 \leq r \leq 1.$$

A derivada da área ψ do quadrado em relação à variável d, coordenada x do segundo ponto de controle, é então,

$$\frac{\partial \psi}{\partial d} = \int_{\mathcal{B}} \int Div\mathbf{V}\; dV = \int_{\partial \mathcal{B}} \mathbf{V} \cdot \mathbf{n}\; dL = \int_0^1 2r\,(1-r)\,(5)\; dr = \frac{10}{6} = 1,666\ldots,$$

pois o campo de velocidades no contorno é a própria segunda função de base da curva. Utilizando

(6.14), obtém-se o mesmo resultado utilizando-se elementos triangulares quadráticos (que descrevem exatamente o campo de velocidades no contorno), ou seja, $\frac{\partial \psi}{\partial d} = 1,666666667$. Entretanto, se o segundo ponto de controle for alterado, mantendo-se o mesmo contorno final (como mostrado na Figura 6.1(b)), as coordenadas dos pontos desse lado passam a ser fornecidas pela expressão

$$\mathbf{x}(r) = \left\{ \begin{array}{c} 5,0 \\ 0,0 \end{array} \right\} (1-r)^2 + \left\{ \begin{array}{c} 5,0 \\ 4,0 \end{array} \right\} 2r(1-r) + \left\{ \begin{array}{c} 5,0 \\ 5,0 \end{array} \right\} r^2 = \left\{ \begin{array}{c} 5,0 \\ 8r - 3r^2 \end{array} \right\}, \qquad 0 \leq r \leq 1.$$

Observa-se que o campo de velocidades também é fornecido pela segunda função de base da curva, portanto espera-se o mesmo resultado. Logo,

$$\frac{\partial \psi}{\partial d} = \int_0^1 2r(1-r)(8-6r)\ dr = \frac{10}{6} = 1,666\ldots$$

O resultado numérico foi de $\frac{\partial \psi}{\partial d} = 1,666585859$ ao invés do resultado exato e, quanto mais deslocado em relação ao centro estiver este ponto de controle, maior o erro obtido. Além disso, trata-se de um erro não controlável e sistemático que se torna maior com a modificação da posição dos pontos de controle, geometrias mais complexas e outros funcionais de performance. O impacto é ainda mais significativo quando o próprio campo de velocidades é usado para atualizar os nós da malha.

Tem-se então o problema de determinar o valor $\tilde{r}_i$ tal que

$$\left\{ \begin{array}{c} \tilde{x}_i \\ \tilde{y}_i \end{array} \right\} = \sum_{j=0}^{n} \mathbf{X}^j R_{j,\chi}(\tilde{r}_i)$$

sendo $\mathbf{X}^j$ as coordenadas dos pontos de controle e $\left[\begin{array}{cc} \tilde{x}_i & \tilde{y}_i \end{array} \right]^T$ as coordenadas do nó. Assim, deve-se resolver a equação vetorial

$$\mathbf{f}(r_i) = \sum_{j=0}^{n} \mathbf{X}^j R_{j,\chi}(r_i) - \left\{ \begin{array}{c} \tilde{x}_i \\ \tilde{y}_i \end{array} \right\} = \mathbf{0}, \qquad D\mathbf{f}(r_i) = \sum_{j=0}^{n} \mathbf{X}^j \frac{\partial}{\partial r} R_{j,\chi}(r_i).$$

Resolve-se essa equação será através do método de Newton-Raphson.

Toma-se a aproximação de Taylor de primeira ordem para $\mathbf{f}(r)$ em torno de r_i^*,

$$\mathbf{f}(r_i) = \mathbf{f}(r_i^*) + \{D\mathbf{f}(r_i^*)\}(r_i - r_i^*) = \mathbf{0}, \tag{6.15}$$

como o ponto de partida para uma forma algorítmica de solução.

Multiplicando (6.15) por $\{D\mathbf{f}(r_i^*)\}^T$,

$$\{D\mathbf{f}(r_i^*)\}^T \{D\mathbf{f}(r_i^*)\}(r_i - r_i^*) + \{D\mathbf{f}(r_i^*)\}^T \mathbf{f}(r_i^*) = \mathbf{0},$$

converte-se a equação vetorial em escalar. Isolando r_i, ou seja,

$$r_i = r_i^* - \frac{\{D\mathbf{f}(r_i^*)\}^T \mathbf{f}(r_i^*)}{\{D\mathbf{f}(r_i^*)\}^T \{D\mathbf{f}(r_i^*)\}},$$

obtém-se a fórmula recursiva

$$r_i^{k+1} = r_i^k - \frac{\left\{D\mathbf{f}\left(r_i^k\right)\right\}^T \mathbf{f}\left(r_i^k\right)}{\left\{D\mathbf{f}\left(r_i^k\right)\right\}^T \left\{D\mathbf{f}\left(r_i^k\right)\right\}}, \quad k+1 \to k, \quad \text{até que } \left|r_i^{k+1} - r_i^k\right| \leq \varepsilon > 0. \tag{6.16}$$

Se houve convergência para $k+1$, adota-se $\tilde{r}_i = r_i^{k+1}$.

Este procedimento deve ser realizado para cada nó da região do contorno associada a variáveis de projeto. Entretanto, este algoritmo é eficiente, pois não há necessidade de resolver sistemas lineares uma vez que o problema para cada nó é desacoplado dos demais e todas as funções envolvidas são conhecidas (as funções de base e suas derivadas são obtidas de maneira semelhante). Testes numéricos mostraram que o algoritmo converge rapidamente mesmo para aproximação $r_i^k = 0,0$ para todos os nós de uma curva (SILVA, 2003).

Aplicando este procedimento ao exemplo do quadrado, obtém-se $\frac{\partial \psi}{\partial d} = 1,666666667$ para qualquer coordenada y, $0,0 < y < 5,0$, do segundo ponto de controle.

É muito importante a aplicação do método anterior em problemas 3D, pois em superfícies não há ordenação preferencial dos nós que permita mesmo uma aproximação linear para as coordenadas paramétricas. Nesse caso, o algoritmo envolve a solução de um sistema linear 2×2 simétrico em

cada nó da região do contorno associada a variáveis de projeto.

6.3 Campos de Velocidades do Interior do Domínio

Neste trabalho estudou-se duas técnicas de geração de campos de velocidades no interior do domínios discretizados em elementos finitos: *método da camada unitária de contorno* Silva (1997, 2003) e o *método de deslocamentos fictícios de contorno* Choi e Chang (1994). A seguir, as principais características desses dois métodos são apresentadas.

6.3.1 Método da Camada Unitária de Contorno

Segundo Silva (2003), esse método apresenta o campo de velocidade não-nulo somente na primeira camada de elementos adjacentes ao contorno parametrizado.

Assume-se um campo de velocidade nos nós do contorno, e assim interpola-se o campo de velocidades dos nós da camada adjacente através das funções de forma lineares dos campos de velocidades do contorno e do campo de velocidades nulas na fronteira interna da camada. Sobre o contorno, o campo de velocidade deve ter a mesma ordem da interpolação da malha de elementos.

Esse método possui como ponto positivo a eficiência computacional, pois não é necessário nenhum cálculo numérico. Soma-se, ainda, o fato de que as expressões de análise de sensibilidade somente precisam ser avaliadas na camada adjacente ao contorno, o que é muito vantajoso em geometrias $3D$.

Entretanto, existe um custo pela possibilidade de variação do domínio em uma única camada de elementos. É provável que sejam produzidos elementos distorcidos ou mesmo perda da consistência geométrica da malha. Por isso, o método possui maior eficiência quanto maior for o refinamento da malha adjacente ao contorno (SILVA, 2003). Dessa forma, quando o critério de distorção é atingido, deve-se regerar automaticamente a malha por um gerador de malha integrado ao processo de otimização.

Porém, segundo o trabalho de Schleupen, Maute e Ramm (2000), na otimização de forma, em todo o processo de procura com derivada fixa (iteração externa), a topologia de malha deve se manter fixa. O que gera inconsistência, e muitos autores como Muñoz-Rojas, Fonseca e Creus (2004) não aprovam o uso deste método, pois acredita-se que uma vez alterada a malha altera-se também o erro do processo de otimização diminuindo em alguns casos a taxa de convergência.

6.3.2 Método de Deslocamentos Fictícios

Esse método consiste em interpretar cada campo de velocidades no contorno como condições de Dirichlet impostas num problema elastostático linear fictício. As seguintes características devem ser colocadas (SILVA, 2003; CHOI; CHANG, 1994):

- o corpo $\mathcal{B}$ é constituído de material elástico linear fictício com módulo de elasticidade unitário e coeficiente de Poison nulo;
- usa-se a mesma discretização da análise de resposta para o campo de deslocamentos, devido à exigência teórica de garantir a regularidade do campo de deslocamentos;
- as velocidades dos nós internos de $\partial\mathcal{B}$ são as incógnitas do problema;
- cada variável de forma dá origem a um campo de velocidades não-nulas no contorno em uma região de $\partial\mathcal{B}$, deixando o resto do contorno associado a velocidades nulas.

Dessa maneira, cada campo é determinado a partir da solução de um problema linear, garantindo assim a exigência teórica de fornecer uma variação linear do domínio $\mathcal{B}$ em relação à respectiva variável de projeto. Cada problema linear corresponde à solução de um sistema de equações lineares da ordem de uma iteração de Newton-Raphson. Assim, pode se afirmar que a precisão usando os campos de velocidades obtidos é praticamente idêntica, sendo em alguns casos um pouco menor devido ao erro numérico da solução do sistema linear.

Como principal característica do método está a possibilidade de originar campos que permitam aplicar com maior flexibilidade a expansão em série de Taylor do campo de velocidades. Isso

permite preservar a qualidade da malha original para intervalos maiores de perturbação, fazendo com que em alguns casos não seja necessário a reconstrução da malha. Quando não existe gerador de malha integrado ao sistema, esse método é o mais adequado.

O método de deslocamentos fictícios de contorno associado à solução iterativa com baixa precisão (métodos baseados em gradiente conjugado (BITTENCOURT; FEIJÓO, 1997)) fornece campos de velocidades com características muito semelhantes às dos campos obtidos pelo método original (sem usar os métodos de gradiente conjugado), porém com custo intermediário entre o mesmo e o método de camada unitária de contorno (SILVA; BITTENCOURT, 2007).

6.4 Algoritmo de Otimização

Utilizou-se do algoritmo de otimização implementado em Silva (2003) que realizou otimização de forma em problemas envolvendo materiais hiperelásticos. O algoritmo utiliza-se do método de Pontos Interiores de Herskovits (HERSKOVITS, 1986), que consiste em um algoritmo de pontos interiores para otimização não-linear sujeita a restrições de igualdade e desigualdade. Maiores detalhes sobre esse método são encontrados e discutidos em Silva (1997).

Os parâmetros do algoritmo de Herskovits são:

- α **:** O parâmetro $\alpha \in (0, 1)$ controla a ineficiência permitida na deflexão da direção de busca original $\mathbf{d}_0$ quando a direção de deflexão $\mathbf{d}_1$ não for uma direção de decréscimo. Dessa forma, quanto menor o valor desse parâmetro menor será o valor da derivada direcional $\mathbf{d} \cdot \mathbf{C}$ ou $(\mathbf{d} \cdot \nabla \varphi)$; quanto maior, maior a declividade da direção de busca. Assim, deve ser ajustado de acordo com o grau de não-linearidade do contorno da região viável, evitando que as estimativas de passo t_0 obtidas por linearização das restrições não sejam seguidamente rejeitadas pelo critério de viabilidade. Isso resulta em passos pequenos e na necessidade de um maior número de avaliações das funções de restrição, reduzindo a eficiência global do processo. Dessa forma, um valor menor de α pode resultar na viabilidade de passos maiores.
- ξ **:** O parâmetro $\xi > 0$ indica a precisão exigida para a solução do problema, ou seja, é usado como critério de parada para o processo iterativo. Seu valor é comparado com $\|\mathbf{d}_0\|^2$ e com a

diferença percentual entre dois valores sucessivos da função objetivo. Dessa forma, deve-se avaliar o significado físico da precisão exigida, levando-se em conta as unidades utilizadas.

- $\bar{\rho}$ **:** A constante $\bar{\rho} > 0$ é utilizada quando a direção $\mathbf{d}_1$ também for uma direção de decréscimo da função objetivo. Essa situação pode ocorrer nas primeiras iterações. Adota-se, em geral, $\bar{\rho} = 1$.

- ν **:** O parâmetro $\nu \in (0, 1)$ controla a diminuição das estimativas de passo na regra de Armijo. Na busca linear mostrada em (SILVA, 1997) este parâmetro é utilizado apenas para fornecer um novo ponto para a interpolação quadrática da função objetivo ou das restrições violadas. Dessa forma, este parâmetro tem pouca influência na evolução do processo iterativo. Valores de $0,8$ ou $0,7$ são geralmente utilizados.

- η **:** A constante $\eta \in (0, 1)$ é utilizada na regra de Armijo para determinar o decréscimo mínimo aceitável da função objetivo na busca linear inexata. Se $\eta = 0$ então nenhum decréscimo é exigido, ao passo que se $\eta = 1$, exige-se um nível de decréscimo que somente pode ser atingido se a função objetivo for linear. Em problemas altamente não-lineares, em geral utilizam-se valores de η próximos de 0, já que a condição de viabilidade costuma ser muito restritiva.

- γ **:** O parâmetro $\gamma \in (0, 1)$ é utilizado para evitar a saturação das restrições na etapa de interpolação linear. Seu valor é importante apenas nas iterações iniciais, pois a certa altura passa a ser atualizada pelo valor de $\|\mathbf{d}_0\|$. Geralmente, toma-se próximo de 0 favorecendo passos maiores.

- ϵ **:** O parâmetro $\epsilon > 0$ é utilizado na regra de atualização das variáveis duais λ_i para forçar seus valores negativos para zero mais rapidamente. Entretanto, como não há um procedimento fechado para essa atualização, não há uma regra para se estimar ϵ. Na implementação de (EVSUKOFF, 1992) é utilizado $\epsilon = 0,1$, deixando claro que valores como $0,01$ e $0,001$ podem ser utilizados, dependendo mais uma vez das unidades utilizadas.

- δ **:** O parâmetro $\delta > 0$ é um dos mais influentes na convergência do algoritmo, pois controla o tipo de atualização das variáveis duais das restrições de desigualdade. Em outras palavras, esse parâmetro julga se o ponto atual está suficientemente próximo do contorno da região viável. Seu valor depende muito do problema e das unidades, sendo necessário avaliar como $\|\mathbf{d}_0\|^2$ evolui ao longo do processo e a precisão exigida para a solução.

7 hp^2fem: Ambiente de Desenvolvimento de Códigos de Elementos Finitos de Alta Ordem

7.1 Introdução

Programas de computador ou *softwares* permitem a criação, visualização e acesso a informações que antes eram inconcebíveis. Isso os torna peça indispensável ao mundo de hoje. Na engenharia, a busca e o uso intensivo de tecnologia faz com que os programas cresçam em termos de tamanho, complexidade, distribuição e importância. É nesse contexto que a *Mecânica Computacional*, uma ramificação da Engenharia Mecânica preocupada em desenvolver modelos matemáticos de sistemas mecânicos complexos, necessita de técnicas numéricas avançadas para solucionar esses modelos e equipamentos (*hardware*) de alto poder computacional.

Uma vez que os *sistemas de software*[1] tornam-se maiores e mais complexos, problemas referentes à especificação e gerenciamento do projeto são muitas vezes ainda maiores que as dificuldades de implementação e desenvolvimento de algoritmos (JACOBSON; BOOCH; RUMBAUGH, 1999; SILVA, 2003).

Assim, a adoção de procedimentos da *Engenharia de Software*[2] permite que todos os *requisitos*[3] de projeto sejam primeiramente extraídos e entendidos para que, em uma fase posterior, os *componentes*[4] sejam implementados e integrados ao sistema. Esses procedimentos formam um corpo organizado de técnicas constituindo um *processo*.

Segundo Jacobson, Booch e Rumbaugh (1999), um processo define o *sujeito*, a *ação*, a *forma*,

[1] Sistemas de software são definidos por Carvalho e Chiossi (2001) como um conjunto com fronteira claramente identificável onde existe uma relação estrutural entre os elementos pertencentes a esse conjunto. A relação entre os elementos de um sistema com os demais elementos do universo devem ser fracos o suficiente para serem desprezados, quando se considera o sistema isolado.

[2] Ainda de acordo com Carvalho e Chiossi (2001), Engenharia de Software é uma disciplina que reúne metodologias e ferramentas utilizadas para tornar um sistema operacional. O objetivo da Engenharia de Software é auxiliar no processo de produção de software, reduzindo o tempo de desenvolvimento e os custos e aumentando a qualidade.

[3] Requisito é uma condição ou capacidade que um sistema tem que satisfazer (KRUCHTEN, 2003).

[4] Em Kruchten (2003), componente de software é definido como um pedaço não-específico de software, um módulo, um pacote ou um subsistema que desempenha uma função específica, possui um limite claro e pode ser integrado numa arquitetura definida. Nesta trabalho, os termos módulo e pacotes serão considerados sinônimos.

o *tempo* e o *espaço* para que um determinado *objetivo* seja atingido. Neste caso, o objetivo é construir um novo sistema de software ou melhorar um já existente.

Para o caso de desenvolvimento de software na área de Mecânica Computacional, de acordo com Silva (2003), é comum concentrar a atenção nas funcionalidades, já que existe uma relação mais ou menos direta entre a função e o algoritmo (ou conjuntos de algoritmos), e muito pouco na estrutura. Os termos estruturais em Engenharia de Software são usualmente denominados de *arquitetura*. É através de uma arquitetura bem definida que surge a base para iteração entre diversos algoritmos e tecnologias, possibilitando a evolução rápida e concisa do sistema como um todo. A boa arquitetura é modular e possui interfaces bem definidas, ou seja, as várias partes do sistema têm contornos destacados, permitindo a rápida detecção de erros ou equívocos. Além disso permite que funcionalidades sejam atualizadas ou adicionadas ao sistema de maneira segura e rastreável.

As arquiteturas devem ser descritas através de modelagem visual (KRUCHTEN, 2003). Modelo é uma simplificação da realidade que descreve, da melhor forma possível, um meio físico por uma perspectiva em particular. A aplicação da modelagem é importante porque permite que o desenvolvedor (ou a equipe de desenvolvimento) visualize, especifique, construa e documente toda a estrutura e comportamento da arquitetura de um sistema. Já a modelagem visual vai um passo à frente, pois fornece meios para facilitar o gerenciamento desses modelos, muitas vezes complexos, ocultando ou expondo detalhes de acordo com as necessidades.

Sabendo dos benefícios que a técnica de modelagem tem mostrado no campo da engenharia, várias empresas do setor tem adotado a modelagem na atividade de desenvolvimento de software. Entre essas empresas destacam-se Lockheed-Martim, Volvo, Ericson e Intel (KRUCHTEN, 2003). Já no campo da Mecânica Computacional, a preocupação da modelagem de sistemas pode ser vista nos trabalhos de Silva (2003), Ferreira (2002), Silva e Bittencourt (2000), Sotelino, Chen e White (1998), Besson e Foerch (1997) e Yu e Kumar (2001).

Ao aplicar orientação por objetos na implementação de software para a solução numérica de problemas, tende-se a dispensar uma atenção maior à relação entre funções e estrutura do sistema, devido às próprias características dessa técnica [5].

[5] A técnica de orientação por objetos é baseada na definição entidades abstratas (classes) que agregam dados e funções com visibilidade controlada (encapsulamento de informações). Um sistema orientado por objetos é composto

De fato, se comparada com a programação orientada por algoritmos, a programação orientada por objetos permite a construção de sistemas mais robustos e ganhos de produtividade, pois aumenta o limite de gerenciamento de sistemas de software mais complexos, mesmo na ausência de técnicas de modelagem ou processo de desenvolvimento.

Mas atingido esse limite, são encontrados os mesmos problemas de manutenção e instabilidade de sistemas. Isto ocorre porque sem recursos de modelagem, gerencia-se o código fonte diretamente, comprometendo a obtenção da arquitetura adequada para o sistema, uma vez que se perde o foco do desenvolvimento, ao se preocupar com detalhes de implementação antes de planejar com mais cuidado as características globais do sistema. Segundo, sem um processo como suporte, corre-se o risco de aplicar modelagem e técnicas de programação sem objetivo ou controle.

O processo de extração de requisitos é de extrema importância. Esse processo transforma as idéias que estão na mente dos usuários em um documento formal. Dessa forma, com a análise do problema, a definição dos objetivos e o conhecimento das restrições é possível gerar um *documento de especificação dos requisitos* que descreve *como* é e *o quê* deve ser feito (CARVALHO; CHIOSSI, 2001). Idealmente, esse documento deve ser completo e consistente.

Entretanto, a entrada para esse processo, geralmente, não é completa e muito menos concisa. Como consequência, o desenvolvimento de software é feito através de uma abordagem iterativa, pois muitas vezes os requisitos mudam ao longo do processo de criação. Esse tipo de abordagem permite diminuir os riscos ou ao menos suavizá-los, porque a integração é geralmente o único momento em que os problemas são identificados e medidas corretivas ou mudanças podem ser realizados sem comprometer o tempo de entrega do produto (KRUCHTEN, 2003).

As linguagens de programação, por exemplo, fornecem condições para que os programas sejam escritos ignorando os detalhes, tais como mecanismo de endereçamento de memória e número de bits usado para representar números ou caracteres. Isso proporciona ao programador uma maior concentração sobre o problema a ser resolvido ao invés do funcionamento do computador (CARVALHO; CHIOSSI, 2001).

por interações entre instâncias de classes (objetos) através de suas interfaces visíveis (públicas). Sistemas orientados por objetos são projetados e construídos através da identificação de classes e a atribuição de responsabilidades e, portanto, a orientação por objetos induz maior preocupação com a estrutura do sistema e sua interação com as funcionalidades.

As linguagens de programação mais utilizadas em Mecânica Computacional são o *Fortram* e o C++. São linguagens de alto desempenho que estão presentes em muitos códigos comerciais tais como ANSYS, HYPERWORKS, ABAQUS, NASTRAN entre outros.

Tanto o C++ como o Fortran são linguagens que precisam de compiladores para transformar o código escrito em umas dessas linguagens em uma lista de instrução de máquina, de tal forma, que o computador execute as tarefas programadas. Além do que, ao compilar o código é necessário o uso de certas bibliotecas do sistema operacional para que o programa funcione corretamente. Isso reduz a portabilidade do código e aumenta os riscos de mal-funcionamento. Assim, surgiram as linguagens de programação denominadas *script* que executam o código diretamente via linha de comando e são de fáceis utilização. Como exemplo, pode-se citar o *TCL*, *Python* e o *MATLAB*.

O MATLAB, desenvolvido pela empresa Mathworks, tornou-se muito presente no meio acadêmico e em algumas indústrias. Devido ao fato de não ser classificada como uma linguagem de alto desempenho, o MATLAB vem sendo utilizado no desenvolvimento de pequenos programas e/ou testes de algoritmos. Porém, desde a versão R2006b, foi inserido no MATLAB suporte à programação orientada por objetos e computação distribuída que permite a sua adoção no contexto de Computação de Alto Desempenho.

Dessa forma, instituições comerciais ou acadêmicas, que queiram desenvolver sistema de software em MATLAB, precisam não apenas focar nos algoritmos numéricos para a solução de uma determinada classe de problema, mas também na organização e definições de características do sistema de software. Por essa razão, nesse trabalho utilizou a *Linguagem de Modelagem Unificada* UML (BOOCH; RUMBAUGH; JACOBSON, 1999; RUMBAUGH; JACOBSON; BOOCH, 1999) e programação orientada por objetos em MATLAB (THE MATHWORKS, Inc, 2009; REGISTER, 2007) como ferramentas de desenvolvimento de um programa de elementos finitos de alta ordem.

Discutem-se, a seguir, conceitos do processo e linguagem de modelagem citados. Por fim, exemplifica-se o uso desses conceitos na definição da arquitetura básica (*framework*) para implementação dos algoritmos de problemas estruturais, como aqueles apresentados no capítulo anterior.

Como características principais, um software de elementos finitos deve ser flexível quanto à implementação de novos métodos e mantenabilidade do programa. Neste sentido, é importante ter

e manter códigos modulares, visando aumentar a facilidade de testá-los, corrigí-los e atualizá-los de acordo com novas necessidades.

7.2 Conceitos de Programação Orientada por Objetos

Uma das tecnologias mais utilizadas atualmente para o desenvolvimento de softwares é a Programação Orientada por Objetos. Ela tem grande potencial para aumentar a produtividade e a mantenabilidade da programação, pois fornece recursos para estruturar, ordenar e por fim, modular o código.

Segundo Silva (2003), na programação orientada a procedimentos tradicionais, existe uma separação entre estruturas de dados e procedimentos de manipulação de dados. Logo, primeiro deve-se desenvolver a estrutura de dados associada ao tipo de aplicação, para depois implementar conjuntos de procedimentos necessários para o tipo de problema a ser resolvido. Como tais funções ficam dependentes dos tipos de dados manipulados, a reutilização de códigos torna-se mais difícil.

Além disso, erros ou mal funcionamento de alguma parte do código (quando alguma alteração requerida é realizada) são mais comuns. Isso se deve ao fato que grande parte das estruturas de dados são acessadas diretamente e manipuladas pelos procedimentos específicos. A programação orientada por objetos consiste em definir e implementar hierarquias de tipos abstratos de dados.

As *classes* são entidades que possuem características em comum. Os objetos são instâncias[6] de classes. Os dados da classe que devem ser armazenados são denominados de *propriedades*. As funções ou implementações de operações, particulares à classe, são denominadas de *métodos* (THE MATHWORKS, Inc, 2009). Já os atributos de classe, de acordo com Lee e Tepfenhart (2001), definem as características ou comportamento de uma classe.

Definem-se agora alguns termos muito utilizados em programação orientada à objetos (LEE; TEPFENHART, 2001; SILVA, 1997; QUATRANI, 2000; HORTON'S, 2006). São termos que devem ser bem entendidos e assimilados pelo programador, com o intuito de evitar equívocos durante a pro-

[6]Em programação orientada por objetos, a palavra instanciar possui o mesmo significado que criar. Dessa forma, instância é uma variável criada para uma dada classe

gramação.

Método : é um conjunto detalhado de operações que um objeto realiza quando um outro objeto chama por um serviço (LEE; TEPFENHART, 2001). Um método é similar a uma função. Um método necessita de uma mensagem para ser acessado e cada método pode somente usar seus próprios conjuntos de dados. Um método em C++ é facilmente identificado pelo sinal :: que está situado entre o nome da classe e o nome do método dessa classe. Os métodos podem ser definidos como público, privado ou protegido.

Atributos : são dados próprios da classe, que são acessados por meio de métodos da classe. Os atributos são definidos também como público, privado ou protegido.

Abstração : consiste na generalização de um sistema de modo a abranger as características mais importantes e desprezar os detalhes específicos desse sistema. Dessa forma, as características relevantes de um objeto de uma classe devem ser definidas para torná-lo diferente dos objetos de outras classes. Podem ser aplicados os mais diversos níveis de abstração, desde a reprodução fiel das características de objetos reais até a definição de classes puramente abstratas.

Uma abstração é eficiente quando sua interface é pública, simples, concisa e independente da estrutura interna da classe. Também deve ser genérica o suficiente para que os objetos da classe possam ser aplicados em múltiplos casos. A abstração está muito ligada ao conceito de *herança* e *polimorfismo* que serão definidos posteriormente.

Encapsulamento : Dado um objeto, um procedimento externo e demais objetos podem não conhecer ou acessar sua estrutura interna. A capacidade de ocultar a estrutura interna dos objetos, deixando visível apenas os componentes de sua interface publica é chamado de *encapsulamento*.

Segundo Liberty (1998), existem três níveis de acesso aos métodos e dados em $C++$: `public`, `protected` e `private`. O acesso `public` permite que os componentes sejam acessados externamente. O nível `protected` engloba dados e procedimentos que podem e devem ser acessados somente pelos métodos originários da classe e das classes derivadas (herança). Já os componentes com acesso `private` podem ser acessados unicamente pelos métodos da própria classe, sendo também invisíveis para as classes derivadas.

É considerado como uma boa prática de programação restringir ao máximo o acesso aos atributos, de forma a isolar detalhes de implementação de uma classe, tornando públicas somente as funções membro que realizam uma interface mínima da classe com outros componentes.

Modulação : permite que um sistema possa ser decomposto em subconjuntos, de modo que possa funcionar com um certo grau de independência ou apenas fracamente dependente. Com o uso de abstração e encapsulamento, um programa pode ser construído de forma que o funcionamento se dê através da interação de objetos pouco acoplados, tornando fácil a atualização e detecção de erros.

Herança : Dada uma classe, a mesma pode ser utilizada para construir subclasses, herdando os dados e métodos da classe antecessora através do recurso de derivação de classes.

Dessa forma, as classes podem ser definidas formando uma estrutura hierárquica. A hierarquia permite conhecer o nível de relação entre as classes, ou seja, é possível determinar a classe "mãe" (classe mais genérica ou abstrata) situada no topo da hierarquia e as classes "filhas" (classes mais específicas) situadas na base. A herança permite que a adição de novas classes derivadas seja o mais transparente e simples possível. Também, pode-se agregar ao código módulos já testados e confiáveis.

Polimorfismo : O polimorfismo pode ser estático (resolvido em tempo de compilação de código) ou dinâmico (resolvido em tempo de execução de código). A sobrecarga de funções é um polimorfismo estático que permite que um programa possa declarar várias funções com o mesmo nome, diferenciando entre si pela quantidade de parâmetros apresentados e por seus respectivos tipos de dados. A sobrecarga de operadores também é um polimorfismo estático que permite que a definição de certos operadores resultem em uma chamada de função que depende dos tipos de dado dos operadores sendo utilizados.

O polimorfismo por herança é um exemplo de polimorfismo dinâmico, no qual ponteiros de uma classe base podem referenciar objetos de classes derivadas, o que permite que uma chamada de função virtual seja resolvida em tempo de execução de código. Ponteiros e referências de uma classe base podem referenciar objetos de qualquer classe derivada de si, o que permite que arranjos e outros containers de um dado tipo possam armazenar ponteiros de diversos tipos de dados, o que não poderia ser feito de outra maneira em C++.

Relações : As relações de um programa orientado à objetos podem ser associação ou agregação.

Elas são usadas para capturar a colaboração entre os objetos necessários para providenciar um serviço para um cliente.

Por associação, pode-se entender com uma relação bidirecional ou unidirecional entre duas classes, isto é, existe uma ligação entre os objetos das classes associadas. Já a relação de agregação é uma forma especializada de associação, na qual tudo faz parte do contexto de relação.

Agregação é conhecido como "parte de" ou uma relação de englobamento. Por exemplo, uma classe que controla todos os atributos do modelo de elementos finitos `DiscreteModel` possui uma relação de agregação com a classe `Nodes`, pois os nós são partes obrigatórias do modelo de elementos finitos.

7.3 Programação Orientada por Objetos em MATLAB

Para usar a Programação Orientada por Objetos em MATLAB, assim como em C++ , é necessário primeiro decompor o seu domínio (sistema) em objetos, modularizando o código o máximo possível. Isso reduz a complexidade do código e permite aumentar a consistência do programa, diminuindo os requerimentos de testes e manutenção. Dessa maneira, os objetos fazem o papel de interface para que ocorra a interação entre os módulos.

A construção de uma classe em MATLAB obedece à seguinte sintaxe:

```
classdef ClasseA                        % nome da classe
  properties                            % propriedade da classe
    Prop1;                              % nome da propriedade
  end
  methods                               % métodos da classe
    function obj = ClasseA(val1)        % construtor da classe
      obj.Prop1 = val1;
    end
    Metodo1(Param);                     % escopo do método
  end
end
```

Nota-se que os componentes básicos utilizados na definição da classe estão claramente separados em blocos. Esses blocos são:

classdef : contém a definição da classe em um arquivo com extensão ".m". Esse arquivo possui o mesmo nome da classe. O bloco começa com a palavra chave **classdef** e encerra-se com a palavra chave **end**;

properties : contém as definições das propriedades da classe. Para cada conjunto de atributos de acesso é necessário criar blocos diferentes. Os blocos são inicializados com a palavra chave **properties** e encerrado com **end**;

methods : neste bloco estão presentes as definições dos métodos da classe. Pode-se, aqui, definir somente o escopo dos métodos como nome do método e os seus parâmetros de entrada e saída (se necessários), ou implementá-los diretamente, como feito no caso do construtor da classe definido entre as palavras chaves **function** e **end**. Assim como o bloco das propriedades, para cada tipo de acesso, um bloco novo deve ser implementado.

Frequentemente, os dados (variáveis) e operações (métodos) estão encapsulados dentro de uma classe para segregar ou evitar que informações e implementações sejam acessadas de forma errônea de um módulo para outro. A modularidade do código é controlada através de três níveis de acesso, como na linguagem C++, ou seja:

- Público (Public): qualquer parte do código pode acessar esta propriedade ou método;
- Protegido (Protected): somente os objetos ou métodos pertencente à classe ou às classes derivadas podem acessar esta propriedade ou método;
- Privado (Private): somente os objetos da classe podem acessar os métodos ou propriedades definidos na classe.

Entretanto, o MATLAB permite uma flexibilidade maior na definição dos níveis de acesso. É possível definir diferentes tipos de controle sobre o acesso para modificação e retorno de uma propriedade ou método. Por exemplo, pode-se definir uma propriedade cujo acesso à modificação

seja protegido, mas o acesso para retornar o seu conteúdo é público. Dessa forma, qualquer método ou objetos de outras classes podem acessar o conteúdo dessa variável, porém não podem modificá-la, a não ser no caso de serem da classe em questão. O padrão do MATLAB é definir atributos das propriedades e métodos como público. A definição de acessos diferentes são realizados na definição da classe. O exemplo a seguir ilustra o processo de definição dos atributos.

```
classdef ClasseA
  properties(SetAccess = protected, GetAccess = public)
    Prop1;
  end
  methods(Access = public)
    function obj = ClasseA(val1)
      obj.Prop1 = val1;
    end
    MetodoSetProp1(Param1);
  end
  methods(Access = protected)
    Metodo2(Param1,Param2);
  end
end
```

No exemplo anterior, a propriedade `Prop1` foi definida somente para leitura, ou seja, a modificação só é permitida via método público `MetodoSetProp1`. O valor de `Prop1` pode ser retornado à qualquer momento via comando `obj.Prop1` no ambiente de comando do MATLAB.

Os construtores no MATLAB, assim como em C++, são definidos como públicos. Dessa forma, basta definir o acesso público como realizado no exemplo. O método público `MetodoSetProp1(Param1)` e o método protegido `Metodo2(Param1,Param2)` estão implementados em arquivos .m de mesmo nome dos métodos.

Do ponto de vista de arquivos, a construção de classes no MATLAB é feita sob uma árvore de diretórios de três maneiras distintas. Na primeira forma, vista na Figura 7.1, a classe é definida em um arquivo .m, cujo nome é o mesmo da classe. Esse arquivo deve conter também as propriedades

e as implementações dos métodos da classe.

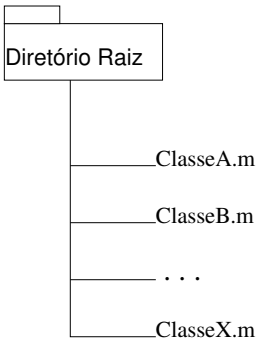


Figura 7.1: Primeira maneira, e mais simples, de agrupar as classes MATLAB em forma de diretórios (THE MATHWORKS, Inc, 2009).

Na segunda forma, agrupa-se a definição de classe e os métodos em arquivos .m separados dentro de um diretório com mesmo nome da classe precedido pelo símbolo de "@" dentro do diretório de trabalho do MATLAB como mostrado na Figura 7.2. Tomando-se como base a Figura 7.2, a definição da classe, as propriedades e o escopo do método `MétodoClasseA` estão no arquivo **ClasseA.m**, enquanto que a implementação do método está no arquivo **MétodoClasseA.m**.

A terceira forma, consiste no agrupamento das várias classes em diretórios precedidos do símbolo "+", chamados de *pacotes*. A Figura 7.3 mostra essa última maneira de compor as classes em estruturas de diretório. Nesse trabalho, escolheu-se a composição baseada em pacotes e múltiplas definições de classe, onde cada método é escrito em arquivos separados, pois permite uma melhor organização do código e documentação.

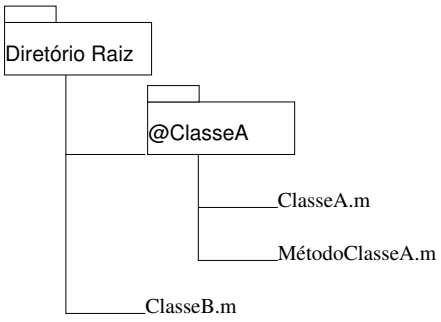


Figura 7.2: Segunda maneira de agrupar as classes MATLAB em forma de diretórios (THE MATHWORKS, Inc, 2009). Desta forma, a classe e os métodos estão definidos em arquivos separados.

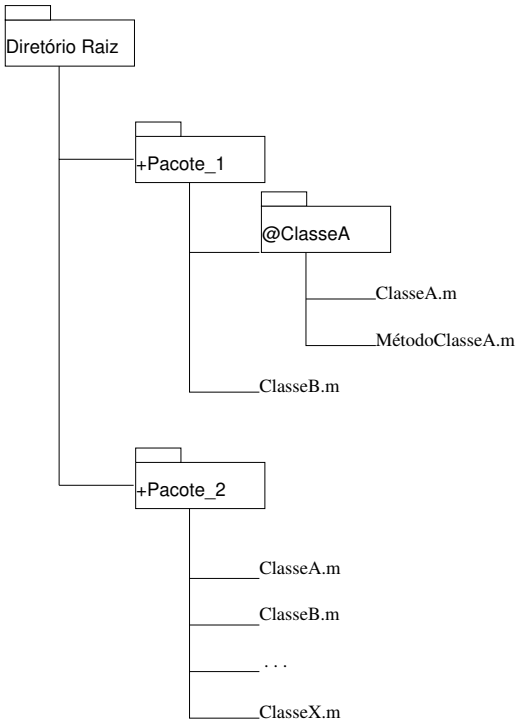


Figura 7.3: Terceira maneira de agrupar as classes MATLAB em forma de diretórios (THE MATHWORKS, Inc, 2009). Essa composição permite modular o código de maneira mais simples e segura.

Apesar do MATLAB possuir suporte a hierarquia, polimorfismo e construção de classes abstratas, não é possível criar métodos virtuais puros como em C++. Essa limitação se deve ao fato que o MATLAB não permite criar métodos de mesmo nome porém com argumentos de entrada e saída diferentes (chamados de métodos virtuais puros em C++). O MATLAB não checa o tipo dos argumentos, mas somente o número deles são conferidos. Dessa maneira, como sugerido em Register (2007), faz-se o uso de classes containers. Essas classes criam uma propriedade denominada vetor de células que permite que objetos de outras classes sejam agrupados e possam acessar seus próprios métodos.

7.4 Modelagem Visual de Software

Na construção de um sistema de software é necessário que se desenvolva e mantenha-se modelos que representem a realidade de todo o sistema. Através de modelos é possível que a realidade seja simplificada, permitindo-se que sistemas complexos sejam dominados e compreendidos em sua totalidade ou algo próximo disso.

Como ferramenta de modelagem, tem-se a *Linguagem de Modelagem Unificada*, (*Unified Modeling Language* – UML), que é uma linguagem de sistemas de software orientada por objetos (KRUCHTEN, 2003; LEE; TEPFENHART, 2001; SCOTT, 2003). Essa linguagem surgiu da fusão e revisão de técnicas já estabelecidas de modelagem de software, o que possibilitou sua adoção como padrão de modelagem.

A UML permite modelar, estruturar requisitos, analisar, projetar e documentar um sistema de software utilizando diagramas. Por ser orientada por objetos, a UML encaixa-se perfeitamente no contexto de desenvolvimento de um programa orientado por objetos. Assim, os conceitos de classe, atributos, operações e seus relacionamentos são facilmente transferidos de um modelo descrito através de diagramas para a linguagem de programação.

O principal diagrama UML utilizado neste trabalho para o desenvolvimento de um sistema de um software de elementos finitos foi o diagrama de classe. Este diagrama descreve as dependências, associações e hierarquias (quando necessário) entre classes, além de suas características internas. As dependências são definidas como linhas tracejadas e indicam quando uma classe uti-

liza objetos de outra classe como atributos, parâmetros de operações ou apenas variáveis locais. As associações são indicadas por linhas contínuas contendo diamantes e/ou setas nas extremidades, que representam conexões de longa duração entre objetos de classes diferentes.

O uso de UML no processo de construção de software pode ser encontrado, com maiores detalhes de outros tipos de diagramas importantes e estudo de casos, em Quatrani (2000), Lee e Tepfenhart (2001). Já em Silva (2003), encontra-se a modelagem completa de um sistema de software de elementos finitos de baixa ordem.

Os elementos apresentados anteriormente formam a base para a programação orientada por objeto no MATLAB. A seguir, apresenta-se a organização e estrutura do código de elementos finitos de alta ordem desenvolvida neste trabalho e em outros trabalhos do grupo.

7.5 Organização do hp^2fem

A primeira estrutura do programa de elementos finitos usada no grupo de estudo foi idealizada por Bittencourt (2000) e organizada usando a ferramenta de engenharia de software IBM RATIONAL ROSE no trabalho de Silva (2003). Aqui, o modelo de elementos finitos foi expandido para dar suporte à alta ordem usando as bases tensorizáveis estudadas em Vazquez (2008), Bargos (2009).

O programa hp^2fem é um programa de elementos finitos de alta ordem orientados por objetos formado por um conjunto de arquivos MATLAB. O programa é composto pelo módulo +Solver, que está dividido em 14 subpacotes como mostrado na Figura 7.4 que serão detalhados a seguir.

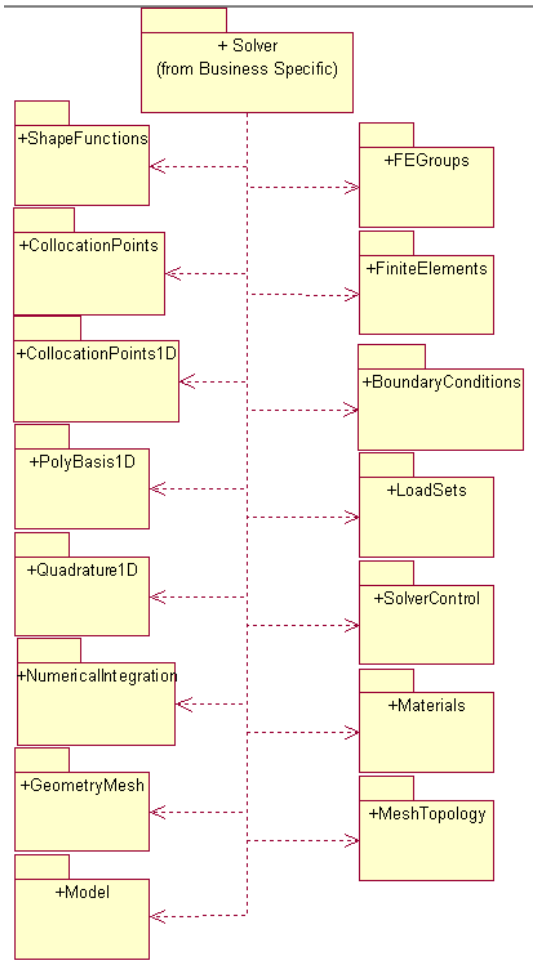


Figura 7.4: Módulos principais do programa hp^2fem.

Os subpacotes são compostos pelas seguintes classes:

+SolverControl : `Solver`, `GlobalSolver` e `ElementElementSolver`;

+Model : `DiscreteModel`, `DOFs`, `Equations` e `Nodes`;

+BoundaryConditions : `BoundaryConditions`, `HDirichletBC` e `NHDirichletBC`;

+LoadSets : `LoadSets` e `LoadSet`;

+FEGroups : `FEGroups` e `FiniteElementGroups`;

+FiniteElement : `FiniteElement`, `GeometricProperties`, `Mapping` e `GradientMeasure`;

+Material : `IsotropicElasticMaterial` e `NeoHookeanMaterial`;

+GeometryMesh : `GeometryMesh`, `GMKeyPoints`, `GMLines` e `GMSurfaces`;

+MeshTopology : `MeshTopology`, `ElementElementTable`, `EquationEquationTable`, `NodeElementTabe`, `NodeNodeTable`, `HighOrder` e `TopologicalIndices`;

+ShapeFunctions :`ShapeFunctions`, `LineShapeFunctions`, `SquareShapeFunctions`, `TriangleShapeFunctions`, `HexaShapeFunctions`, `TetraShapeFunctions`;

+NumericalIntegration :`NumericalIntegration`, `LineNmericalIntegration`, `SquareNumericalIntegration`, `TriangleNumericalIntegration`, `HexaNumericalIntegration`, `TetraNumericalIntegration`;

+CollocationPoints : `CollocationPoints`, `LineCollocationPoints`, `SquareCollocationPoints`, `TriangleCollocationPoints`, `HexaCollocationPoints` e `TetraCollocationPoints`;

+CollocationPoints1D : `CollocationPoints1D`, `GaussJacobiCollocationPoints1D` e `NewtonCotesCollocationPoints1D`;

+PolyBasis1D :`Polynomials1D`, `Hermite1D`, `Jacobi1D`, `LagrangeJacobi1D`, `Lobatto1D`, `StandaedLagrange1D` e `TruncatedLagrange1D`;

+Quadrature1D :`Quadraure1D`, `GaussJacobiQuadrature1D` e `NewtonCotesQuadrature1D`;

As principais funcionalidades e relações de cada um dos subpacotes e de suas respectivas classes serão explicadas a seguir.

O subpacote +SolverControl é responsável pelas entidades de armazenamento do sistema global, procedimentos para sua construção e diretrizes de solução de modelos lineares e não-lineares. A classe principal desse subpacote, `Solver`, inicializa o procedimento de leitura de um arquivo de entrada ASCII, com extensão .def, que contém os parâmetros de solução tais como: tipo de análise (estática, dinâmica e projeção), método de solução do sistema linear, algoritmo de solução de problema linear ou não-linear, tipo de operador (simétrico ou não-simétrico) e tipo de solver (global ou elemento-elemento).

Conforme a opção selecionada para o tipo de solver empregado, um objeto é inicializado para a classe de solução correta. Caso o modelo empregado necessite de um solver global, uma instância para a classe `GlobalSolver` é criado. Será a classe `GlobalSolver` ou `ElementElementSolver` a responsável pelos procedimentos de criação do sistema global de solução e sua posterior solução.

A classe `DiscreteModel` é inicializada pelo arquivo ASCII com extensão .fem, o qual contém as definições de malha, e também pelo arquivo ".def", que possui as definições do tipo de problema e informações sobre as condições de contorno e carregamento. Assim, esta classe tem a condição de armazenar o modelo numérico em termos de nós, elementos, condições de contorno, condições de carregamento, graus de liberdade do modelo, topologia da malha e relação malha-geometria. Esse armazenamento é feito através de relações de agregações com cada uma das classes especializadas, como representado na Figura 7.5. As instâncias para as classes de nós (`Nodes`), grupos de elementos finitos (`FEGroups`), equações (`Equations`), graus de liberdade `DOFs`, condições de contorno (`BoundaryConditions`), condições de carregamento `LoadSets`, topologia da malha (`MeshTopology`) e relação malha-geometria (`GeometryMesh`) são dadas, respecti-

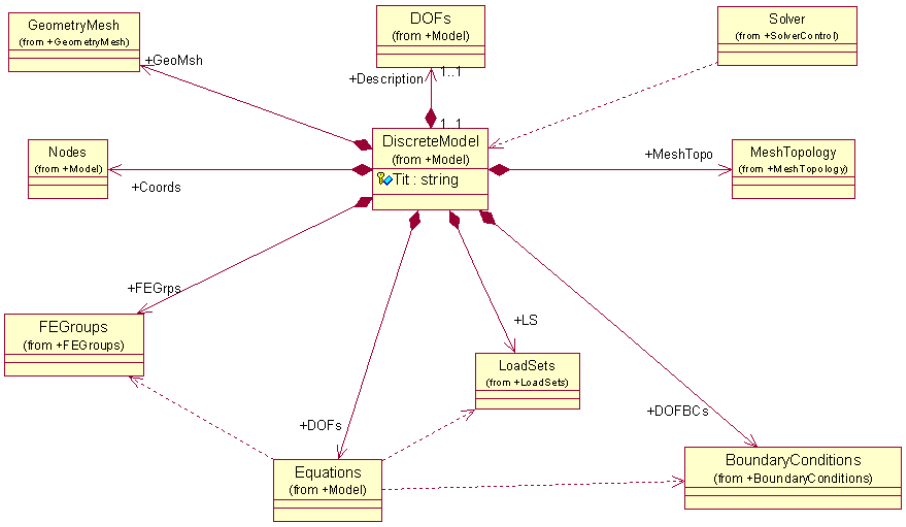


Figura 7.5: Relacionamento entre a classe `DiscreteModel` e as demais classes do programa hp^2fem .

vamente, pelas classes `Coords`, `FEGrps`, `Eqs`, `DOFs`, `DOFBCs`, `LS`, `MeshTopo` e `GeoMsh`.

As definições das propriedades e comportamento do material estão definidas na classe `Material`. Essa classe aplica o conceito de classe container, para definir uma pseudo-relação de derivação com as classes específicas para cada modelo de material. Por enquanto, o programa somente dá suporte aos modelos de materiais elásticos isotrópico linear e neo-Hookeano compressível, armazenados respectivamente pelas classes `IsotropicElasticMaterial` e `NeoHookeanMaterial`. O tipo de material é conhecido no momento de execução do programa a partir da classe `Material`, a qual inicializa um objeto para uma das classes especializadas, conforme mostrado na Figura 7.6. O tensor de tensão de Cauchy e o segundo tensor de tensão de Piola-Kirchhoff, explicados no Capítulo 4, estão implementados nas classes `IsotropicElasticMaterial` e `NeoHookeanMaterial` e são chamados corretamente pela classe `Material` de acordo com o tipo de material e cinemática escolhidos.

`BoundaryCondition` funciona de forma análoga a classe `Material`. A parir do tipo da condição de contorno definida no arquivo .def, a qual pode ser uma condição homogênea (condições nulas) ou não-homogênea (não-nulas) de Dirichlet, cria-se objeto para a classe de armazenamento e manipulação especializada. Assim, para uma condição não-homogênea de Dirichlet, tem-se um objeto, denotado por `NHDBC`, para a classe `NHDirichletBC` (veja Figura 7.7). No caso de uma

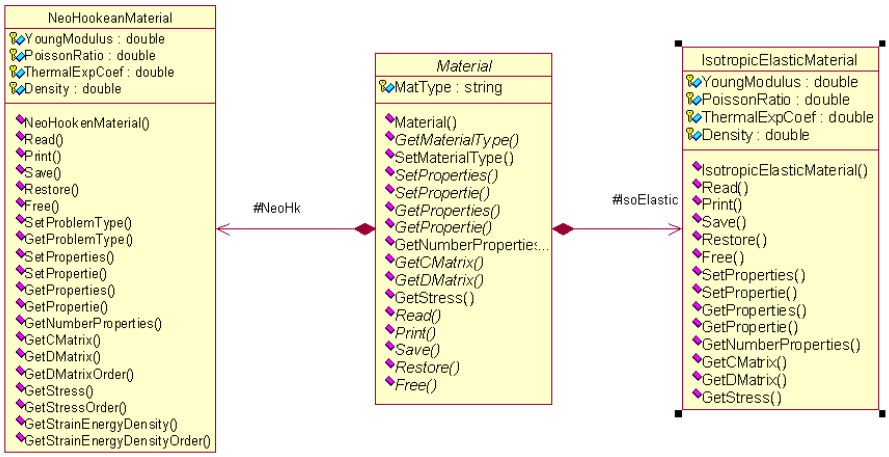


Figura 7.6: Diagrama da classe `Material`.

condição homogênea, cria-se o objeto `HDBC` da classe `HDirichletBC`.

As condições de contorno podem ser aplicadas às entidades de geometria (ponto, linha e superfície) e entidades de malha (nós, arestas e faces). Aplicando-se sobre a geometria, as condições de contorno são transferidas para as entidades de malha via relação malha-geometria estabelecida pelas classes contidas no subpacote +GeometryMesh. Por exemplo, ao se aplicar uma condição de travamento sobre uma superfície, essa informação é automaticamente transferida para as faces dos elementos que compõe essa superfície. É óbvio que essa relação malha-geometria depende diretamente da incidência dos elementos contidos em uma dada superfície. Essa informação é armazenada e manipulada pelas classes do subpacote +MeshTopology. Os diagramas das classes contidas em +MeshTopology e +GeometricMesh estão nas Figuras 7.8 e 7.9, respectivamente.

Um objeto `LoadSet` representa um caso de carregamento em problemas lineares e um passo de carregamento em problemas não-lineares. Logo, os casos ou passos de carregamento são armazenados pela classe `LoadSets` na forma de um array de objeto da classe `LoadSet`. Assim como na forma de aplicação de condição de contorno, os carregamentos podem ser aplicados sobre as entidades de geometria e/ou malha. A Figura 7.10 apresenta o diagrama da classe `LoadSets` e suas relações com outras classes do programa hp^2fem.

As condições de contorno não-homogênea e carregamentos podem ser aplicados de forma

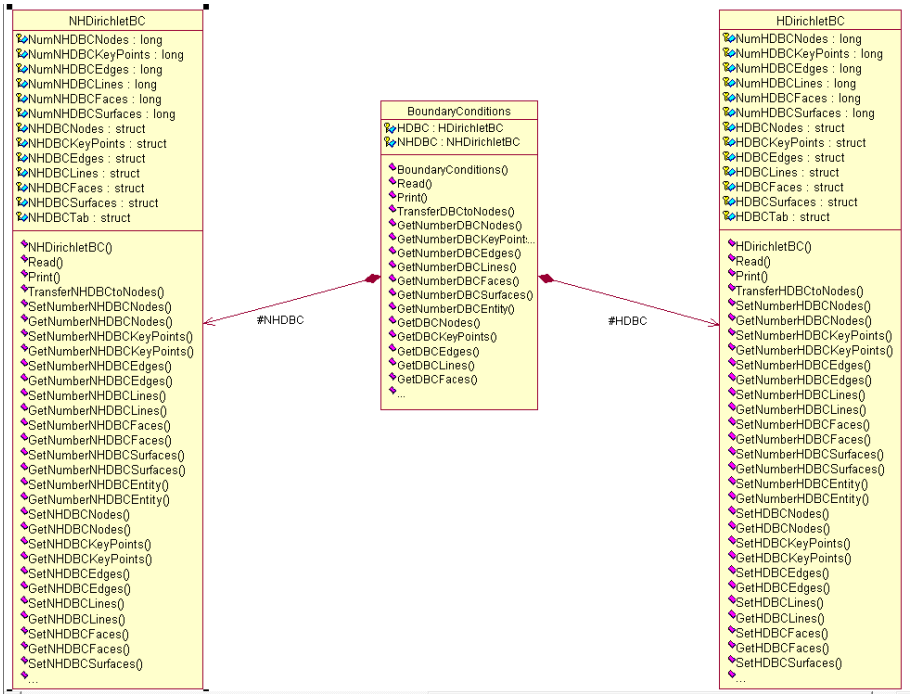


Figura 7.7: Diagrama da classe `BoundaryConditions`.

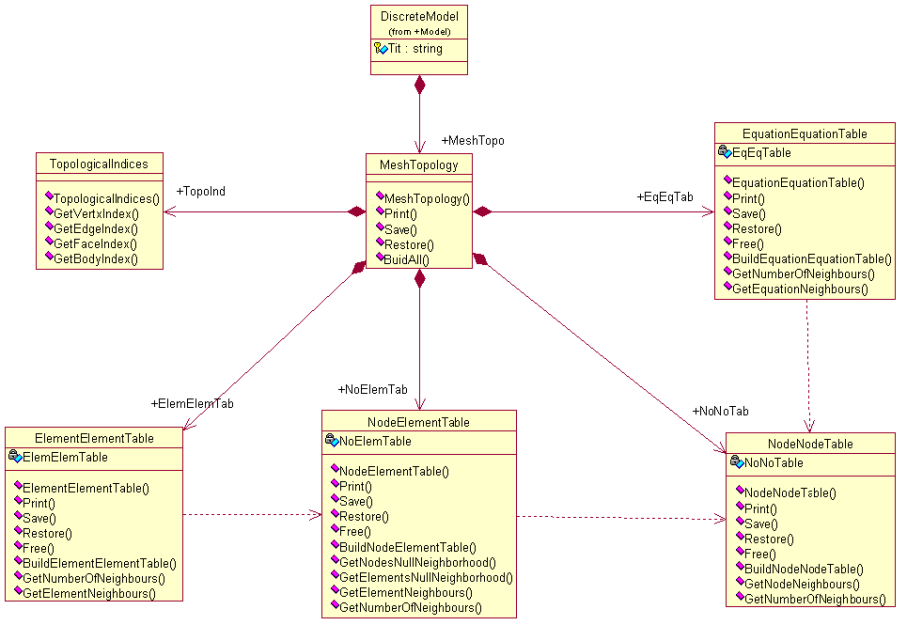


Figura 7.8: Diagrama da classe `MeshTopology`.

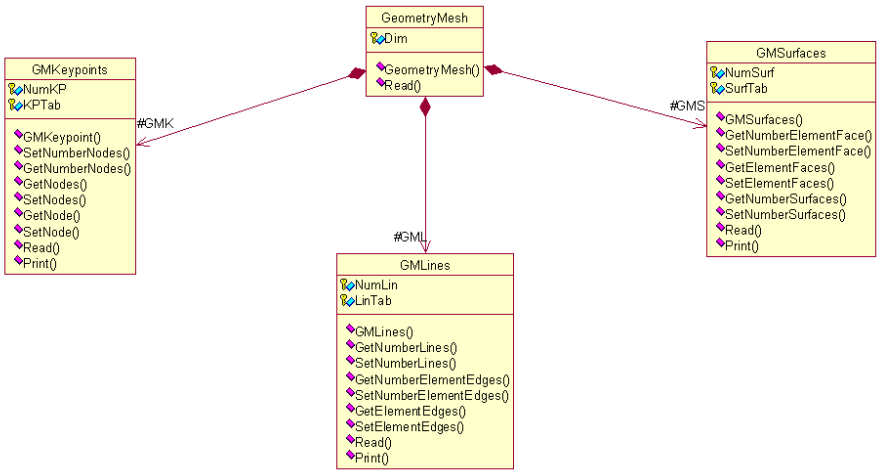


Figura 7.9: Diagrama da classe `GeometricMesh`.

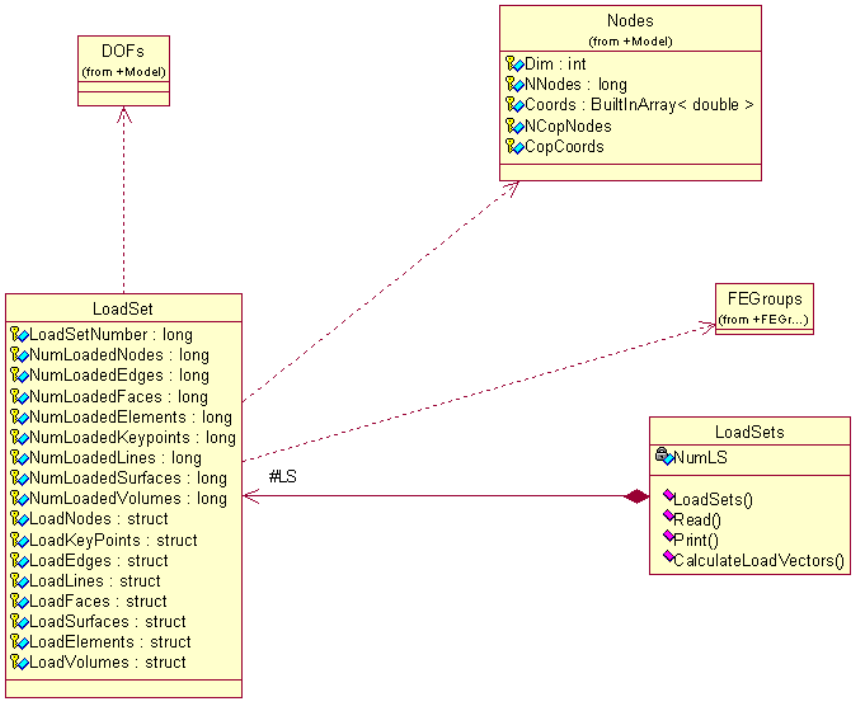


Figura 7.10: Diagrama da classe `LoadSets`.

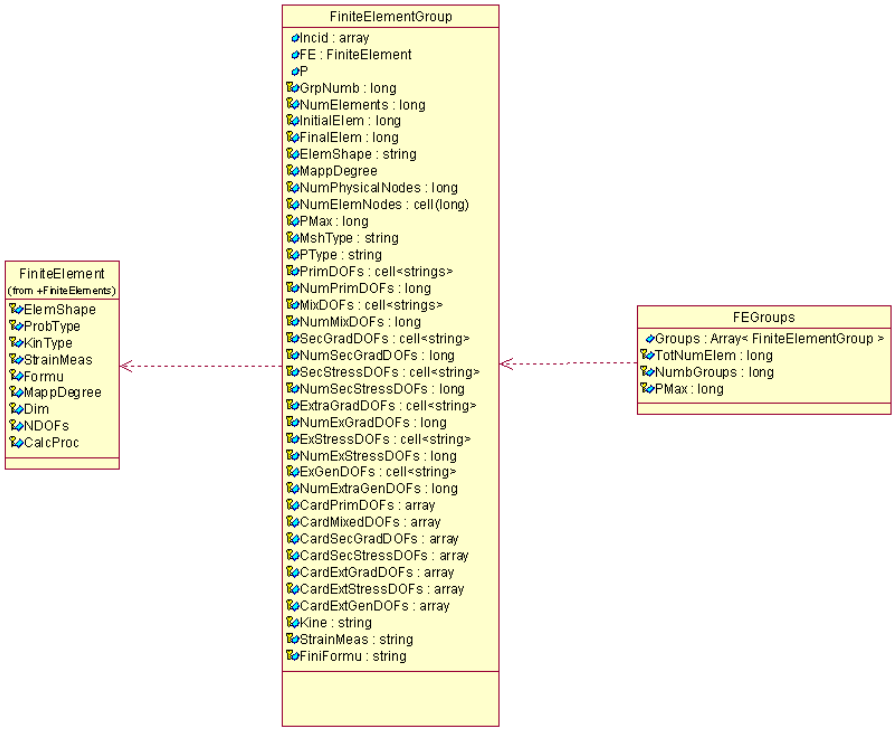


Figura 7.11: Classes para representação e manipulação do conjunto de elementos do modelo discreto.

simbólica. Usando o recurso de manipulação simbólica do MATLAB é possível converter uma função literal em um valor numérico. Essa conversão é feita dentro da classe de elementos finitos `FiniteElement`.

A classe `FEGroups` representa o conjunto de todos os elementos do modelo discreto (Figura 7.11), sendo seu principal atributo um array de objetos `FiniteElementGroup`. Cada objeto `FiniteElementGroup` define um grupo de elementos de mesmo tipo (formato e esquemas de interpolação e integração), material e propriedades geométricas. Seus principais atributos são a incidência do grupo e um objeto `FE` para `FiniteElement`, o qual pode ser inicializado como qualquer uma de suas especializações (tipo de problema e forma de elemento).

Já a classe `FiniteElement` implementa as operações realizadas sobre o elemento, excluindo-se aquelas referentes ao material. As dependências desta classe são mostradas no diagrama da figura 7.12, ou seja, realiza as principais operações a nível elementar. Os gradientes

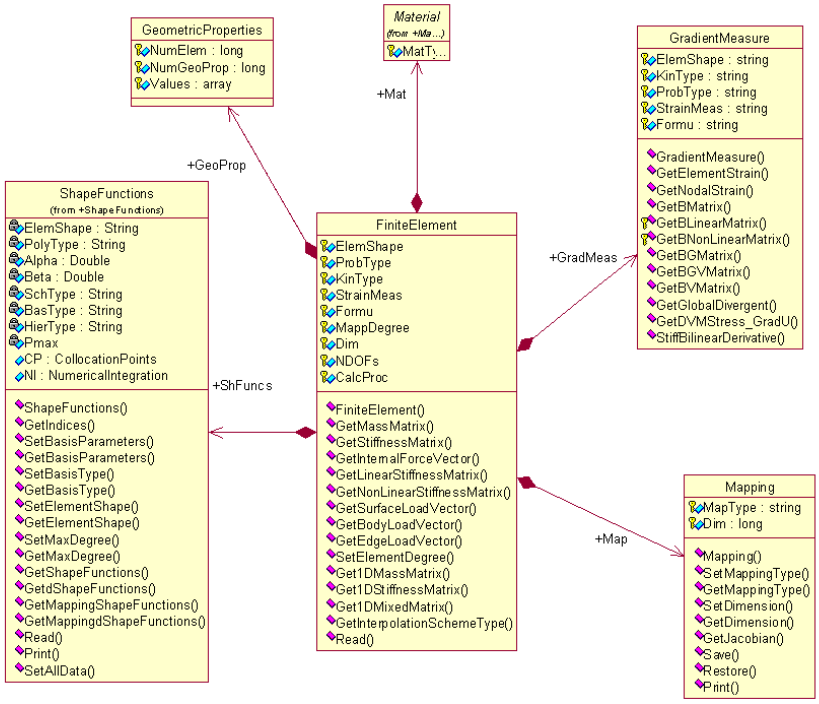


Figura 7.12: Diagrama da classe `FiniteElement`.

de deformação de Green-Lagrange e Almansi-Hamel e o Jacobiano são calculados pelas classes `GradientMeasure` e `Mapping`, respectivamente.

As propriedades geométricas, quando aplicáveis a uma determinada análise, são armazenadas na classe `GeometricProperties`. As funções de interpolação de um elemento são construídas de acordo com a forma do elemento através da tensorização das funções de interpolação 1D, como visto em Bargos (2009) e Vazquez (2008). Assim, para gerar as funções para um quadrado de grau 4, primeiro são geradas as funções de uma linha de grau 4. O diagrama de função de interpolação está na Figura 7.13.

De forma análoga, os pontos de colocação e integração são também tensorizados a partir do modelo unidimensional, de acordo com a ordem da função de interpolação. As Figuras 7.14 e 7.15 ilustram os diagramas das classes `NumericalIntegration` e `CollocationPoints`, respectivamente.

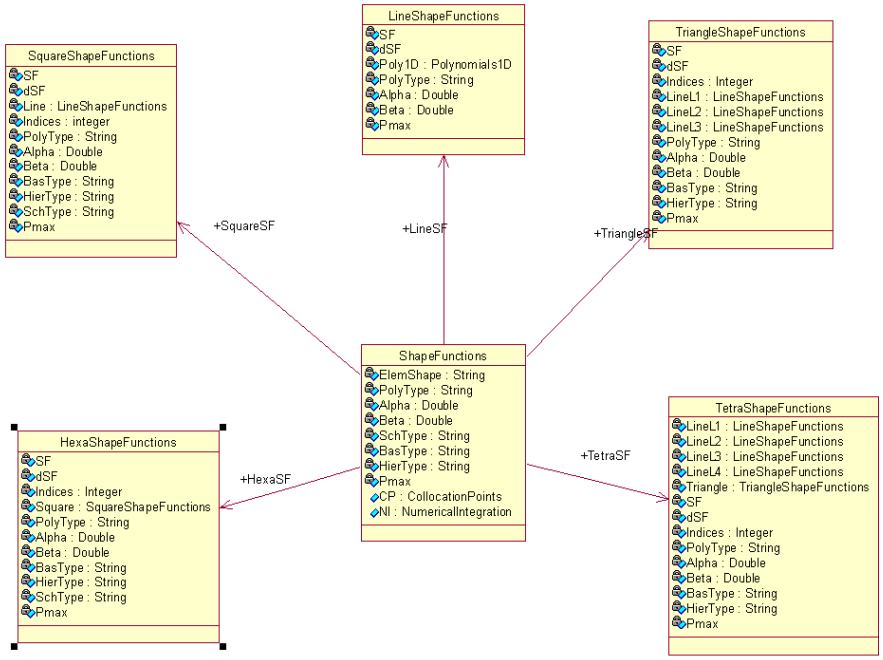


Figura 7.13: Diagrama da classe `ShapeFunction`.

As classes `PolynomialBais1D`, `CollocationPoints1D` e `Quadrature1D`, mostradas nas Figuras 7.16 à 7.18, fornecem a base polinomial empregada pelas funções de interpolação, o esquema de criação dos pontos de colocação e a regra de integração empregada no âmbito da formulação 1D. Essas classes permitem que as funções de forma, os pontos de colocação e a regra de integração 2D e 3D sejam facilmente derivadas através de produto tensorial da formulação 1D. Estas classes foram baseadas nos trabalhos Vazquez (2008) e Bargos (2009).

7.6 Funcionamento do Programa para Solução de Problemas de Grandes Deformações e Contato

O programa hp^2fem é alimentado por dois arquivos ASCII. O primeiro arquivo tem extensão ".fem" e possui as informações de malha, tais como dimensão do domínio, número de nós, incidência de elementos, tipo de elemento e relação malha-geometria. Por relação malha-geometria, entende-se definir, por exemplo, para uma determinada superfície quais são as faces de elementos que a compõe. O segundo arquivo, de extensão ".def", contém as informações sobre as diretivas de

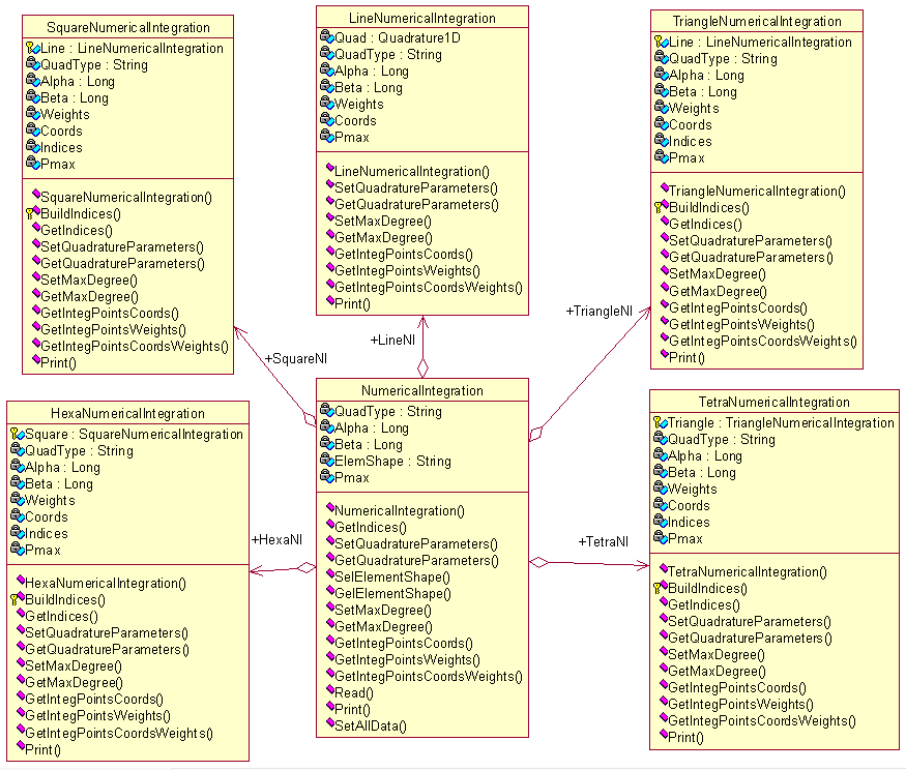


Figura 7.14: Diagrama da classe `NumericalIntegration`.

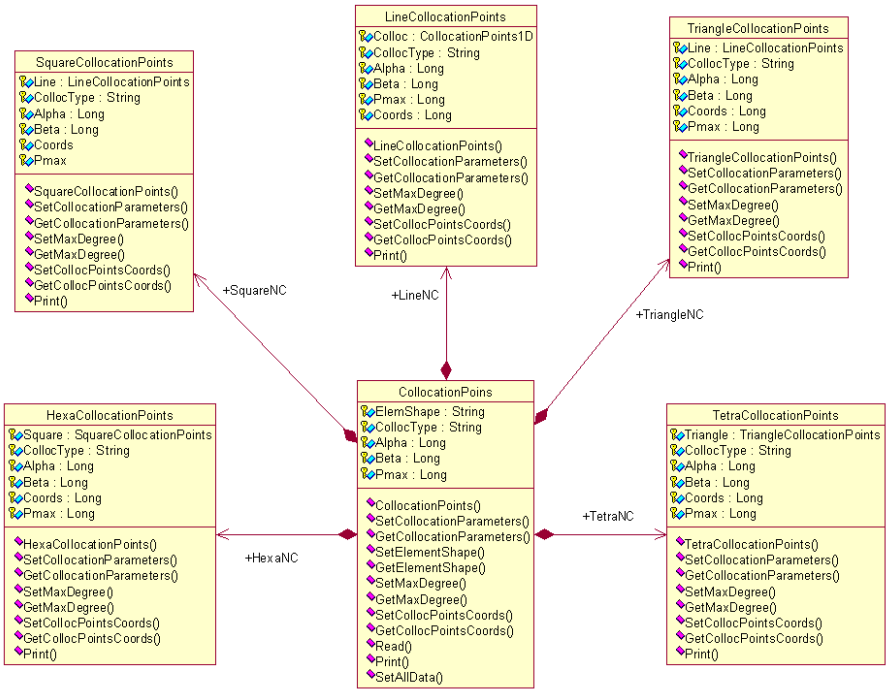


Figura 7.15: Diagrama da classe `CollocationPoints`.

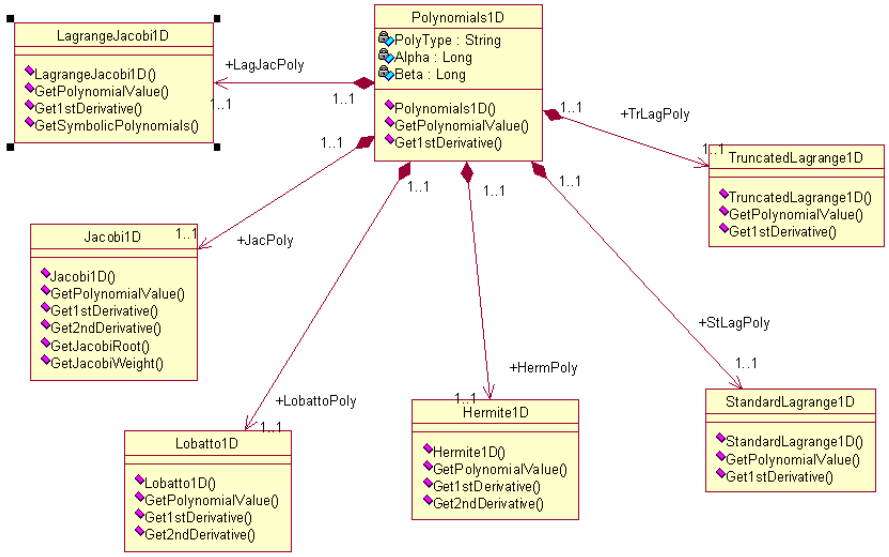


Figura 7.16: Diagrama da classe `PolynomialBasis1D`.

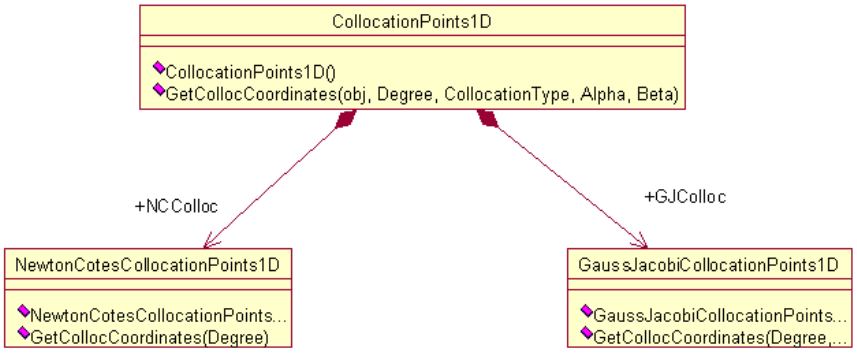


Figura 7.17: Diagrama da classe `CollocationPoints1D`.

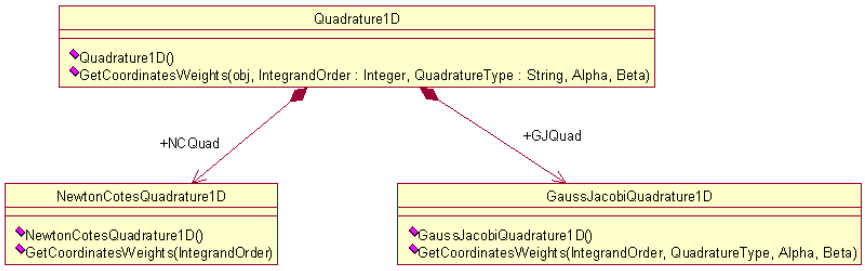


Figura 7.18: Diagrama da classe `Quadrature1D`.

solução e as propriedades físicas para cada grupo tais como propriedades do material, condição de contorno, carregamentos, propriedades geométricas, cinemática e graus de liberdade.

Utiliza-se o programa Gid (CIMME, 2008) para pré-processamento e gerar dois arquivos. No entanto, o programa não é compatível com a geração de malha de elementos de alta ordem, sendo assim, necessários alguns recursos que permitam o hp^2fem transformar uma malha de baixa ordem para uma de alta ordem. Tendo por base uma malha linear, uma base de interpolação, o tipo de mapeamento e o grau do polinômio, pode-se gerar os nós para uma malha de qualquer grau P desejado.

7.6.1 GiD

O GiD é um programa de pré e pós-processamento que possui uma interface gráfica simples usada para construir geometrias de modo hierárquico, ou seja, uma entidade de nível maior é gerada sobre uma entidade de nível menor. Por exemplo, uma linha (entidade superior) é composta por dois pontos (entidade de nível mais baixo). Também, todos os materiais, condições e parâmetros de solução são aplicados sobre as entidades geométricas, e somente após a concepção da malha é que estas definições são passadas para os nós e elementos. Esta é, também, uma característica importante para a otimização de forma, quando há necessidade de geração automática de malha, após a modificação da geometria devido a minimização das variáveis de projeto.

Quando deseja-se usar o GiD com um programa específico escrito por um usuário, é necessário pré-definir algumas informações que serão trocadas entre os programas. Para isso, faz-se uso da customização, e consiste em criar um conjunto de arquivos que possibilite ao GiD interpretar o que está sendo solicitado e traduzir no formato requerido pelo pacote de otimização e solução.

Definem-se três níveis de customização, como explicado a seguir.

Nível 1

É o método mais simples de customização. Nesse nível, criam-se apenas arquivos de definição de material, definição de problema e condições aplicáveis à geometria, tais como condições de

contorno e carregamentos. Essas informações são recuperadas por arquivos denominados "templates" que possuem extensão ".bas". Esses arquivos são escritos em uma linguagem própria do GiD (CIMME, 2008). São empregados ainda para escrever os arquivos de saída do GiD nos formatos dos arquivos de entrada do programa. O GiD já vem com alguns arquivos templates para conversão para os solvers comerciais mais utilizados no mercado, como por exemplo, Nastran e Ansys.

Nível 2

Aqui, a customização é via programação *TCL* (WELCH, 1999) em conjunto com os templates. Essa combinação permite que sejam construídas rotinas mais avançadas para recuperação e manipulação, principalmente, de dados da topologia da geometria, relacionamentos entre as entidades geométricas e as entidades de malha. Essas informações são muito utilizadas para conversão de carregamentos distribuídos aplicados sobre a geometria em carregamentos nodais, e também, na entrada de dados para o hp^2fem.

Existe, contudo, mais um nível de customização do GiD, pois, até o momento, não se pode ter controle sobre a aparência do software tal como disposição dos menus, conteúdos das janelas de diálogos e influência sobre os eventos de seleção.

Nível 3

Neste nível de customização, tanto a aparência, disposição e conteúdo de menus e janelas são alterados, além de se obter maior controle sobre eventos de seleção usando mouse e diálogos das seleções realizadas. Para tanto, usa-se da linguagem *TK* (WELCH, 1999) que é responsável pelas alterações visuais do programa. É possível construir um outro software como por exemplo o *CompassFEM 8.0.8 R3* da empresa Compass Ingeniería y sistemas (COMPASS, 2008).

7.6.2 Organização dos Arquivos do Programa GiD

Até agora, consideraram-se os aspectos da customização, mas não sobre o significado e organização dos arquivos, de modo que o GiD tenha acesso e funcione com um pré e pos-processador personalizado. O GiD é composto de arquivos cada qual com uma funcionalidade

específica. Todos esses arquivos estão dentro de um diretório específico, denominado aqui de **problem_type_hpfem.gid**, e possuem o mesmo nome do diretório, isto é, *problem_type_hpfem.xxx*, sendo *xxx* a extensão do arquivo com a respectiva funcionalidade. As extensões e as funcionalidades dos arquivos são definidas como:

mat : contêm as definições de materiais do modelo. Por enquanto, estão disponíveis apenas informações para materiais elásticos, como módulo de elasticidade, coeficiente de Poison, coeficiente de expansão térmica e densidade. As propriedades do material são adicionadas diretamente à entidade geométrica. Apenas depois do processo de geração de malha, estas propriedades são transferidas para os nós e elementos. O conteúdo deste arquivo está disponível no Apêndice A.2;

prb : arquivo contendo as opções para parâmetros que regem o modelo de elementos finitos, tais como título da simulação, quantidade e tipo de graus de liberdade, parâmetros de solução numérica, parâmetros do algoritmo de otimização, tipo de problema, cinemática e dependência do tempo. No Apêndice A.2, encontram-se as definições da customização;

cnd : nesse arquivo são colocadas todas as possíveis condições aplicáveis ao modelo. Todas elas são aplicadas às entidades geométricas selecionadas pelo usuário. O GiD permite que duas palavras chaves sejam definidas para informar qual tipo de entidade geométrica será aplicada determinada condição, e depois, qual entidade de malha essas condições serão transferidas. Essas palavras chaves são `CONDTYPE` e `CONDMESHTYPE`, respectivamente. Maiores informações estão no Apêndice A.2;

geo : arquivo onde está definida a geometria, isto é, todas as informações topológicas das entidades geométricas estão escritas nesse arquivo.

tcl : contêm rotinas em código *TCL* que possibilita obter informações sobre condições aplicadas sobre geometria e quais entidades de malha compartilham dessa informação.

bas : os arquivos com extensão ".bas", também denominados de "templates", são responsáveis pela conexão direta de dados entre o GiD e o programa. Estes arquivos funcionam como conversores entre padrões. No presente trabalho, têm-se os seguintes arquivos templates: FEM.bas; LOD.bas; DEF.bas; e BAT.bas, e todos geram arquivos de saída no formato ASCII. As funcionalidades desses arquivos são:

- **FEM.bas**: escreve o arquivo de malha no formato FEM, que são informações sobre dimensão do problema, número de nós, coordenadas dos nós, número de elementos, tipo de elemento e incidência dos elementos;
- **DEF.bas**: gera o arquivo contendo as definições do problema tais como número e tipo do grau de liberdade, algoritmo de solução do sistema de equação, precisão da solução, cinemática do problema, tipo do problema e condições de contorno. A extensão de saída deste arquivo é ".def";
- **BAT.bas**: gera um arquivo de comandos capaz de refazer todo o projeto de otimização. Esse arquivo batch contêm todas as definições do projeto que permite ao GiD regerar todo o projeto, sem a necessidade de iniciar a interface gráfica e repetir todas as seleções feitas anteriormente. Isto é necessário, pois permite regerar a malha quando o processo de otimização requerer, além de facilitar o preparo de arquivos de exemplo e garantir a portabilidade.
- **OPT.bas**: agrupa as informações sobre a parametrização das variáveis de projeto, funções objetivos, restrições de igualdade e desigualdade e parâmetros usados no algoritmo de otimização;

Para entender melhor o processo de customização para a criação de um conjunto próprio de dados, a Figura 7.19 mostra a árvore de arquivos do GiD usada neste trabalho.

Todos estes arquivos, anteriormente mencionados, são sempre invocados pelo GiD quando deseja-se gerar as definições de um modelo de elementos finitos. A seguir, exemplifica o uso de programa para otimização de forma.

7.6.3 Funcionamento do Módulo de Otimização

Em um processo de otimização é necessário que tanto o programa de otimização quanto o programa de construção de geometria e geração automática de malha possuam conexão automática para a troca de dados.

O primeiro passo é criar ou importar uma geometria que será otimizada para dentro do pro-

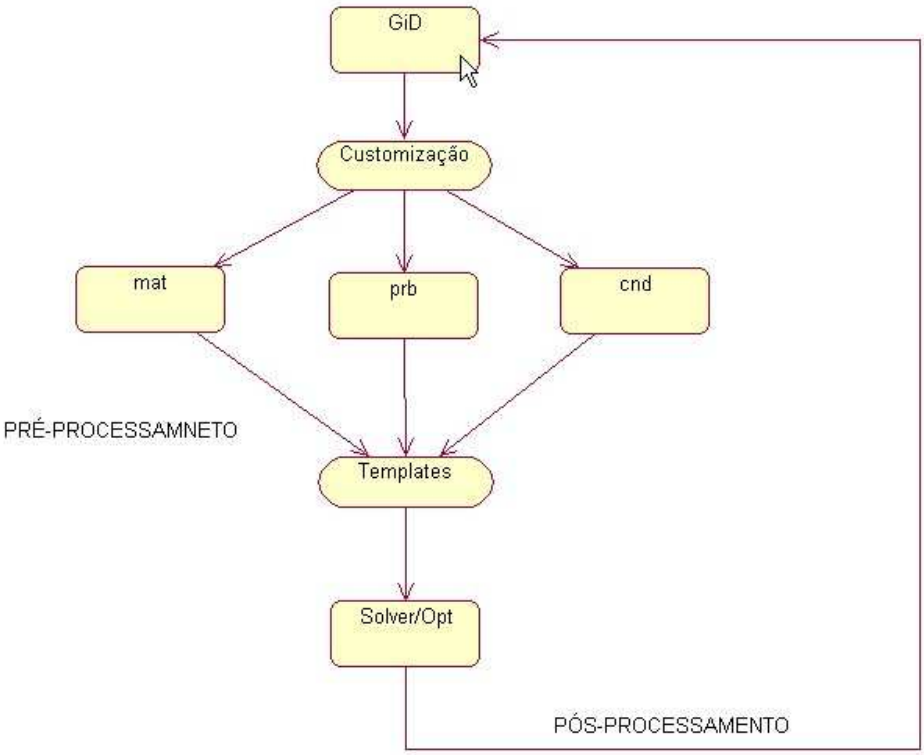


Figura 7.19: Árvore de arquivos para a customização do GiD.

grama GiD. A seguir, identificam-se as possíveis regiões a serem otimizadas, isto é, escolhem-se as variáveis de projeto que devem ser necessariamente parametrizadas usando curvas e superfícies NURBS. Caso essas regiões não sejam parametrizadas, o GiD permite converter qualquer entidade para uma entidade NURBS como mostrado em CIMME (2008). A seguir aplicam-se todas as condições de contorno, carregamentos , parâmetros de solução e otimização e em seguida via "templates"são gerados os arquivos de entrada para o programa de otimização.

Os conteúdos dos arquivos “.def” e “.fem” já foram discutidos no relatório anterior, tornando assim necessário para esse relatório a discussão sobre o arquivo “.opt”. Esse arquivo contém as seguintes informações:

Inicia-se o programa de otimização escolhendo o nome do projeto e dispara-se o processo. O programa irá checar a validade dos arquivos de entrada e em caso de erro avisa ao usuário e aborta o programa. Caso os arquivos estejam corretos, o processo de otimização é iniciado e se for necessário regerar a malha, o programa GiD é invocado e a malha é gerada e devolvida para o processo.

É importante lembrar que uma vez iniciado o procedimento de otimização, todo o procedimento de comunicação entre o otimizador e o GiD é feito de forma transparente baseado na troca de arquivos ASCII.

Quando se atinge o critério de convergência da otimização, o programa interrompe as iterações e escreve a solução em um arquivo do banco de dados que é recuperado com o auxílio de um programa, chamado `ReadOpt`.

A Figura 7.20 ilustra as relações de dependências entre as componentes que atuam no ambiente de otimização proposto.

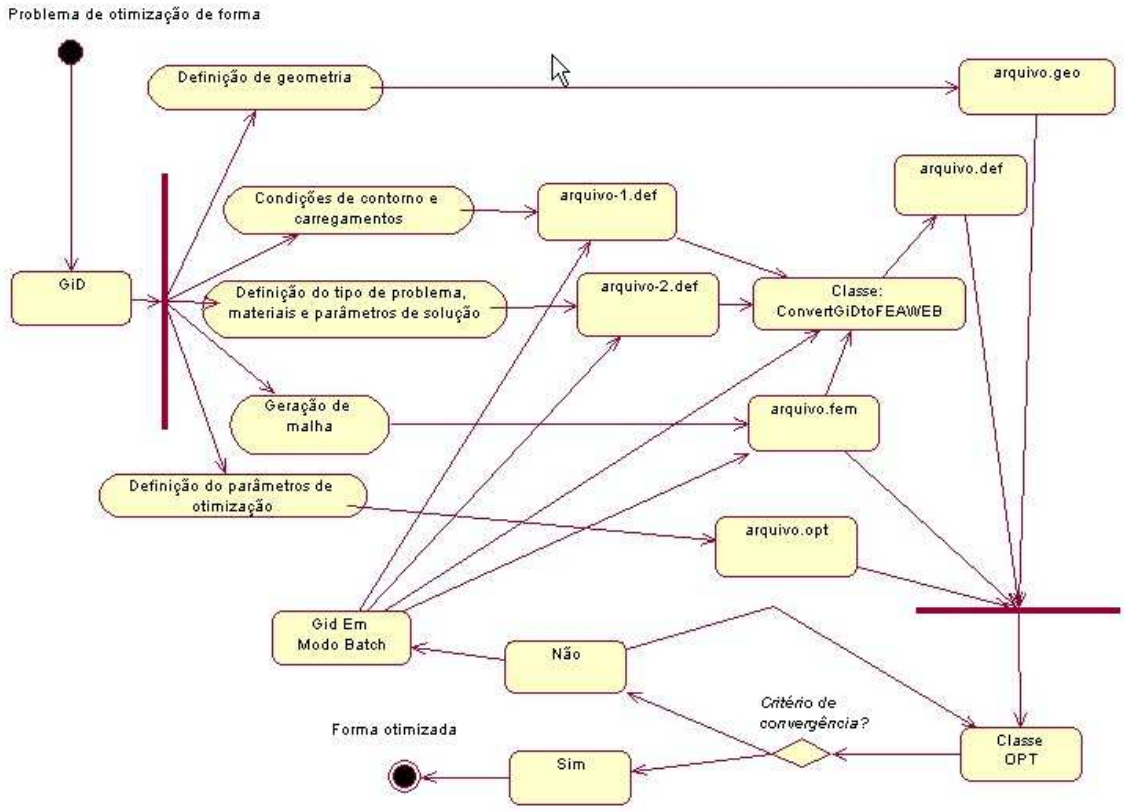


Figura 7.20: Diagrama esquemático do funcionamento do pacote de otimização usando o GiD como pré-processador

8 Resultados e Discussão

Para validação do programa de elementos finitos hp^2fem para problemas estruturais são apresentados sies ciclos de testes.

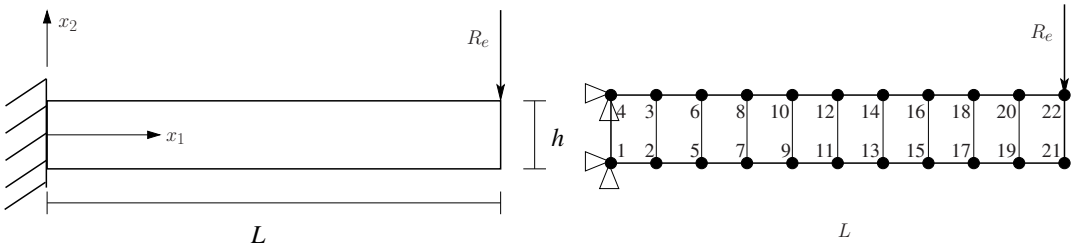
O primeiro ciclo testa os modelos 2D e 3D usando funções de interpolação lineares. Os resultados de deslocamento e tensão obtidos serão comparados com o programa comercial ANSYS e o programa escrito em fortran FLagSHyP (BONET; WOOD, 2008). O segundo ciclo de teste compara os resultados obtidos usando a formulação Lagrangiana total. No terceiro ciclo, os critérios de convergência serão analisados. Já o quarto ciclo verificará os modelos 2D e 3D usando elementos finitos de alta ordem e os resultados obtidos serão comparados com a resposta analítica. O quinto ciclo abordará o problema de otimização de forma estrutural. E finalmente, o problema de contato será bordado no sexto ciclo de teste.

8.1 Problemas de Grandes Deformações

8.1.1 Teste 1- Viga engastada submetida a um carregamento concentrado

Seja a viga representada pela Figura 8.1(a), tendo comprimento $L = 10$m e seção retangular com altura $h = 0,5$m e largura $t = 1$m. Ela está submetida a uma carga concentrada $R_e = 10$N atuando na extremidade direita. Esse carregamento é aplicado em 10 passos de carregamentos. Já a extremidade esquerda está engastada. O modelo de elementos finitos consiste em 10 quadrados isoparamétricos para estado plano de tensão de grau 1, usando 2×2 pontos de integração de Gauss, como mostrado na Figura 8.1(b). Utiliza-se o modelo de material neo-Hookeano compressível e as propriedades são $E = 1000$Pa e $\nu = 0,25$.

Para validar o programa hp^2fem, comparam-se os valores dos deslocamentos do nó 21 com os resultados obtidos usando os programas Ansys e o FlagSHyP. Como critério de convergência, adotou-se aquele da equação (4.128), sendo a tolerância $\epsilon_f = 10^{-6}$.



(a) Viga sujeita a um carregamento concentrado.

(b) Malha de elementos finitos para a viga sujeita a um carregamento concentrado na extremidade direita.

Figura 8.1: Modelo do teste 1.

A Tabela 8.1 mostra os resultados obtidos para o deslocamento do nó 21 para os 10 passos de carregamento. No programa hp^2fem foi usada a formulação Lagrangiana total e o programa FLagShyP utiliza a formulação Lagrangiana atualizada, não abordada nesta tese ma que pode ser vista em Bonet e Wood (2008). Já o Ansys não disponibiliza nenhuma informação quanto à formulação empregada.

Tabela 8.1: Validação do algoritmo de Newton-Raphson.

Passos	hp^2fem		FLagSHyP		Ansys	
	u_1 [m]	u_2 [m]	u_1 [m]	u_2 [m]	u_1 [m]	u_2 [m]
1	-0,1199	-1,1333	-0,1200	-1,1330	-0,1195	-1,1315
2	-0,3761	-2,1885	-0,3760	-2,1880	-0,3749	-2,1855
3	-0,7247	-3,1101	-0,7250	-3,1100	-0,7227	-3,1069
4	-1,1181	-3,8824	-1,1180	-3,8820	-1,1153	-3,8789
5	-1,5209	-4,5161	-1,5210	-4,5160	-1,5182	-4,5138
6	-1,9116	-5,0327	-1,9120	-5,0330	-1,9090	-5,0314
7	-2,2794	-5,4550	-2,2790	-5,4550	-2,2771	-5,4545
8	-2,6202	-5,8027	-2,6200	-5,8030	-2,6353	-5,8185
9	-2,9334	-6,0915	-2,9330	-6,0920	-2,9467	-6,1047
10	-3,2203	-6,3340	-3,2200	-6,3340	-3,2286	-6,3427

Na Tabela 8.2, agrupou-se os erros relativos entre o programa hp^2fem e os demais programas. Percebe-se por essa tabela que o maior erro relativo presente nos 10 passos de carregamento não foi superior a 0,6%. Dessa forma, conclui-se que o programa desenvolvido apresenta uma solução satisfatória.

Tabela 8.2: Erros relativos dos deslocamentos calculados

Passos	hp^2fem/FLagSHyP		hp^2fem/Ansys	
	Erro u_1 [%]	Erro u_2 [%]	Erro u_1 [%]	Erro u_2 [%]
1	0,08	0,03	0,34	0,16
2	0,03	0,02	0,32	0,14
3	0,05	0,00	0,27	0,10
4	0,01	0,01	0,26	0,09
5	0,01	0,00	0,18	0,05
6	0,02	0,01	0,13	0,03
7	0,02	0,00	0,10	0,01
8	0,01	0,01	0,57	0,27
9	0,01	0,01	0,45	0,22
10	0,01	0,00	0,26	0,14

Usando o GiD como pós-processador dos valores dos deslocamentos nodais obtidos pelo hp^2fem, a configuração deformada final da viga é mostrada na Figura 8.2, juntamente com os valores do deslocamento resultante. Percebe-se que a viga sofreu grandes deslocamentos, resultado do processo de grandes deformações. A Figura 8.3 mostra o gráfico da distribuição do número de

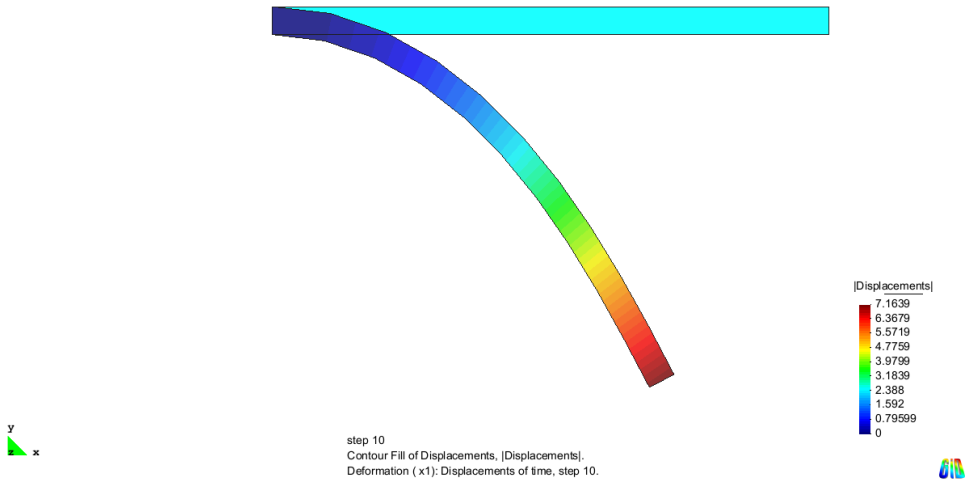


Figura 8.2: Visualização da deformação da viga.

iterações de Newton-Raphson para cada passo de carregamento usado pelos três programas. Como se pode perceber não ocorre uma grande discrepância entre o número de iterações de Newton-Raphson entre os três programas. Com base nesse primeiro teste, pode-se afirmar que o programa

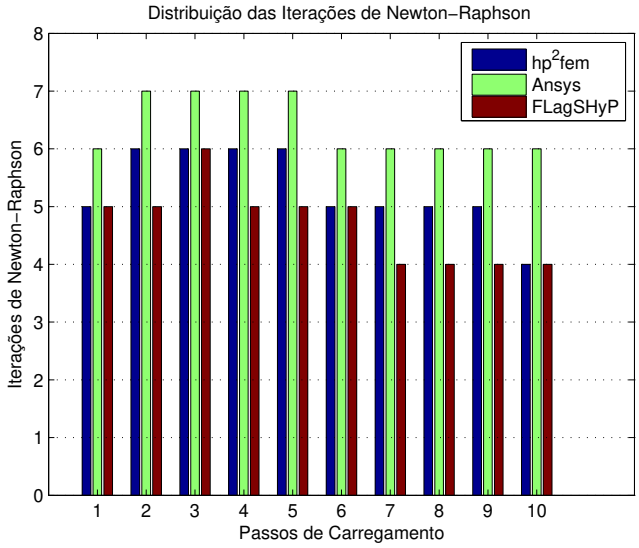
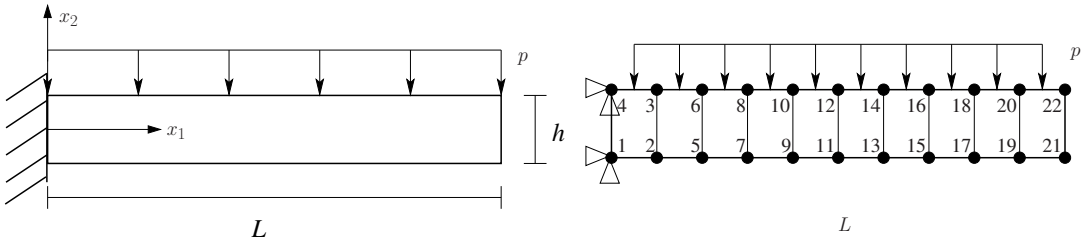


Figura 8.3: Número total de iterações de Newton-Raphson para cada passo de carregamento.

hp^2fem para um problema de grandes deformações usando elemento finito de grau 1 e estado plano de deformação, submetido a um carregamento concentrado, está validado.

8.1.2 Teste 2- Viga engastada submetida à uma pressão uniforme

No teste 2, o interesse está em validar o algoritmo de cálculo do carregamento dependente da deformação. Para isso, utiliza-se uma pressão $p = 1\mathrm{N/m}^2$ atuando na viga ilustrada na Figura 8.4(a). As medidas geométricas, as condições de contorno, as propriedades do material e o modelo de elementos finitos (veja Figura 8.4(b)) são iguais aos do teste 1. Comparam-se os valores dos deslocamentos do nó 21 contra os valores obtidos pelo Ansys e FLagSHyP, calculados para os 10 passos de carregamento.



(a) Viga engastada sujeita a uma pressão uniformemente distribuída.

(b) Malha de elementos finitos para a viga sujeita a uma pressão uniformemente distribuída.

Figura 8.4: Modelo do teste 2.

Tabela 8.3: Comparação dos deslocamentos do nó 21 obtidos no teste 2.

Passos	hp^2fem		FLagSHyP		Ansys	
	u_1 [m]	u_2 [m]	u_1 [m]	u_2 [m]	u_1 [m]	u_2 [m]
1	-0,0249	-0,4306	-0,0250	-0,4306	-0,0249	-0,4305
2	-0,0709	-0,8594	-0,0710	-0,8594	-0,0709	-0,8592
3	-0,1378	-1,2848	-0,1380	-1,2850	-0,1377	-1,2845
4	-0,2253	-1,7054	-0,2250	-1,7050	-0,2252	-1,7051
5	-0,3329	-2,1199	-0,3330	-2,1200	-0,3327	-2,1194
6	-0,4602	-2,5268	-0,4600	-2,5270	-0,4600	-2,5267
7	-0,6063	-2,9248	-0,6060	-2,9250	-0,6061	-2,9250
8	-0,7706	-3,3129	-0,7710	-3,3130	-0,7705	-3,3133
9	-0,9521	-3,6898	-0,9520	-3,6900	-0,9522	-3,6905
10	-1,1501	-4,0544	-1,1500	-4,0540	-1,1503	-4,0557

Na Figura 8.5, mostram-se os valores do deslocamento resultante e a configuração deformada final da viga.

Tabela 8.4: Erro relativo dos deslocamentos do nó 21 obtidos no teste 2.

Passos	hp^2fem/FLagSHyP		hp^2fem/Ansys	
	Erro u_1 [%]	Erro u_2 [%]	Erro u_1 [%]	Erro u_2 [%]
1	0,08	0,03	0,34	0,16
1	0,33	0,01	0,05	0,02
2	0,11	0,00	0,06	0,02
3	0,12	0,02	0,06	0,02
4	0,14	0,02	0,07	0,02
5	0,02	0,01	0,07	0,02
6	0,03	0,01	0,04	0,00
7	0,05	0,01	0,02	0,01
8	0,06	0,00	0,01	0,01
9	0,01	0,01	0,00	0,02
10	0,00	0,01	0,02	0,03

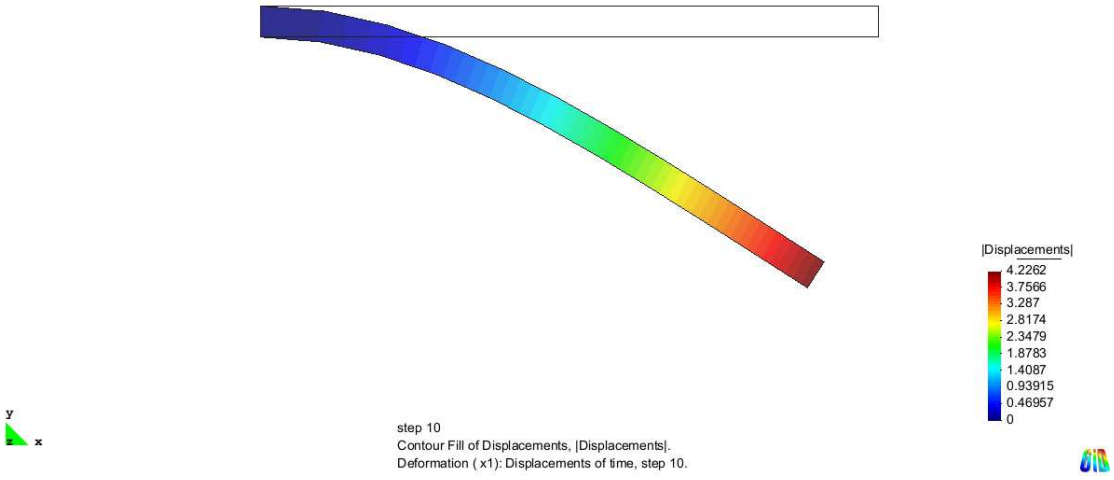


Figura 8.5: Deformação da viga sujeita à um carregamento distribuído.

O programa FLagSHyP usa um procedimento de simetrização da matriz de rigidez[1] da carga de pressão, por essa razão, a solução do sistema de equações é facilitado acarretando em um número menor de iterações. Porém, a matriz não é consistente podendo influenciar no resultado final. Já o programa Ansys não faz menção alguma quanto ao procedimento de cálculo da matriz de rigidez da pressão. A comparação entre os números de iteração para cada passo de carregamento dos três programas analisados é mostrado na Figura 8.6.

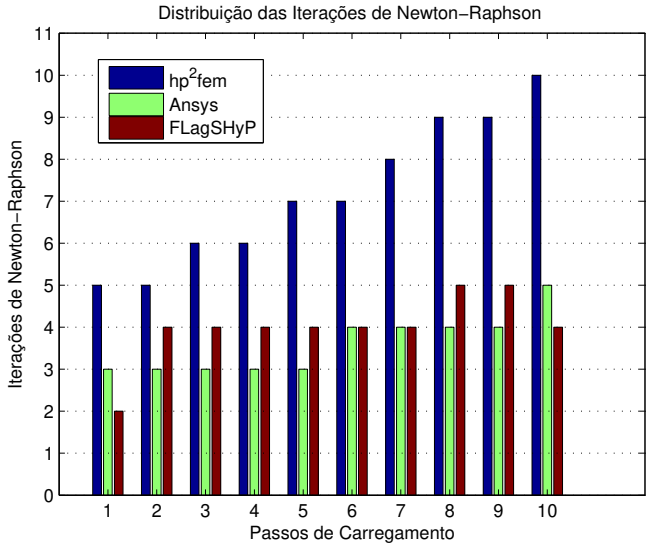
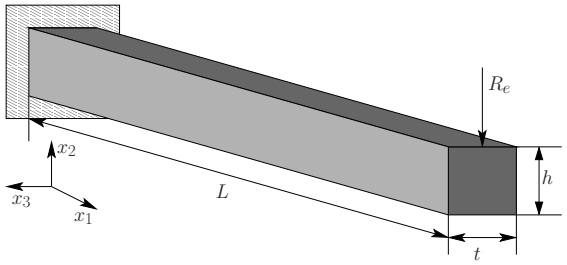


Figura 8.6: Número total de iterações de Newton-Raphson para cada passo de carregamento.

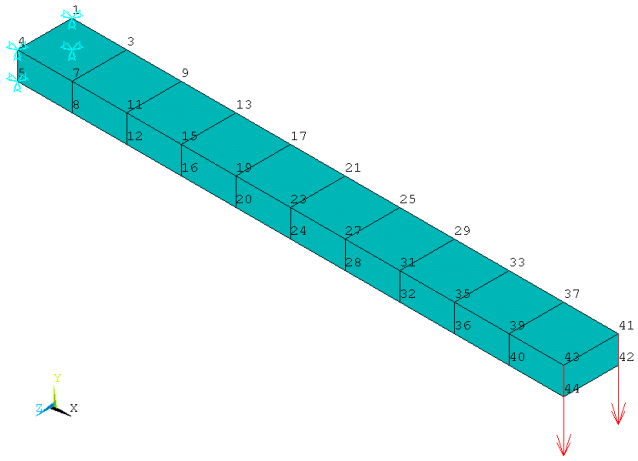
[1](BONET; WOOD, 2008) assume que os pontos que estão no contorno da superfície onde se aplica a pressão são fixos ou prescritos. Desta forma, o deslocamento $\Delta\mathbf{u}$ e o deslocamento virtual $\delta\mathbf{u}$ são nulos no contorno da superfície com condição de pressão aplicada.

8.1.3 Teste 3 - Viga 3D submetida a um carregamento concentrado

Seja a viga tridimensional representada na Figura 8.7(a), submetida a uma força concentrada $R_e = 10$N na extremidade direita. Impõe-se a condição de travamento na extremidade esquerda da viga. A medidas da viga e as propriedades do material neo-Hookeano compressível são: $L = 10$m, $t = 1,0$m, $h = 0,5$m, $E = 10000$Pa e $\nu = 0,25$. Usa-se uma malha de 10 hexaedros de 8 nós (veja Figura 8.7(b)). O carregamento é igualmente distribuído entre os nós 43 e 41. Sendo o nó 42 simétrico ao nó 44, comparam-se apenas os deslocamentos do nó 42 calculado com o programa hp^2fem contra os valores obtidos pelos programas Ansys e FLagSHyP.



(a) Viga engastada sujeita a uma força concentrada.



(b) Malha de elementos finitos para a viga sujeita a uma carga concentrada na extremidade direita.

Figura 8.7: Modelo do teste 3.

A Tabela 8.5 mostra os valores dos deslocamentos nas três direções obtidos pelos três programas. Nota-se que o programa hp^2fem apresentou resultado satisfatório quando comparado aos outros dois programas. Calculando-se o erro relativo da solução obtida pelo hp^2fem, como mostrado na Tabela 8.6, é possível afirmar que o programa desenvolvido hp^2fem está validado para problemas tridimensionais com malha de hexaedros linear.

Tabela 8.5: Comparação dos deslocamentos do nó 42 obtidos no teste 2.

Passos	hp^2fem			FLagSHyP			Ansys		
	u_1 [m]	u_2 [m]	u_3 [m]	u_1 [m]	u_2 [m]	u_3 [m]	u_1 [m]	u_2 [m]	u_3 [m]
1	-0,1237	-1,1541	-3,70E-05	-0,1240	-1,1540	-3,70E-05	-0,1232	-1,1523	-3,69E-05
2	-0,3881	-2,2257	-6,76E-05	-0,3880	-2,2260	-6,76E-05	-0,3868	-2,2226	-6,75E-05
3	-0,7465	-3,1575	-9,04E-05	-0,7460	-3,1570	-9,04E-05	-0,7443	-3,1541	-9,02E-05
4	-1,1489	-3,9341	-1,05E-04	-1,1490	-3,9340	-1,05E-04	-1,1456	-3,9304	-1,05E-04
5	-1,5589	-4,5684	-1,14E-04	-1,5590	-4,5680	-1,14E-04	-1,5558	-4,5662	-1,13E-04
6	-1,9548	-5,0834	-1,17E-04	-1,9550	-5,0830	-1,17E-04	-1,9518	-5,0823	-1,16E-04
7	-2,3262	-5,5031	-1,15E-04	-2,3260	-5,5030	-1,15E-04	-2,3234	-5,5029	-1,15E-04
8	-2,6693	-5,8476	-1,10E-04	-2,6690	-5,8480	-1,10E-04	-2,6668	-5,8482	-1,10E-04
9	-2,9839	-6,1334	-1,03E-04	-2,9840	-6,1330	-1,03E-04	-2,9951	-6,1456	-1,04E-04
10	-3,2714	-6,3729	-9,38E-05	-3,2710	-6,3730	-9,38E-05	-3,2799	-6,3824	-9,36E-05

Tabela 8.6: Erro relativo dos deslocamentos do nó 42 obtidos no teste 3.

Passos	hp^2fem/FLagSHyP			hp^2fem/Ansys		
	Erro u_1 [%]	Erro u_2 [%]	Erro u_3 [%]	Erro u_1 [%]	Erro u_2 [%]	Erro u_3 [%]
1	0,27	0,01	0,00	0,37	0,16	0,14
2	0,04	0,01	0,01	0,35	0,14	0,16
3	0,06	0,01	0,00	0,30	0,11	0,16
4	0,01	0,00	0,04	0,29	0,09	0,21
5	0,01	0,01	0,04	0,20	0,05	0,25
6	0,01	0,01	0,01	0,15	0,02	0,32
7	0,01	0,00	0,00	0,12	0,00	0,42
8	0,01	0,01	0,03	0,09	0,01	0,56
9	0,00	0,01	0,03	0,37	0,20	0,91
10	0,01	0,00	0,00	0,26	0,15	0,15

E pós-processando os dados obtidos do programa hp^2fem, mostra-se-se na Figura 8.8 a comparação das geometrias não-deformada e deformada para a viga analisada.

Para verificar o desempenho do algoritmo de Newton-Raphson, compara-se na Figura 8.8, o número de iterações para cada passo de carregamento entre os três programas.

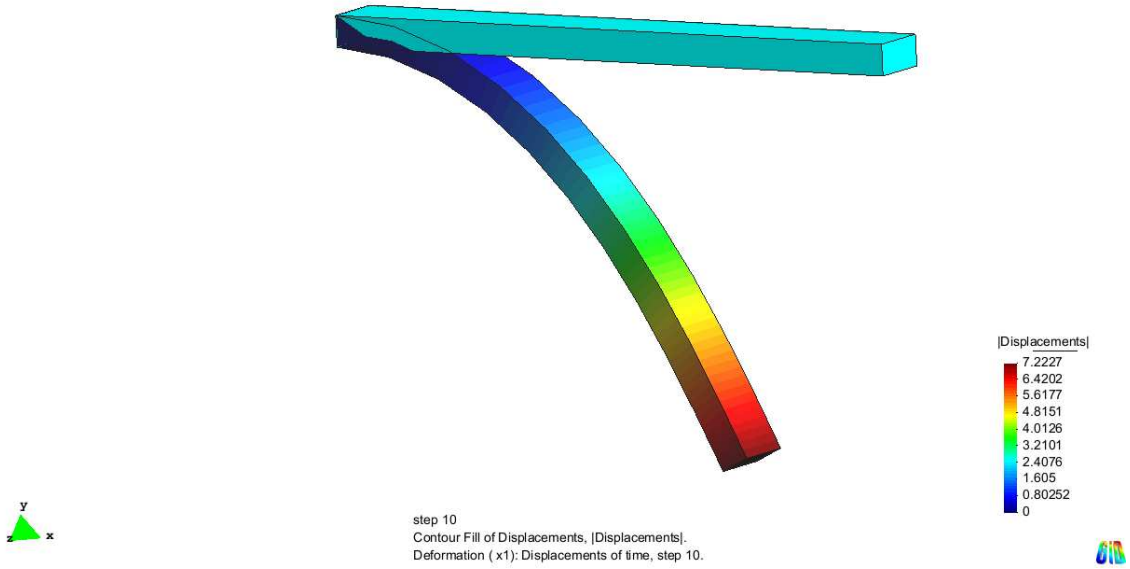


Figura 8.8: Deformação da viga 3D sujeita a um carregamento concentrado.

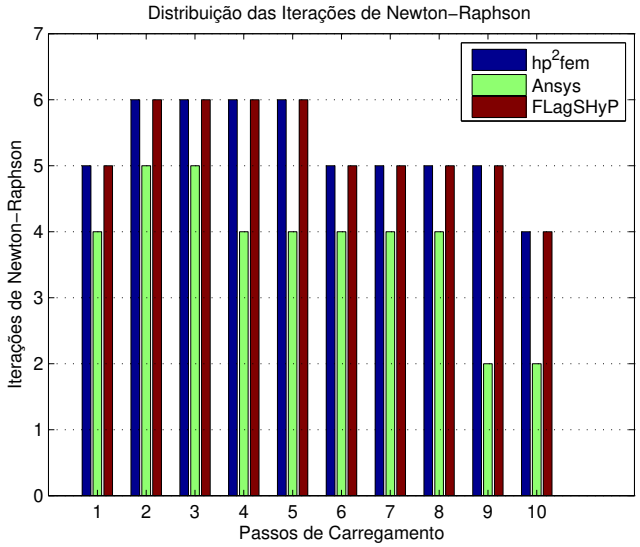
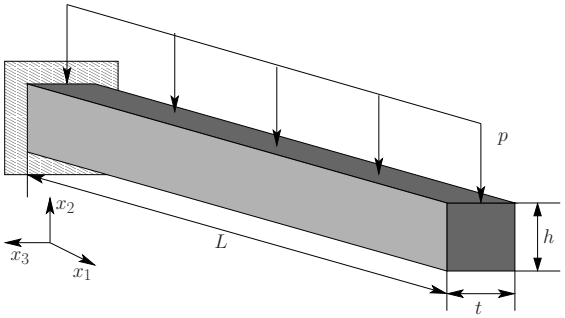


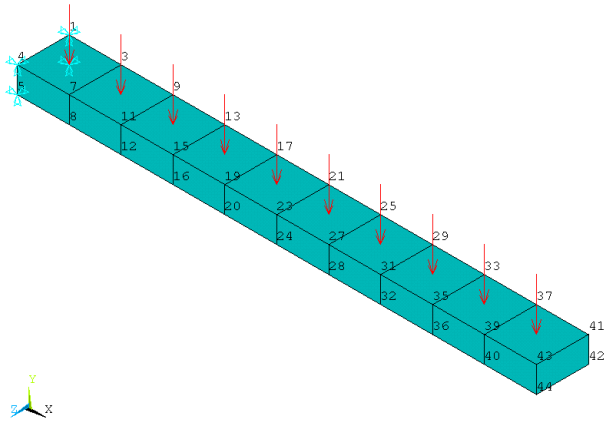
Figura 8.9: Número total de iterações de Newton-Raphson para cada passo de carregamento.

8.1.4 Teste 4 - Viga 3D sujeita a uma pressão uniforme

No teste 4, aplica-se uma pressão uniformemente distribuída $p = 1\mathrm{N/m}^2$ atuando no topo da viga como mostrado na Figura 8.10(a). As medidas geométricas, as condições de contorno, as propriedades do material e o modelo de elementos finitos (veja Figura 8.10(b)) são iguais ao do teste 3. Novamente, comparam-se os os deslocamentos do nó 42 contra os valores obtidos pelo Ansys e FLagSHyP.



(a) Viga engastada sujeita à uma pressão uniformemente distribuída.



(b) Malha de elementos finitos para a viga sujeita à uma pressão uniformemente distribuída.

Figura 8.10: Modelo do teste 4.

Os resultados para os deslocamentos do nó 42 são mostrados na Tabela 8.7. Percebe-se que o resultado obtido pelo programa hp^2fem é muito próximo aos valores calculados pelo Ansys e FLagSHyP. Isso é verificado na Tabela 8.8, onde nota-se que o erro relativo da solução fornecida pelo programa desenvolvido é abaixo de 0,5%.

Tabela 8.7: Comparação dos deslocamentos do nó 42 obtidos no teste 4.

Passos	hp2fem			FLagSHyP			Ansys		
	u_1 [m]	u_2 [m]	u_3 [m]	u_1 [m]	u_2 [m]	u_3 [m]	u_1 [m]	u_2 [m]	u_3 [m]
1	-0,0256	-0,4385	-1,23E-6	-0,0256	-0,4385	-1,23E-6	-0,0256	-0,4384	-1,23E-6
2	-0,0731	-0,8749	-2,46E-6	-0,0730	-0,8749	-2,47E-6	-0,0730	-0,8747	-2,46E-6
3	-0,1422	-1,3077	-3,70E-6	-0,1420	-1,3080	-3,70E-6	-0,1420	-1,3074	-3,70E-6
4	-0,2326	-1,7355	-4,93E-6	-0,2330	-1,7350	-4,93E-6	-0,2324	-1,7350	-4,93E-6
5	-0,3438	-2,1566	-6,16E-6	-0,3440	-2,1570	-6,16E-6	-0,3436	-2,1564	-6,16E-6
6	-0,4753	-2,5698	-7,39E-6	-0,4750	-2,5700	-7,39E-6	-0,4750	-2,5696	-7,39E-6
7	-0,6263	-2,9736	-8,63E-6	-0,6260	-2,9740	-8,63E-6	-0,6259	-2,9735	-8,63E-6
8	-0,7959	-3,3668	-9,86E-6	-0,7960	-3,3670	-9,86E-6	-0,7955	-3,3670	-9,86E-6
9	-0,9833	-3,7482	-1,11E-5	-0,9830	-3,7480	-1,11E-5	-0,9829	-3,7487	-1,11E-5
10	-1,1874	-4,1168	-1,23E-5	-1,1870	-4,1170	-1,23E-5	-1,1871	-4,1176	-1,23E-5

Tabela 8.8: Erro relativo dos deslocamentos do nó 42 obtidos no teste 4.

Passos	hp^2fem/FLagSHyP			hp^2fem/Ansys		
	Erro u_1 [%]	Erro u_2 [%]	Erro u_3 [%]	Erro u_1 [%]	Erro u_2 [%]	Erro u_3 [%]
1	0,04	0,01	0,03	0,06	0,02	0,00
2	0,07	0,00	0,01	0,08	0,02	0,02
3	0,12	0,02	0,00	0,09	0,02	0,02
4	0,17	0,03	0,01	0,09	0,02	0,01
5	0,04	0,02	0,00	0,07	0,01	0,01
6	0,07	0,01	0,01	0,07	0,01	0,01
7	0,05	0,01	0,00	0,06	0,00	0,01
8	0,01	0,01	0,00	0,05	0,01	0,02
9	0,03	0,01	0,02	0,04	0,01	0,02
10	0,04	0,01	0,04	0,03	0,02	0,03

Como feito anteriormente, a geometria deformada final é mostrada na Figura 8.11, juntamente com a distribuição do deslocamento nodal resultante.

A performance do algoritmo de Newton-Raphson implementado no hp^2fem é comparado aos outros programas analisados na Figura 8.12. Percebe-se que o hp^2fem apresentou uma performance

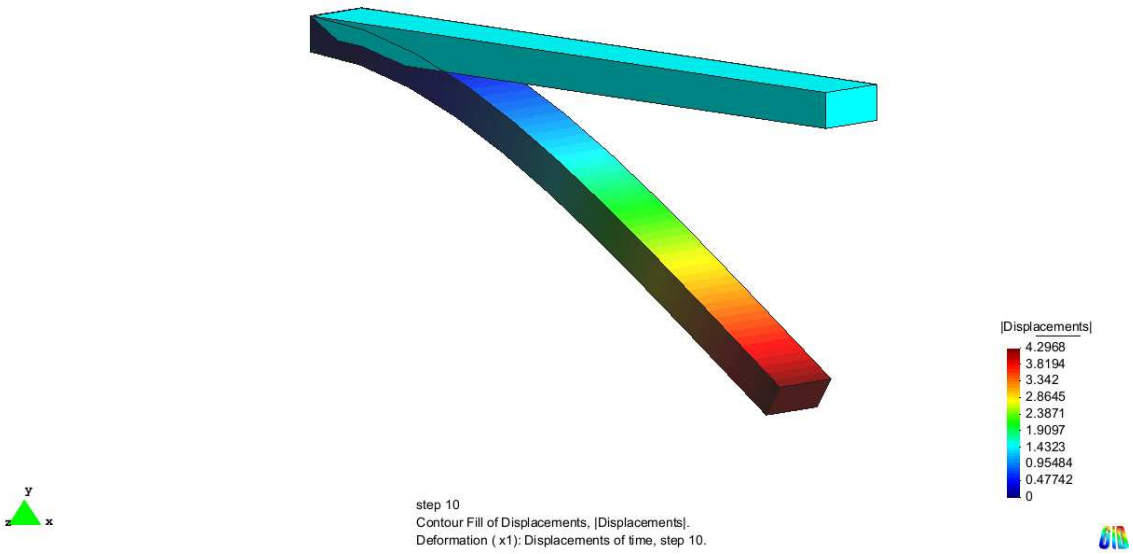


Figura 8.11: Visualização da deformação da viga 3D sujeita à um carregamento distribuído.

boa para solução do problema proposto.

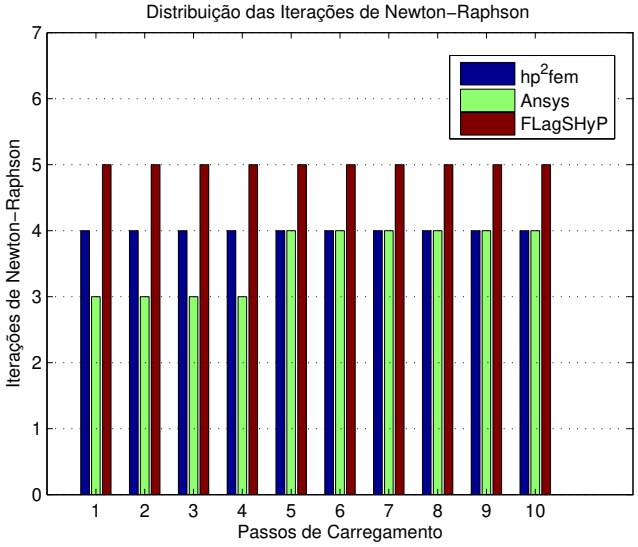


Figura 8.12: Número total de iterações de Newton-Raphson para cada passo de carregamento.

8.1.5 Teste 5 - Estudo da convergência do MEF-AO

Neste teste, o interesse está em aplicar o MEF-AO em problemas de grandes deformações. Para isso, realiza-se o estudo de convergência proposto por (DONG; YOSIBASH, 2009). Baseando-se em um problema fictício cuja solução analítica é conhecida, estuda-se a consistência do MEF-AO e o comportamento do erro entre as soluções aproximada e analítica. As bases de comparação dos erros são as normas L^2 e L^∞ da diferença apresentada entre as duas soluções.

O problema estudado consiste em um cubo que inicialmente ocupa um domínio $0 \leq X_1 \leq 1$, $0 \leq X_2 \leq 1$ e $0 \leq X_3 \leq 1$, conforme mostrado na Figura 8.13. As propriedades do material são módulo de elasticidade E e coeficiente de Poisson ν. A equação constitutiva para o material Neo-Hookeano é dado por (4.54). Sabe-se que a face $X_1 = 0$ está engastada e que nas outras faces aplica-se o campo de tração $\mathbf{t} = (t_1, t_2, t_3)$ não dependente da deformação dado por

$$\begin{cases} t_1 = n_1 \left\{ Aa^2 \cos(aX_1)\mu - \dfrac{\mu - \lambda \ln\left[Aa^2 \cos(aX_1)\right]}{Aa^2 \cos(aX_1)} \right\}, \\ t_2 = n_2 \left\{ \lambda \ln[Aa^2 \cos(aX_1)] \right\}, \\ t_3 = n_3 \left\{ \lambda \ln[Aa^2 \cos(aX_1)] \right\}, \end{cases} \tag{8.1}$$

sendo $\mathbf{n} = n_1, n_2, n_3$ o vetor unitário normal ao plano da superfície tracionada na configuração inicial. As constantes A e a são utilizadas para proporcionar consistência de unidades do problema. Já λ e ν são os coeficientes de Lamé apresentados nas equações (4.53a) e (4.53b). Aplica-se, em conjunto, uma força de corpo $\rho\mathbf{f} = (f_1, f_2, f_3)$ dada por

$$\begin{cases} f_1 = -\dfrac{\sin(aX_1)\left\{\lambda + \mu - \lambda \ln[A(1 - 2\sin^2(\frac{aX_1}{2}))] - \lambda \ln(a^2) + \mu A^2 a^4 \cos^2(aX_1)\right\}}{-Aa\cos^2(aX_1)}, \\ f_2 = 0, \\ f_3 = 0. \end{cases} \tag{8.2}$$

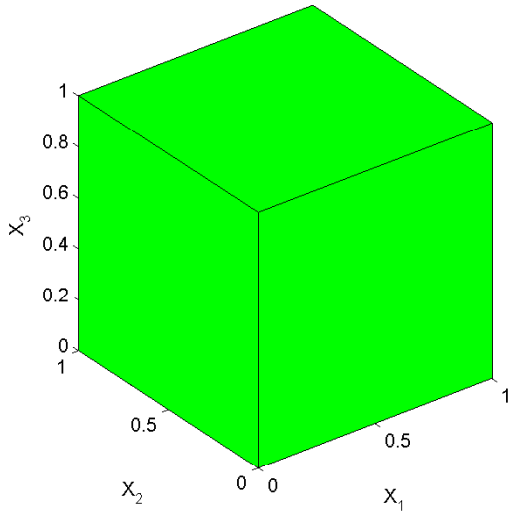


Figura 8.13: Hexaedro usado no teste 6.

A solução analítica para esse problema é

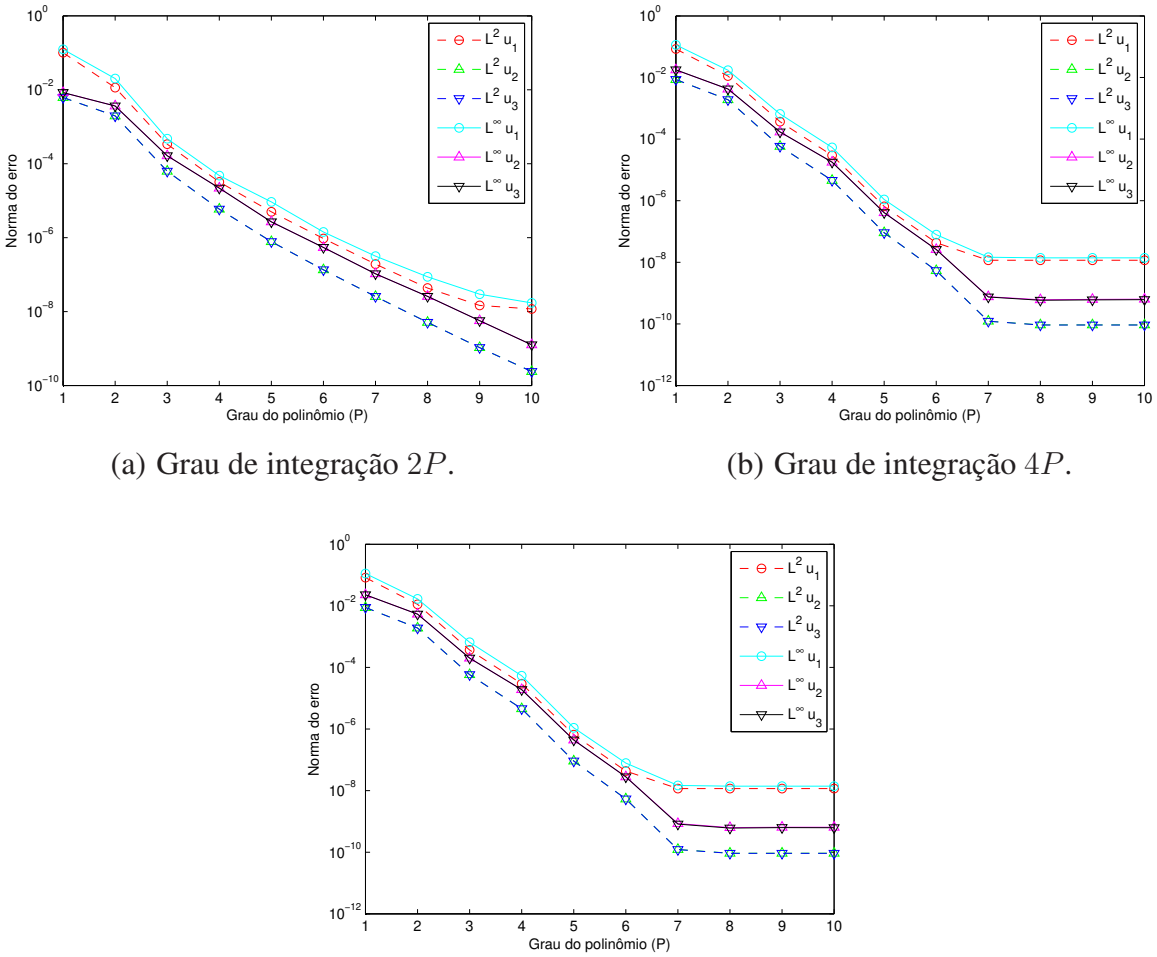
$$\begin{cases} u_1 = A\sin(a\,X_1) - X_1, \\ u_2 = 0,0, \\ u_3 = 0,0, \end{cases} \tag{8.3}$$

As funções de forma para o elemento são da família de Lagrange, utilizando-se de pontos de colocação e integração seguindo os esquemas de Gauss-Lobatto-Legendre e Gauss-Legendre, respectivamente, estudadas e implementadas em Bargos (2009), Vazquez (2008).

Para mostrar o decaimento da norma do erro, utilizou-se um elemento hexaedro e variou-se a ordem desse elemento de 1 à 10. Para cada ordem do elemento, computou-se as normas L^2 e L^∞ do erro entre a solução analítica dada por (8.3) com a solução aproximada calculada usando (4.118). As Figuras 8.14 apresentam os resultados para a convergência espacial das normas L^2 e L^∞ do erro para o problema, utilizando-se os seguintes valores para as constantes apresentadas anteriormente

$$A = 1,9m^{-1}, \;\; a = 1,0m, \;\; E = 1000Pa, \;\; \nu = 0,3$$

e variando a ordem de integração de 2P a 6P, sendo P a ordem da função de forma.



(a) Grau de integração $2P$.

(b) Grau de integração $4P$.

(c) Grau de integração $6P$.

Figura 8.14: Convergência espacial para as normas L^2 e L^∞ do erro para o cubo do teste 6. As normas estão em função da ordem do elemento.

Nota-se que o aumento do grau do polinômio de interpolação permite que o erro diminua consideravelmente. Porém, percebe-se, também, que o aumento demasiado do número de pontos de integração não possibilita uma melhor taxa de convergência. Pelas Figuras 8.14(b) e 8.14(c), é possível afirmar que não existe grandes ganhos com o uso da ordem de integração $6P$ se comparado ao caso $4P$ e que o uso de grau de interpolação maior que 7 não produz aumento da convergência espacial. Dessa forma, pelos custos computacionais (tempo e alocação de memória), prefere-se o

uso de ordem de integração $2P$.

Este estudo foi realizado para testar a convergência do método utilizando uma condição de carregamento não polinomial.

8.1.6 Teste 6

O teste 6 é usado para comparar os resultados obtidos entre os refinamentos p e o h para o problema de solução teórica apresentado no teste 8.1.5. Procurou-se fazer uma correspondência entre o grau das funções de forma utilizadas e o número de graus de liberdade. Por exemplo, uma malha de 1 hexaedro de grau 2 possui o mesmo número de nós e, consequentemente, a mesma quantidade de graus de liberdade que uma malha de 8 hexaedros de grau 1. Dessa forma, com o auxílio da Tabela 8.9 é possível construir uma relação entre a malha de grau 1 e a sua correspondente de 1 elemento de grau P.

Tabela 8.9: Equivalências em termos de número de nós e graus de liberdade para uma malha de hexaedro de grau $P > 1$ e uma malha de grau 1 usando polinômios de Lagrange.

Grau P da função de forma	Número de elementos	Número total de nós	Número total de graus de liberdade	Número equivalente de elementos de grau 1
2	1	27	81	8
4	1	125	375	64
6	1	343	1029	216
8	1	729	1536	512
10	1	1331	3993	1000

A Figura 8.15 mostra a convergência baseada no refinamento h. Percebe-se que comparando a Figura 8.15 com as Figuras 8.14(a) à 8.14(c), a convergência apresentada pelo aumento do grau da função de interpolação possui um efeito maior na convergência espacial. O aumento do grau de integração e o refinamento h possuem efeitos secundários nesse caso.

Como este problema possui solução suave, é esperado que a versão-p do MEF possua uma taxa de convergência mais alta (convergência exponencial) que a do refinamento h (convergência algebrica). Isto é facilmente verificado ao compara-se os resultados do teste 5 (refinamento p) como

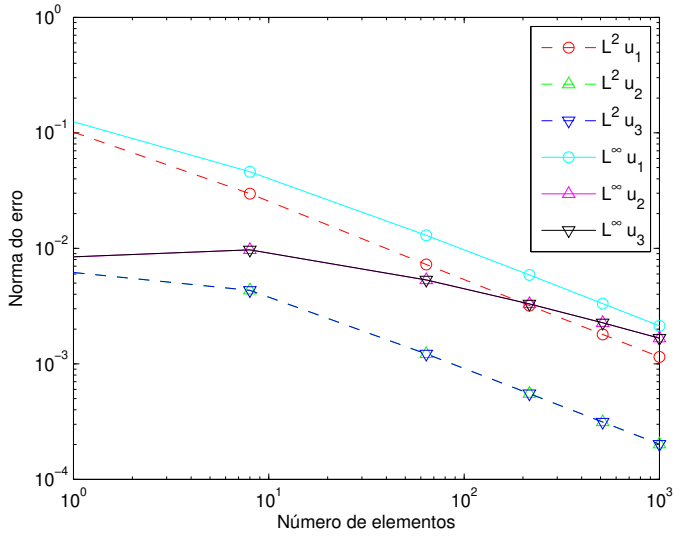


Figura 8.15: Aplicação do refinamento h para convergência espacial das normas L^2 e L^∞ do erro para o mesmo cubo do teste 5. As normas estão em função do número de elementos.

o teste 6 (refinamento h).

8.1.7 Teste 7 - Teste de convergência em refinamento hp

Neste teste, utiliza-se do mesmo problema definido no Teste 5, porém verifica-se a convergência não somente quanto ao refinamento p, mas também utilizando o refinamento hp. Para isso, utilizam-se quatro malhas 1, 8, 64 e 216 de elementos. O objetivo é verificar qual malha atinge uma convergência L^2 superior a 10^{-8} utilizando um grau menor de função de interpolação. A Tabela 8.10 mostra os resultados obtidos.

Tabela 8.10: Convergência usando refinamento hp para o teste 7.

Número de Elementos	Grau P da Função	Número de Equações	Norma L^2			Norma L^∞		
			u_1	u_2	u_3	u_1	u_2	u_3
1	10	1331	1,78E-08	2,65E-08	4,99E-09	2,31E-08	2,31E-08	2,31E-08
8	7	3375	1,16E-08	1,26E-10	1,26E-10	1,56E-08	7,18E-10	7,18E-10
64	4	4913	2,31E-08	2,23E-09	2,23E-09	3,90E-08	1,30E-08	1,30E-08
216	3	20577	2,10E-07	2,82E-08	2,82E-08	5,05E-07	2,15E-07	2,15E-07

Do teste 6, percebe-se que o refinamento h não melhora significativamente a taxa de conver-

gência para as normas adotadas. Por outro lado, o teste 7 mostrou que ao utilizar-se do refinamento hp é possível obter uma convergência excelente para um grau menor porém usando um maior número de elementos. Dentre as três malhas proposta, pode-se afirmar que a malha de 64 elementos é a que possui uma melhor relação número de equações e número de elementos.

8.2 Problemas de Otimização de Forma

A biela mostrada na Figura 8.16 foi utilizada para validar o programa junto com o programa GiD. O problema de otimização consiste em otimizar o volume da biela sujeita a uma máxima energia de deformação de 0,2 kNcm. Os nós da superfície A estão engastados e aplicou-se um carregamento distribuído na superfície B como ilustrado na Figura 8.16.

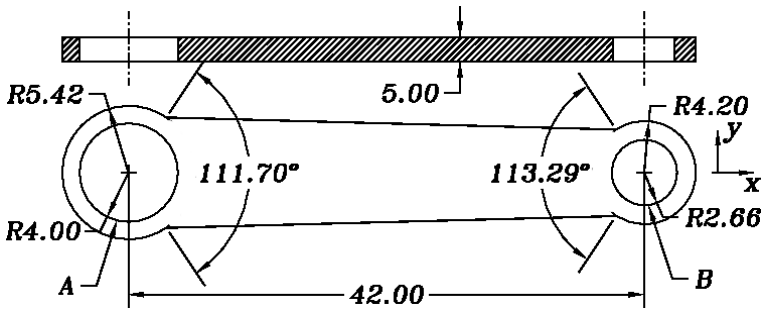


Figura 8.16: Dimensões [cm] do projeto inicial ($E = 21,0 \times 10^3$ kN/cm^2, $\nu = 0,3$, $\rho = 7,81 \times 10^{-3}$kg/cm^3). Os nós da superfície A estão engastados. A força distribuída ($F_x = 0,12$ kN/cm^2, $F_y = 0,06$ kN/cm^2, $F_z = 0,04$ kN/cm^2) é aplicada na superfície B (SILVA, 2003).

As superfícies 21 e 23, mostradas na Figura 8.17, foram parametrizadas como superfícies NURBS, e as curvas de número 1, 2, 15 e 16 são curvas NURBS. A função objetivo é o volume da biela e as variáveis de projeto estão definidas na Tabela 8.11 e numeradas de 1 a 20 como mostrado na Figura 8.17. O volume inicial é $1764.48cm^3$ com uma tensão máxima de von Mises de $5,036kN/cm^2$ como mostrado na Figura 8.18.

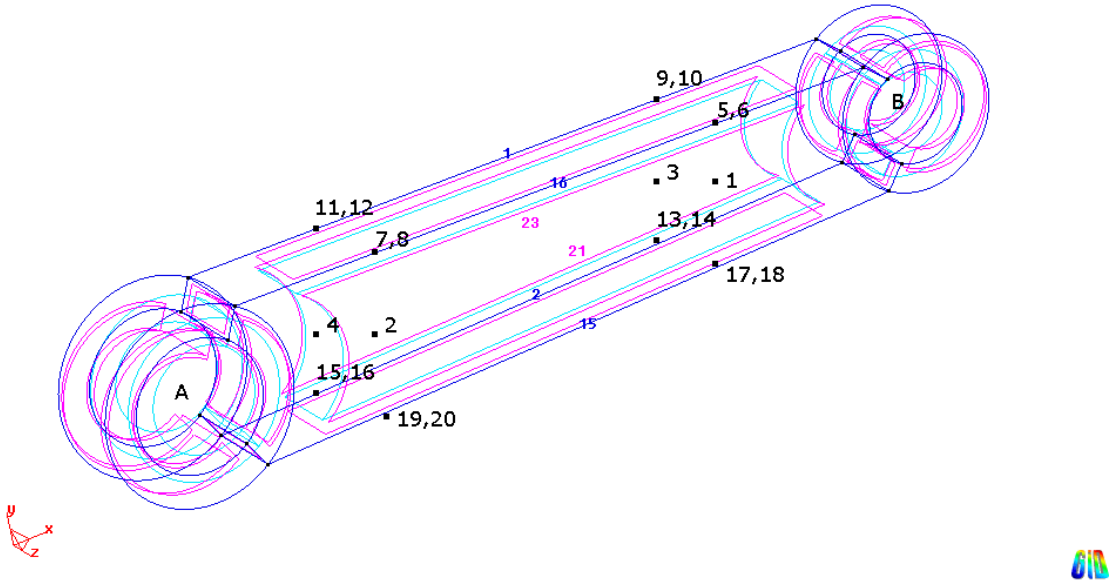


Figura 8.17: Variáveis de projeto (em negrito de 1 a 20). O objetivo é minimizar o volume com restrição na máxima energia de deformação (0,2 kNcm).

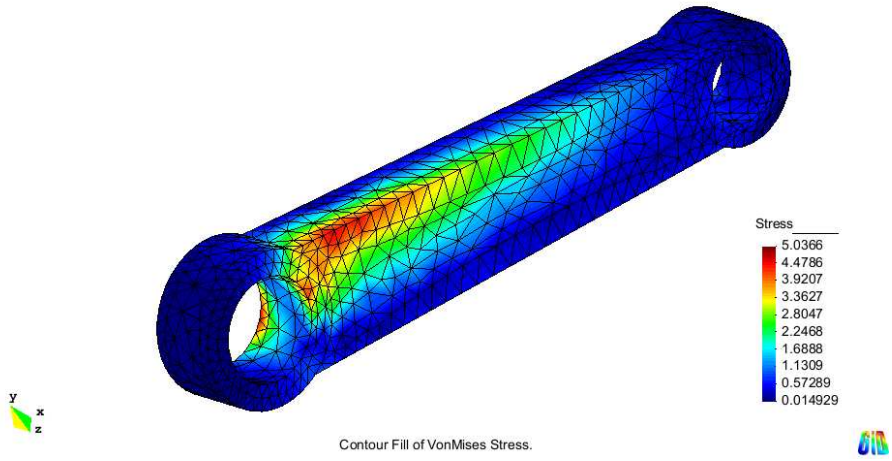


Figura 8.18: Diagrama de tensão equivalente de von Mises atuante na biela antes do processo de otimização. Unidade em [kN/cm^2].

Tabela 8.11: Variáveis de projeto usadas na otimização da biela definida na Figura 8.16.

Variável de projeto	Entidade geométrica/ID	Ponto de controle	Direção de controle	Valor inicial [cm]	Limite inferior [cm]	Limite superior [cm]
1	Superfície 21	(2,4)	Z	5,00	2,50	5,50
2	Superfície 21	(3,4)	Z	5,00	2,50	5,50
3	Superfície 23	(2,4)	Z	0,00	-0,50	2,50
4	Superfície 23	(3,4)	Z	0,00	-0,50	2,50
5	Curva 16	2	Y	15,24	13,60	15,50
6	Curva 16	2	Z	5,00	4,00	5,50
7	Curva 16	3	Y	14,75	13,60	15,50
8	Curva 16	3	Z	5,00	4,00	5,50
9	Curva 1	2	Y	15,24	13,60	15,50
10	Curva 1	2	Z	0,00	-0,50	1,00
11	Curva 1	3	Y	14,75	13,60	15,50
12	Curva 1	3	Z	0,00	-0,50	1,00
13	Curva 2	3	Y	6,76	6,00	8,40
14	Curva 2	3	Z	0,00	-0,50	1,00
15	Curva 2	2	Y	7,25	6,00	8,40
16	Curva 2	2	Z	0,00	-0,50	1,00
17	Curva 15	3	Y	6,76	6,00	8,40
18	Curva 15	3	Z	5,00	4,00	5,50
19	Curva 15	2	Y	7,25	6,00	8,40
20	Curva 15	2	Z	5,00	4,00	5,50

8.2.1 Otimização de Forma usando o Método da Camada unitária de Contorno

Para validar a conexão entre os programa e o GiD , empregou-se o método da camada unitária de contorno, com os parâmetros de otimização mostrados na Tabela 8.12. Usou-se o método do gradiente conjugado com pré-condicionador de Gauss-Seidel com precisão numérica de 10^{-4} para resolver os sistemas lineares de equações. O problema foi discretizado em uma malha de 5988 tetraedros e 1672 nós.

Tabela 8.12: Parâmetros do algoritmo de Heskovits usados na otimização da biela usando o método da camada unitária.

Parâmetro	Valor
α	$0,50$
β	$0,20$
ϵ	$0,10$
ξ	$5,00 \times 10^{-5}$
δ	$0,20$
$\bar{\rho}$	$1,00$
ν	0,60
γ	0,40

Também, foram utilizados os valores 25 e 5 para o número máximo de busca linear após a imposição inicial e o número máximo de análises na busca linear, respectivamente.

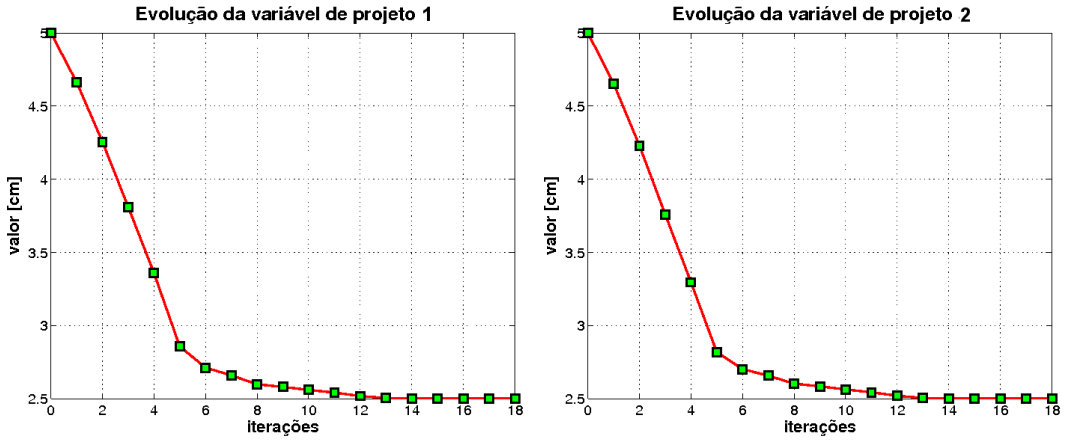
O processo de otimização precisou regerar a malha 3 vezes durante as 18 iterações necessárias para que o critério de otimização fosse satisfeito. A Tabela 8.13 e a Figura 8.19 mostram o resultado ótimo para as variáveis de projeto de 1 à 5 e o decréscimo do volume.

As geometrias inicial e final, após o processo de otimização são mostradas na Figura 8.20. A tensão de von Mises e o deslocamento máximo resultante para a configuração final são ilustrados pelas Figuras 8.21 e 8.22, respectivamente.

A porcentagem de redução do volume foi de $26,88\%$

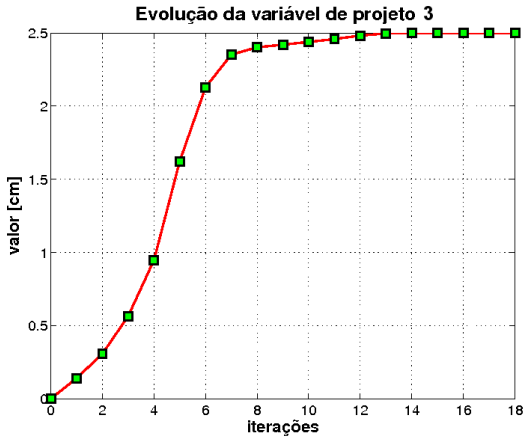
Tabela 8.13: Valores das variáveis de projeto de 1 à 5 e do volume para o problema definido pela Figura 8.16 usando o método da camada unitária. Unidade em $[cm]$.

Iterações	Variável 1	Variável 2	Variável 3	Variável 4	Variável 5	Volume
0	5,00000000	5,00000000	0,00000000	0,00000000	15,24121200	1764,48388927
1	4,66237408	4,65243754	0,13948532	0,14984680	14,43751953	1576,49370420
2	4,25513239	4,22968848	0,30668435	0,33539802	13,98396702	1511,93974896
3	3,81043100	3,75922568	0,56355910	0,62723937	14,06896748	1456,56238668
4	3,35782486	3,29506941	0,94719863	1,06139171	13,78758699	1410,26590879
5	2,85624733	2,81802776	1,62213728	1,78496309	13,88488736	1381,80904653
6	2,71176788	2,70171624	2,12831403	2,21398524	13,81837954	1353,03705201
7	2,65880863	2,65652572	2,35132561	2,33797501	13,80527315	1335,36116079
8	2,59819520	2,60225786	2,40149932	2,39144412	13,77457964	1317,32936741
9	2,57966845	2,58364734	2,41901076	2,41122887	13,75514781	1311,40582177
10	2,55971256	2,56266131	2,43830029	2,43330857	13,74735147	1305,59843332
11	2,54044209	2,54288343	2,45769442	2,45395985	13,69422171	1300,03420332
12	2,51752417	2,51906866	2,48164435	2,47941609	13,63997115	1294,27792350
13	2,50377011	2,50364318	2,49629168	2,49655290	13,60909058	1292,40097663
14	2,50151399	2,50152813	2,49850711	2,49849083	13,60779055	1291,29238849
15	2,50044711	2,50044805	2,49953966	2,49953335	13,60226153	1290,53150689
16	2,50017020	2,50016778	2,49983528	2,49983995	13,60093504	1290,35287995
17	2,50006363	2,50006624	2,49993271	2,49992838	13,60031944	1290,25717995
18	2,50002549	2,50002473	2,49997566	2,49997687	13,60014439	1290,22676752

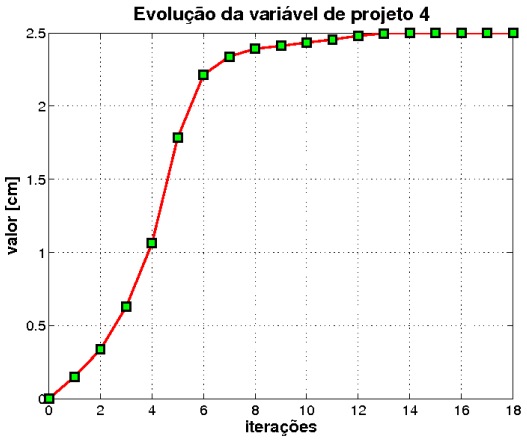


(a) Variável 1.

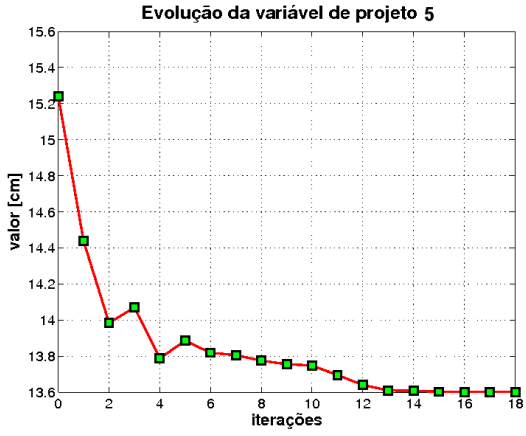
(b) Variável 2.



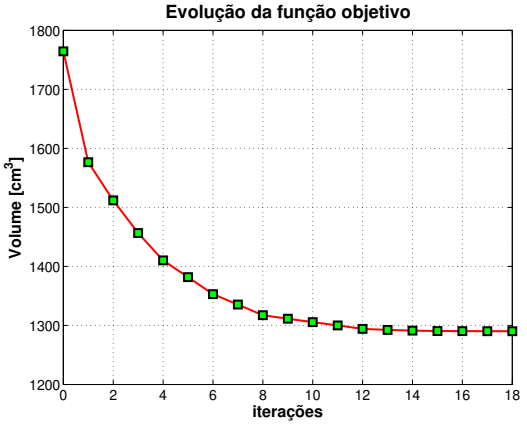
(c) Variável 3.



(d) Variável 4.

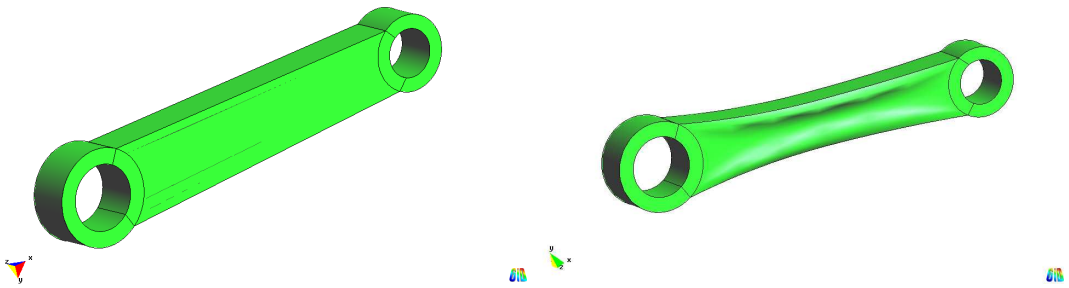


(e) Variável 5.



(f) Função objetivo

Figura 8.19: Evolução das variáveis de projeto e da função objetivo para o exemplo da biela tridimensional definida na Figura 8.16 usando o método da camada unitária de contorno.



(a) Volume inicial da biela 3D (b) Volume final da biela 3D

Figura 8.20: Comparação entre os volumes inicial e final da biela do problema de otimização definido na Figura 8.16.

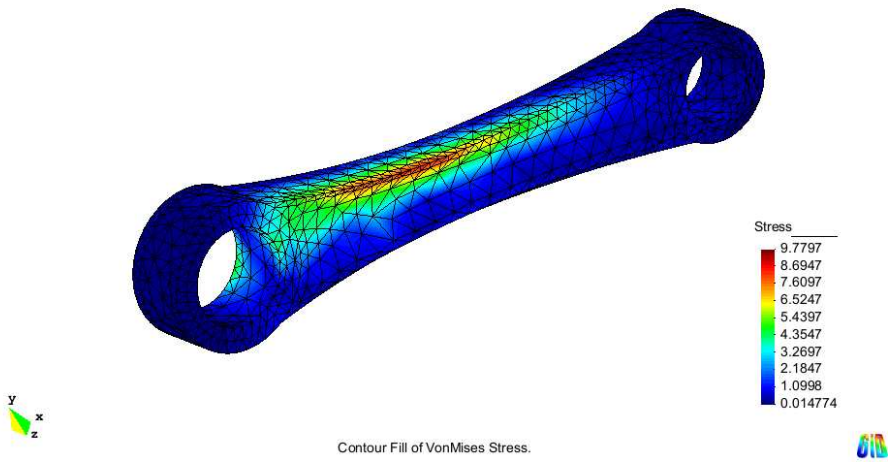


Figura 8.21: Tensões equivalentes de von Mises atuante na biela do problema definido na Figura 8.16 após o processo de otimização. Unidade em $[kN/cm^2]$.

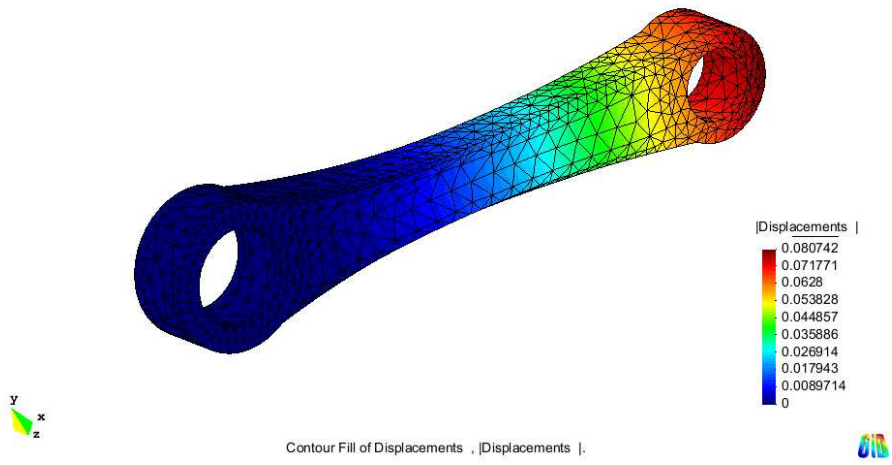


Figura 8.22: Deslocamentos resultantes atuante na biela do problema definido na Figura 8.16 após o processo de otimização. Unidade em $[cm]$.

8.2.2 Otimização de Forma usando o Método de Deslocamentos Fictícios do Contorno

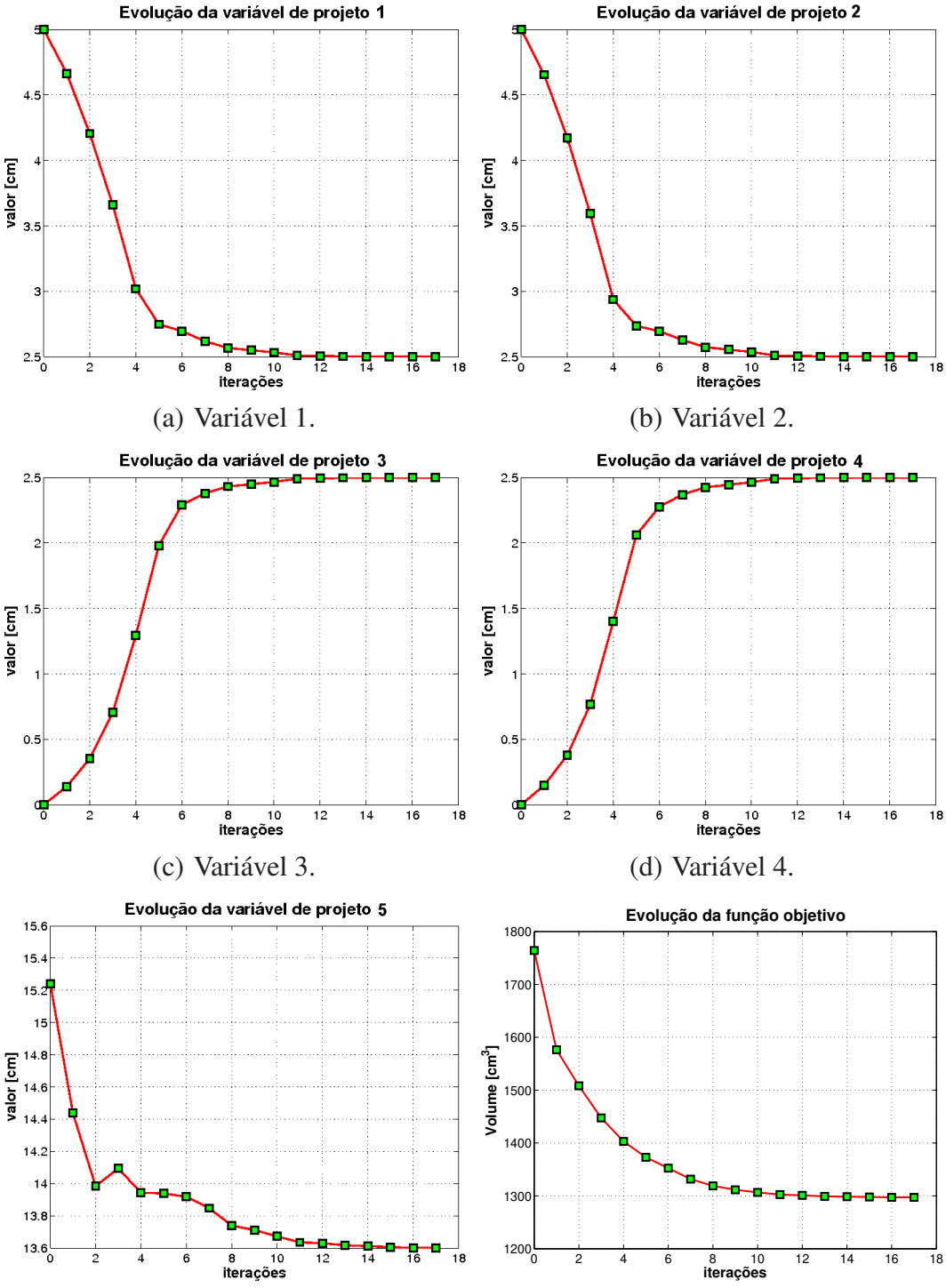
Usou-se o mesmo problema definido para o método da camada unitária de contorno. A parametrização foi mantida, assim como os valores dos parâmetros de Herskovits do algoritmo de otimização. Utilizou-se também o gradiente conjugado com pré-condicionador de Gauss-Seidel com precisão de 10^{-4}. Para evitar regeração da malha, usou-se uma malha mais refinada, o que gerou um total de 2898 nós e 11541 elementos tetraedros. Para essa configuração, o problema atingiu a convergência com 17 iterações.

A Tabela 8.14 e a Figura 8.23 mostram os valores obtidos durante o processo de otimização para as variáveis de projeto de 1 a 5, e também para a função objetivo, funcional de volume.

Como nesse processo não foi preciso regerar a malha, as geometrias inicial e final, após o processo de otimização são mostradas na Figura 8.24 e contêm as definições de malhas para as duas configurações.

Tabela 8.14: Valores das variáveis de projeto de 1 à 5 e do volume para o problema definido pela Figura 8.16 usando o método dos deslocamentos fictícios. Unidades em [cm].

Iterações	Variável 1	Variável 2	Variável 3	Variável 4	Variável 5	Volume
0	5,00000000	5,00000000	0,00000000	0,00000000	15,24121200	1764,17608472
1	4,66377063	4,65447199	0,13899456	0,14745068	14,43741234	1576,38301794
2	4,20482302	4,17378025	0,35157252	0,37699581	13,98376044	1508,30059812
3	3,65806545	3,59311398	0,70700022	0,76469599	14,09418257	1447,51929766
4	3,01856405	2,93724559	1,29467298	1,40047079	13,94243246	1402,97973966
5	2,74791638	2,73560915	1,97969223	2,06018832	13,93892731	1373,09262992
6	2,69531846	2,69502601	2,29187689	2,27593534	13,91739419	1352,41185842
7	2,61842063	2,62783978	2,37906818	2,36886897	13,84718518	1331,64819369
8	2,56657834	2,57326329	2,43098721	2,42382122	13,73920956	1318,88811190
9	2,55010434	2,55411728	2,44838145	2,44397437	13,71080329	1311,60816867
10	2,53310858	2,53588538	2,46578336	2,46273495	13,67208299	1306,56599383
11	2,50970167	2,50971290	2,49023383	2,49020704	13,63473634	1302,52082956
12	2,50647817	2,50637986	2,49347143	2,49349548	13,62903776	1301,01265939
13	2,50267617	2,50259448	2,49729790	2,49731914	13,61661752	1299,06001996
14	2,50185804	2,50181070	2,49813259	2,49814759	13,61128953	1298,48697956
15	2,50096869	2,50095847	2,49903109	2,49903479	13,60551399	1297,88155595
16	2,50002779	2,50003305	2,49997349	2,49997264	13,60000647	1297,22922861
17	2,50001942	2,50001977	2,49998070	2,49998067	13,60009434	1297,21760952



(a) Variável 1.

(b) Variável 2.

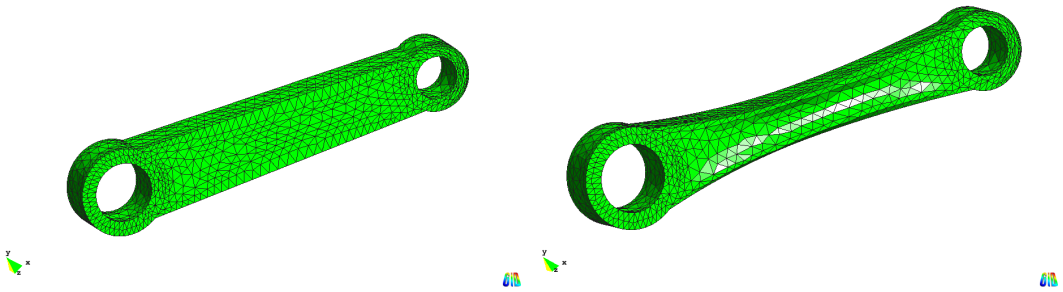
(c) Variável 3.

(d) Variável 4.

(e) Variável 5.

(f) Função objetivo.

Figura 8.23: Evolução das variáveis de projeto e da função objetivo para o exemplo da biela tridimensional definida na Figura 8.16 usando o método dos deslocamentos fictícios.



(a) Volume inicial da biela 3D (b) Volume final da biela 3D

Figura 8.24: Comparação entre os volumes inicial e final da biela do problema de otimização definido na Figura 8.16. O processo de otimização não necessitou regerar a malha, assim, usam-se as malhas para comparação das formas.

As Figuras 8.21 e 8.22 ilustram a tensão de von Mises e o deslocamento máximo resultante para a configuração final, respectivamente.

Com esse método foi possível obter uma redução de volume de $26,47\%$,

O método da camada unitária apresentou um resultado de otimização ligeiramente superior ao apresentado pelo método dos deslocamentos fictícios. Entretanto, o primeiro método necessita obrigatoriamente de um gerador de malha integrado ao programa de otimização. Isso não é mandatório no método dos deslocamentos fictícios, pois usando uma malha refinada foi possível obter a solução ótima sem regeração de malha, ocasionando em alto custo computacional ao se calcular as expressões para o campo de velocidades. Mas de qualquer forma, o resultado apresentado é satisfatório mesmo usando precisão numérica mais baixa para a solução iterativa. Porém , existem discordâncias com relação a aplicação do método das camadas unitárias.

Pelos resultados obtidos, é possível afirmar que os dois métodos podem ser aplicados para os problemas lineares de otimização de componentes mecânicos satisfatoriamente.

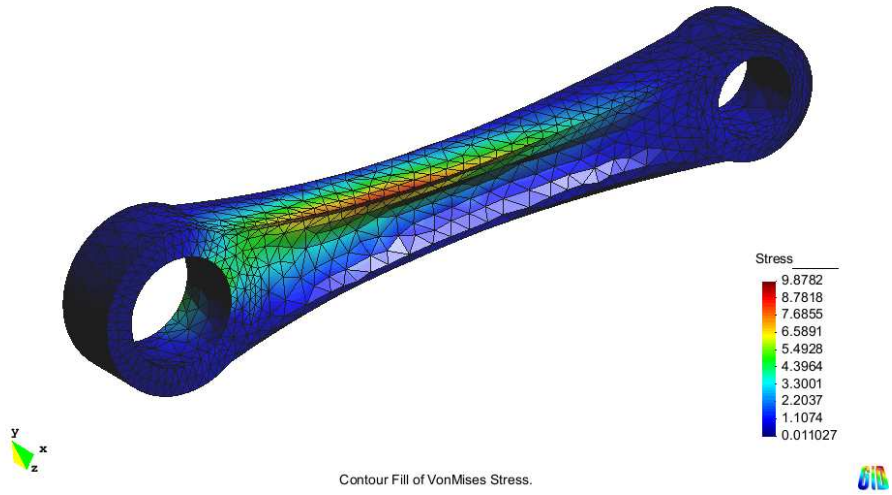


Figura 8.25: Tensões equivalentes de von Mises atuante na biela do problema definido na Figura 8.16 após o processo de otimização. Unidade em [kN/cm^2]

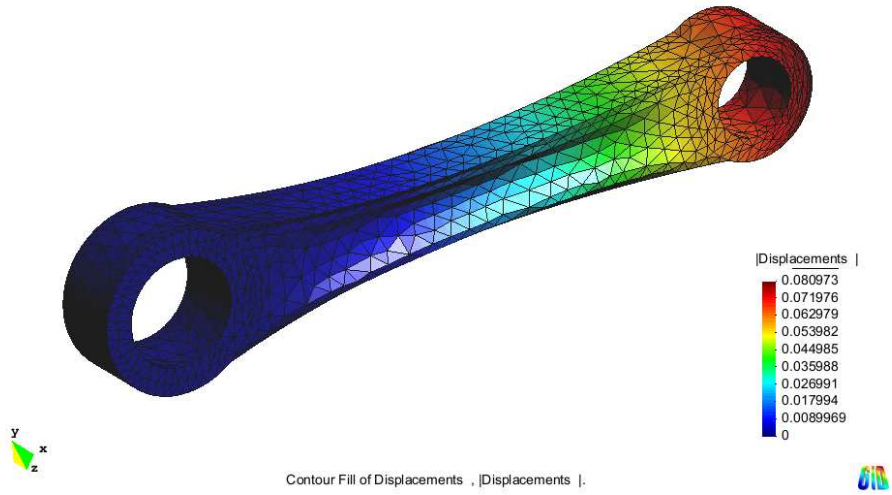


Figura 8.26: Deslocamentos resultantes atuante na biela do problema definido na Figura 8.16 após o processo de otimização usando o método dos deslocamentos fictícios. Unidade em [cm]

8.3 Problemas de Contato

Para validar o modelo de contato, alguns exemplos clássicos serão analisados e discutidos, comparando-os com a solução teórica, quando existir, e com a solução numérica usando o software comercial ANSYS versão 11.

8.3.1 Problema de Signorini

O primeiro problema a ser analisado é o problema clássico de Signorini baseado em (SIMO; LAURSEN, 1992) que consiste em um bloco elástico deslizando contra uma base rígida como ilustrado na Figura 8.27. O bloco, que possui dimensões $4m \times 2m$, é ao mesmo tempo tracionado na direção x e pressionado contra a base na direção y. O módulo de elasticidade é $E = 1000N/m^2$ e o coeficiente de Poisson é $\nu = 0,3$. O coeficiente de atrito adotado é $\mu = 0,5$ e os parâmetros de penalidades são $k_n = 10^8 N/m^2$ e $k_t = 10^4 N/m^2$. Foi utilizada uma malha de 200 elementos planos de 4 nós em estado plano de tensão, 20 elementos contatores isoparamétricos e 20 elementos alvos isoparamétricos com 2 nós por elemento. O primeiro e o último elementos da superfície contatora estão com deslocamentos na direção x restringido, ou seja os nós de números 1 e 2, como mostrado na Figura 8.28.

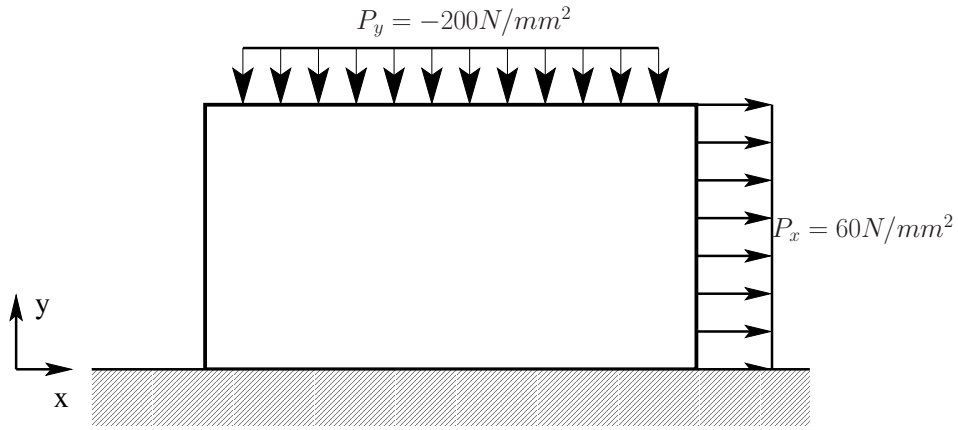


Figura 8.27: Bloco elástico em contato com fundação rígida (SIMO; LAURSEN, 1992).

O exemplo foi resolvido usando o programa hp^2fem com o módulo de contato adicionado

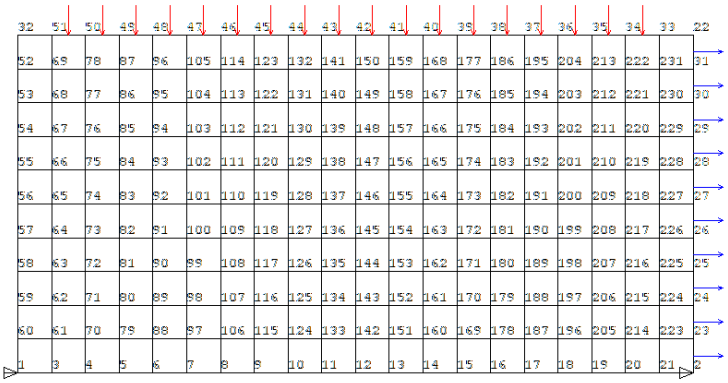


Figura 8.28: Malha e condições de contorno para o problema do exemplo 1.

à base do programa e com o software comercial ANSYS 11.0 com os mesmos dados e usando a discretização segmento-segmento com o elemento `CONTA172`. Segundo Simo e Laursen (1992), é necessário apenas um passo de carregamento para resolver o problema usando o *Método das Penalidades* para esses dados. Os resultados obtidos de força de atrito e deslocamentos na direção x e y estão nas Tabelas 8.15 e 8.16, respectivamente. Foi necessário apenas uma iteração de Newton-Raphson para uma precisão de $0,005$ baseada na norma da força desbalanceada. Todos os nós estão em condição de adesão.

Pelos resultados apresentados nas Tabelas 8.15 e 8.16, pode-se afirmar que o algoritmo desenvolvido para solucionar problema de contato com atrito via Método das Penalidades com discretização em elementos isoparamétricos está validado. O erro máximo obtido foi de 0% tanto para as forças de contato como para os deslocamentos dos nós em contato.

Tabela 8.15: Exemplo 1: Força de atrito e força normal obtidas usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força de atrito			Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]	hp^2fem [N]	ANSYS [N]	Erro [%]
1	7,2396	7,2396	0,00	268,7806	268,7800	0,00
2	-19,5080	-19,5080	0,00	589,6289	589,6300	0,00
3	30,3270	30,3270	0,00	139,6729	139,6700	0,00
4	30,8835	30,8830	0,00	146,7771	146,7800	0,00
5	24,3557	24,3560	0,00	146,5009	146,5000	0,00
6	17,7276	17,7280	0,00	152,1024	152,1000	0,00
7	10,8559	10,8560	0,00	156,8912	156,8900	0,00
8	3,7927	3,7926	0,00	161,3807	161,3800	0,00
9	-3,3650	-3,3650	0,00	164,8895	164,8900	0,00
10	-10,5468	-10,5470	0,00	167,4507	167,4500	0,00
11	-17,7299	-17,7300	0,00	169,1069	169,1100	0,00
12	-24,9383	-24,9380	0,00	169,9879	169,9900	0,00
13	-32,2280	-32,2280	0,00	170,2268	170,2300	0,00
14	-39,6665	-39,6660	0,00	169,9730	169,9700	0,00
15	-47,3115	-47,3120	0,00	169,3885	169,3900	0,00
16	-55,1964	-55,1960	0,00	168,8435	168,8400	0,00
17	-63,3298	-63,3300	0,00	168,6394	168,6400	0,00
18	-71,7174	-71,7170	0,00	170,9157	170,9200	0,00
19	-80,4823	-80,4820	0,00	174,3679	174,3700	0,00
20	-90,1594	-90,1590	0,00	197,8402	197,8400	0,00
21	-74,6255	-74,6250	0,00	205,8402	205,8400	0,00

Tabela 8.16: Exemplo 1: Deslocamentos dos nós em contato usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	u_x			u_y		
Nó	hp^2fem [10^{-3}m]	ANSYS [10^{-3}m]	Erro [%]	hp^2fem [10^{-6}m]	ANSYS [10^{-6}m]	Erro [%]
1	0,0000	0,0000	0,00	-3,0990	-3,0990	0,00
2	0,0000	0,0000	0,00	-7,1073	-7,1073	0,00
3	-3,4258	-3,4258	0,00	-1,1532	-1,1532	0,00
4	-3,1312	-3,1312	0,00	-1,5122	-1,5122	0,00
5	-2,4307	-2,4307	0,00	-1,4508	-1,4508	0,00
6	-1,7764	-1,7764	0,00	-1,5240	-1,5240	0,00
7	-1,0875	-1,0875	0,00	-1,5687	-1,5687	0,00
8	-0,3802	-0,3802	0,00	-1,6150	-1,6150	0,00
9	0,3363	0,3363	0,00	-1,6499	-1,6499	0,00
10	1,0547	1,0547	0,00	-1,6755	-1,6755	0,00
11	1,7728	1,7728	0,00	-1,6919	-1,6919	0,00
12	2,4930	2,4930	0,00	-1,7006	-1,7006	0,00
13	3,2212	3,2212	0,00	-1,7028	-1,7028	0,00
14	3,9644	3,9644	0,00	-1,7001	-1,7001	0,00
15	4,7286	4,7286	0,00	-1,6938	-1,6938	0,00
16	5,5170	5,5170	0,00	-1,6885	-1,6885	0,00
17	6,3301	6,3301	0,00	-1,6824	-1,6824	0,00
18	7,1706	7,1706	0,00	-1,7139	-1,7139	0,00
19	8,0222	8,0222	0,00	-1,7004	-1,7004	0,00
20	9,1202	9,1202	0,00	-2,0962	-2,0962	0,00
21	9,2313	9,2313	0,00	-1,3769	-1,3769	0,00

8.3.2 Dois Blocos Elásticos em Contato

Considera-se agora o exemplo entre dois blocos elásticos em contato como mostrado na Figura 8.29. Ambos os blocos possuem o mesmo material, isto é, $E_1 = E_2 = 10^8 Pa$ e $\nu_1 = \nu_2 = 0,3$. O coeficiente de atrito para a interface do contato é de $\mu = 0,5$ e a força aplicada é de $P = 10000N$. Os parâmetros de penalidades adotados foram $k_n = 10^{10} Pa$ e $k_t = 10^{10} Pa$ para o ANSYS, e $k_n = 10^{10} Pa$ e $k_t = 10^8 Pa$ para o algoritmo implementado no hp^2fem. Considerou-se somente metade do modelo para simulação e aplicou-se condição de simetria como mostrado na Figura 8.30. Foram utilizados 173 elementos de 4 nós em estado de tensão e 26 elementos de contato (13 contatores e 13 alvos). A malha é ajustada para não violar a hipótese da formulação via elementos isoparamétricos em contato. O problema foi resolvido no ANSYS 11.0 usando elemento `PLANE42` e `CONTA172` para comparação dos resultados. Usou-se apenas um passo de carregamento.

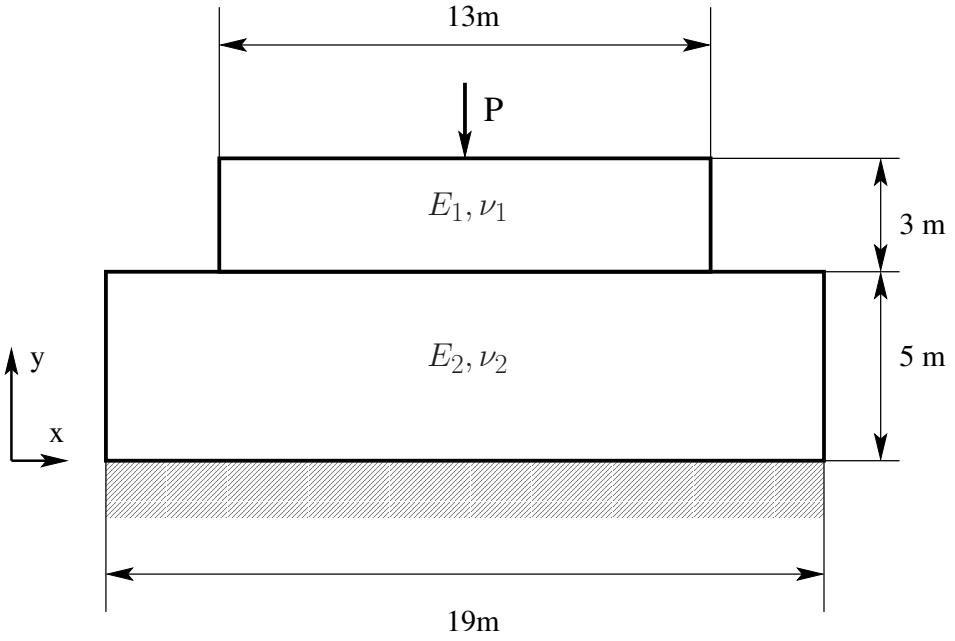


Figura 8.29: Dois blocos elásticos em contato.

A Tabela 8.17 contêm os resultados para força de contato normal e de atrito obtidos usando o ANSYS 11.0 e o algoritmo de contato usando elemento isoparamétrico implementado no hp^2fem. A mesma comparação de resultados foi realizada para os deslocamentos na direção x e y e estão na Tabela 8.18.

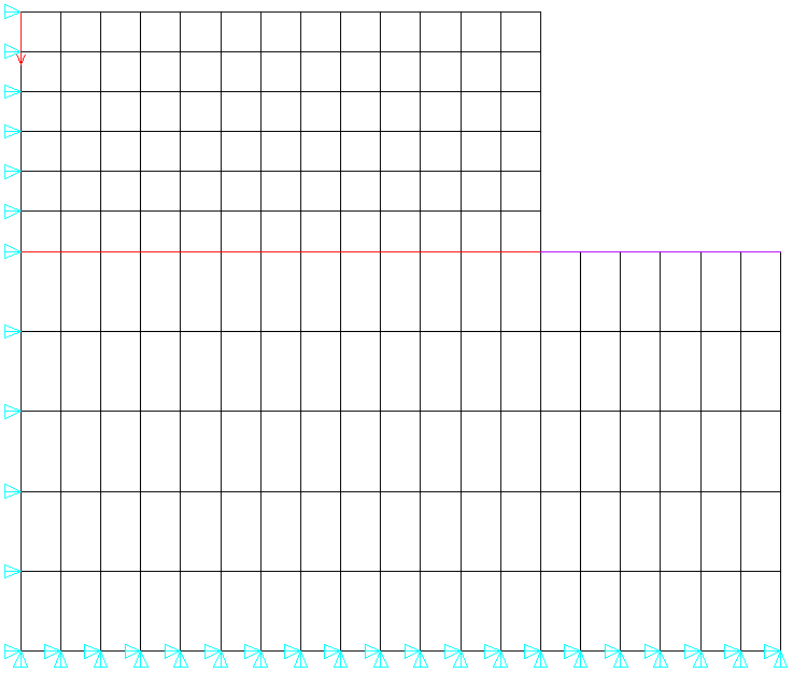


Figura 8.30: Malha e condição de contorno para o problema do exemplo 2.

Tabela 8.17: Exemplo 2: Força de atrito e força normal obtidas usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força de atrito			Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]	hp^2fem [N]	ANSYS [N]	Erro [%]
121	-70,31689	-70,78200	0,66	2216,34372	2216,60000	0,01
122	0,00000	0,00000	0,00	0,00000	0,00000	0,00
123	-335,27538	-334,60000	0,20	2110,03103	2110,30000	0,01
124	-747,65357	-742,11000	0,74	1841,56215	1841,70000	0,01
125	-749,97160	-749,98000	0,00	1499,94319	1500,00000	0,00
126	-577,32114	-577,37000	0,01	1154,64228	1154,70000	0,00
127	-427,68428	-427,73000	0,01	855,36856	855,47000	0,01
128	-309,44856	-309,48000	0,01	618,89712	618,96000	0,01
129	-218,71047	-218,72000	0,00	437,42093	437,44000	0,00
130	-139,49455	-139,48000	0,01	278,98911	278,97000	0,01
131	-47,48689	-47,43600	0,00	94,97377	94,87100	0,00
132	0,00000	0,00000	0,00	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00	0,00000	0,00000	0,00

Tabela 8.18: Exemplo 2: Deslocamentos dos nós em contato usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	u_x			u_y		
Nó	hp^2fem [10^{-5}m]	ANSYS [10^{-5}m]	Erro [%]	hp^2fem [10^{-5}m]	ANSYS [10^{-5}m]	Erro [%]
121	0,00000	0,00000	0,00	-6,55244	-6,55380	0,02
122	2,87139	2,87320	0,06	1,33590	1,33790	0,15
123	0,52988	0,53245	0,48	-6,39004	-6,39120	0,02
124	0,97661	0,98119	0,47	-5,94371	-5,94440	0,01
125	1,45056	1,45350	0,20	-5,30133	-5,30150	0,00
126	1,89866	1,90090	0,12	-4,55350	-4,55340	0,00
127	2,22417	2,22620	0,09	-3,78028	-3,78010	0,00
128	2,43372	2,43560	0,08	-3,03932	-3,03900	0,01
129	2,56670	2,56850	0,07	-2,35928	-2,35890	0,02
130	2,66502	2,66680	0,07	-1,74613	-1,74560	0,03
131	2,75907	2,76090	0,07	-1,19624	-1,19560	0,05
132	2,82930	2,83110	0,06	-0,61255	-0,61157	0,16
133	2,86010	2,86190	0,06	0,03547	0,03680	3,76
134	2,86999	2,87180	0,06	0,68790	0,68955	0,24

Percebe-se pelos resultados apresentados que o algoritmo proposto consegue convergir para uma solução próxima do ANSYS usando um valor de penalidade tangencial 100 vezes menor (existe um fator multiplicador dentro do algoritmo do ANSYS versão 11.0 que não é explicada no manual do usuário), evitando-se, assim, o mal condicionamento da matriz de rigidez. O erro máximo obtido foi de 3,76% para o deslocamento na direção y para o nó 133. Já para as forças de contato, o erro máximo obtido foi de 0,74% para a força de atrito. Assim, pode-se afirmar que o modelo de contato 2D para pequenas deformações usando elementos isoparamétricos para as condições de dois corpos deformáveis é válido.

A Problema de Hertz

Para validar o algoritmo para elemento de contato nó-segmento testou-se o problema de Hertz (HERTZ, 1881). Considere o contato sem atrito entre uma esfera elástica e um anteparo rígido. Há solução teórica como mostrado em (TIMOSHENKO; WOINOWSKY-KRIEGER, 1959). Utiliza-se apenas a metade do modelo para simular o problema, como ilustrado na Figura 8.31 e aplica-se as condições de simetria. O módulo de elasticidade da esfera é $E = 1000N/mm^2$ e o coeficiente de Poisson é $\nu = 0,3$. Primeiro, utiliza-se uma carga utilizada $P = 60\pi N$ e depois uma carga $P = 30\pi N$ para comparar o efeito da carga no posicionamento do arco de contato. Foram utilizadas duas malhas para verificação do problema. As malha 1 e 2 utilizadas na solução são mostradas nas Figura 8.33(c) e 8.33(d), respectivamente.

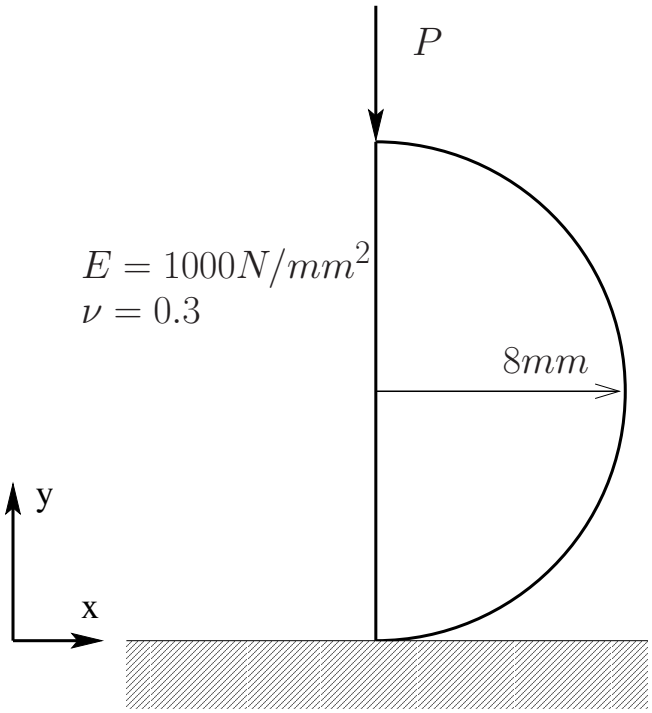
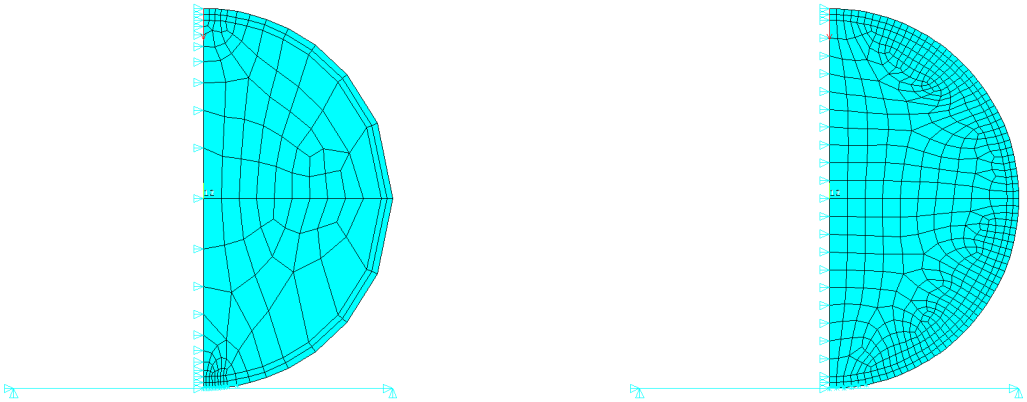


Figura 8.31: Esfera elástica em contato com fundação rígida (ANSYS INC., 2008).

A Tabela 8.33 fornece os valores teóricos de máxima pressão de contato normal e o meio arco de contato para o contato entre uma esfera e um anteparo rígido. As seguintes equações dadas



(a) Malha 1. (b) Malha 2.

Figura 8.32: Malhas e condições de contorno para o problema do exemplo 2.

por Timoshenko e Woinowsky-Krieger (1959) forma utilizadas.

$$Q_0 = \frac{3P}{2\pi a^2} \text{ (máxima pressão de contato normal),} \tag{8.4a}$$

$$a = \sqrt[3]{\frac{3\pi}{4}\frac{PR}{E}} \text{ (meio arco de contato).} \tag{8.4b}$$

Tabela 8.19: Exemplo 3: Resultado teórico para o problema de contato entre uma esfera elástica e um anteparo rígido.

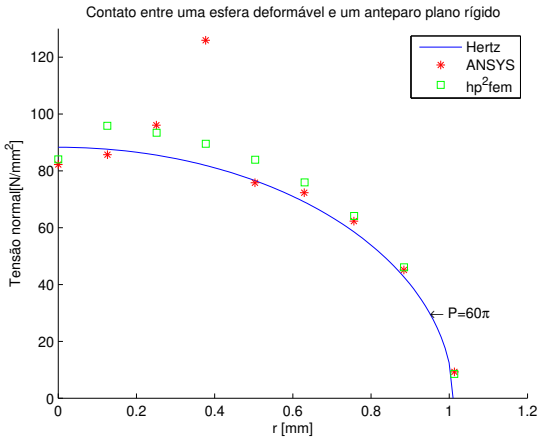
$P\ [N]$	$Q_0\ [N/mm^2]$	$a\ [mm]$
60π	88,2904	1,0096
30π	70,0761	0,8013

Pode-se determinar a pressão de contato normal Q para qualquer distancia r dentro do arco de contato. Basta usar a seguinte equação fornecida por (PEDERSEN, 2006)

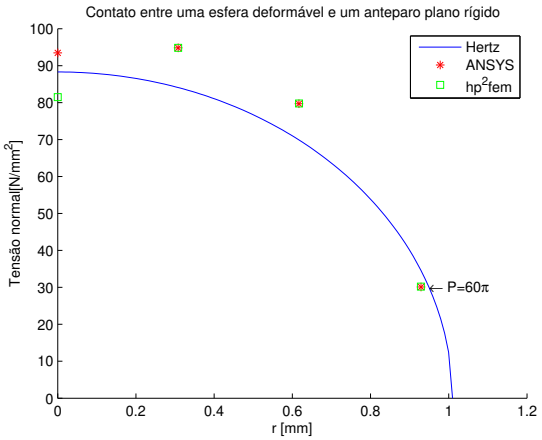
$$Q = Q_0\sqrt{1 - \left(\frac{r}{a}\right)^2} = -\sigma_y, \tag{8.5}$$

sendo σ_y a tensão nodal na direção y.

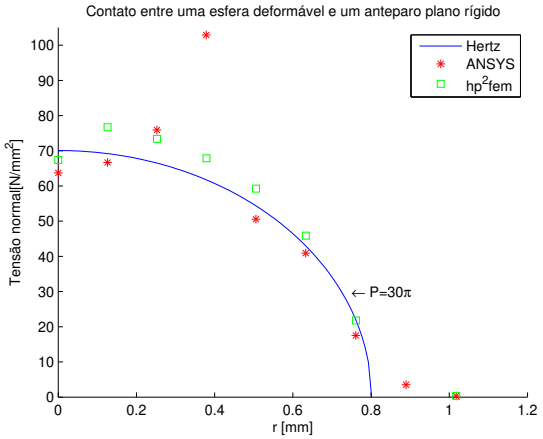
A equação (8.5) foi usada para traçar os gráficos da solução teórica contra a solução numérica obtida pela implementação nó-segmento do hp^2fem e o programa ANSYS.



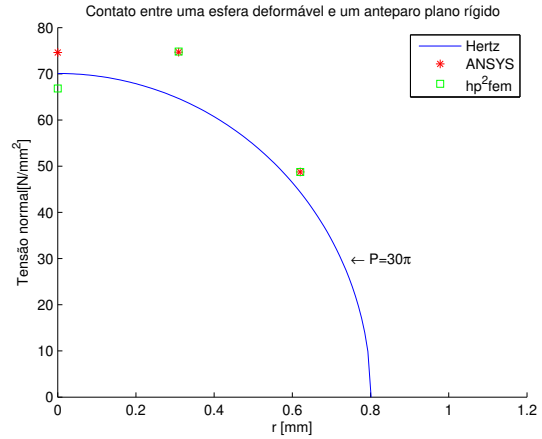
(a) Resultado para Malha 1 ($P = 60\pi$).



(b) Resultado para Malha 2 ($P = 60\pi$).



(c) Resultado para Malha 1 ($P = 30\pi$).



(d) Resultado para Malha 2 ($P = 30\pi$).

Figura 8.33: Comparação entre a solução teórica de Hertz do problema do exemplo 2 com o algoritmo proposto nó-segmento e o ANSYS 11.0.

Pelos resultados apresentados, pode-se perceber que o algoritmo apresentado funciona tão bem quanto o programa comercial ANSYS e no caso da malha 1 o resultado foi melhor. Nota-se que para um melhor resultado é necessário um refinamento h próximo a região de contato como feito no trabalho de Fancello (1993).

B Dois Blocos elásticos em Contato com Atrito

Considera-se o mesmo problema da seção 8.3.2, usando agora a discretização nó-segmento. Para validar o algoritmo proposto para esta situação, considera-se um primeiro o caso sem atrito e depois para três valores de atrito $\mu_1 = 0,1$, $\mu_2 = 0,3$ e $\mu = 0,5$. Como feito nos exemplos anteriores, compara-se a solução da força normal e de atrito obtidas pelo hp^2fem com o ANSYS, e no caso onde $\mu = 0,5$ compara-se o resultado obtido com a discretização usando elementos isoparamétricos. A Tabela 8.20 compara os valores de tensão normal obtidas com o ANSYS e o hp^2fem. Como o erro máximo obtido foi de 0,22% pode-se afirmar que o algoritmo está funcionando corretamente para o caso de problema de contato sem atrito. Foram necessárias 4 iterações de Newton-Raphson para um valor de convergência $7,5593 \times 10^{-6}$ no programa hp^2fem e 4 iterações para o ANSYS com valor de convergência $0,4217$ de acordo com o seu critério de parada.

Tabela 8.20: Exemplo 4: Força de contato normal obtida usando $\mu = 0,0$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]
121	604,75246	604,77000	0,00
122	0,00000	0,00000	0,00
123	1151,38720	1151,40000	0,00
124	1005,58031	1005,60000	0,00
125	813,06290	813,13000	0,01
126	616,48926	616,60000	0,02
127	438,88282	439,02000	0,03
128	277,82751	277,98000	0,05
129	91,99655	92,20100	0,22
130	0,00000	0,00000	0,00
131	0,00000	0,00000	0,00
132	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00

Conforme mostrado na Tabela 8.21 para $\mu = 0,1$, o erro máximo obtido, tanto para a força de atrito tangencial como para a força normal, foi de 0,04%. Pode-se assim, concluir que para um coeficiente de atrito baixo, o algoritmo está funcionando corretamente, mesmo tendo um parâmetro de penalidade tangencial 100 vezes inferior ao utilizado no ANSYS.

Tabela 8.21: Exemplo 4: Força de atrito e força normal obtidas usando $\mu = 0,1$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força de atrito			Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]	hp^2fem [N]	ANSYS [N]	Erro [%]
121	0,00000	0,00000	0,00	591,08012	591,09000	0,00
122	0,00000	0,00000	0,00	0,00000	0,00000	0,00
123	-112,80601	-112,81000	0,00	1128,06008	1128,10000	0,00
124	-98,88943	-98,88900	0,00	988,89434	988,89000	0,00
125	-80,14961	-80,15100	0,00	801,49612	801,51000	0,00
126	-61,05890	-61,06000	0,00	610,58898	610,60000	0,00
127	-44,07357	-44,07400	0,00	440,73571	440,74000	0,00
128	-29,38973	-29,38900	0,00	293,89726	293,89000	0,00
129	-14,52547	-14,52000	0,04	145,25465	145,20000	0,04
130	0,00000	0,00000	0,00	0,00000	0,00000	0,00
131	0,00000	0,00000	0,00	0,00000	0,00000	0,00
132	0,00000	0,00000	0,00	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00	0,00000	0,00000	0,00

Tabela 8.22: Exemplo 4: Força de atrito e força normal obtidas usando $\mu = 0,3$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força de atrito			Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]	hp^2fem [N]	ANSYS [N]	Erro [%]
121	0,00000	0,00000	0,00	565,10433	564,04000	0,19
122	0,00000	0,00000	0,00	0,00000	0,00000	0,00
123	-324,64776	-324,43000	0,07	1082,15919	1081,40000	0,07
124	-286,21604	-286,32000	0,04	954,05347	954,39000	0,04
125	-232,84954	-232,82000	0,01	776,16514	776,08000	0,01
126	-178,46811	-178,41000	0,03	594,89370	594,70000	0,03
127	-131,18456	-131,17000	0,01	437,28186	437,24000	0,01
128	-92,65915	-92,70300	0,05	308,86383	309,01000	0,05
129	-60,04468	-60,17800	0,22	200,14893	200,59000	0,22
130	-24,40466	-24,76100	1,46	81,34887	82,53500	1,46
131	0,00000	0,00000	0,00	0,00000	0,00000	0,00
132	0,00000	0,00000	0,00	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00	0,00000	0,00000	0,00

Já para o coeficiente de atrito $\mu = 0,3$, o erro máximo obtido foi de 1,46% para ambas as forças de contato, conforme Tabela 8.22. Apenas o nó 121 está em situação de adesão. O nós 123 ao 130 estão em situação de deslizamento. Como era esperado, ocorre uma dissipação da força de contato normal devido à presença do atrito, que vai aumentando proporcionalmente ao valor de μ.

A Tabela 8.23 mostra os resultados para um coeficiente de atrito $\mu = 0,5$. Nota-se que o erro máximo obtido para a força de atrito é superior a 214%. Esse erro ocorre porque no software ANSYS os nós 123 e 124 estão em condição de adesão, enquanto que para o algoritmo implementado no hp^2fem, esses mesmos nós estão em condição de deslizamento. A posição inicial de contato dos nós contatores 123 e 124 com os seus respectivos elementos alvos eram dadas pela coordenada convectiva normalizada $\bar{\xi} = 0,0$. Durante o processo de deformação, a coordenada convectiva passa abruptamente para $\bar{\xi} = 1,0$, ou seja, o contato passa a ser com o elemento alvo da esquerda. Essa troca de valor ocasiona uma descontinuidade no método de *return mapping* para atualização das forças de contato, o que gera a situação permanente de deslizamento. Acredita-se que uma alteração no posicionamento desses nós na malha pode solucionar o problema.

Tabela 8.23: Exemplo 4: Força de atrito e força normal obtidas usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM e ANSYS 11.0.

	Força de atrito			Força normal		
Nó	hp^2fem [N]	ANSYS [N]	Erro [%]	hp^2fem [N]	ANSYS [N]	Erro [%]
121	0,00000	0,00000	0,00	544,41244	551,95000	1,37
122	0,00000	0,00000	0,00	0,00000	0,00000	0,00
123	-525,72464	-167,30000	214,24	1051,44927	1049,80000	0,16
124	-172,78261	-356,43000	51,52	927,47776	918,72000	0,95
125	-374,13277	-374,95000	0,22	748,26554	749,89000	0,22
126	-288,71062	-289,36000	0,22	577,42125	578,71000	0,22
127	-214,73933	-214,74000	0,00	429,47867	429,47000	0,00
128	-155,58096	-155,52000	0,04	311,16191	311,04000	0,04
129	-109,53021	-109,47000	0,05	219,06041	218,93000	0,06
130	-70,49958	-70,48200	0,02	140,99916	140,96000	0,03
131	-25,14752	-25,24600	0,39	50,29504	50,49300	0,39
132	0,00000	0,00000	0,00	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00	0,00000	0,00000	0,00

A Tabela 8.24 mostra a comparação entre os resultados obtidos através das discretizações

usando elementos isoparamétricos e nó-segmento. Nota-se que os valores obtidos são diferentes, porém existe uma certa multiplicidade com relação aos valores. Os valores obtidos pela discretização nó-segmento, com exceção dos nós 123 e 124, são aproximadamente a metade dos mesmos obtidos via elementos isoparamétricos.

Tabela 8.24: Exemplo 4: Força de atrito e força normal obtidas usando $\mu = 0,5$ e o Método das Penalidades nos programas hp^2FEM usando as duas discretizações implementadas

	Força de atrito			Força normal		
Nó	hp^2fem [N] nó-segmento	hp^2fem [N] isoparamétrico	Erro [%]	hp^2fem [N] nó-segmento	hp^2fem [N] isoparamétrico	Erro [%]
121	0,00000	-70,31689	100,00	544,41244	2216,34372	75,44
122	0,00000	0,00000	0,00	0,00000	0,00000	0,00
123	-525,72464	-335,27538	56,80	1051,44927	2110,03103	50,17
124	-172,78261	-747,65357	76,89	927,47776	1841,56215	49,64
125	-374,13277	-749,97160	50,11	748,26554	1499,94319	50,11
126	-288,71062	-577,32114	49,99	577,42125	1154,64228	49,99
127	-214,73933	-427,68428	49,79	429,47867	855,36856	49,79
128	-155,58096	-309,44856	49,72	311,16191	618,89712	49,72
129	-109,53021	-218,71047	49,92	219,06041	437,42093	49,92
130	-70,49958	-139,49455	49,46	140,99916	278,98911	49,46
131	-25,14752	-47,48689	47,04	50,29504	94,97377	47,04
132	0,00000	0,00000	0,00	0,00000	0,00000	0,00
133	0,00000	0,00000	0,00	0,00000	0,00000	0,00
134	0,00000	0,00000	0,00	0,00000	0,00000	0,00

A versão do hp^2fem de baixa ordem possui suporte à solução de problemas de contato em pequenas deformações usando duas discretizações com características distintas. A primeira, mais simples usando elementos isoparamétricos, permite somente a solução de problemas com malhas casadas. Entretanto, podem ser usados elementos de alta ordem, desde que o grau dos elementos do par de contato seja o mesmo. Caso contrário, tem-se a necessidade de usar os *Métodos Mortar*.

O problema de contato com atrito com ambos os corpos deformáveis na discretização nó-segmento é o mais complicado de ser implementado, pois não se tem garantia da condição de unicidade dado pela busca do contato. Em algumas situações, é possível ter dois nós alvos para o mesmo nó contator. Nesses casos, é necessário a aplicação de uma suavização da região de contato, usando funções de interpolação C^1 contínuas, tais como NURBS, Beziér e Hermite.

9 Conclusão e Perspectivas Futuras

Nesta tese, mostraram-se vários aspectos da construção e formulação de uma arquitetura orientada por objetos para MEF-AO e foram discutidos os benefícios da aplicação da arquitetura na solução de problemas de grandes deformações, problema de contato e otimização de forma. As conclusões obtidas são apresentadas a seguir:

- Este trabalho desenvolveu e implementou em linguagem MatLab uma arquitetura orientada por objetos para MEF-AO denominada de hp^2fem;
- Aplicou o MEF-AO na solução de problemas de mecânica estrutural não-linear;
- O código é modular e flexível, permitindo a adição de novos tipos de problemas;
- O programa hp^2fem provê prototipagem de códigos e elaboração de testes de algoritmo de maneira rápida e confiável. A ferramenta de depuração do MatLab auxilia a execução das tarefas de teste; Adicionalmente, em conjunto com os recursos de manipulação simbólica e resolução de sistemas lineares, deixa o programador focado no desenvolvimento do modelo de elementos finitos;
- A adoção de carregamento dependente da deformação aumenta a complexidade da modelagem em elementos finitos, devido à não-simetria da matriz de rigidez, e consequentemente eleva o custo computacional para solucionar esse problema;
- As funções de interpolação de alta ordem possibilitam a obtenção de soluções mais precisas que as obtidas via refinamento h;
- O refinamento hp obtém melhores resultados de convergência que o método p. O método p deve ser utilizado para uma primeira análise e depois o usuário deve refinar via método-h a região de interesse de cálculo;
- Escrevendo o código de elementos finitos em função de operadores, tais como o tensores de tensão (Cauchy e Piola-Kirchhoff), tensores de deformação (Green-Lagrange) e os operadores cinemáticos (gradiente de deformação, tensor à direita e à esquerda de Cauchy) tornou o código modular e reutilizável para outros tipos de problemas a serem estudados no futuro;

- A partir dos resultados obtidos, pode-se afirmar que o hp^2fem apresenta solução satisfatória para problemas de grandes deformações tanto para baixa ordem como para alta ordem;
- O algoritmo de contato 2D funciona de forma satisfatória e permite a adição do modelo 3D;
- A conexão do gerador de malha GiD com o hp^2fem e o otimizador de forma funciona bem. Esta conexão é feita através de scripts TCL/TK que permite que através da interface gráfica do GiD um problema de mecânica estrutural seja criado e convertido para um formato que o hp^2fem entende;
- A comparação entre os métodos das camadas unitárias e o método dos deslocamentos fictícios foi realizado, da mesma forma que Silva (2003), mostrando que os métodos são equivalentes em termos de resultado. Porém, é necessário tomar cuidado com o método das camadas unitárias, uma vez que não se avalia o erro da sensibilidade e de acordo com Schleupen, Maute e Ramm (2000), a malha deve ser mantida fixa no processo de otimização;
- Além disso, o código desenvolvido permite a migração de forma suave para uma linguagem de programação mais robusta, tal como C++, que permitirá desenvolver exemplos maiores em qualquer tipo de plataforma. Esse trabalho já está sendo desenvolvido por Costa (2012);
- O desenvolvimento deste programa permitiu o crescimento acadêmico do grupo, abrindo o caminho para desenvolvimento de novos projetos por pessoas que tinham pouco contato com mecânica estrutural e/ou computação científica.

Vários aspectos podem ser considerados como possíveis linhas de pesquisa em próximos trabalhos:

1. Implementar problema de contato usando grandes deformações, seguindo os trabalhos de Yang (2009) , Fischer e Wriggers (2006);
2. Implementar o método mortar, que permite que regiões (subdomínios) com funções de interpolação de graus diferentes) sejam compatibilizadas com apresentado nos trabalhos de Wohlmuth (2000a).
3. Aplicar o MEF-AO em otimização estrutural;

4. Estudar e implementar mecanismo de paralelização do código de elementos finitos;

5. Traduzir o código para uma linguagem mais robusta, tal como o C++;

REFERÊNCIAS

ACHARJEE, S.; ZABARAS, N. The continuum sensitivity method for the computational design of three-dimensional deformation process. *Computer Methods in Applied Mechanics and Engineering*, v. 195, p. 6822–6842, 2006.

AL-DOJAYLI, M.; MEGUID, S. Accurate modeling of contact using cubic splines. *Finite Elements in Analysis and Design*, v. 38, p. 337–352, 2002.

ALTAIR ENGINEERING, INC. *Hypermesh version 10.0 user guide*. [S.l.], 2008.

ANSYS INC. *Release 11.0 Documentation for Ansys*. [S.l.], 2008.

ARORA, J. S. *Introduction to Optimum Design*. 2nd. ed. USA: Elsevier inc, 2004.

BABUSKA, I. The *p* and *h-p* versions of the finite element method: the state of the art. In: DWOYER, D. L.; HUSSAINI, M. Y.; VOIGT, R. G. (Ed.). *Finite Elements: Theory and Applications*. New York: Springer-Verlag, 1988.

BABUSKA, I. et al. Efficient preconditioning for the *p*-version finite element method in two dimensions. *SIAM Journal on Numerical Analysis*, v. 28, n. 3, p. 624–661, 1991.

BABUSKA, I.; GUO, B. Q. The *h-p* version of the finite element method for domains with curved boundaries. *SIAM Journal on Numerical Analysis*, v. 25, n. 4, p. 837–861, 1988.

BABUSKA, I.; SURI, M. The optmal convergence rate of the p-version of the finite element method. *SIAM J. Numer. Anal.*, v. 24, p. 750–776, 1987.

BABUSKA, I.; SURI, M. The *p* and *h-p* versions of the finite element method, an overview. *Computer Methods in Applied Mechanics and Engineering*, v. 80, p. 5–26, 1990.

BABUSKA, I.; SZABÓ, B. A.; KATZ, I. N. The p-version of the finite element method. *SIAM J. Numer. Anal.*, v. 18, n. 3, p. 515–545, 1981.

BAŞAR, Y.; WEICHERT, D. *Nonlinear Continuum Mechanics of Solids*: Fundamental mathematical and physical concepts. 1st. ed. Germany: Springer, 2000.

BARGOS, F. F. *Implementação de Elementos Finitos de Alta Ordem baseado em Produto Tensorial*. Dissertação (Mestrado) — Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2009.

BATHE, K. *Finite Element Procedures*. Englewood Cliffs: Prentice Hall, 1996.

BATHE, K.-J.; CHAUGHARY, A. A solution method for planar axisymmetric contact problems. *International Journal for Numerical Method Engineering*, v. 21, p. 65–88, 1985.

BELEGUNDU, A.; ARORA, J. A study of mathematical programming methods for structural optimization. Part I: Theory. *International Journal for Numerical Methods in Engineering*, v. 21, p. 1583–1599, 1985.

BELEGUNDU, A.; ARORA, J. A study of mathematical programming methods for structural optimization. Part II: Numerical results. *International Journal for Numerical Methods in Engineering*, v. 21, p. 1601–1623, 1985.

BELYTSCHKO, T.; LIU, W. K.; MORAN, B. *Nonlinear Finite Elements for Continua and Structures*. 1st. ed. USA: John Wiley & Sons, 2000.

BENNETT, J.; BOTKIN, M. Structural shape optimization with geometric description and adaptive mesh refinement. *AIAA Journal*, v. 23, n. 3, p. 458–464, 1985.

BERNARDI, C.; MADAY, Y.; PATERA, A. T. A new nonconforming approach to domain decomposition: The mortar element method. In: BREZIS, H.; LIONS, J. L. (Ed.). *Nonlinear Partial Differential Equations and their Applications*. First. Paris: Pitman, 1994, (Collège de France Seminar Volume XI). p. 13–51.

BESSON, J.; FOERCH, R. Large scale object-oriented finite element code design. *Computer Methods in Applied Mechanics and Engineering*, v. 142, n. 1-2, p. 165–187, 1997.

BESTLE, D.; EBERHARD, P. Optimization of a contact surface. *Struct Multidisc Optim*, v. 25, p. 339–342, 2003.

BHATTI, M. A. *Advanced Topics in Finite Element Analysis of Structures*: With MATHEMATICA® AND MATLAB® COMPUTATIONS. 1st. ed. New York: John Wiley and Sons, inc, 2006.

BITTENCOURT, E.; CREUS, G. J. Finite element analysis of three-dimensional contact and impact in large deformation problems. *Computers and Structures*, n. 69, p. 219–234, 1998.

BITTENCOURT, M. Using C++ templates to develop finite element classes. *Engineering Computations*, v. 17, n. 7, p. 775–788, 2000.

BITTENCOURT, M. Fully tensorial nodal and modal shape functions for triangles and tetrahedra. *International Journal For Numerical Methods In Engineering*, v. 63, n. 11, p. 1530–1558, 2005.

BITTENCOURT, M.; FEIJÓO, R. Análise comparativa de métodos diretos e iterativos para a solução de sistemas de equações. *Revista Internacional de Métodos Numéricos y Diseño em Ingenieria*, v. 13, n. 2, p. 123–148, 1997.

BITTENCOURT, M. L.; Nogueira Jr., A. C. Spectral/hp finite elements applied to linear and non-linear structural elastic problems. *Latin American Journal of Solids and Structures*, v. 4, p. 61–85, 2007.

BITTENCOURT, M. L.; VAZQUEZ, T. G. Tensor-based gauss-jacobi numerical integration for high-order mass and stiffness matrices. *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING*, v. 79, p. 599–638, 2009.

BITTENCOURT, M. L.; VAZQUEZ, T. G.; VAZQUEZ, M. G. Construction of shape functions for theh- andp-versions of the fem using tensorial product. *International Journal for Numerical Methods in Engineering*, v. 71, p. 529–563, 2007.

BONET, J.; WOOD, R. D. *Nonlinear Continuum Mechanics fo Finite Element Analysis*. 2nd. ed. New York: Cambridge University Press, 2008.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *The Unified Modeling Language User Guide*. Reading (MA): Addison Wesley, 1999.

CARVALHO, A. M. B. R.; CHIOSSI, T. C. dos S. *Introdução à Engenharia de Software*. Primeira. [S.l.]: Editora da Unicamp, 2001.

CEA, J. Numerical methods of shape optimal design. In: HAUG, E.; CEA, J. (Ed.). *Optimization of Distributed Parameter Structures*. Alphen aan den Rijn: Sijthoff & Noordhoff, 1981. p. 1049–1087.

CHOI, K.; CHANG, K. A study of design velocity field computation for shape optimal design. *Finite Elements in Analysis and Design*, v. 15, n. 4, p. 317–341, 1994.

CHOI, K.; HAUG, E. Shape design sensitivity analysis of elastic structures. *Journal of Structural Mechanics*, v. 11, p. 231–269, 1983.

CHOI, K.; KIM, N. H. *Structural Sensitivity Analysis and Optimization 1*. Berlin: Springer, 2005.

CHOI, K.; KIM, N. H. *Structural Sensitivity Analysis and Optimization 2*. Berlin: Springer, 2005.

CHOI, K.; SANTOS, J. Design sensitivity analisys for nonlinear structural systems part I: Theory. *International Journal for Numerical Methods in Engineering*, v. 24, n. 1, p. 2039–2055, 1987.

CHOI, K.; SEONG, H. A domain method for shape design sensitivity analysis of built-up structures. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 1–15, 1986.

CHRISTENSEN, P. W.; KLARBRING, A. *An Introduction to Structural Optimization*. USA: Springer, 2009.

CIMME. *GiD 9.0.6-Reference Manual*. Spain, 2008. Disponível em: <http://gid.cimne.upc.es/>. Acesso em: 01 abril. 2010.

COMPASS. *CompassFem 8.0.8 R3-Reference Manual*. Spain, 2008. Disponível em: <http://www.compassis.com/compass-site/en/introduccion/index.html>. Acesso em: 15 junho. 2008.

COSTA, G. L. V. hp^2*FEM - Uma Aqrquitetura de Software* p *Não-Uniforme para o Método de Elementos Finitos de Alta Ordem*. Dissertação (Mestrado) — Faculdade de Engenharia Mecânica, UNICAMP, Campinas-SP, Agosto 2012.

COURANT, R. Variational methods for the solution of problems of equilibrium and vibration. *Bulletin of the American Mathematical Society*, v. 49, p. 1–23, 1943.

DING, Y. Shape optimization of structures: A literature survey. *Computers & Structures*, v. 24, n. 6, p. 985–1004, 1986.

DONG, S.; YOSIBASH, Z. A paralell spectral element method for dynamic three-dimensional nonlinear elasticity problems. *Computers and Structures*, v. 87, p. 59–72, 2009.

DRIEMEIER, L. *Aplicação Do Conceito de Derivada Topológica Na Otimização Estrutural de Problemas Da Elasticidade.* Dissertação (Mestrado) — DPM/FEM/UNICAMP, Campinas, Fevereiro 2002.

EVGRAFOV, A.; PATRIKSSON, M. Stochastic structural topology optimization: discretization and penalty function approach. *Struct Multidisc Optim*, v. 25, p. 174–188, 2003.

EVSUKOFF, A. *Sobre a Introdução de um Algoritmo de Ponto Interior no Ambiente de Projeto de Engenharia*. Dissertação (Mestrado) — COPPE/UFRJ, Rio de Janeiro, 1992.

FANCELLO, E. *Análise de sensibilidade, geração automática de malhas e o método dos elementos finitos na otimização de forma em problemas de contato e mecânica da fratura*. Tese (Doutorado) — COPPE/UFRJ, 1993.

FANCELLO, E.; HASLINGER, J.; FEIJÓO, R. Numerical comparison between two cost functions in contact shape optimization. *Structural Optimization*, v. 9, n. 1, p. 57–68, 1995.

FERREIRA, W. *Desenvolvimento de Ferramentas Computacionais Para Análise Estrutural Em Fadiga e Geração de Malhas de Elementos Finitos*. Dissertação (Mestrado) — DPM/FEM/UNICAMP, February 2002.

FISCHER, K. A.; WRIGGERS, P. Frictionless 2d contact formulation for finite deformations based on the mortar method. *Computer Mechanics*, v. 36, p. 226–244, 2005.

FISCHER, K. A.; WRIGGERS, P. Mortar based frictional contact formulation for higher order interpolations using the moving cone. *Computer Methods Applied in Mechanics and Engineering*, v. 195, p. 5020–5036, 2006.

FURLAN, F. A. C. *Métodos locais de integração explícito e implícito aplicados ao método de elementos finitos de alta ordem*. Dissertação (Mestrado) — Faculdade de Engenharia Mecânica, UNICAMP, Campinas-SP, Julho 2011.

GIANNAKOPOULOS, A. E. The return mapping method for the integraton of friction constitutive relations. *Computers and Structures*, v. 32, n. 1, p. 157–167, 1989.

HAFTKA, R.; ADELMAN, H. Recent developments in structural sensitivity analysis. *Structural Optimization*, v. 1, n. 3, p. 137–151, 1989.

HAFTKA, R.; GRANDHI, R. Structural shape optimization — a survey. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 91–106, 1986.

HAUG, E.; CHOI, K.; KOMKOV, V. *Design sensitivity analysis of structural systems*. Orlando: Academic Press, 1986. (Mathematics in Science and Engineering, v. 177).

HEEGE, A.; ALART, P. A frictional contact element for strongly curved contact problems. *International Journal for Numerical Methods in Engineering*, v. 39, p. 165–184, 1996.

HEISSERER, U. et al. On volumetric locking-free behavior of p-version finite elements under finite deformations. *Communications in Numerical Methods in Engineering*, v. 24, p. 1019–1032, 2008.

HERSKOVITS, J. A two-stage feasible directions algorithm for nonlinear constrained optimization. *Mathematical Programming*, v. 36, p. 19–38, 1986.

HERSKOVITS, J.; COELHO, C. An interior points algorithm for structural optimization problems. In: BREBBIA, C.; HERNANDEZ, S. (Ed.). *Computer Aided Optimum Design of Structures: Recent Advances - Proceedings of the First International Conference, Southampton, June, 1989*. Southampton: Computational Mechanics Publications - Springer Verlag, 1989. p. 231–241.

HERSKOVITS, J. et al. Contact shape optimization: a bilevel programming approach. *Struct Multidisc Optim*, v. 20, p. 214–221, 2000.

HERTZ, H. Über die berührung fester elastischer körper. *Journal für die reine und angewandte Mathematik*, v. 92, p. 156–171, 1881.

HILD, P. Éléments finis non conformes pour un problème de contact unilatéral avec frottement. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, v. 324, n. 6, p. 707 – 710, 1997.

HOLZAPFEL, G. A. *Nonlinear Solid Mechanics*. 1st. ed. England: Willey, 2000.

HORTON'S, I. *Beginning Visual C++ 2005*. Indianapolis, IN: Wiley Publishing, Inc., 2006.

HRENNIKOFF, H. Solution of problems in elasticity by framework method. *Journal of Applied Mechanics*, A8, n. 1, p. 169–175, 1941.

HUGHES, T. J. R. *The Finite Element Method*: Linear static and dynamic finite element analysis. Mineola, New York: Dover Publications, Inc., 2000.

JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. *The Unified Software Development Process*. Reading (MA): Addison Wesley, 1999.

JEREMIC, B.; XENOPHONTOS, C. Application of the p-version of the finite element method to elastoplasticity with localization of deformation. *Communication in Numerical Methods In Engineering*, v. 15, p. 867–876, 1999.

JOHNSON, C. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge: Cambridge University Press, 1990.

JU, S. H.; ROWLANDS, R. E. A three-dimensional frictional contact element whose stiffness matrix is symmetric. *ASME Journal of Applied Mechanics*, v. 66, p. 460–467, June 1999.

JU, S. H.; STONET, J. J.; ROWLANDS, R. E. A new symmetric contact element stiffness matrix for frictional contact problems. *Computers and Structures*, v. 54, n. 2, p. 289–301, June 1995.

KARNIADAKIS, G. E.; SHERWIN, S. J. *Spectral/hp Element Methods for CFD*. Oxford: Oxford University Press, 1999.

KARNIADAKIS, G. E.; SHERWIN, S. J. *Spectral/hp Element Methods for Computaional Fluid Dynamics*. 2nd. ed. Oxford: Oxford University Press, 2005.

KIKUSCHI, N.; ODEN, J. T. *Contact Problems in Elasticity: A Study of Variational Inequalities and Finite Element Methods*. Philadelphia: SIAM, 1988.

KIM, N.; CHOI, K.; CHEN, J. Die shape design optimization of sheet metal stamping process using meshfree method. *International Journal for Numerical Methods in Engineering*, v. 51, p. 1385–1405, 2001.

KIM, N.; CHOI, K.; CHEN, J. S. Shape design sensitivity analisys and optimization of elasto-pasticity with frictional contact. *AIAA Journal*, v. 38, n. 9, p. 1742–1753, 2000.

KIM, N.; PARK, Y.; CHOI, K. Optimization of a hyper-elastic structure with multibody contact using continuum-based shape design sensitivity analysis. *Structural and Multidisciplinaty Optimization*, v. 21, n. 3, p. 196–208, 2001.

KIM, N. H. *Shape Design Sensitivity Analysis and Optimization of Nonlinear Static/Dynamic Structures with Contact/Impact*. Tese (Doutorado) — The University of Iowa, May 1999.

KIM, N. H.; KYUNG, K. Y.; CHOI, K. K. A material derivative approach in design sensitivity analysis of three-dimensional contact problems. *International Journal of Solids and Structures*, v. 39, p. 2087–2108, 2002.

KONYUKHOV, A.; GENE, M. W.; WALL, W. A. A finite deformation mortar contact formulation using a prima-dual active set strategy. *International Journal for Numerical Methods in Engineering*, v. 79, p. 1354–1391, 2009.

KORELC, J. z e; KRISTANIé, N. Evaluation of design velocity field by direct differentiation of symbolically parameterized mesh. In: nATE, E. O.; OWEN, D. R. J. (Ed.). Barcelona: CIMNE, 2005. p. 1–4.

KOVáCS, B.; SZABó, F.; SZOTA, G. A generalized shape optimization procedure for the solution of linear partial diferential equations with application to multidisciplinary optimization. *Struct Multidisc Optim*, v. 21, p. 327–331, 2001.

KRUCHTEN, P. *Introdução ao RUP*: Rational unified process. Quarta. Rio de Janeiro: Editora Ciência Moderna Ltda, 2003.

LAURSEN, T. *Computational Contact and Impact Mechanics*. Berlin: Springer, 2002.

LAURSEN, T.; SIMO, J. Algorithmic symemtrization of coulomb frictional problems using augmented lagrangians. *Comp. Meth. Appl. Mecha. Engrg.*, n. 108, p. 133–146, 1993.

LEE, R. C.; TEPFENHART, W. M. *UML and C++*: A pratical guide to oject-oriented development. Second. New Jersey: Prentice-Hall, 2001.

LI, W. et al. An evolutionary approach to elastic contact optimization of frames structures. *Finite Elements in Analysis and Design*, v. 40, p. 31–81, 2003.

LI, W. et al. An evolutionary shape optimizaton procedure for contct problems in mechanical designs. *Proceeding of the Institution of Mechanical Engineers-C*, v. 217, p. 435–446, 2003.

LIBERTY, J. *Aprenda em 24 horas C++*. Rio de Janeiro: Campus, 1998.

MCDEVITT, T.; LAURSEN, T. A. A mortar finite element formulation for frictional contact problems. *International Journal For Numerical Methods in Engineering*, v. 48, p. 1525–1547, 2000.

MIJAR, A. R.; ARORA, J. S. Review of formulations for elstostatic frictional contact problems. *Struct. Multidisc. Optim.*, v. 20, n. 3, p. 167–189, November 2000.

MIJAR, A. R.; ARORA, J. S. Study of variational inequality an equality formulations for elastostatic friccional contact problems. *Archives of Comptational Methods in Engineering*, v. 7, n. 4, p. 387–449, 2000.

MIJAR, A. R.; ARORA, J. S. An augmented lagrangian optimization method for contact analysis problems, 1: formulation and algorithm. *Struct. Multidisc. Optim.*, v. 28, p. 99–112, 2004.

MIJAR, A. R.; ARORA, J. S. An augmented lagrangian optimization method for contact analysis problems, 2: numerical evaluation. *Struct. Multidisc. Optim.*, v. 28, p. 113–126, 2004.

MIJAR, A. R.; ARORA, J. S. Return mapping procedure for frictional force calculation: Some insights. *Journal of Engineering Mechanics*, p. 1004–1012, October 2005.

MIRISOLA, M. H. B. *Otimização da Superfície de Contato do Olhal de uma Biela Utilizando Elementos Finitos*. Dissertação (Mestrado) — Faculdade de Engenharia Mecânica, UNICAMP, Campinas-SP, Novembro 2009.

MOK, D. et al. Algorithmic aspects of deformation dependent loads in non-linear statitic finite element analysis. *Engineering Computations*, v. 16, n. 5, p. 601–618, 1999.

Muñoz-Rojas, P. A.; FONSECA, J. S. O.; CREUS, G. J. A modified finite difference sensitivity analysis method allowing remeshing in large strain path-dependent problems. *International Journal for Numerical Methods*, n. 61, p. 1049–1071, 2004.

ODEN, J. T. A general theory of finite elements; ii applications. *International Journal for Numerical Methods in Engineering*, v. 1, p. 247–259, 1969.

ODEN, J. T. *Finite Elements of Nonlinear Continua*. New York: McGraw-Hill, 1972. (Advanced Engineering Series).

OGDEN, R. W. *Non-Linear Elastic Deformations*. 1st. ed. USA: Dover Publications, 1997.

PáCZELT, I.; MRóZ, Z. On optimal contact shapes genertaed by wear. *International Journal for Numerical Methods in Engineering*, v. 63, p. 1250–1287, 2005.

PáCZELT, I.; MRóZ, Z. Optimal shapes of contact interfaces due to sliding wear in the steady relative motion. *International Journal of Solids and Structures*, v. 44, p. 895–925, 2007.

PáCZELT, I.; SZABó, T. Solution of contact optimization problems of cylindrical bodies using hp-fem. *International Journal for Numerical Methods in Engineering*, v. 53, p. 123–146, 2002.

PANAGIOTOPOULOS, P. *Inequaliy Problems in Mechanics and Applications*. Germany: Birkhäuser Boston Inc, 1985.

PEDERSEN, P. A direct analysis of elastic contact using super elements. *Computer Mechanics*, v. 37, p. 221–231, 2006.

PIEGL, L.; TILLER, W. *The NURBS Book*. Second. Berlin: Springer, 1997.

PUSO, M. A.; LAURSEN, T. A. A mortar segment-to-segment contact method for large deformation solid mechanics. *Computer Methods Applied in Mechanics and Engineering*, v. 193, p. 601–629, 2004.

PUSO, M. A.; LAURSEN, T. A. A mortar segment-to-segment frictional contact method for large deformations. *Computer Methods Applied in Mechanics and Engineering*, v. 193, p. 4891–4913, 2004.

QUATRANI, T. *Visual Modeling With Rational Rose 2000 and UML*. First. New Jersey: Addison-Wesley, 2000.

REDDY, J. N. *An Introduction to Nonlinear Finite Element Analysis*. First. New York: Oxford University Press, 2006.

REFAAT, M. H.; MEGUID, S. A. On the elastic solution of frictional contact problems using variational inequalities. *Int. J. Mech. Sci. Vol*, v. 36, n. 4, p. 329–342, 1994.

REFAAT, M. H.; MEGUID, S. A. On the modeling of frictional contact problems using variational inequalities. *Finite Elements in Analysis and Design*, n. 19, p. 89–101, 1995.

REFAAT, M. H.; MEGUID, S. A. A new strategy for the solution of frictional contact problems. *International Journal for Numerical Methods in Engineering*, v. 43, p. 1053–1068, 1998.

REGISTER, A. H. *A Guide to MATLAB® Object-Oriented Programming*. 1st. ed. [S.l.]: Chapman and Hall, 2007.

REKTORYS, K. *Variational Methods In Mathematics, Science And Engineering*. Second. London-England: D. Reidel Publishing Company, 1980.

ROGERS, D. F.; ADAMS, J. A. *Mathematical Elements for Computer Graphics*. 2nd. ed. Mexico: McGraw-Hill, 1990.

RUMBAUGH, J.; JACOBSON, I.; BOOCH, G. *The Unified Modeling Language Reference Manual*. Reading (MA): Addison Wesley, 1999.

SCHLEUPEN, A.; MAUTE, K.; RAMM, E. Adaptive fe-procedures in shape optimization. *Struct. Multidisc. Optim.*, n. 19, p. 282–302, 2000.

SCHMIT, L. Structural synthesis - its genesis and development. *AIAA Journal*, v. 19, n. 10, p. 1249–1263, 1981.

SCOTT, K. *O Processo Unificado Explicado*. Primeira. [S.l.]: Bookman, 2003.

SEONG, H.; CHOI, K. Boundary-layer approach to shape design sensitivity analysis. *Mechanics of Structures and Machines*, v. 15, n. 2, p. 241–263, 1987.

SERPA, A. L. *Problema de contato com atrito utilizando o Método do Lagrangiano Aumentado*. Tese (Doutorado) — Faculdade de Engenharia Mecânica, UNICAMP, Campinas-SP, Novembro 1996.

SHERWIN, S.; PEIRó, J. Mesh generation in curvilinear domain using high-order elements. *International Journal for Numerical Methods in Engineering*, v. 53, p. 207–223, 2002.

SIGNORINI, A. Sopra alcune quetioni di elastostatica. *Atti della Società Italiana per il Progresso delle Scienze*, 1933.

SIGNORINI, A. Questioni di elasticità non linearizzata e semilinearizzata. *Rend. de Matematica*, v. 5, n. 18, p. 95–139, 1959.

SILVA, C. *Otimização Estrutural e Análise de Sensibilidade Orientadas Por Objetos*. Dissertação (Mestrado) — DPM/FEM/UNICAMP, Campinas, 1997.

SILVA, C. *Análise de Sensibilidade, Algoritmos de Otimização e Orientação por Objetos em Hiperelasticidade Não-Linear.* Tese (Doutorado) — DPM/FEM/UNICAMP, Campinas, 2003.

SILVA, C.; BITTENCOURT, M. Análise de sensibilidade e otimização estrutural em problemas elastostáticos lineares. In: ABCM. *Proceedings of 14th Brazilian Congress of Mechanical Engineering*. Bauru, 1997.

SILVA, C.; BITTENCOURT, M. Aspects of three-dimensional structural shape optimization. In: ABCM. *Proceedings of 15th Brazilian Congress of Mechanical Engineering*. Águas de Lindóia, 1999. p. 1–10.

SILVA, C.; BITTENCOURT, M. Design sensitivity analysis expressions in elastic problems (in portuguese). *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería*, v. 5, n. 2, p. 243–268, 1999.

SILVA, C.; BITTENCOURT, M. Expressões de análise de sensibilidade em problemas elásticos. *Revista Internacional de Métodos Numéricos para Cálculo y Diseño en Ingeniería*, v. 5, n. 2, p. 243–268, 1999.

SILVA, C.; BITTENCOURT, M. An object-oriented structural optimization program. *Structural and Multidisciplinary Optimization*, v. 20, n. 2, p. 154–166, October 2000.

SILVA, C. A. C.; BITTENCOURT, M. L. Velocity fields using nurbs with distortion control for structural shape optimization. *Structural and Multidisciplinary Optimization*, v. 33, n. 2, p. 1004–1012, February 2007.

SIMO, J.; LAURSEN, T. An augmented lagrangian treatment of contact problems involving friction. *Computers and Structures*, v. 42, n. 1, p. 97–116, January 1992.

SOLíN, P. ˇ; SEGETH, K.; EL, I. D. z. *High-Order Finite Element Methods*. First. [S.l.]: Chapman & Hall/CRC, 2004.

SOTELINO, E.; CHEN, W.; WHITE, D. Future challenges for simulation in structural engineering. In: IDELSOHN, E. O. I.; DVORKIN, E. (Ed.). *Computational Mechanics - New Trends and Applications*. Barcelona: CIMNE, 1998. p. 1–18.

SOUZA NETO, E. A. de; PERIĆ, D.; OWEN, D. R. J. *Computational Methods for Plasticity*: Theory and applications. 1st. ed. USA: Wiley, 2009.

SPENCER, A. J. M. *Continuum Mechanics*. 1st. ed. USA: Dover Publications, 2004.

STUPKIEWICZ, S. et al. A finite deformation mortar contact formulation using a prima-dual active set strategy. *Computer Methods in applied mechanics and engineering*, v. 191, p. 3555–3581, 2002.

SZABÓ, B.; BABUSKA, I. *Finite Element Analysis*. [S.l.]: John Wiley & Sons, 1991.

THE MATHWORKS, Inc. *User Guide*. [S.l.], 2009.

TIMOSHENKO, S.; WOINOWSKY-KRIEGER, S. *Theory of Plates and Shells*. New York: McGraw-Hill, 1959.

TUR, M.; FUENMAYOR, F. J.; WRIGGERS, P. A mortar based frictional contact formulation for large deformations using lagrange multipliers. *Computer Methods Applied in Mechanics and Engineering*, v. 198, p. 2860–2873, 2009.

TURNER, M. J. et al. Stiffness and deflection analysis of complex structures. *Journal ofthe Aeronautical Sciences*, v. 23, p. 805–823, 1956.

VANDERPLAATS, G. Structural optimization - past, present and future. *AIAA Journal*, v. 20, n. 7, p. 992–1000, 1982.

VAZQUEZ, M. G. *Construção de Funções de Interpolação para as versões h e p do MEF através de Produto Tensorial*. Dissertação (Mestrado) — Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2004.

VAZQUEZ, T. *Funções de Interpolação e Regras de Integração Tensorizáveis para o Método dos Elementos Finitos de Alta Ordem*. Tese (Doutorado) — Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2008.

VONDRáK, V.; KOZUBEK, T. s. Parallel solution of contact shape optimization problems based on total feti domain decomposition method. *Struct. Multidisc. Optim.*, v. 42, p. 955–964, 2009.

VOS, P. E. J. *From h to p Efficiently*: Optimising the implementation of spectral/hp element methods. Tese (Doutorado) — Department of Aeronautics Imperial College London, 2011.

WELCH, B. B. *Pratical Programming in Tcl and Tk*. 3rd. ed. USA: Prentice Hall, 1999.

WOHLMUTH, B. I. *Discretization Methods and Iterative Solvers Based on Domain Decomposition*: Lecture notes in computational science and engineering. First. Heildelberg: Springer-Verlag, 2000.

WOHLMUTH, B. I. A mortar finite element method using dual spaces for the lagrange multiplier. *SIAM Journal on Numerical Analysis*, v. 38, p. 989–1012, 2000.

WRIGGERS, P. *Computational Contact Mechanics*. 2nd. ed. Netherlands: Springer, 2006.

YANG, B. *Mortar Finite Element Methods for Large Deformation Contact Mechanics*. First. USA: VDM Verlag Dr. Müller Aktiengesellschaft & Co., 2009.

YANG, R.; CHOI, K. Accuracy of finite element based shape design sensitivity analysis. *Journal of Structural Mechanics*, v. 13, n. 2, p. 223–239, 1985.

YAO, T.; CHOI, K. 3D shape optimal design and automatic finite element regridding. *International Journal for Numerical Methods in Engineering*, v. 28, n. 2, p. 369–384, 1989.

YOSIBASH, Z. et al. Axisymmetric pressure boundary loading for finite deformation analysis using p-fem. *Computer Methods in Applied Mechanics and Engineering*, v. 196, p. 1261–127, 2007.

YU, L.; KUMAR, A. An object-oriented modular framework for implementing the finite element method. *Computers & Structures*, v. 79, p. 919–928, 2001.

ZHANG, W.-H.; BECKERS, P.; FLEURY, C. A unified parametric design approach to structural shape optimization. *International Journal For Numerical Methods in Engineering*, v. 38, p. 2283–2292, 2000.

ZIENKIEWICZ, O.; CAMPBELL, J. Shape optimization and sequential linear programming. In: GALLAGHER, R.; ZIENKIEWICZ, O. (Ed.). *Optimum Structural Design - Theory and Applications*. New York: John Wiley, 1973. p. 109–125.

ZIENKIEWICZ, O. C.; CHEUNG, Y. K. *The Finite Element Method in Structural and Continuum Mechanics*. First. [S.l.]: McGraw-Hill Publishing Company Ltd, 1967.

ZIENKIEWICZ, O. C.; TAYLOR, R. L. *The Finite Element Method*. Fourth. [S.l.]: McGraw-Hill International Editions, 1989.

ZOLÉSIO, J. The material derivative (or speed) method for shape optimization. In: HAUG, E.; CEA, J. (Ed.). *Optimization of Distributed Parameter Structures*. Alphen aan den Rijn: Sijthoff & Noordhoff, 1981.

APÊNDICE A Elasticidade Plana

Basicamente, na elasticidade plana, a análise resume-se ao plano formado pelos eixos X_1 e X_2. Existem dois tipos de problemas em elasticidade plana: *estado plano de tensão* e *estado plano de deformação*. A seguir, são apresentadas as formulações para esses dois tipos de problemas empregados na engenharia através do MEF.

A.1 Estado Plano de Deformação

No caso de problemas de estado plano de deformação, não existe deformação na direção do eixo X_3, ou melhor, no sentido da espessura. O gradiente de deformação pode ser expresso da seguinte forma

$$\mathbf{F} = \begin{bmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & 0 \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{A.1}$$

Entretanto a melhor forma de expressar o gradiente de deformação é levar em consideração somente as componentes X_1 e X_2 (e suas contrapartidas espaciais x_1 e x_2). Desta forma, escreve-se o gradiente de deformação na forma 2D ($\mathbf{F}_p$) como

$$\mathbf{F}_p = \begin{bmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} \end{bmatrix}. \tag{A.2}$$

As equações constitutivas podem ser escritas em função do gradiente de deformação $\mathbf{F}_p$, isto é, o segundo tensor de tensão de Piola-Kirchhoff e o tensor de tensão de Cauchy são dados por

$$\mathbf{S} = \lambda[\ln(\det \mathbf{F}_p)\mathbf{C}_p^{-1} + \mu(\mathbf{I} - \mathbf{C}_p^{-1})], \tag{A.3}$$

$$\boldsymbol{\sigma} = \frac{\lambda}{\det \mathbf{F}_p} \ln(\det \mathbf{F}_p)\mathbf{I} + \frac{\mu}{\det \mathbf{F}_p}(\mathbf{b} - \mathbf{I}), \tag{A.4}$$

sendo $\mathbf{C} = \mathbf{F}_p^T\mathbf{F}_p$ e $\mathbf{b} = \mathbf{F}_p\mathbf{F}_p^T$, respectivamente o tensor à direita e à esquerda de Cauchy-Green.

A componente de tensão referente ao eixo X_3 pode ser expressa na descrição material S_{33} ou na descrição espacial σ_{33} como

$$S_{33} = \lambda \ln(\det \mathbf{F}_p), \tag{A.5}$$

$$\sigma_{33} = \frac{\lambda}{\det \mathbf{F}_p} \ln(\det \mathbf{F}_p). \tag{A.6}$$

A.2 Estado Plano de Tensão

Para o problema de estado plano de tensão, a situação é um pouco mais complicada. Algumas considerações são feitas a seguir com relação à componente referente a tensão na direção de X_3.

O gradiente de deformação possui a seguinte forma

$$\begin{bmatrix} \frac{\partial x_1}{\partial X_1} & \frac{\partial x_1}{\partial X_2} & 0 \\ \frac{\partial x_2}{\partial X_1} & \frac{\partial x_2}{\partial X_2} & 0 \\ 0 & 0 & F_{33} \end{bmatrix}. \tag{A.7}$$

O determinante do gradiente de deformação pode ser expresso por

$$\det \mathbf{F} = F_{33} \det \mathbf{F}_p. \tag{A.8}$$

Deve-se, em primeiro lugar, determinar F_{33} a partir da condição $\sigma_{33} = 0$ (condição de estado plano de tensão). Partindo do modelo tridimensional, é possível identificar a seguinte relação

$$\sigma_{33} = \frac{1}{F_{33} \det \mathbf{F}_p}[\lambda \ln(F_{33} \det \mathbf{F}_p) + \mu(F_{33}^2 - 1)] = 0 \tag{A.9}$$

Sabendo que em um problema de estado plano de tensão a espessura precisa ser pequena, e que o termo F_{33} é tende a 1. Usando a série de Taylor nota-se que próximo de 1 pode-se afirmar

segundo (BHATTI, 2006) que

$$(F_{33} - 1) \approx 2\ln(F_{33}). \tag{A.10}$$

Usando a simplificação (A.10) na equação (A.9), obtém-se a seguinte expressão

$$\ln(F_{33}) = -\frac{\lambda}{\lambda + 2\mu}\ln(\det \mathbf{F}_p). \tag{A.11}$$

Agora, fazendo uso da relação (A.8), chega-se na expressão

$$\ln(\det \mathbf{F}) = \ln F_3 3 + \ln(\det \mathbf{F}_p) = -\frac{\lambda}{\lambda + 2\mu}\ln(\det \mathbf{F}_p) + \ln(\det(\mathbf{F}_p)). \tag{A.12}$$

Logo,

$$\ln(\det \mathbf{F}) = \frac{2\mu}{\lambda + 2\mu}\ln(\det \mathbf{F}_p) \equiv \ln(\det \mathbf{F}_p^{\gamma}) \Rightarrow \det \mathbf{F} = \det \mathbf{F}_p^{\gamma} \tag{A.13}$$

sendo

$$\gamma = \frac{2\mu}{\lambda + 2\mu}. \tag{A.14}$$

Finalizando, as equações constitutivas para um problema de estado plano de tensão pode ser escrita da seguinte maneira

$$\mathbf{S} = \lambda[\ln(\det \mathbf{F}_p^{\gamma})\mathbf{C}_p^{-1} + \mu(\mathbf{I} - \mathbf{C}_p^{-1})], \tag{A.15}$$

$$\boldsymbol{\sigma} = \frac{\lambda}{\det \mathbf{F}_p^{\gamma}}\ln(\det \mathbf{F}_p)\mathbf{I} + \frac{\mu}{\det \mathbf{F}_p}(\mathbf{b} - \mathbf{I}), \tag{A.16}$$

Considerando-se $\gamma = 1$, as equações constitutivas anteriores ficam iguais as equações encontradas para o problema de estado plano de deformação.

APÊNDICE B Obtenção de Soluções Analíticas para Problemas de Elasticidade

Para a validação da solução aproximada usando o MEF-AO, faz-se um estudo de convergência baseado na comparação dos resultados obtidos variando-se o grau das funções de forma com uma solução analítica existente, como exemplificado a seguir.

Suponha um cubo que ocupe inicialmente uma configuração material $0 \leq X_1 \leq 1$, $0 \leq X_2 \leq 1$ e $0 \leq X_3 \leq 1$, com as propriedades do material, módulo de elasticidade E e coeficiente de Poisson ν, conhecidas. Primeiro, define-se uma solução analítica em termos das coordenadas materiais ou espaciais. Por exemplo, pode-se definir o seguinte campo de deslocamento $\mathbf{u} = (u_1; u_2; u_3)$ na descrição material

$$\begin{cases} u_1 = A \sin(a\, X_1) - X_1, \\ u_2 = 0,0, \\ u_3 = 0,0, \end{cases} , \tag{B.1}$$

sendo A e a parâmetros constantes. É importante notar que este campo de deslocamento implica na condição de contorno de Dirichlet $(u_1, u_2, u_3) = (0, 0, 0)$ na face $X = 0$.

O segundo passo é determinar a equação constitutiva apropriada ao problema. Por exemplo, para o caso de um material neo-Hookeano, definido na seção 4.3.5, utiliza-se a equação (4.54) na descrição material ou a equação (**??**) na descrição espacial. No caso do campo de deslocamento proposto, emprega-se a equação (4.54) para determinar o segundo tensor de tensão de Piola-Kirchhoff $\mathbf{S}$. Porém, no caso da descrição material é necessário transformar o tensor de tensão $\mathbf{S}$ no primeiro tensor de tensão de Piola-Kirchhoff $\mathbf{P}$. Essa transformação é dada pela equação $\mathbf{P} = \mathbf{FS}$, sendo $\mathbf{F}$ o gradiente de deformação.

Utilizando a equação de equilíbrio estática na descrição material

$$\text{Div}\mathbf{P} + \rho_0 \mathbf{f} = \mathbf{0} \rightarrow \rho_0 \mathbf{f} = -\text{Div}\mathbf{P} \tag{B.2}$$

sendo ρ_0 a densidade inicial, determina-se a seguinte carga de corpo para o campo (B.1)

$$\begin{cases} f_1 = -\dfrac{\sin(aX_1)\left\{\lambda+\mu-\lambda\ln[A(1-2\sin^2(\frac{aX_1}{2}))]-\lambda\ln(a^2)+\mu A^2a^4cos^2(aX_1)\right\}}{-Aa(\cos^2(aX_1)}, \\ f_2 = 0, \\ f_3 = 0. \end{cases}, \tag{B.3}$$

sendo λ e μ os coeficientes de Lamé definidos pelas equações (4.53a) e (4.53b), respectivamente.

Por último, determinam-se as forças de tração aplicadas nas faces livres, ou seja,

$$\mathbf{t} = \mathbf{P}\mathbf{n}, \tag{B.4}$$

sendo $\mathbf{n} = [n_1; n_2; n_3]$ o vetor normal à superfície tracionada. Dessa forma, a força de tração aplicada em cada uma das superfícies é dada por

$$\begin{cases} t_1 = n_1\left\{Aa^2\cos(aX_1)\mu - \dfrac{\mu-\lambda\ln\left[Aa^2\cos(aX_1)\right]}{Aa^2\cos(aX_1)}\right\}, \\ t_2 = n_2\left\{\lambda\ln[Aa^2\cos(aX_1)]\right\}, \\ t_3 = n_3\left\{\lambda\ln[Aa^2\cos(aX_1)]\right\}, \end{cases} \tag{B.5}$$

Os vetores normais às superfícies tracionadas estão resumidos na Tabela B.1

Tabela B.1: Definição dos vetores normais às superfícies tracionadas do exemplo estudado.

Face	vetor normal ($\mathbf{n}$)
$X_1 = 1$	$\mathbf{n} = [1; 0; 0]$
$X_2 = 0$	$\mathbf{n} = [0; -1; 0]$
$X_2 = 1$	$\mathbf{n} = [0; 1; 0]$
$X_3 = 0$	$\mathbf{n} = [0; 0; -1]$
$X_3 = 1$	$\mathbf{n} = [0; 0; 1]$

Assim, pode-se realizar o teste de convergência espacial para o MEF-AO, como foi apresentado no teste 6 da Seção 8.

APÊNDICE C Normas

Do ponto de vista matemático, a norma de um entidade matemática é uma quantidade que pode representar, em alguns casos, uma medida de comprimento, tamanho ou a extensão dessa entidade. Como exemplo de normas, têm-se as normas L^2 e L^∞.

A norma L^2 de uma função $\mathbf{u}(\mathbf{x})$ é representada por um produto interno da seguinte forma

$$\|\mathbf{u}\|_{L^2} = \sqrt{\int_a^b (\mathbf{u}(\mathbf{x}) \cdot \mathbf{u}(\mathbf{x}))dx}. \tag{C.1}$$

Quando se deseja calcular a norma L^2 do erro $\mathbf{e}(\mathbf{x})$ entre as soluções exata $\mathbf{u}(\mathbf{x})$ e aproximada $\mathbf{u}_{\text{mef}}(\mathbf{x})$, utiliza-se a seguinte equação

$$\|\mathbf{e}\|_{L^2} = \sqrt{\sum_{i=1}^{\text{nel}} \int_{\Omega_e} [\mathbf{u}(\mathbf{x}) - \mathbf{u}_{\text{mef}}(\mathbf{x})]dx}. \tag{C.2}$$

A expressão anterior é calculada para todos os pontos de integração do elemento e multiplicadas pelas ponderações. A norma do quadrado do erro no elemento é a soma das contribuições de cada elemento. A norma do erro para uma malha é a soma das contribuições de cada elemento.

A norma L^∞ é dada por

$$\|\mathbf{u}\|_{L^\infty} = \max \mathbf{u}(\mathbf{x}). \tag{C.3}$$

A partir daí, a norma L^∞ do erro da aproximação é dada por

$$\|\mathbf{e}\|_{L^\infty} = \max \mathbf{e}(\mathbf{x}). \tag{C.4}$$

Nesse caso, toma-se o máximo do erro em ponto de integração do elemento. Para uma malha, a norma do máximo é obtida tomando-se o máximo dos erros calculados para cada elemento.

APÊNDICE D Arquivo de Definições de Material

Listagem D.1: Arquivo de definição de material usado no GiD

```
BOOK:LINEAR
NUMBER: 1 MATERIAL: ELISO
QUESTION: E
VALUE: 2.07e11
QUESTION: NU
VALUE: 0.3
QUESTION: ALPHA
VALUE: 1.0
QUESTION: DENSITY
VALUE: 7810
END MATERIAL
BOOK:NON-LINEAR
NUMBER: 2 MATERIAL: MOONEY_RIVELN
QUESTION: E
VALUE: 20.7e3
QUESTION: NU
VALUE: 0.3
QUESTION: ALPHA
VALUE: 1.0
QUESTION: DENSITY
VALUE: 7.81e-3
END MATERIAL
```

APÊNDICE E Arquivo de Definições do Problema

Listagem E.1: Arquivo de definição das características do problema usado no GiD

```
PROBLEM DATA
BOOK: Problem Data
QUESTION: TITLE
VALUE: Title
QUESTION: Num_of_Load_Cases
VALUE: 1
QUESTION: Number_of_Primary_Fields
VALUE: 1
QUESTION: Primary_Fields(UX,UY,UZ)
VALUE: UX
QUESTION: Number_of_Mixed_Fields
VALUE: 0
QUESTION: Mixed_Fields(PRESSURE,TEMPERATUE)
VALUE: 0
QUESTION: Num_Secondary_Fields
VALUE: 0
QUESTION: Secondary_Fields(See_Documentation)
VALUE: 0
QUESTION: Number_of_Problem_Type(max4)
VALUE: 1
QUESTION: (1)Group_of_elements/Problem_Type
VALUE: Serendipty Plane_Stress
QUESTION: (2)Group_of_elements/Problem_Type
VALUE: 0
QUESTION: (3)Group_of_elements/Problem_Type
VALUE: 0
QUESTION: (4)Group_of_elements/Problem_Type
VALUE: 0
QUESTION: Number_of_Groups_with_Geometric_properites(max4)
VALUE: 0
```

```
QUESTION: (1)Number_of_Geometric_Properties/Value(s)
VALUE: 0
QUESTION: (2)Number_of_Geometric_Properties/Value(s)
VALUE: 0
QUESTION: (3)Number_of_Geometric_Properties/Value(s)
VALUE: 0
QUESTION: (4)Number_of_Geometric_Properties/Value(s)
VALUE: 0
BOOK: Solution Directives
QUESTION: Time_Dependency#CB#(STATIC,DYNAMIC_TRANSIENT,DYNAMIC_STEADY)
VALUE: STATIC
QUESTION: Kinematics#CB#(INFINITESIMAL,FINITE)
VALUE: FINITE
QUESTION: Solution_Algorithm#CB#(LINEAR_SOLUTION,NEWTON_RAPHSON)
VALUE: LINEAR_SOLUTION
QUESTION: Number_of_Load_Steps
VALUE: 1
QUESTION: Max_Iteration
VALUE: 100
QUESTION: Precision
VALUE: 0.0001
QUESTION: Linear_System_Solution#CB#(GAUSS,CGGS)
VALUE: CGGS
QUESTION: Max_Iteration_LS
VALUE: 10000
QUESTION: Precision_LS
VALUE: 0.0001
END PROBLEM DATA
INTERVAL DATA
BOOK: Interval Data
QUESTION: Load_Case_Num
VALUE: 1
END INTERVAL DATA
```

APÊNDICE F Arquivo de Condições do Problema

Listagem F.1: Arquivo de condições de problema usado no GiD

```
BOOK: Constraints
NUMBER: 1 CONDITION: Point-ElimDOF
CONDTYPE: over points
CONDMESHTYPE: over nodes
QUESTION: Elim_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UZ_DOF?#CB#(0,1)
VALUE: 0
END CONDITION
NUMBER: 2 CONDITION: Line-ElimDOF
CONDTYPE: over lines
CONDMESHTYPE: over nodes
QUESTION: Elim_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UZ_DOF?#CB#(0,1)
VALUE: 0
END CONDITION
NUMBER: 3 CONDITION: Surface-ElimDOF
CONDTYPE: over surfaces
CONDMESHTYPE: over nodes
QUESTION: Elim_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: Elim_UZ_DOF?#CB#(0,1)
VALUE: 0
```

```
END CONDITION
NUMBER: 4 CONDITION: Point-PrescDOF
CONDTYPE: over points
CONDMESHTYPE: over nodes
QUESTION: Presc_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UX_=
VALUE: 0.0
QUESTION: Presc_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UY_=
VALUE: 0.0
QUESTION: Presc_UZ_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UZ_=
VALUE: 0.0
END CONDITION
NUMBER: 5 CONDITION: Line-PrescDOF
CONDTYPE: over lines
CONDMESHTYPE: over nodes
QUESTION: Presc_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UX_=
VALUE: 0.0
QUESTION: Presc_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UY_=
VALUE: 0.0
QUESTION: Presc_UZ_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UZ_=
VALUE: 0.0
END CONDITION
```

```
NUMBER: 6 CONDITION: Surface-PrescDOF
CONDTYPE: over surfaces
CONDMESHTYPE: over nodes
QUESTION: Presc_UX_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UX_=
VALUE: 0.0
QUESTION: Presc_UY_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UY_=
VALUE: 0.0
QUESTION: Presc_UZ_DOF?#CB#(0,1)
VALUE: 0
QUESTION: UZ_=
VALUE: 0.0
END CONDITION
BOOK: Loads
NUMBER: 7 CONDITION: Point-Load
CONDTYPE: over points
CONDMESHTYPE: over nodes
QUESTION: Load_Case
VALUE: 1
QUESTION: FUX_=
VALUE: 0.0
QUESTION: FUY_=
VALUE: 0.0
QUESTION: FUZ_=
VALUE: 0.0
END CONDITION
NUMBER: 8 CONDITION: Line-Load
CONDTYPE: over lines
CONDMESHTYPE: over face elements
QUESTION: Load_Case
```

```
VALUE: 1
QUESTION: FUX_=
VALUE: 0.0
QUESTION: FUY_=
VALUE: 0.0
QUESTION: FUZ_=
VALUE: 0.0
QUESTION: System?#CB#(G,L)
VALUE: G
END CONDITION
NUMBER: 9 CONDITION: Surface-Load
CONDTYPE: over surfaces
CONDMESHTYPE: over face elements
QUESTION: Load_Case
VALUE: 1
QUESTION: FUX_=
VALUE: 0.0
QUESTION: FUY_=
VALUE: 0.0
QUESTION: FUZ_=
VALUE: 0.0
QUESTION: System?#CB#(G,L)
VALUE: G
END CONDITION
NUMBER: 10 CONDITION: Volume-Load
CONDTYPE: over volumes
CONDMESHTYPE: over body elements
QUESTION: Load_Case
VALUE: 1
QUESTION: ACEL_UX_=
VALUE: 0.0
QUESTION: ACEL_UY_=
VALUE: 0.0
```

```
QUESTION: ACEL_UZ_=
VALUE: 0.0
END CONDITION
BOOK: Geometry Information
NUMBER: 11 CONDITION: Volume-FEGroups
CONDTYPE: over volumes
CONDMESHTYPE: over body elements
QUESTION: FEGroup
VALUE: 1
END CONDITION
```