TESE DEFENDIDA POR LUCIANO SANTOS DRIEMEIER E APROVADA PELA COMISSÃO JULGADORA EM 26/02/2002

ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

Aplicação do Conceito de Derivada Topológica na Otimização Estrutural de Problemas da Elasticidade

Autor: Eng. Luciano Santos Driemeier
Orientador: Prof. Dr. Marco Lúcio Bittencourt

08/02

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

Aplicação do Conceito de Derivada Topológica na Otimização Estrutural de Problemas da Elasticidade

Autor: Eng. Luciano Santos Driemeier
Orientador: Prof. Dr. Marco Lúcio Bittencourt

Curso: Engenharia Mecânica
Área de Concentração: Mecânica dos Sólidos

Dissertação de mestrado acadêmico apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para a obtenção do título de Mestre em Engenharia Mecânica.

CAMPINAS, 2002
SP - BRASIL

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

D831a

Driemeier, Luciano Santos
Aplicação do conceito de derivada topológica na otimização estrutural de problemas da elasticidade / Luciano Santos Driemeier.--Campinas, SP: [s.n.], 2002.

Orientador: Marco Lúcio Bittencourt
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Otimização estrutural. 2. Análise de sensibilidade. 3. Engenharia de software. 4. Métodos dos elementos finitos. I. Bittencourt, Marco Lúcio. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
COMISSÃO DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

DISSERTAÇÃO DE MESTRADO ACADÊMICO

Aplicação do Conceito de Derivada Topológica na Otimização Estrutural de Problemas da Elasticidade

Autor: Eng. Luciano Santos Driemeier
Orientador: Prof. Dr. Marco Lúcio Bittencourt

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

Prof. Dr. Francisco Antonio Menezes
DE/FEC

Prof. Dr. Janito Vaqueiro Ferreira
DMC/FEM

Campinas, 26 de fevereiro de 2002.

Dedicatória

Como minha irmã, às pessoas de minha vida: Hélio, Anita, Larissa, Marta, Carina e Marina.

Sempre tive em minha família um grande guia moral e espiritual. Admiro e agradeço meus pais, Hélio e Anita, pelas profundas lições de honestidade, humildade, amor e trabalho que recebi durante toda minha vida. Tenho o orgulho e privilégio de poder contar com minhas duas irmãs, Larissa e Marta, que sempre foram para mim um exemplo de sucesso pessoal e profissional. Obrigado Carina por me mostrar que a vida ainda está baseada nas coisas simples e no amor. Devo a vocês tudo o que tenho.

Agradecimentos

Agradeço à FAPESP e à UNICAMP, sem as quais nada teria sido possível. Sinto-me privilegiado em poder ter usufruído do apoio financeiro e estrutural de duas instituições tão confiáveis e valiosas para nosso país.

Ao amigo Eng. MSc Antônio André Novotny pelas valiosas discussões e principalmente pela confiança ao ceder os resultados originais de sua futura tese de doutorado. Os conceitos de Análise de Sensibilidade à Mudança de Topologia (ASMT) apresentados nos Capítulos 2 e 3 foram desenvolvidos por A.A. Novotny para sua futura tese de doutorado, que será intitulada *Análise de Sensibilidade à Mudança de Topologia*, sob orientação dos Professores R. A. Feijóo, E. Taroco do LNCC e C. Padra do Centro Atómico Bariloche (Argentina). Os resultados fundamentais (formulação matemática da ASMT) foram gentilmente cedidos pelo A.A. Novotny em caráter excepcional tendo em vista colaborações realizadas em ocasiões anteriores entre A.A. Novotny, Prof. M.L. Bittencourt e Prof. R.A. Feijóo.

Ao meu amigo e orientador Prof. M.L. Bittencourt pelas incansáveis correções do meu texto.

Ao meu grande amigo Cláudio Alessandro Carvalho Silva pelas valiosas dicas para conclusão do meu trabalho.

Aos meus dois também grandes amigos Wallace e Alberto que dedicaram preciosas horas de seus tempos a mim, ajudando-me a concluir o trabalho.

A todos os amigos da FEM (colegas, professores e funcionários), os quais seria impossível citar separadamente. A convivência com as pessoas na UNICAMP sempre foi um grande aprendizado.

E, é claro, aos meus pais, às minhas irmãs e à Carina por todo o apoio sem o qual nada teria sentido.

*O*nde há dúvida, há liberdade.
Provérbio Latino

Resumo

DRIEMEIER, Luciano Santos, *Aplicação do Conceito de Derivada Topológica na Otimização Estrutural de Problemas da Elasticidade*, Campinas : Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2002. 106p. Dissertação (Mestrado)

O presente trabalho tem o objetivo de aplicar o conceito de derivada topológica na obtenção da topologia ótima em problemas da elasticidade. A derivada topológica fornece a sensibilidade de uma função custo, definida em todo o domínio, à criação de um furo. Nesse trabalho, a obtenção da derivada topológica está baseada nos conceitos de análise de sensibilidade à mudança de forma. Dessa forma, utilizam-se dois conceitos distintos para a realização de otimização topológica. No trabalho, apresentam-se resultados de otimização em domínios bi e tridimensionais. Esses resultados apresentaram ótima concordância quando comparados com soluções analíticas e resultados da literatura para domínios mais complexos. Mostra-se que a derivada topológica é uma poderosa ferramenta para obtenção da topologia ótima em problemas da elasticidade. Deve-se ressaltar que esse conceito pode ser aplicado em outros problemas da engenharia, tais como corrosão, microestrutura e fadiga.

Palavras Chave

- Derivada Topológica, Análise de Sensibilidade à Mudança de Forma, Otimização Topológica, Métodos dos Elementos Finitos, C++, Engenharia de Software

Abstract

DRIEMEIER, Luciano Santos, *Application of the Topological Derivative to the Structural Optimization of Elastic Problems*, Campinas : Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2002. 106p. Dissertação (Mestrado)

This work deals with the application of the topological derivative to obtain the optimal topology of elastic problems. The topological derivative defines the sensitivity of a cost function, defined over the domain, to the creation of a hole. In this work, the topological derivative is obtained based on shape design sensitivity analysis concepts. In this way, two distinct concepts are connected to perform the topology optimization. Results for the topological optimization of two-dimensional and three-dimensional domains are presented for several cases. These results are in total concordance when compared with analytical solutions and results from the literature for complex domains. The topological gradient is a powerful concept for obtaining optimal topology in elasticity problems. It is important to emphasize that this concept may be applied to other engineering problems such as corrosion, microstructure and fatigue.

Keywords

- Topological Gradient, Shape Design Sensitivity Analysis, Topological Optimization, Finite Element Method, C++, Software Engineering

Conteúdo

Lista de Figuras

Nomenclatura

Letras Latinas

$a(\cdot,\cdot)$ - operador bilinear, limitado e coercivo

b - forças de corpo

e - erro na aproximação

f - excitações no contorno

$g_T(\cdot,\cdot,...,\cdot)$ - expressão generalizada para o integrando de uma integral no contorno que pode expressar a análise de sensibilidade à mudança de forma de uma vasta classe de problemas de engenharia

$l(\cdot)$ - funcional linear limitado

u - variável primal de um problema

u_h - aproximação para a variável primal u de um problema

$\mathbf{n}$ - vetor normal a um ponto

$\mathbf{p}$ - vetor de variáveis de projeto

$\mathbf{r}$ - vetor resíduo

$\mathbf{u}$ - vetor de variáveis primais de um problema

$\mathbf{v}(\mathbf{x})$ - descrição espacial da velocidade

$\mathbf{x}^\tau$ - posição do ponto material $\mathbf{x}$ no tempo τ

$G_T(\hat{\mathbf{x}})$ - valor da derivada topológica em um ponto $\hat{\mathbf{x}}$ obtida em relação a um domínio inicial com furo

$G_T^*(\hat{\mathbf{x}})$ - valor da derivada topológica em um ponto $\hat{\mathbf{x}}$ obtida em relação a um domínio inicial sem furo

$H^n(\cdot)$ - espaço de Hilbert de ordem n

U - espaço das funções admissíveis

V - espaço das variações admissíveis

V' - espaço dual de V

$X^\tau(\mathbf{x},\tau)$ - função que determina a posição do ponto material $\mathbf{x}$ no tempo τ

$\mathbf{F}$ - vetor de carga nodal generalizado

$\mathbf{K}$ - matriz de rigidez global

$\mathcal{B}$ - região do espaço euclidiano associada a um corpo, domínio de um problema

$\mathcal{B}_0$ - domínio original de um problema

$\partial\mathcal{B}_0$ - contorno do domínio original de um problema

$\mathcal{B}_\tau$ - domínio perturbado de um problema

$\partial\mathcal{B}_\tau$ - contorno do domínio perturbado de um problema

$\mathcal{G}(\cdot, \mathbf{u}(\cdot))$ - forma geral de um funcional $\psi(\cdot)$
$\pounds_\tau$ - Lagrangeano de um problema na configuração perturbada
$\mathcal{T}$ - trajetória de um corpo

Letras Gregas

ς - vetor de incógnitas do problema adjunto
$\psi(\cdot)$ - funcional de um problema de otimização
$\dot{\psi}(\cdot)$ - variação do funcional de um problema de otimização
Φ_s - descrição espacial de um campo
Ψ_m - descrição material de um campo
Ω - domínio aberto e limitado
Γ - contorno de um determinado domínio
Γ_D - contorno submetido a condições de Dirichlet
Γ_N - controno submetido a condições de Neumann

Capítulo 1

Introdução

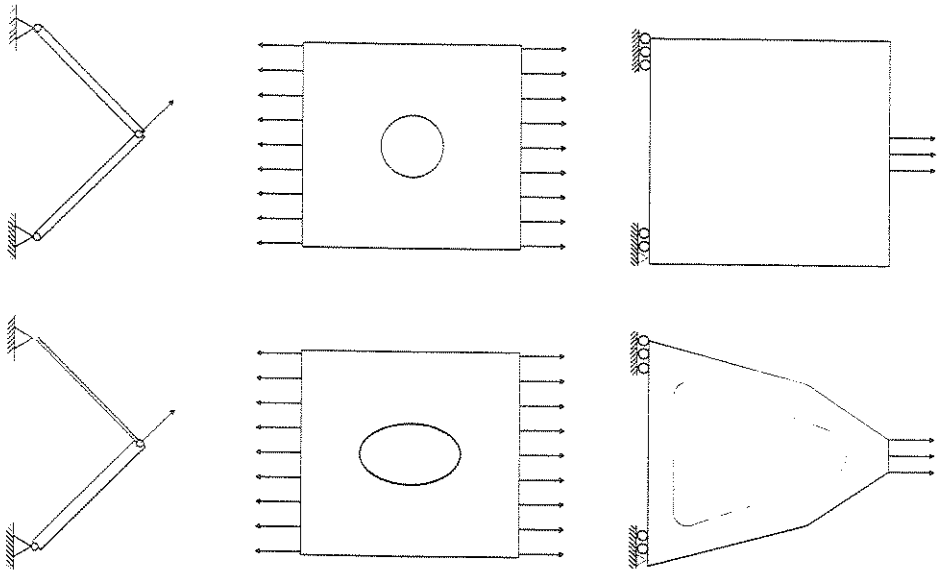
Durante muito tempo, o projeto foi considerado um processo baseado quase unicamente na experiência da equipe de trabalho envolvida. Conseqüentemente, o progresso nos mais diversos campos foi caracterizado por uma evolução lenta, consistindo na melhoria gradual de projetos já existentes em termos de atividades de tentativa e erro.

Em um ambiente com pouca demanda por inovações é razoável supor que os novos projetos sejam constituídos de pequenas alterações em trabalhos anteriores. Dessa forma o questionamento se a metodologia de tentativa e erro é a melhor linha de ação para a solução do problema, torna-se uma preocupação secundária. Porém, a agressiva competição tecnológica e a crescente exigência por novidades tornam essa abordagem inadequada. As demandas por alta qualidade, economia de energia e matéria-prima, baixo impacto ambiental e pequeno tempo de desenvolvimento exigem a criação de produtos para os quais inexiste experiência anterior.

Uma alternativa que tem apresentado sucesso é a metodologia de projeto baseada em otimização, que busca soluções ótimas possíveis direcionadas por critérios objetivos. Em outras palavras, procura-se sistematizar a 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.

Nos últimos anos, a questão de como se obter automaticamente a geometria ótima de um domínio em análise tem despertado muito interesse. Existem diferentes maneiras de se tratar essa questão. Na Figura 1.1, encontra-se a ilustração de uma possível divisão dos problemas de *otimização estrutural.* Na Figura 1.1a, tem-se um exemplo simples de otimização, na qual somente a seção tranversal da estrutura é otimizada. Uma maneira já consagrada na literatura de se obter a geometria ótima de um domínio, ilustrada na Figura 1.1b, é a otimização de forma. O incoveniente de se utilizar a otimização de forma é que seu resultado é obtido através de modificações na fronteira de uma geometria que deve ser previamente definida. Já a otimização topológica, como mostrado na Figura 1.1c, busca a topologia[1] ótima de um problema sem quaisquer hipóteses prévias sobre a geometria inicial.



(a) Otimização de tamanho. (b) Otimização de forma. (c) Otimização topológica.

Figura 1.1: Tipos de otimização (Cea et al., 1998).

A seguir, apresenta-se uma revisão bibliográfica do problema de otimização estrutural

[1]Forma viável que o domínio pode tomar.

com enfoque na otimização topológica.

1.1 Revisão Bibliográfica

A otimização topológica é um assunto relativamente novo e em rápida expansão no campo da mecânica estrutural. Um grande estímulo ao desenvolvimento dessa área está relacionado ao fato que resultados obtidos na otimização topológica podem resultar em economias bem mais significativas, quando comparadas com as otimizações de tamanho ou mesmo de forma (ver Figura 1.1). Devido à sua complexidade, os progressos na otimização topológica são freqüentemente interrompidos por inconsistências conceituais e confusões de terminologia. Dessa forma, necessita-se rever sistematicamente seus principais conceitos e estabelecer uma terminologia comum. Nessa revisão bibliográfica, será proposta uma terminologia baseada em (Rozvany, 2001), que permite fragmentar de maneira precisa os problemas de otimização topológica. A fragmentação facilita a compreensão dos tipos de métodos existentes bem como possibilita situar o trabalho aqui desenvolvido.

A otimização topológica pode ser dividida em dois grupos principais: otimização de *layout* (LO[2]) e otimização de forma generalizada (GSO[3]) ou otimização de forma com topologia variável. A *otimização de layout* se refere a problemas de redes de membros estruturais com pequenas frações de volume. A otimização de forma generalizada se refere a problemas com grandes frações de volume, nos quais se otimizam simultaneamente a topologia e a forma das fronteiras internas de um meio contínuo sólido, poroso ou compósito. As soluções podem ser exatas ou aproximadas para problemas dos dois grupos definidos.

Os princípios da teoria de otimização de *layout* foram desenvolvidos há quase 100 anos, na virada do século passado, no contexto de treliças. Esses princípios se devem a um versátil inventor australiano chamado Michell. A teoria de peso mínimo para treliças de Michell (1904) incluía, em sua essência, muito do que hoje é denominada *teoria do layout ótimo*. Muitas extensões da teoria de Michell foram exploradas por pesquisadores da equipe de

[2]Da língua inglesa, *Layout Optimization*.
[3]Da língua inglesa, *Generalized Shape Optimization*.

Rozvany nos anos 70. Publicações sobre essas extensões podem ser encontradas em (Rozvany, 1972a; Rozvany, 1972b). Posteriormente, na mesma linha de pesquisa, Prager, em conjunto com Rozvany, escreveu muitos artigos importantes, podendo-se citar (Prager e Rozvany, 1977; Rozvany e Prager, 1977). A teoria do *layout* foi consideravelmente generalizada nos anos 80 e 90, recentemente para estruturas com vários carregamentos e restrições em (Rozvany, 1992; Rozvany e Birker, 1994).

O primeiro passo na direção da otimização de forma generalizada (*GSO*) foi um artigo de Rossow e Taylor (Rossow e Taylor, 1973). Apesar do resultado da proposta de Rossow e Taylor não resultar em topologias com vazios, Taylor, em comunicações privadas com Rozvany, afirmou que, já no começo dos anos 70, sua intenção era obter cavidades ou furos em algumas áreas de um disco sendo otimizado. A maior contribuição de Taylor foi, provavelmente, a primeira conceituação da otimização de forma generalizada na história da mecânica.

O progresso na otimização de forma generalizada foi desencadeado por dois importantes acontecimentos:

Primeiro, uma nova classe de estruturas ótimas surgiu no começo dos anos 80. Foi encontrado em (Cheng e Olhoff, 1981) que otimizações de discos sólidos, baseadas em elementos finitos, resultavam em um sistema de tiras estruturais. Prager observou, em uma pequena carta antes de sua morte em 1980, que a distribuição dessas tiras era quase sempre idêntica à encontrada para redes de membros estruturais na otimização de *layout*. Similarmente, em outros trabalhos, regiões sólidas, vazias e porosas foram encontradas em projetos de seção tranversal ótima para torção na plasticidade.

Segundo, alguns estudos matemáticos estabeleceram microestruturas ótimas para discos perfurados. Mudanças na microestrutura implicam uma modificação na relação constitutiva do material. Dessa forma, relações constitutivas podem ser modificadas através da criação de materiais laminados que reduzem a massa e a resistência mecânica dos elementos. Logo, através de mudanças nas microestruturas, pode-se estabelecer em um problema discretizado, influências ponderadas dos elementos na massa e resistência mecânica do componente estru-

tural.

A partir desses resultados, derivaram-se os tensores de rigidez homogeneizados para as microestruturas ótimas estabelecidas (inicialmente com coeficiente de Poisson nulo) e a topologia ótima exata foi obtida para discos axissimétricos em flexão. Em um próximo passo, esses resultados foram estendidos para problemas com coeficiente de Poisson não nulo e discos compósitos de dois materiais. Todos os tensores de rigidez obtidos pelas técnicas de homogeneização foram os mesmos obtidos pelos princípios da mecânica ou da engenharia.

Uma outra linha de métodos está baseada no conceito de derivada topológica (Novotny et al., 2000; Novotny et al., 2001a; Novotny et al., 2002d; Novotny et al., 2002b; Novotny et al., 2002a; Novotny et al., 2002c; Novotny et al., 2001b). A Derivada Topológica (DT) fornece a sensibilidade de uma função custo, definida em todo o domínio de um problema, à criação de um furo. Schumacher introduziu a primeira definição da derivada topológica em (Schumacher, 1995). O objetivo de Schumacher era introduzir um pequeno furo no domínio e aumentá-lo através das ferramentas clássicas de otimização de forma. Esse procedimento tornou-se conhecido como o *método bolha.*

Em (Eschenauer et al., 1994) uma variação do método é proposta através da inserção iterativa de bolhas no domínio. Cada bolha inserida tem sua forma ótima determinada, resultando em uma estrutura topologicamente otimizada. Segundo essa abordagem, o cálculo de gradientes requer expandir em série de potências a solução do problema em um anel definido no domínio. Este fato depende fortemente do tipo de operador diferencial do problema e da possibilidade de expandi-lo em série, o que pode ser impraticável. Assim, apesar de sua generalidade, essa abordagem pode se tornar restritiva. Em (Cea et al., 1998) foi proposta uma forma heurística para calcular a derivada topológica via análise de sensibilidade à mudança de forma, dando bons resultados em alguns casos particulares.

Recentemente, em (Novotny et al., 2000; Novotny et al., 2001a), foi formalmente demonstrada a relação entre a derivada topológica e os conceitos de análise de sensibilidade à mudança de forma. Através da aplicação em problemas de elasticidade e de Poisson, a derivada topológica mostrou-se uma poderosa ferramenta na obtenção da topologia ótima.

1.1.1 Terminologia para Otimização Topológica

Para que se possa classificar os tipos de otimização topológica, faz-se necessária uma terminologia concisa. Para tanto, descreve-se nessa seção a terminologia encontrada em (Rozvany, 2001). Com isso, será possível mostrar mais eficientemente os principais métodos existentes para otimização de forma generalizada (*GSO*).

Primeiramente, define-se uma terminologia para classificação dos tipos de elementos encontrados em problemas de otimização topológica.

- **S** - sólido - elemento totalmente preenchido por material;
- **E**[4] - vazio - elemento que não contém material;
- **P** - poroso - elemento que contém material e vazios (cavidades);
- **C** - compósito - elemento que contém mais de um material e não contém vazios;
- **CP** - compósito-poroso - elemento que contém mais de um material e vazios.

Da mesma forma, define-se uma terminologia para definição dos tipos de topologia que podem ser encontradas.

- **ISE** (*Isotropic-Solid or Empty finite elements*) - topologias que contêm um conjunto de elementos finitos que podem estar completamente vazios ou preenchidos por materiais isotrópicos;
- **ASE** (*Anisotropic material, Solid or Empty finite elements*) - topologias que contêm um conjunto de elementos finitos que podem estar completamente vazios ou preenchidos por materiais anisotrópicos;
- **ISEP**, **ISEC** e **ISECP** (*Isotropic based material, Solid, Empty or Porous finite elements*) - topologias que contêm um conjunto de elementos finitos que podem apresentar porções vazias e preenchidas por materiais isotrópicos. As variações **ISEC** e

[4] Da língua inglesa, *Empty*.

ISECP têm o mesmo significado permitindo, respectivamente, a presença de mais de um matérial (compósito) e mais de um material associado a porosidades (compósito-poroso).

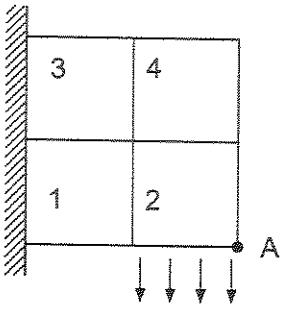
As topologias em análise nessa dissertação são do tipo *ISE*. A seguir, serão descritas as principais características e os principais métodos existentes para obtenção de topologias *ISE* ótimas.

1.1.2 Problemas de Topologia *ISE*

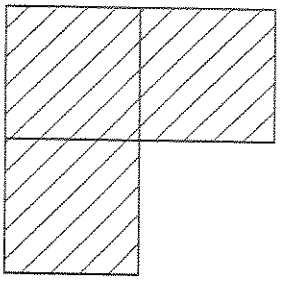
Os problemas de topologia *ISE* mais simples que podem ser encontrados são os de determinar a distribuição ótima **0-1** (i.e., vazio ou preenchido) de um único material em um domínio definido. Um exemplo elementar desse tipo está ilustrado na Figura 1.2a.

Para esse exemplo, tem-se quatro elementos finitos quadrados em estado plano de tensão. Os elementos 1 e 3 estão engastados pela sua aresta esquerda. O elemento 2 está submetido a um carregamento uniformemente distribuído em sua aresta inferior. O carregamento deve ser transmitido para os suportes de forma que o deslocamento no ponto A (ver Figura 1.2a) seja minimizado, qualquer elemento possa ser sólido ou vazio e a razão entre os volumes final e inicial não exceda $0,75$.

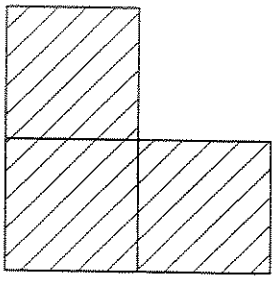
Para a razão proposta de $0,75$ existem quatro soluções possíveis. Uma solução possível mas inviável está ilustrada na Figura 1.2b. A solução ótima para o problema está ilustrada na Figura 1.2c. Uma das soluções possíveis, mas não ótima, está ilustrada na Figura 1.2d. Um problema de otimização isotrópico **0-1**, com N número de elementos finitos, possui 2^N soluções possíveis das quais, como ilustrado, algumas podem ser inviáveis. O número de soluções possíveis para um problema *ISE* com mais de um material formando os elementos é $(n+1)^N$, sendo n o número de materiais distintos encontrados. Observa-se que quando o número de elementos cresce, torna-se inviável a análise de todas as soluções possíveis visando obter a solução ótima. Esse fato estimulou o desenvolvimento de métodos para otimização topológica.



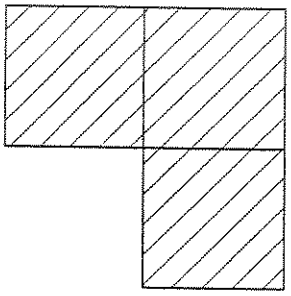
(a) Definição do problema.



(b) Solução inviável.



(c) Solução ótima.



(d) Solução não ótima.

Figura 1.2: Problema simples envolvendo elementos isotrópicos sólidos ou vazios (Rozvany, 2001).

1.1.3 Principais Métodos para Otimização de Problemas de Topologia *ISE*

Nesta seção, descrevem-se os principais métodos encontrados para resolução de problemas de topologia *ISE*. Como principais métodos, destacam-se o **SIMP** (*Solid Isotropic Microstructures with Penalizaion for intermediate densities*), o **OMP** (*Optimal Microstructures with Penalization for intermediate densities*), o **NOM** (*NonOptimal Microstructures* ou *Near Optimal Microstructures*) e o **DDP** (*Dual Discrete Programming*).

Alguns métodos emergentes na otimização de forma generalizada se enquadram melhor

em um grupo separado denominado **SERA** (*Sequential Element Rejections and Admissions method*). Nesse grupo, encontram-se métodos como o **ESO** (*Evolutionary Structural Optimization*), o **BESO** (*Bidirectional ESO*) e o **GSD** (*Generalized Stress Design*). Podem ser citados outros métodos importantes encontrados na literatura, como o **GA** (*Genetic Algorithms*) e o **ABG** (*Adaptative Biological Growth*). Segue uma breve descrição de alguns dos métodos citados acima, bem como das características do grupo *SERA*.

SIMP. Neste método, assume-se que uma variável de topologia, como por exemplo a espessura t de um disco, possa variar continuamente ($0 < t < t_0$) ao longo de um domínio Ω. O domínio é então submetido a um algoritmo de otimização, através de algum critério ótimo ou programação matemática. Pela hipótese da espessura variar continuamente ao longo do domínio, algumas espessuras intermediárias devem resultar do processo de otimização. Essas espessuras intermediárias, que não são interessantes para o resultado, são eliminadas através da criação de funções de penalização. As funções de penalização forçam as espessuras intermediárias a convergirem para 0 ou t_0.

OMP. Neste método, as microestruturas ótimas para cada elemento finito do problema em análise são encontradas a partir das restrições de projeto e da função objetivo a ser minimizada. Para a obtenção de um resultado satisfatório, as densidades intermediárias obtidas são penalizadas, de maneira a serem eliminadas.

NOM. Neste método, são usadas microestruturas não ótimas ou quase ótimas para cada elemento finito do problema em análise. Porém, não são usadas penalizações para densidades intermediárias. O fato da microestrutura não ser ótima assegura um certo grau de penalização fixo, pois há menos parâmetros sendo modificados nas microestruturas, o que resulta na obtenção de resultados mais homogêneos.

SERA. Os métodos pertencentes a esse grupo são denominados "mata-forte", gerando novas topologias para as estruturas em análise através da eliminação de elementos que estão abaixo de um valor definido para algum critério. Esses critérios podem ser tensão, densidade de energia ou qualquer outro parâmetro de resposta. Os métodos desse grupo são variações interessantes das técnicas de critério ótimo *FSD* (*Fully Stressed Design*) e *UED* (*Uniform*

Energy Distribution). O critério *FSD* baseia-se na idéia intuitiva da obtenção do ótimo através da saturação da tensão. Portanto, cada membro da estrutura, em pelo menos um caso de carregamento, está submetido à tensão máxima ou deslocamento máximo prescrito. O critério *UED* baseia-se na obtenção de uma distribuição uniforme de energia na estrutura em análise. Pode-se mostrar que esse critério, em alguns casos particulares, resulta em estruturas com rigidez otimizada.

ESO. Neste método, elementos da topologia inicial do problema são removidos, formando cavidades permanentes (irreversíveis) na estrutura. Como a distribuição de tensão muda consideravelmente entre dois estágios do processo iterativo, pode haver a necessidade de reinserção de um elemento removido. Nesse método não é possível a reinserção de elementos.

BESO. Neste método, a rejeição do elemento é um processo reversível, ou seja, um elemento removido pode ser reinserido. Por um lado, essa reversibilidade resolve uma limitação do *ESO*. Porém já foi demonstrado que o método pode ser insuficiente em retificar rejeições incorretas, além de poder não obter sucesso tentando conectar grandes áreas ou volumes onde muito material foi removido.

Alguns dos métodos anteriores (i.e., SIMP, OMP, NOM e DDP) caracterizam a topologia por uma densidade de material a ser determinada. As cavidades correspondem a regiões de densidade nula e o domínio é identificado por uma região com densidade não-nula. A topologia final é obtida aplicando-se técnicas de homogeinização. Estes métodos são aplicáveis apenas a problemas de Mecânica dos Sólidos e consideram somente condições de contorno naturais homogêneas nos furos.

Os métodos baseados no conceito de gradiente topológico não possuem tais restrições. Particularmente, os trabalhos (Novotny et al., 2000; Novotny et al., 2001a) são importantes, pois permitem acoplar a derivada topológica com a análise de sensibilidade à forma de maneira simples e bastante geral.

1.2 Geração de Geometrias

Uma etapa essencial em qualquer análise computacional de engenharia é a definição do problema. O primeiro passo para se definir um problema consiste em gerar sua geometria. Dessa forma, devido à sua relevância, um dos trabalhos desenvolvidos nessa dissertação é um ambiente de geração e correção de geometrias.

As geometrias são importantes tanto na fabricação quanto no projeto de produtos. Dessa forma, grande quantidade de recursos são destinadas ao desenvolvimento de aplicativos de *CAD* (*Computer Aided Design*) e *CAE* (*Computer Aided Engineering*). Dois grandes desafios encontrados por usuários de aplicativos *CAD/CAE* são a incompatibilidade entre formatos e a presença de inconsistências nas geometrias que inviabilizam a geração de malhas. Ferramentas relacionadas a esses dois desafios são tratadas nessa dissertação.

Devido à grande quantidade de aplicativos *CAD* existentes no mercado, uma questão cada vez mais importante é a viabilização do intercâmbio entre geometrias geradas em aplicativos diferentes. Alguns formatos padrões, como o *IGES* e o *STEP*, foram criados nesse sentido. Porém, estes formatos ainda são insuficientes para garantir a total intercambiabilidade das geometrias. Estima-se que sejam gastos anualmente no mundo cerca de US$1 bilhão com problemas de conversão e correção de arquivos CAD.

A geometria, depois de definida, pode apresentar uma série de inconsistências que inviabilizam seu uso em aplicativos *CAE*. Problemas comuns, como pontos e linhas duplicados ou interseções mal definidas entre arestas, impedem a geração de malhas e, conseqüentemente, a análise do problema. Nesse sentido, é de extrema importância o desenvolvimento de ferramentas capazes de abordar sistematicamente esses problemas.

Abordando estes aspectos brevemente descritos nos parágrafos anteriores, foi desenvolvido nesse trabalho um ambiente de geração de geometrias. Nesse ambiente, além de gerar novas geometrias, é possível importar e exportar arquivos *CAD* entre alguns dos formatos disponíveis no mercado, bem como executar processos de correção sobre estas geometrias. Estas funcionalidades serão destacadas na Seção 5.5.

1.3 Programação Orientada a Objetos

A programação orientada a objetos e a linguagem *C++* serão brevementes discutidas nesta seção.

O conceito de orientação a objetos independe da linguagem de programação adotada. Atualmente, existe uma grande quantidade de linguagens orientadas a objetos, como por exemplo, o *Java* e o *C++*. As linguagens orientada a objetos têm suas origens anteriores a 1960, firmando-se e estabelecendo-se na década de 80. A partir de então, obviamente dependendo da aplicação, tornaram-se uma unanimidade entre os programadores experientes. Esse trabalho de otimização será implementado seguindo a filosofia de orientação a objetos e utilizando a linguagem de programação *C++*.

A programação orientada a objetos difere da programação procedural em sua essência. Enquanto na programação procedural, preocupa-se em desenvolver funções capazes de resolver um determinado problema, na programação orientada a objetos a preocupação está em como descrever esse problema através de uma estrutura de dados. Neste caso, a estrutura de dados criada agrupa as funcionalidades necessárias para solucão do problema. Pode-se perceber, em uma análise mais apurada, que a programação orientada a objetos é muito mais exigente em termos conceituais que a procedural. Isso ocorre porque seu passo fundamental está na fragmentação e abstração dos conceitos envolvidos no problema proposto (Booch, 1991).

Uma vez definida uma melhor estrutura de dados, ou seja, um conjunto de abstrações que modela um problema proposto, basta definir como esta estrutura interage entre si. Os modos de interagir das classes são conhecidos como métodos. Esses métodos implementam as funcionalidades previstas para cada uma das abstrações. Estando esses métodos da estrutura de dados bem definidos, basta reunir o conjunto de abstrações para solucionar um problema. Este fato reforça que a essência da programação orientada a objetos está na fragmentação e abstração dos conceitos envolvidos.

Os dois parágrafos acima descrevem, de maneira bem simplificada, o conceito de orientação a objetos. A estrutura de dados abstraída para representar um problema é formada

por um conjunto de classes. Cada classe possui uma interface, ou seja, um conjunto de métodos, que permite o acesso às suas funcionalidades. A instância de uma classe, ou seja, a declaração e inicialização de uma classe, é o que se chama objeto. Um programa possui uma quantidade de objetos, ou seja, instâncias de classes, que são capazes de resolver um problema.

O *C++* é uma linguagem de alto nível e foi escolhida para desenvolvimento dos programas propostos nessa dissertação. Apesar de ser uma linguagem de difícil assimilação, devido a quantidade de recursos disponíveis, é a mais adequada para desenvolvimento de aplicativos extensos e confiáveis. Não é objetivo desta seção descrever as características da linguagem, bem como sua sintaxe. Uma referência abrangente e de qualidade pode ser encontrada em (Graham, 1991).

Com o descrito até aqui, pode-se justificar o uso do conceito de orientação a objetos e da linguagem de programação *C++*. Para aplicações extensas e que são atualizadas com freqüência, como é o caso da aplicação desenvolvida nessa dissertação, o uso destes conceitos e linguagem, sem a menor dúvida, estão justificados. A vantagem da implementação que segue a filosofia descrita é evidente, pois a fragmentação do problema permite atualização, revisão e introdução constante de novas classes e funcionalidades. Essas novas classes tem sua parcial independência do resto código, interessando para o resultado final apenas a clareza de sua interface.

1.4 Objetivos

Um dos objetivos dessa dissertação é aplicar o conceito de derivada topológica em problemas elásticos bi e tridimensionais. A derivada topológica é obtida através da análise de sensibilidade à mudança de forma, que está descrita na Seção 2.5, conforme as expressões mostradas no Capítulo 3. Nessa dissertação não foi feita nenhum desenvolvimento sobre o conceito de derivada topológica. Emprega-se simplesmente os conceitos elaborados nos seguintes trabalhos (Novotny et al., 2000; Novotny et al., 2001a; Novotny et al., 2002d; Novotny et al., 2002b; Novotny et al., 2002a; Novotny et al., 2002c; Novotny et al., 2001b) que foram cedidos

gentilmente pelos autores.

Para obtenção dos resultados foi utilizado um algoritmo simples descrito na Seção 3.4, implementado em *C++*, seguindo os procedimentos de engenharia de *software* descritos no Capítulo 5.

De acordo com a nomenclatura definida na revisão bibliográfica (Seção 1.1.1), serão tratados problemas de topologia *ISE*. O trabalho pode ser considerado, devido à forma como as informações obtidas da derivada topológica foram utilizadas, um método pertencente ao grupo *SERA*. Porém, é muito importante ressaltar, como ficará mais claro no desenvolvimento do texto, que a derivada topológica é um conceito amplo, que de forma nenhuma está relacionado às técnicas de critério ótimo *FSD* (*Fully Stressed Design*) ou *UED* (*Uniform Energy Distribution*).

Um outro objetivo desse trabalho é implementar um ambiente de geração de geometrias. Este ambiente está baseado no modelador geométrico *ACIS*. Suas características, bem como as do modelador geométrico, são encontradas no Capítulo 5.

1.5 Organização do Texto

A dissertação está dividida em seis capítulos. O primeiro capítulo descreveu e situou o trabalho desenvolvido, bem como importantes introduções aos conceitos de implementação. No Capítulo 2, encontra-se a descrição da análise de sensibilidade, mostrando as principais características e expressões que podem ser encontradas na análise de sensibilidade. No Capítulo 3, mostram-se as relações entre a derivada topológica e a análise de sensibilidade à mudança de forma e o algoritmo implementado. No Capítulo 4, consideram-se os resultados obtidos para alguns problemas da elasticidade, assim como comentários sobre o algoritmo e os resultados. No Capítulo 5, mostram-se as ferramentas de engenharia de *software* utilizadas na implementação da derivada topológica e do ambiente de geração de geometrias. Finalmente, no Capítulo 6, apresentam-se as conclusões e algumas propostas de continuidade de trabalhos relacionados à otimização através da derivada topológica.

Capítulo 2

Análise de Sensibilidade

Grande parte dos esforços na pesquisa em mecânica computacional estão direcionados na determinação do comportamento, ou seja, da resposta, de um corpo quando submetido a um conjunto de forças externas. Quando se considera um sistema como esse, formado por um corpo e ações externas, uma outra questão de igual importância pode ser proposta: o que acontece quando algumas das características desse sistema são alteradas?

A grande motivação para essa questão vem dos problemas de otimização, nos quais se busca um projeto ótimo baseado em critérios objetivamente definidos. Isso ocorre porque nos processos de otimização, o caminho entre o projeto inicial e o projeto ótimo pode ser encurtado através do conhecimento da sensibilidade do modelo a mudanças em suas variáveis. Nesse sentido, a Análise de Sensibilidade (AS) relaciona os efeitos da variação de parâmetros com a resposta do modelo (Haug et al., 1986).

Este capítulo apresenta, inicialmente, definições gerais sobre Problemas de Valor de Contorno (PVC) e aproximações do mesmo. Posteriormente, introduz-se o conceito geral de análise de sensibilidade. Após esta breve introdução, será apresentada a sua formulação considerando problemas nos quais as variáveis de projeto não influenciam a forma do domínio. Na Seção 2.5, aborda-se a análise de sensibilidade à mudança de forma (ASMF), na qual se toma o próprio domínio como variável de projeto. A formulação desenvolvida para ASMF será utilizada no Capítulo 3 para a obtenção da derivada topológica.

2.1 Definições Matemáticas

O conteúdo dessa seção está baseado nos trabalhos (Novotny et al., 2000; Novotny et al., 2001a).

2.1.1 Problema de Valor de Contorno

Um exemplo de problema de valor de contorno (PVC) está ilustrado na Figura 2.1. Seja um domínio aberto e limitado $\Omega \subset \Re^N$, cujo contorno $\Gamma = \Gamma_N \cup \Gamma_D$ com $\Gamma_N \cap \Gamma_D = \emptyset$ é suficientemente regular, isto é, admite a existência de um vetor normal n em quase todo ponto, exceto possivelmente em um conjuto de medida nula. Os índices utilizados para definir os contornos, N e D, encontrados em Γ_N e Γ_D, denotam, respectivamente, o contorno no qual se encontram condições de Neumann e o contorno no qual se encontram condições de Dirichlet. Fisicamente, tem-se um corpo Ω, submetido a excitações f em Γ_N e b em Ω e com restrições na variável primal u no contorno Γ_D.

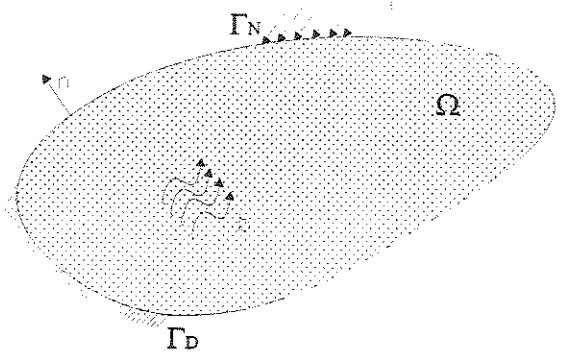


Figura 2.1: Representação de um problema de valor de contorno genérico (Novotny et al., 2000; Novotny et al., 2001a).

Esse problema pode ser escrito em uma forma variacional, o que permite trabalhar em espaços topologicamente mais fracos e estabelecer condições de existência e unicidade para solução. A forma fraca de um problema elíptico de valor de contorno pode ser escrita na

seguinte forma geral:

Encontrar $u \in U$, *tal que*

$$a(u, v) = l(v), \qquad \forall\, v \in V, \tag{2.1}$$

sendo $a(\cdot, \cdot) : U \times V \rightarrow \Re$ um operador bilinear, limitado e coercivo e $l(\cdot)$ um funcional linear limitado, ou seja, $l \in V'$, sendo V' o espaço dual de V. Além do mais, o espaço das variações admissíveis V e o espaço das funções admissíveis U podem ser definidos, respectivamente, como

$$V = \{v \in H^n(\Omega) \mid v = 0 \text{ sobre } \Gamma_D\},$$
$$U = \{u \in H^n(\Omega) \mid u = g \text{ sobre } \Gamma_D\},$$

sendo $H^n(\cdot)$ um espaço de Hilbert de ordem n, definido em um dado domínio. Dessa forma, pela característica do espaço, está-se resolvendo um problema de valor de contorno de ordem $2n$.

Quando as condições de contorno de Dirichlet (as condições impostas no contorno Γ_D) não forem homogêneas, o espaço das funções admissíveis U trata-se, na realidade, de uma variedade linear definida em $H^n(\Omega)$. No caso do operador $a(\cdot, \cdot)$ ser simétrico, os cálculos da análise de sensibilidade podem ser simplificados. Essa simetria do operador será muito importante para obtenção da derivada topológica na Seção 3.3.2.

Nos trabalhos (Oden e Carey, 1983; Oden e Reddy, 1976), abordam-se as questões de equivalência entre formulações diferenciais e variacionais. Já os problemas de existência e unicidade da solução das equações escritas na forma abstrata são dados pelo Teorema Generalizado de Lax-Milgran, também demonstrado nas referências citadas.

2.1.2 Solução Aproximada

Soluções analíticas para problemas de valor de contorno são, em geral, impraticáveis. Os problemas de valor de contorno envolvem equações diferenciais de equilíbrio, equações constitutivas e condições de carregamento e contorno. Encontrar soluções analíticas para esses

problemas somente é possível quando não existem complexidades, como por exemplo, geometrias sofisticadas. Nos casos onde há presença de complexidades, os métodos numéricos são aplicados para obtenção de soluções aproximadas. O método mais difundido atualmente é o Método dos Elementos Finitos (MEF), o qual está fundamentado na discretização do meio contínuo (Cook et al., 1991; Bathe, 1996).

Em geral, os métodos de busca de soluções aproximadas para problemas de valor de contorno baseiam-se na reconstrução do problema, de modo que a solução aproximada pertença a uma classe restrita de funções. Mais especificamente, o MEF pode ser visto como uma maneira sistemática e geral de reconstruir famílias de subespaços $U_h \subset U$, ou seja, consiste em resolver o seguinte problema aproximado:

Encontrar a solução aproximada $u_h \in U_h \subset U$, *tal que*

$$a(u_h, v_h) = l(v_h), \qquad \forall\, v_h \in V_h \subset V. \tag{2.2}$$

Se o espaço das variações admissíveis discretizado V_h coincide com o espaço das funções admissíveis discretizado U_h, ou seja, se $V_h \equiv U_h$ são topologicamente equivalentes, a aproximação resultante é conhecida como o método de Bubnov-Galerkin, ou simplesmente, método de Galerkin.

Esta última forma de se colocar as equações de estado (2.2) conduz às equações de elementos finitos que, no caso de problemas elípticos de valor de contorno, podem ser reescritas de maneira geral como

$$\mathbf{K}\bar{\mathbf{u}}_h = \mathbf{F}, \tag{2.3}$$

sendo $\bar{\mathbf{u}}_h$ o vetor de incógnitas nodais generalizados, $\mathbf{K}$ a matriz de rigidez global e $\mathbf{F}$ o vetor de carregamento nodal generalizado.

É importante observar que u_h obtido pelo Método de Galerkin é a melhor aproximação possível para solução exata de u no espaço U_h. A afirmação anterior é verdadeira porque é possível mostrar que nesse tipo de aproximação o erro $e = u - u_h$ é ortogonal ao subespaço V_h.

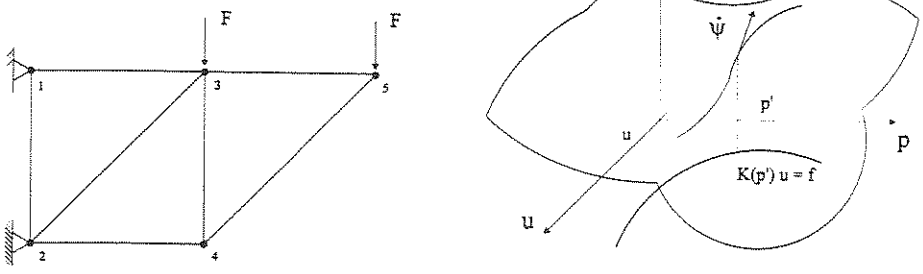
2.2 Conceito Geral de Sensibilidade

O conteúdo dessa seção está baseado nos trabalhos (Silva, 1997; Fancello, 1993).

Um exemplo simples de análise de sensibilidade pode ser ilustrado através da treliça da Figura 2.2a. Considera-se essa treliça em regime elástico linear e dependente de um vetor $\mathbf{p}$ de variáveis de projeto. Esse vetor $\mathbf{p}$ pode conter, por exemplo, as áreas das seções tranversais A_i e os módulos de elasticidade E_i para cada uma das barras, além das coordenadas x_j dos nós. O equilíbrio dessa estrutura pode ser expresso pelo seguinte sistema linear

$$\mathbf{K}(\mathbf{p})\mathbf{u} = \mathbf{f}(\mathbf{p}), \tag{2.4}$$

sendo $\mathbf{K}$ a matriz de rigidez global, $\mathbf{u}$ o vetor de deslocamentos nodais e $\mathbf{f}$ o vetor de forças externas. Deve-se notar que $\mathbf{u} = \mathbf{u}(\mathbf{p})$, ou seja, a resposta da estrutura depende das variáveis de projeto.



(a) Treliça submetida a forças. (b) Análise de Sensibilidade.

Figura 2.2: Ilustração da AS em um problema de treliça (Fancello, 1993).

Suponha que se deseja determinar a variação do trabalho realizado pela força F no nó 5 quando as variáveis de projeto $\mathbf{p}$ são modificadas. Para tanto, define-se uma função relativa ao trabalho da força F

$$\psi(\mathbf{u}, \mathbf{p}) = \mathbf{u}_5(\mathbf{p}) \cdot \mathbf{F}, \tag{2.5}$$

que depende de $\mathbf{p}$ através da solução de $\mathbf{u}(\mathbf{p})$.

Do ponto de vista da teoria da otimização, ψ é a função objetivo e $\mathbf{Ku} = \mathbf{f}$ é uma restrição de igualdade no domínio $(\mathbf{u}, \mathbf{p})$. Nesse caso, a AS significa determinar a variação de ψ quando se caminha sobre a curva definida pela equação de estado (2.4), como ilustrado na Figura 2.2b.

Em casos gerais, definem-se funcionais dependentes das variáveis de projeto, as quais estão relacionadas a parâmetros discretos do problema (e.g., propriedades geométricas e de material) ou mesmo à forma do domínio. A AS objetiva obter a variação dos funcionais quando ocorrem alterações nas variáveis de projeto.

A AS pode ser feita de duas maneiras: desenvolvendo as expressões dos gradientes em relação às equações de sistemas já discretizados ou obtê-las diretamente das equações do contínuo, as quais são então discretizadas para efeito de cálculo. Nos problemas de elasticidade linear, essas duas abordagens são equivalentes tanto teórica quanto numericamente. Entretanto, pode-se demonstrar que em certas situações as duas formulações não são equivalentes, e que em diversos casos, a formulação discreta conduz a resultados imprecisos (Choi e Twu, 1989). A formulação contínua da análise de sensibilidade é a mais indicada para problemas nos quais a forma do componente é considerada uma variável (ASMF).

2.3 Formulação Discreta da Análise de Sensibilidade

O conteúdo dessa seção está baseado em (Silva, 1997; Silva e Bittencourt, 1999).

Considere uma estrutura com número finito de graus de liberdade, material elástico e submetida a pequenas deformações. De acordo com o descrito na Seção 2.1.2, quando a solução de um problema é aproximada pelo método dos elementos finitos, a representação do sistema linear que leva à solução aproximada pode ser escrita, de maneira simplificada, como

$$\mathbf{K}(\mathbf{p})\mathbf{u} = \mathbf{f}(\mathbf{p}), \tag{2.6}$$

sendo $\mathbf{u} = \mathbf{u}(\mathbf{p})$ e $\mathbf{p} = \{p_1, \ldots, p_n\}$ o vetor das n variáveis de projeto discretas.

Pode-se propor um funcional usualmente utilizado em problemas de otimização estru-

tural, na seguinte forma geral

$$\psi(\mathbf{p}) = \mathcal{G}(\mathbf{p}, \mathbf{u}(\mathbf{p})). \tag{2.7}$$

Esse funcional depende das variáveis de projeto tanto de forma explícita quanto de forma implícita, através da solução $\mathbf{u}$ da equação de estado (2.6). Através da AS, pode-se determinar a dependência total desse tipo de funcional em relação às variáveis $\mathbf{p}$ de um projeto parametrizado. Para tanto, escreve-se uma variação do funcional (2.7) como

$$\begin{aligned} \dot{\psi}(\mathbf{p}) &= \nabla_{\mathbf{p}}\psi \cdot d\mathbf{p} = \frac{\partial \mathcal{G}}{\partial \mathbf{p}} \cdot d\mathbf{p} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot d\mathbf{u} = \frac{\partial \mathcal{G}}{\partial \mathbf{p}} \cdot d\mathbf{p} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot (\nabla_{\mathbf{p}}\mathbf{u}) d\mathbf{p} \\ &= \left[\frac{\partial \mathcal{G}}{\partial \mathbf{p}} + (\nabla_{\mathbf{p}}\mathbf{u})^T \frac{\partial \mathcal{G}}{\partial \mathbf{u}}\right] \cdot d\mathbf{p} \Longrightarrow \nabla_{\mathbf{p}}\psi = \left[\frac{\partial \mathcal{G}}{\partial \mathbf{p}} + (\nabla_{\mathbf{p}}\mathbf{u})^T \frac{\partial \mathcal{G}}{\partial \mathbf{u}}\right]. \end{aligned} \tag{2.8}$$

Na equação acima, os termos $\dfrac{\partial \mathcal{G}}{\partial \mathbf{p}}$ e $\dfrac{\partial \mathcal{G}}{\partial \mathbf{u}}$ podem ser determinados diretamente, pois se conhece a forma do funcional $\mathcal{G}$. Como $\mathbf{u}$ é definido de forma implícita pela solução do sistema (2.6), sua derivabilidade em relação a $\mathbf{p}$ não é imediata. Dessa forma, o problema reside na determinação de $\nabla_{\mathbf{p}}\mathbf{u}$.

Admitindo a existência da derivada $\nabla_{\mathbf{p}}\mathbf{u}$, dois métodos, o direto e o adjunto, podem ser empregados na sua determinação.

2.3.1 Método Direto

Considerando a existência de $\nabla_{\mathbf{p}}\mathbf{u}$, define-se um vetor resíduo $\mathbf{r}$ de (2.6) como

$$\mathbf{r} = \mathbf{K}(\mathbf{p})\mathbf{u} - \mathbf{f}(\mathbf{p}) = 0 \tag{2.9}$$

Derivando a função resíduo acima em relação a cada componente p_k $(k = 1, ..., n)$ do vetor $\mathbf{p}$, tem-se o conjunto de equações

$$\frac{\partial \mathbf{r}}{\partial p_k} = \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} + \mathbf{K}\frac{\partial \mathbf{u}}{\partial p_k} - \frac{\partial \mathbf{f}}{\partial p_k} = 0, \qquad k = 1, ..., n. \tag{2.10}$$

Rearranjando os termos

$$\mathbf{K}\frac{\partial \mathbf{u}}{\partial p_k} = \frac{\partial \mathbf{f}}{\partial p_k} - \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}, \qquad k = 1, ..., n. \tag{2.11}$$

Resolvendo os n sistemas de equações lineares em (2.11), obtém-se $\nabla_{\mathbf{p}}\mathbf{u}$, pois cada solução de (2.11) fornece uma coluna de $\nabla_{\mathbf{p}}\mathbf{u}$. Observa-se que uma vez fatorada a matriz K, obtém-se as derivadas $\dfrac{\partial \mathbf{u}}{\partial p_k}$ por substituição. Após determinar $\nabla_{\mathbf{p}}\mathbf{u}$, basta utilizá-lo na

expressão (2.8) para calcular a sensibilidade de ψ em relação às variáveis de projeto.

O cálculo do gradiente do funcional ψ, pelo método direto, pode ser resumido nos seguintes passos:

1. Determinar $\frac{\partial \mathbf{K}}{\partial p_k}$, $\frac{\partial \mathbf{f}}{\partial p_k}$, $\frac{\partial \mathcal{G}}{\partial p_k}$ $(k = 1, ..., n)$ e $\frac{\partial \mathcal{G}}{\partial \mathbf{u}}$;

2. Obter $\nabla_{\mathbf{p}}\mathbf{u}$ através da equação (2.11), solucionando n sistemas lineares com mesma matriz $\mathbf{K}$;

3. Calcular as componentes de $\nabla_{\mathbf{p}}\psi$ através de

$$\frac{\partial \psi}{\partial p_k} = \frac{\partial \mathcal{G}}{\partial p_k} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot \frac{\partial \mathbf{u}}{\partial p_k}. \tag{2.12}$$

2.3.2 Método Adjunto

Seja o vetor $\varsigma \in \Re^N$ e considere o produto escalar de ς por (2.11)

$$\varsigma \cdot \mathbf{K}\frac{\partial \mathbf{u}}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}. \tag{2.13}$$

Pode-se reescrever a equação acima como

$$\mathbf{K}^T\varsigma \cdot \frac{\partial \mathbf{u}}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}. \tag{2.14}$$

Definindo o problema adjunto como

$$\mathbf{K}^T\varsigma = \frac{\partial \mathcal{G}}{\partial \mathbf{u}}, \tag{2.15}$$

obtém-se ς pela solução do sistema linear (2.15).

Substituindo (2.15) em (2.14), tem-se

$$\frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot \frac{\partial \mathbf{u}}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}, \tag{2.16}$$

Substituindo agora (2.16) em (2.12), obtém-se a sensibilidade de ψ em relação ao vetor $\mathbf{p}$ através da expressão

$$\frac{\partial \psi}{\partial p_k} = \frac{\partial \mathcal{G}}{\partial p_k} + \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}. \tag{2.17}$$

O cálculo do gradiente do funcional ψ, pelo método adjunto, pode ser resumido nos seguintes passos:

1. Determinar $\frac{\partial \mathbf{K}}{\partial p_k}$, $\frac{\partial \mathbf{f}}{\partial p_k}$, $\frac{\partial \mathcal{G}}{\partial p_k}$ $(k = 1, ..., n)$ e $\frac{\partial \mathcal{G}}{\partial \mathbf{u}}$;

2. Resolver o problema adjunto (2.15);

3. Calcular as componentes de $\nabla_{\mathbf{p}}\psi$ através de (2.17)

$$\frac{\partial \psi}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} + \frac{\partial \mathcal{G}}{\partial p_k}.$$

2.3.3 Comparação entre os Métodos Direto e Adjunto

A precisão dos resultados obtidos pelos métodos adjunto e direto são equivalentes em problemas lineares (Haug et al., 1986). As diferenças que justificam a escolha de um dos métodos, estão baseadas em seus custos computacionais. Embora não afetando a precisão dos resultados, o número de operações realizadas em cada um dos métodos pode diferir bastante em uma mesma aplicação. Dessa forma, para se escolher o método mais eficiente, deve-se comparar o número de sistemas lineares a serem resolvidos, bem como o número de vetores que devem ser armazenados para solução de um problema.

No método direto, calcula-se o gradiente $\nabla_{\mathbf{p}}\mathbf{u}$, através da solução de n (número de variáveis de projeto) sistemas lineares de mesma matriz de ordem N (dimensão da discretização do problema), independentemente do número de funcionais do problema. O método adjunto, por sua vez, resolve um sistema linear de dimensão N, por funcional.

Como em otimização o número de funcionais ativos é sempre menor ou igual a n para um projeto viável, o método adjunto será, de forma geral, mais eficiente. A eficiência do método adjunto é enfatizada se o algoritmo de minimização utilizado na otimização for capaz de reconhecer o conjuto de restrições potencialmente ativas (Silva, 1997).

2.4 Formulação Contínua da Análise de Sensibilidade

O conteúdo dessa seção está baseado em (Silva, 1997; Silva e Bittencourt, 1999).

Diferentemente da formulação discreta, no caso da formulação contínua, as equações são tomadas em sua forma original, deduzindo-se analiticamente as expressões de sensibilidade, as quais são resolvidas numericamente. Assim como foi feito para a formulação discreta, primeiramente o problema será tratado com variáveis de projeto controlando apenas características discretas do problema variacional, enquanto o domínio $\mathcal{B}$ permanece fixo.

Considere problemas elastostáticos lineares propostos na forma variacional, como o encontrado em (2.1). No caso em que o domínio $\mathcal{B}$ permanece fixo, $a(u,v)$ e $l(v)$ dependem diretamente do vetor de variáveis $\mathbf{p}$ e a solução u é um campo vetorial tal que $u = u(\mathbf{x}, \mathbf{p})$. Demonstra-se em (Haug et al., 1986) que as formas bilinear $a(u,v)$ e linear $l(v)$ são diferenciáveis com respeito ao projeto $\mathbf{p}$, mantendo-se u fixo. Da mesma maneira, u e ∇u são diferenciáveis com relação a $\mathbf{p}$.

Dessa forma, definindo $\delta\mathbf{p}$ como uma variação no projeto atual $\mathbf{p}$, tem-se que as derivadas direcionais de $a(\cdot,\cdot)$ e $l(\cdot)$ e u são dadas por

$$\dot{a}_{\mathbf{p}}(u,v) = Da_{\mathbf{p}}(u,v)[\delta\mathbf{p}] = \lim_{\tau\to 0} \frac{a_{\mathbf{p}+\tau\delta\mathbf{p}}(u,v) - a_{\mathbf{p}}(u,v)}{\tau} = \nabla_{\mathbf{p}} a(u,v)\cdot\delta\mathbf{p}, \tag{2.18}$$

$$\dot{l}_{\mathbf{p}}(v) = Dl_{\mathbf{p}}(v)[\delta\mathbf{p}] = \lim_{\tau\to 0} \frac{l_{\mathbf{p}+\tau\delta\mathbf{p}}(v) - l_{\mathbf{p}}(v)}{\tau} = \nabla_{\mathbf{p}} l(v)\cdot\delta\mathbf{p}, \tag{2.19}$$

$$\dot{u}(\mathbf{x},\mathbf{p}) = Du(u,v)[\delta\mathbf{p}] = \lim_{\tau\to 0} \frac{u(\mathbf{x},\mathbf{p}+\tau\delta\mathbf{p}) - u_{\mathbf{p}}(\mathbf{x},\mathbf{p})}{\tau} = \nabla_{\mathbf{p}} u(\mathbf{x},\mathbf{p})\cdot\delta\mathbf{p}. \tag{2.20}$$

Um conjunto de funcionais de performance estrutural, tal como, por exemplo, volume, deslocamentos, tensões e energia de deformação, podem ser expressos na seguinte forma geral (Haug et al., 1986)

$$\psi(\mathbf{p}) = \int_{\mathcal{B}} \mathcal{G}(u, \nabla u, \mathbf{p})\, dV, \qquad u = u(\mathbf{x},\mathbf{p}), \qquad \mathbf{x}\in\mathcal{B}, \tag{2.21}$$

sendo $\mathcal{G}$ um funcional continuamente diferenciável com relação aos seus argumentos e $\mathcal{B}$ fixo.

A seguinte derivada direcional, na direção de uma variação $\delta\mathbf{p}$ do projeto, é válida (Haug et al., 1986)

$$\dot{\psi}(\mathbf{p}) \equiv D\psi(\mathbf{p})[\delta\mathbf{p}] = D\left\{\int_{\mathcal{B}} \mathcal{G}(u(\mathbf{x},\mathbf{p}), \nabla u(\mathbf{x},\mathbf{p}), \mathbf{p})\, dV\right\}[\delta\mathbf{p}] = \nabla\psi\cdot\delta\mathbf{p}. \tag{2.22}$$

Como o domínio é fixo, pode-se passar a operação de diferenciação para dentro da integral.

Portanto, aplicando a regra da cadeia (Silva e Bittencourt, 1999) vem que

$$\begin{aligned}\dot{\psi}(\mathbf{p}) &= \int_{\mathcal{B}} D\left\{\mathcal{G}(u(\mathbf{x},\mathbf{p}),\nabla u(\mathbf{x},\mathbf{p}),\mathbf{p})\right\}[\delta\mathbf{p}]\, dV \\ &= \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u} + \mathcal{G}_{,\mathbf{p}}\cdot\dot{\mathbf{p}})\, dV. \end{aligned} \tag{2.23}$$

Em geral, u é determinado numericamente pelo MEF. Os termos $\dot{u}$ e $\nabla\dot{u}$ são desconhecidos e, para determiná-los, pode-se empregar os métodos direto ou adjunto.

2.4.1 Método Direto

No método direto, a variação $\dot{u}$ é obtida da derivada total de (2.1), ou seja, (Silva e Bittencourt, 1999)

$$[a(u,v)]^{\cdot} = [l(v)]^{\cdot} \Rightarrow \dot{a}_{\mathbf{p}}(u,v) + a(\dot{u},v) = \dot{l}_{\mathbf{p}}(v) \rightarrow a(\dot{u},v) = \dot{l}_{\mathbf{p}}(v) - \dot{a}_{\mathbf{p}}(u,v). \tag{2.24}$$

A equação acima dá origem a um novo problema variacional de mesma forma bilinear (2.1), cuja solução é justamente a derivada $\dot{u}$ do campo de deslocamentos. A partir dessa solução, determina-se $\nabla\dot{u}$, obtendo-se então $\dot{\psi}(\mathbf{p})$ através da expressão (2.23).

Observa-se, no entanto, que os termos do lado direito em (2.24) dependem da variação $\delta\mathbf{p}$. Assim, o método direto não é interessante. Dessa forma, deseja-se obter expressões explícitas para as derivadas em função de $\delta\mathbf{p}$.

2.4.2 Método Adjunto

No método adjunto, elimina-se a dependência direta de $\dot{\psi}(\mathbf{p})$ em relação a $\dot{u}$ e $\nabla\dot{u}$. Para isso, introduz-se um problema auxiliar, o qual possui as mesmas restrições em deslocamentos do problema original, mas é sujeito a um *carregamento adjunto* que depende do funcional $\psi(\mathbf{p})$. Observa-se que os dois primeiros termos em (2.23), constituem-se em uma forma linear e limitada em $\dot{u}$ (Silva e Bittencourt, 1999),

$$l^{adj}(\dot{u}) = \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u})\, dV. \tag{2.25}$$

Considerando a forma bilinear $a(\cdot,\cdot)$ com variáveis adequadas, pode-se escrever um novo problema variacional correspondente a $l^{adj}(\dot{u})$, ou seja,

$$a(\varsigma,\dot{u}) = \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u})\; dV. \tag{2.26}$$

Devido à simetria de $a(\cdot,\cdot)$ e utilizando (2.24) e (2.26), tem-se que (Silva e Bittencourt, 1999)

$$a(\varsigma,\dot{u}) = a(\dot{u},\varsigma) = \dot{l}_{\mathbf{p}}(\varsigma) - \dot{a}_{\mathbf{p}}(u,\varsigma) = \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u})\; dV. \tag{2.27}$$

Substituindo a equação acima em (2.23), tem-se uma nova expressão da derivada do funcional $\psi(\mathbf{p})$

$$\dot{\psi}(\mathbf{p}) = \dot{l}_{\mathbf{p}}(\varsigma) - \dot{a}_{\mathbf{p}}(u,\varsigma) + \int_{\mathcal{B}} \mathcal{G}_{,\mathbf{p}}\cdot\dot{\mathbf{p}}\; dV. \tag{2.28}$$

Dessa forma, a determinação de $\dot{\psi}(\mathbf{p})$ depende apenas do estado adjunto ς. Para isso, define-se o seguinte problema variacional adjunto (Silva e Bittencourt, 1999):

Determinar o campo de deslocamentos ς *tal que*

$$a(\varsigma,v) = \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot v + \mathcal{G}_{,\nabla u}\cdot\nabla v)\; dV, \qquad \forall v \in V. \tag{2.29}$$

Das propriedades de diferenciabilidade assumidas para $\mathcal{G}$, o termo independente é um funcional linear e contínuo. Portanto, garante-se a existência e a unicidade do estado adjunto ς pelo teorema de Lax-Milgram (Haug et al., 1986).

Dessa forma, tem-se a seguinte expressão para a derivada parcial do funcional de performance ψ em relação a um parâmetro p_k não relacionado à forma do domínio do problema estrutural (Silva e Bittencourt, 1999)

$$(\nabla_{\mathbf{p}}\psi)_k = \frac{\partial\psi}{\partial p_k} = \frac{\partial l}{\partial p_k}(\varsigma) - \frac{\partial a}{\partial p_k}(u,\varsigma) + \int_{\mathcal{B}} \frac{\partial\mathcal{G}}{\partial p_k}\; dV, \qquad k = 1,...,n. \tag{2.30}$$

Observa-se que não é mais necessário avaliar $\nabla\dot{u}$. Mas se deve resolver um problema adjunto para todo funcional dependente de u. As expressões para alguns funcionais, encontradas em (Silva e Bittencourt, 1999), estão transcritas no Apêndice A.

2.5 Análise de Sensibilidade à Mudança de Forma

O conteúdo dessa seção está baseado nos trabalhos (Silva, 1997; Fancello, 1993; Novotny et al., 2000; Novotny et al., 2001a).

Nas Seções 2.3 e 2.4, a análise de sensibilidade foi apresentada em casos nos quais o vetor $\mathbf{p}$, que parametrizava características do problema, não afetava a geometria do domínio. Entretanto, como ficará mais claro no Capítulo 3, a aplicação da análise de sensibilidade nessa dissertação requer a avaliação de funcionais e suas sensibilidades em casos onde o próprio domínio $\mathcal{B}$ é uma variável. Nesses casos, emprega-se a análise de sensibilidade à mudança de forma.

A formulação contínua da análise de sensibilidade à mudança de forma está baseada na definição de uma transformação que descreve a variação de um domínio em torno do domínio original. Essa transformação é descrita por um campo de velocidades, conhecido como *ação de mudança de forma*, usado na determinação da derivada material ou derivada total do funcional (ver Seção 2.5.2).

Apresentam-se a seguir alguns conceitos gerais necessários para a compreensão da análise de sensibilidade à mudança de forma baseados em (Gurtin, 1981; Silva, 1997).

2.5.1 Descrição do Movimento de um Corpo

Embora não seja possível associar intrinsecamente regiões do espaço euclidiano pontual $\mathcal{E}$ a um corpo, convenciona-se adotar uma dessas regiões, denotada por $\mathcal{B}$, como *configuração de referência* de um corpo. Na configuração de referência, associam-se pontos materiais do corpo com suas posições em $\mathcal{B}$. Dessa maneira, tem-se que o corpo $\mathcal{B}$ é formalmente uma região regular do espaço euclidiano, sendo os pontos $\mathbf{x} \in \mathcal{B}$ chamados *pontos materiais.*

Um movimento de um corpo $\mathcal{B}$, como ilustrado na Figura 2.3, é uma função tal que

$$\begin{aligned} X^\tau : \mathcal{B} \times \Re &\to \mathcal{E} \\ (\mathbf{x}, \tau) &\to \mathbf{x}^\tau, \end{aligned} \tag{2.31}$$

onde, para um τ fixo, tem-se uma transformação injetora suave.

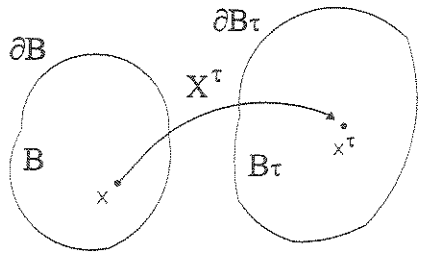


Figura 2.3: Movimento mapeado através de uma seqüência de deformações.

Na transformação, $\mathbf{x}^\tau = X^\tau(\mathbf{x}, \tau)$ é a posição do ponto material $\mathbf{x}$ no tempo τ, enquanto $\mathcal{B}_\tau = X^\tau(\mathcal{B}, \tau)$ é a região ocupada pelo corpo em τ. Para cada τ fixo, $X^\tau(\mathbf{x}, \tau)$ representa uma deformação de $\mathcal{B}$. Dessa forma, um movimento é uma seqüência de deformações em função do parâmetro τ, ver Figura 2.3.

Pode-se definir a trajetória $\mathcal{T}$ de um ponto material $\mathbf{x} \in \mathcal{B}$ como sendo o conjunto de posições assumidas por $\mathbf{x}$ ao longo do tempo. Esse conjunto de posições é originado pelo movimento, podendo-se escrever

$$\mathcal{T} = \{(\mathbf{x}^\tau, \tau) \mid \mathbf{x}^\tau \in \mathcal{B}_\tau,\ \tau \in \Re\}. \tag{2.32}$$

Como X^τ é um parâmetro injetor para todo τ, tem-se que a transformação inversa $X(\cdot, \tau) \equiv [X^\tau(\cdot, \tau)]^{-1}$, $X(\cdot, \tau) : \mathcal{B}_\tau \longrightarrow \mathcal{B}$ existe, e é tal que

$$X^\tau(X(\mathbf{x}^\tau, \tau), \tau) = \mathbf{x}^\tau \text{ e } X(X^\tau(\mathbf{x}, \tau), \tau) = \mathbf{x}. \tag{2.33}$$

Dado $(\mathbf{x}^\tau, \tau) \in \mathcal{T}$, então $\mathbf{x} = X(\mathbf{x}^\tau, \tau)$ é o ponto material que ocupa a posição espacial $\mathbf{x}^\tau$ no tempo τ.

2.5.2 Descrições Material e Espacial

Qualquer campo associado com o movimento pode ser expresso em função do ponto material e do tempo (*descrição material*) ou em função da trajetória (*descrição espacial*). Dado o campo material $(\mathbf{x}, \tau) \longrightarrow \Phi(\mathbf{x}, \tau)$, define-se sua descrição espacial Φ_s como

$$\Phi_s(\mathbf{x}^\tau, \tau) = \Phi(X(\mathbf{x}^\tau, \tau), \tau). \tag{2.34}$$

Da mesma forma, define-se a descrição material Ψ_m do campo espacial $(\mathbf{x}^\tau, \tau) \longrightarrow \Psi(\mathbf{x}^\tau, \tau)$ como

$$\Psi_m(\mathbf{x}, \tau) = \Psi(X^\tau(\mathbf{x}, \tau), \tau). \tag{2.35}$$

Dado um campo material $\Phi(\mathbf{x}, \tau)$, denota-se por

$$\dot{\Phi}(\mathbf{x}, \tau) = \frac{\partial}{\partial \tau}\Phi(\mathbf{x}, \tau), \tag{2.36}$$

a *derivada material em relação ao tempo*, ou seja, a derivada com respeito a τ mantendo o ponto $\mathbf{x}$ fixo. Analogamente,

$$\nabla\Phi(\mathbf{x}, \tau) = \nabla_{\mathbf{x}}\Phi(\mathbf{x}, \tau), \tag{2.37}$$

é o gradiente com respeito a $\mathbf{x}$, mantendo τ fixo, chamado de *gradiente material* de Φ.

Para um campo espacial $\Psi(\mathbf{x}^\tau, \tau)$, a *derivada espacial em relação ao tempo* (derivada em relação a τ mantendo $\mathbf{x}^\tau$ fixo) é dada por

$$\Psi'(\mathbf{x}^\tau, \tau) = \frac{\partial}{\partial \tau}\Psi(\mathbf{x}^\tau, \tau). \tag{2.38}$$

Por sua vez, o gradiente com respeito a $\mathbf{x}^\tau$ para um τ fixo (*gradiente espacial*) é expresso como

$$\text{grad } \Psi(\mathbf{x}^\tau, \tau) = \nabla_{\mathbf{x}^\tau}\Psi(\mathbf{x}^\tau, \tau). \tag{2.39}$$

O divergente de um campo vetorial ou tensorial também pode ser definido em termos materiais ou espaciais, sendo denotados, respectivamente, por Div e div.

Também pode ser definida a *derivada material no tempo de um campo espacial* Ψ como

$$\dot{\Psi} = [(\Psi_m)^{\cdot}]_s = \frac{\partial}{\partial \tau}\Psi(X^\tau(\mathbf{x}, \tau), \tau)\Big|_{\mathbf{x}=X(\mathbf{x}^\tau, \tau)}, \tag{2.40}$$

sendo na verdade a *derivada total* do campo em relação ao tempo τ. Observa-se que $\dot{\Psi}$ representa a derivada em relação ao tempo de Ψ, mantendo o ponto material fixo. Dessa maneira, para calcular $\dot{\Psi}$ toma-se a descrição material Ψ_m, determina-se a derivada no tempo e retorna-se à descrição espacial.

2.5.3 Variação da Geometria do Domínio

A grande maioria dos modelos mecânicos associados a problemas de valor de contorno são formulados através de equações diferenciais definidas ponto a ponto em um domínio $\mathcal{B}$ ou, de forma mais geral, por equações integrais sobre $\mathcal{B}$. Assim, perturbações nesse domínio produzem, necessariamente, alterações tanto nos integrandos quanto no próprio domínio de integração. A análise de sensibilidade à mudança de forma determina a variação das características associadas ao problema devido a modificações no domínio.

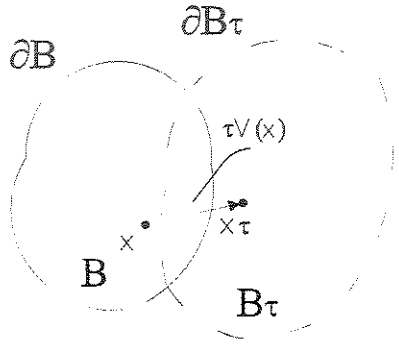


Figura 2.4: Variação do domínio e contorno de um problema.

Seja o corpo $\mathcal{B} \subset \Re^N$ tal como definido na Seção 2.1.1. Considere que este corpo sofra uma perturbação τ[1], representada pelo mapeamento suave e inversível dado em (2.31).

Dessa forma, pode-se escrever os novos domínio $\mathcal{B}_\tau$ e contorno $\partial\mathcal{B}_\tau$ perturbados, conforme ilustrado na Figura 2.4, como

$$\begin{aligned} \mathcal{B}_\tau &= \{\mathbf{x} \in \Re^N \mid \exists \mathbf{x} \in \mathcal{B},\ \mathbf{x}^\tau = X(\mathbf{x},\tau),\ \mathbf{x}_0 = 0 \text{ e } \mathcal{B}_0 = \mathcal{B}\}, \\ \partial\mathcal{B}_\tau &= \{\mathbf{x} \in \Re^N \mid \exists \mathbf{x} \in \partial\mathcal{B},\ \mathbf{x}^\tau = X(\mathbf{x},\tau),\ \mathbf{x}_0 = 0 \text{ e } \partial\mathcal{B}_0 = \partial\mathcal{B}\}. \end{aligned}$$

Sendo $X^\tau(\mathbf{x},\tau)$ um transformação suave e contínua, pode-se expandi-la em série de Taylor em torno de τ_0, ou seja,

$$X^\tau(\mathbf{x},\tau) = X^\tau(\mathbf{x},\tau_0) + \frac{\partial}{\partial\tau}X^\tau(\mathbf{x},\tau_0)(\tau-\tau_0) + o(\tau^2). \tag{2.41}$$

[1]Agora, τ representa uma perturbação qualquer no domínio, e não mais o tempo como nas Seções 2.5.1 e 2.5.2.

Na expressão anterior, o campo vetorial

$$\mathbf{v}(\mathbf{x}) = \frac{\partial X^\tau}{\partial \tau}(\mathbf{x}, \tau)$$

é definido, no contexto de Mecânica do Contínuo, como a descrição espacial da velocidade. No contexto de AS, a expressão anterior representa uma *ação de mudança de forma* ou *velocidade de mapeamento do projeto* (Haug et al., 1986; Novotny et al., 2000).

Observa-se ainda que

$$X^\tau(\mathbf{x}, \tau) = \mathbf{x}^\tau \text{ e } X^\tau(\mathbf{x}, 0) = \mathbf{x}. \tag{2.42}$$

Fazendo $\tau_0 = 0$ em (2.41) e empregando as relações anteriores, tem-se

$$\mathbf{x}^\tau = \mathbf{x} + \tau \mathbf{v}(\mathbf{x}, 0) + o(\tau^2).$$

Logo, para toda perturbação τ suficientemente pequena, ou seja, desprezando os termos de ordem mais alta $o(\tau^2)$, $\mathbf{x}^\tau$ pode ser escrito como

$$\mathbf{x}^\tau = \mathbf{x} + \tau \mathbf{v}(\mathbf{x}). \tag{2.43}$$

Pela forma em que $X^\tau(\cdot, \tau)$ foi definida, tem-se que o problema elíptico de valor de contorno (2.1), dado na configuração de referência, também deve ser satisfeito quando escrito na configuração perturbada $\mathcal{B}_\tau$, para qualquer τ, tomando a seguinte forma:

Encontrar $u_\tau \in U_\tau := U(\mathcal{B}_\tau)$, *tal que*

$$a_\tau(u_\tau, v_\tau) = l_\tau(v_\tau), \qquad \forall\ v_\tau \in V_\tau = V(\mathcal{B}_\tau). \tag{2.44}$$

2.5.4 Sensibilidade de Funcionais Definidos em um Domínio Variável

Deseja-se agora estabelecer a sensibilidade de uma função custo $\psi(\mathcal{B}_\tau)$ em relação a uma perturbação τ do domínio original. A sensibilidade de ψ é dada pela seguinte derivada

$$\left.\frac{d}{d\tau}\psi(\mathcal{B}_\tau)\right|_\tau = \lim_{\tau \to 0} \frac{\psi(\mathcal{B}_\tau) - \psi(\mathcal{B}_0)}{\tau}. \tag{2.45}$$

Para se calcular a derivada anterior, necessita-se de uma expressão explícita de ψ em função de τ. De forma análoga a (2.21), pode-se expressar a função custo ψ de forma geral

como

$$\psi\left(\mathcal{B}_\tau\right) := \int_{\mathcal{B}_\tau} \mathcal{G}(\tau, u_\tau) d\mathcal{B}_\tau = \Psi_\tau(\tau, u_\tau), \tag{2.46}$$

sendo u_τ uma função implícita de τ através da equação de estado (2.44) e $\mathcal{G}$ é uma representação da função custo.

Assim, a derivada de $\Psi_\tau(\tau, u_\tau)$ em relação a perturbação τ é dada por (Haug et al., 1986; Novotny et al., 2000)

$$\left.\frac{d}{d\tau}\Psi_\tau(\tau, u_\tau)\right|_\tau = \lim_{\tau \to 0} \frac{\Psi_\tau(\tau, u_\tau) - \Psi_0(0, u)}{\tau}. \tag{2.47}$$

Em (Haug et al., 1986; Silva, 1997), o desenvolvimento das expressões para o cálculo da derivada anterior foi feito manipulando-se diretamente as formas bilinear $a(\cdot, \cdot)$, linear $l(\cdot)$ e a equação (2.46). Uma outra alternativa para se obter a análise de sensibilidade à mudança de forma é empregar o conceito de Lagrangeano. Nesse caso, a equação de estado (2.44) é tratada como uma restrição ao problema de minimização.

A formulação através do lagrangeano foi adotada em (Novotny et al., 2000), sendo mais simples que aquela apresentada em (Haug et al., 1986; Silva, 1997). A seguir, transcreve-se a dedução apresentada em (Novotny et al., 2000).

O cálculo da derivada em (2.47) pode ser escrito como

$$\begin{cases} \text{Calcular } \left.\dfrac{d}{d\tau}\Psi_\tau(\tau, u_\tau)\right|_{\tau=0} \\ \text{Sujeito a } a_\tau(u_\tau, v_\tau) = l_\tau(v_\tau), \qquad \forall\, v_\tau \in V_\tau,\ \tau \geq 0. \end{cases}$$

O método do Lagrangeano, consiste em relaxar a restrição do problema, neste caso a equação de estado (2.44). Assim, o Lagrangeano na configuração perturbada $\mathcal{B}_\tau$ é dado por

$$\pounds_\tau(\tau, u_\tau, \lambda_\tau) = \Psi_\tau(\tau, u_\tau) + a_\tau(u_\tau, \lambda_\tau) - l_\tau(\lambda_\tau). \tag{2.48}$$

Uma vez que a equação de estado (2.47) é satisfeita para toda perturbação τ, basta calcular a derivada total do Lagrangeano $\pounds_\tau$ em relação ao parâmetro τ para obter, automaticamente, a sensibilidade da função custo Ψ_τ em relação a τ, ou seja,

$$\frac{d\pounds_\tau}{d\tau} = \frac{d\Psi_\tau}{d\tau} = \frac{\partial \pounds_\tau}{\partial \tau} + \left\langle \frac{\partial \pounds_\tau}{\partial u_\tau}, \frac{du_\tau}{d_\tau} \right\rangle + \left\langle \frac{\partial \pounds_\tau}{\partial \lambda_\tau}, \frac{d\lambda_\tau}{d_\tau} \right\rangle. \tag{2.49}$$

Denominando

$$\dot{u}_\tau = \frac{du_\tau}{d\tau} \quad \text{e} \quad \dot{\lambda}_\tau = \frac{d\lambda_\tau}{d\tau}, \tag{2.50}$$

a derivada do Lagrangeano em relação a λ_τ na direção de $\dot{\lambda}_\tau$ fica

$$\left\langle \frac{\partial \pounds_\tau}{\partial \lambda_\tau}, \dot{\lambda}_\tau \right\rangle = \left\langle \frac{\partial a_\tau}{\partial \lambda_\tau}, \dot{\lambda}_\tau \right\rangle - \left\langle \frac{\partial l_\tau}{\partial \lambda_\tau}, \dot{\lambda}_\tau \right\rangle = a_\tau(u_\tau, \dot{\lambda}_\tau) - l_\tau(\dot{\lambda}_\tau). \tag{2.51}$$

Como $\dot{\lambda}_\tau$ é um elemento arbitrário de V_τ, ao se anular (2.51) para todo $\dot{\lambda}_\tau \in V_\tau$, recupera-se a equação de estado do problema, ou seja,

Encontrar $u_\tau \in U_\tau$ *tal que*

$$\left\langle \frac{\partial \pounds_\tau}{\partial \lambda_\tau}, \dot{\lambda}_\tau \right\rangle = a_\tau(u_\tau, \dot{\lambda}_\tau) - l_\tau(\dot{\lambda}_\tau) = 0, \qquad \forall \ \dot{\lambda}_\tau \in V_\tau. \tag{2.52}$$

Por outro lado, a derivada do Lagrangeano em relação a u_τ na direção $\dot{u}_\tau$ pode ser escrita como

$$\left\langle \frac{\partial \pounds_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle = \left\langle \frac{\partial \Psi_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle + \left\langle \frac{\partial a_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle = \left\langle \frac{\partial \Psi_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle + a_\tau(\dot{u}_\tau, \lambda_\tau). \tag{2.53}$$

Da mesma forma, observa-se que $\dot{u}_\tau$ é um elemento arbitrário de V_τ e que $p_\tau \in U_\tau$. Logo, pode-se escolher λ_τ de maneira a eliminar o termo dado pela equação (2.53). Esta escolha de λ_τ conduz a chamada *equação adjunta* do problema, a qual, devido a simetria da forma bilinear $a_\tau(\cdot,\cdot)$, pode ser escrita como

Encontrar $\lambda_\tau \in U_\tau$ *tal que*

$$\left\langle \frac{\partial \pounds_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle = \left\langle \frac{\partial \Psi_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle + a_\tau(\lambda_\tau, \dot{u}_\tau) = 0, \qquad \forall \ \dot{u}_\tau \in V_\tau. \tag{2.54}$$

Como os termos (2.52) e (2.54) se anulam, conclui-se que a derivada total do Lagrangeano coincide com sua derivada parcial, ou seja,

$$\frac{d}{d\tau}\pounds_\tau(\tau, u_\tau, \lambda_\tau) = \frac{\partial}{\partial \tau}\pounds_\tau(\tau, u_\tau, \lambda_\tau). \tag{2.55}$$

A equação (2.55) leva a importante conclusão de que, para u_τ solução da equação de estado e λ_τ solução da equação adjunta, basta calcular a derivada parcial do Lagrangeano em relação a τ para se obter a sensibilidade da função custo à mudança de forma (2.47). Além do mais, o cálculo da derivada do Lagrangeano pode ser reescrito da seguinte e já consagrada maneira:

Seja $u_\tau \in U_\tau$ solução da equação de estado, isto é,

$$a_\tau(u_\tau, \dot{\lambda}_\tau) = l_\tau(\dot{\lambda}_\tau), \qquad \forall \ \dot{\lambda}_\tau \in V_\tau. \tag{2.56}$$

Seja $\lambda_\tau \in U_\tau$ solução da equação adjunta, dada por

$$a_\tau(\lambda_\tau, \dot{u}_\tau) = -\left\langle \frac{\partial \Psi_\tau}{\partial u_\tau}, \dot{u}_\tau \right\rangle, \qquad \forall \ \dot{u}_\tau \in V_\tau. \tag{2.57}$$

Assim, considerando a equação (2.55), a análise de sensibilidade à mudança de forma resulta em

$$\frac{d}{d\tau}\Psi_\tau(\tau, u_\tau) = \frac{\partial}{\partial \tau}\mathcal{L}_\tau(\tau, u_\tau, \lambda_\tau) = \frac{\partial}{\partial \tau}\Psi_\tau(\tau, u_\tau) + \frac{\partial}{\partial \tau}a_\tau(u_\tau, \lambda_\tau) - \frac{\partial}{\partial \tau}l_\tau(\lambda_\tau). \tag{2.58}$$

Para uma vasta classe de problemas de engenharia a análise de sensibilidade à mudança de forma dada pela equação (2.58) pode ser escrita em função de uma integral no contorno Γ_ϵ, a qual, geralmente, toma a forma

$$\left.\frac{d}{d\tau}\Psi_\tau(\tau, u_\tau)\right|_{\tau=0} = \int_\Gamma g_T(\sum, \mathbf{n}, \mathbf{v}, c.c.)\ d\Gamma_\epsilon, \tag{2.59}$$

sendo $\mathbf{n}$ o vetor unitário normal (exterior) à superfície Γ_ϵ, $\mathbf{v}$ a velocidade de mudança de forma dada pela equação (2.42), *c.c.* representa as condições de contorno em Γ_ϵ e o tensor $\sum$ pode ser interpretado com uma generalização do Tensor Momento Energia de Eshelby (Eshelby, 1975).

Outro importante resultado, já demonstrado em (Fancello et al., 1991), é que somente a velocidade $\mathbf{v}$ na direção normal à fronteira Γ é significativa nos cálculos de sensibilidade. Este resultado baseia-se na idéia de que somente a componente normal da velocidade, ou seja v_n, é que efetivamente produz uma mudança na forma do corpo. Sendo assim, considerando que $\mathbf{v}(\mathbf{x}) = v_n\mathbf{n}$, a Equação (2.59) pode ser reescrita como

$$\left.\frac{d}{d\tau}\Psi_\tau(\tau, u_\tau)\right|_{\tau=0} = v_n \int_\Gamma g_T(\sum, \mathbf{n}, c.c.)\ d\Gamma_\epsilon, \tag{2.60}$$

para v_n constante em Γ_ϵ.

Finalmente, basta mostrar como a expressão de análise de sensibilidade à mudança de forma (2.60) pode ser utilizada para se obter a derivada topológica, relacionando portanto ambos os conceitos. Isso está feito no próximo capítulo.

Capítulo 3

Derivada Topológica

Neste capítulo, apresenta-se a definição da derivada topológica e sua relação com a ASMF. Posteriormente, deduz-se a expressão da derivada topológica para problemas elásticos lineares tomando a energia de deformação como função custo.

Deve-se ressaltar que não houve nesse trabalho o desenvolvimento de nenhum conceito relacionado à derivada topológica e sua obtenção através da análise de sensibilidade. Aqui apenas foram apresentados e utilizados os conceitos desenvolvidos e gentilmente cedidos por A.A. Novotny. Esses resultados originais cedidos farão parte da tese de doutorado de A.A. Novotny intitulada *Análise de Sensibilidade à Mudança de Topologia*, sob orientação dos Professores R. A. Feijóo, E. Taroco do LNCC e C. Padra do Centro Atómico Bariloche (Argentina).

3.1 Definição de Derivada Topológica

Considere o domínio $\Omega \in \Re^n (n = 2 \text{ ou } 3)$ e seu contorno Γ, conforme ilustrado na Figura 3.1. Seja u_Ω a solução de uma equação diferencial parcial definida em Ω e $\psi(\Omega)$ uma função custo na forma

$$\psi(\Omega) = \Psi(\Omega, u_\Omega). \tag{3.1}$$

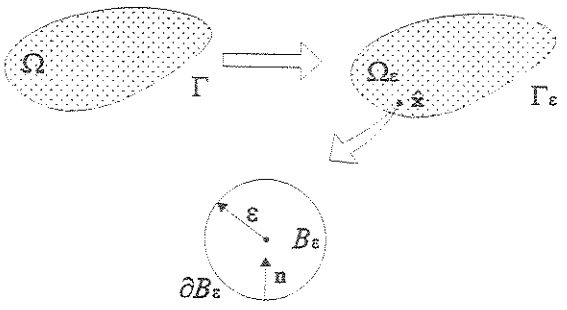


Figura 3.1: Furo criado no domínio (Novotny et al., 2000; Novotny et al., 2001a).

Seja B_ε um pequeno furo de raio ε centrado no ponto $\hat{\mathbf{x}} \in \Omega$, cujo contorno é denotado por ∂B_ε. Podem ser definidos dois domínios, o domínio original sem furo Ω e o domínio Ω_ε com um pequeno furo B_ε e contorno denotado por $\Gamma_\varepsilon = \Gamma \cup \partial B_\varepsilon$ (ver Figura 3.1). A derivada topológica, quando existe, é definida por (Schumacher, 1995; Novotny et al., 2000)

$$G_T^*(\hat{\mathbf{x}}) := \lim_{\varepsilon \to 0} \frac{\psi(\Omega_\varepsilon) - \psi(\Omega)}{f(\varepsilon)}, \tag{3.2}$$

sendo $f(\varepsilon)$ uma função negativa, monotônica e decrescente, que representa uma medida do furo B_ε ($f(\varepsilon) \to 0$ quando $\varepsilon \to 0$) .

Observa-se novamente que a derivada topológica fornece a sensibilidade de uma função custo definida em um problema à criação de um furo no domínio. O grande incoveniente de se trabalhar com a equação (3.2) é que, quando o furo é criado, não é mais possível estabelecer um mapeamento um para um entre os domínios Ω_ε e Ω, ver Figura 3.1. Em outras palavras, não existe mais um *homeomorfismo* entre os domínios Ω_ε e Ω, que se encontram em espaços topológicos diferentes. Por causa dessa inconsistência, a derivada topológica não pode ser obtida de forma convencional.

Para se contornar este problema, propõe-se em (Novotny et al., 2000) que já exista um furo B_ε no domínio inicial. Neste caso, o domínio final seria obtido a partir de uma perturbação $B_{\varepsilon+\delta\varepsilon}$ do furo B_ε. Dessa forma, o homeomorfismo entre os domínios inicial e perturbado é preservado.

Assumindo-se um problema inicial que já contenha o furo B_ε, ou seja, partindo-se de Ω_ε, pode-se causar uma pequena perturbação δ_ε em B_ε, de modo a originar o furo $B_{\varepsilon+\delta\varepsilon}$.

Para o domínio $B_{\varepsilon+\delta\varepsilon}$, tem-se o contorno $\Gamma_{\varepsilon+\delta\varepsilon} = \Gamma \cup \partial B_{\varepsilon+\delta\varepsilon}$, como ilustrado na Figura 3.2. A derivada topológica pode então ser redefinida da seguinte maneira (Novotny et al., 2000)

$$G_T(\hat{\mathbf{x}}) := \lim_{\substack{\varepsilon \to 0 \\ \delta\varepsilon \to 0}} \frac{\psi(\Omega_{\varepsilon+\delta\varepsilon}) - \psi(\Omega_\varepsilon)}{f(\varepsilon + \delta\varepsilon) - f(\varepsilon)}. \tag{3.3}$$

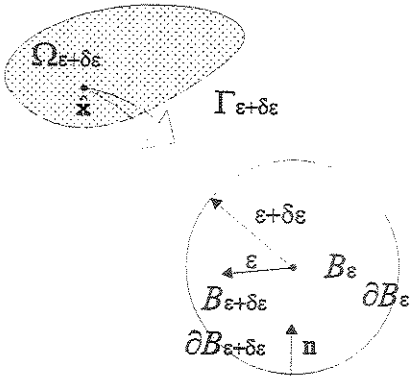


Figura 3.2: Incremento do furo criado no domínio (Novotny et al., 2000; Novotny et al., 2001a).

Esta última definição da derivada topológica (3.3), fornece, a rigor, apenas a informação da sensibilidade da função custo a um aumento do furo e não a sensibilidade do problema à criação do furo. Assim sendo, não se tem mais um indicativo se um furo deve ser efetivamente criado, conforme a definição original (3.2). Por outro lado, entende-se que expandir um furo de raio ε, quando $\varepsilon \to 0$, nada mais é que criá-lo.

Conforme já mencionado, a grande vantagem desta última forma de escrever a derivada topológica é que o homeomorfismo entre os domínios original e modificado é preservado. Entretanto, não se pode afirmar que ambas as definições (3.2) e (3.3) são equivalentes, sem uma prova matemática completa que estabeleça a relação entre as mesmas. A prova que estabelece a relação das definições é parte da demonstração de um teorema, proposto em (Novotny et al., 2000), que está transcrito no Anexo B.

3.2 Derivada Topológica via Análise de Sensibilidade à Mudança de Forma

Seja a função custo $\Psi(\cdot)$ definida nos domínios $\Omega_\varepsilon = \Omega - B_\varepsilon$ e $\Omega_{\varepsilon+\delta\varepsilon} = \Omega - B_{\varepsilon+\delta\varepsilon}$. Considerando os conceitos de análise de sensibilidade à mudança de forma apresentados na Seção 2.5, tem-se

$$\begin{aligned} \Omega_{\varepsilon+\delta\varepsilon} &= \mathcal{B}_\tau \Longrightarrow \Omega_\varepsilon = \mathcal{B}_0, \\ \Gamma_{\varepsilon+\delta\varepsilon} &= \partial\mathcal{B}_\tau \Longrightarrow \Gamma_\varepsilon = \partial\mathcal{B}_0. \end{aligned} \tag{3.4}$$

Observa-se que apenas o furo B_ε sofre uma perturbação δ_ε.

A partir da equação (2.43) e considerando que $\mathbf{v}(\mathbf{x}) = v_n\mathbf{n}$, vem que

$$\mathbf{x}^\tau = \mathbf{x} + \tau v_n \mathbf{n}. \tag{3.5}$$

Dessa forma, é possível associar a perturbação $\delta\varepsilon$ com o parâmetro τ. A partir das equações (3.4) e (3.5) e observando que $\delta\varepsilon = \|\mathbf{x}^\tau - \mathbf{x}\|$ para $\mathbf{x} \in \partial B_\varepsilon$ e $\mathbf{x}^\tau = X^\tau(\mathbf{x}, \tau) \in \partial B_{\varepsilon+\delta\varepsilon}$, tem-se que

$$\delta_\varepsilon = \|\tau v_n \mathbf{n}\| = \tau \left| v_n \right|. \tag{3.6}$$

A relação entre a derivada topológica e os conceitos de análise de sensibilidade à mudança de forma está determinada no seguinte teorema (Novotny et al., 2000):

Seja o gradiente topológico dado pela Equação (3.2), *tal que* $0 < G_T(\hat{\mathbf{x}}) < \infty$, *então o limite com* $\varepsilon \to 0$ *existe e pode ser escrito como*

$$G_T^*(\hat{\mathbf{x}}) = G_T(\hat{\mathbf{x}}) = \lim_{\varepsilon\to 0} \frac{1}{f'(\varepsilon)\left|v_n\right|} \left.\frac{d\psi(\Omega_\tau)}{d\tau}\right|_{\tau=0}. \tag{3.7}$$

A prova do teorema está dividida em duas partes. A primeira parte está relacionada à equivalência das definições (3.2) e (3.3). A segunda parte mostra que a derivada topológica definida em (3.3) existe e é dada por (3.7). A partir desta equivalência, tem-se um primeiro passo na direção de viabilizar o uso de todo o ferramental matemático desenvolvido na análise de sensibilidade à mudança de forma na obtenção da derivada topológica. Para isso, basta substituir a sensibilidade do funcional $\psi(\mathcal{B}_\tau)$ em (3.7) pelas expressões de análise de sensibilidade à mudança de forma, as quais podem ser obtidas usando os conceitos apresentados na Seção 2.5.

O ponto central a ser enfatizado nesse teorema é que a determinação da sensibilidade da função custo à criação de furos pode ser calculada no domínio sem furo Ω, devido à equivalência entre as definições (3.3) e (3.2). Os furos serão criados nas iterações do algoritmo de otimização topológica a ser dado posteriormente. Assim, não é necessário expandir a equação de estado em uma série de potências como feito em Schumacher (Schumacher, 1995). Além disso, a própria existência da derivada topológica é garantida pelo teorema anterior e pela já estabelecida teoria da ASMF (Haug et al., 1986).

Lembrando que $\psi(\Omega_\tau) = \Psi(\tau, u_\tau)$ e que $\Gamma_{\varepsilon+\delta\varepsilon} = \Gamma_\tau$, pode-se, a partir do resultado do teorema, de (2.60) e de uma escolha adequada de $f(\varepsilon)$, tal que $0 < G_T(\hat{\mathbf{x}}) < \infty$, escrever a derivada topológica da seguinte forma

$$G_T(\hat{\mathbf{x}}) = \lim_{\varepsilon\to 0} \frac{1}{f'(\varepsilon)\,|v_n|} \frac{d}{d\tau}\Psi_\tau(\tau, u_\tau)\Big|_{\tau=0} = \lim_{\varepsilon\to 0} \frac{sign\ (v_n)}{f'(\varepsilon)} \int_{\Gamma_\varepsilon} g_T(\mathbf{x})\ d\Gamma_\varepsilon. \tag{3.8}$$

Finalmente, como a parcela do contorno $\Gamma_\varepsilon = \Gamma \cup \partial B_\varepsilon$ que efetivamente está submetida à perturbação $\delta\varepsilon$ (ou $\tau\,|v_n|$) é relativa ao contorno do furo B_ε, isto é ∂B_ε, então a equação (3.8) pode ser escrita, considerando uma expansão no furo ($sign\ (v_n) = -1$), como

$$G_T(\hat{\mathbf{x}}) = -\lim_{\varepsilon\to 0} \frac{1}{f'(\varepsilon)} \int_{\partial B_\varepsilon} g_T(\mathbf{x})\ d\partial B_\varepsilon. \tag{3.9}$$

Assim sendo, para obter a expressão final da derivada topológica dada pela equação (3.7), basta calcular $g_T(\sum, \mathbf{n}, c.c.)$. O tensor $\sum$ depende estritamente da função custo escolhida e de cada problema em análise (equação de estado), e posteriormente calcular o limite com $\varepsilon \to 0$.

3.3 A Derivada Topológica na Elasticidade Linear

A aplicação do conceito de derivada topológica possibilita melhorar o desempenho de componentes estruturais de maneira sistemática. Para mostrar como obter através da derivada topológica, a topologia ótima de um componente, nesta seção será descrita a formulação de um problema elástico linear de valor no contorno e como obter a derivada topológica para este problema.

3.3.1 Definição do Problema

A forma forte do problema de elasticidade linear no domínio Ω_ε, com um pequeno furo B_ε centrado em $\hat{\mathbf{x}} \in \Omega$, é dado por

$$\begin{cases} \text{div}(\mathbf{C}\nabla\mathbf{u}^s) + \mathbf{b} = \mathbf{0} \quad \text{em} \quad \Omega_\varepsilon, \\ \mathbf{u} = \mathbf{0} \quad \text{em} \quad \Gamma_D, \\ (\mathbf{C}\nabla\mathbf{u}^s)\mathbf{n} = \mathbf{f} \quad \text{em} \quad \Gamma_N, \end{cases} \tag{3.10}$$

sendo $\mathbf{C}$ o tensor de elasticidade de quarta ordem para um material elástico, linear e isotrópico; $\mathbf{E} = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T) = \nabla\mathbf{u}^s$ é o tensor de pequenas deformações; $\mathbf{b}$ é o campo vetorial das forças de corpo; $\mathbf{f}$ é o campo vetorial das forças superficiais no contorno Γ_N; e $\mathbf{T} = \mathbf{C}\nabla\mathbf{u}^s$ é a equação constitutiva relacionando o tensor de tensões de Cauchy $\mathbf{T}$ e o tensor de pequenas deformações $\mathbf{E} = \nabla\mathbf{u}^s$. Considera-se aqui o caso de condições de contorno essenciais homogêneas, $\mathbf{u} = \mathbf{0}$, em Γ_D. Adota-se ainda, a condição de Neumann homogênea no furo, ou seja, $(\mathbf{C}\nabla\mathbf{u}^s)\mathbf{n} = \mathbf{0} \quad \text{em} \quad \partial B_\varepsilon$.

A forma fraca ou variacional, associada ao problema de valor de contorno 3.10, é obtida multiplicando-se a equação diferencial do problema por uma função $\mathbf{v}$ pertencente ao espaço das variações admissíveis V e integrando-se ao longo do domínio Ω_ε, ou seja,

$$\int_{\Omega_\varepsilon} [\text{div}(\mathbf{C}\nabla\mathbf{u}^s) + \mathbf{b}] \cdot \mathbf{v} \; d\Omega_\varepsilon = 0, \qquad \forall\, \mathbf{v} \in V. \tag{3.11}$$

Logo,

$$\int_{\Omega_\varepsilon} [\text{div}(\mathbf{C}\nabla\mathbf{u}^s) \cdot \mathbf{v} + \mathbf{b} \cdot \mathbf{v}] \; d\Omega_\varepsilon = 0, \qquad \forall\, \mathbf{v} \in V. \tag{3.12}$$

O primeiro termo do integrando anterior pode ser expresso como (Gurtin, 1981)

$$\text{div}(\mathbf{C}\nabla\mathbf{u}^s) \cdot \mathbf{v} = \text{div}[(\mathbf{C}\nabla\mathbf{u}^s)\mathbf{v}] - (\mathbf{C}\nabla\mathbf{u}^s) \cdot \nabla\mathbf{v} \tag{3.13}$$

Substituindo a relação anterior em (3.12), tem-se que

$$\int_{\Omega_\varepsilon} [\text{div}[(\mathbf{C}\nabla\mathbf{u}^s)\mathbf{v}] - (\mathbf{C}\nabla\mathbf{u}^s) \cdot \nabla\mathbf{v} + \mathbf{b} \cdot \mathbf{v}] \; d\Omega_\varepsilon = 0. \tag{3.14}$$

Aplicando-se agora o teorema da divergência ao primeiro termo vem que

$$\int_{\Omega_\varepsilon} \text{div}[(\mathbf{C}\nabla\mathbf{u}^s)\mathbf{v}] \; d\Omega_\varepsilon = \int_{\partial\Omega_\varepsilon} (\mathbf{C}\nabla\mathbf{u}^s)^T\mathbf{v} \cdot \mathbf{n} \; d\partial\Omega_\varepsilon = \int_{\partial\Omega_\varepsilon} (\mathbf{C}\nabla\mathbf{u}^s)\mathbf{n} \cdot \mathbf{v} \; d\partial\Omega_\varepsilon, \tag{3.15}$$

sendo $\mathbf{n}$ o campo vetorial dos vetores normais nos pontos do contorno $\partial\Omega_\varepsilon$.

Substituindo (3.14) e (3.15) e empregando as condições de contorno indicadas em (3.10), obtém-se a forma fraca do problema de elasticidade linear

$$\int_{\Omega_\varepsilon} \mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{v}^s \, d\Omega_\varepsilon = \int_{\Omega_\varepsilon} \mathbf{b}\cdot\mathbf{v} \, d\Omega_\varepsilon + \int_{\Gamma_N} \mathbf{f}\cdot\mathbf{v} \, d\Gamma_N, \qquad \forall\, \mathbf{v}\in V. \tag{3.16}$$

Pode-se escrever a expressão anterior na forma abstrata (2.1), isto é,

$$a(\mathbf{u},\mathbf{v}) = l(\mathbf{v}), \qquad \forall\, \mathbf{v}\in V, \tag{3.17}$$

sendo as formas bilinear $a(\cdot,\cdot)$ e linear $l(\cdot)$ dadas, nesse caso, por

$$\begin{aligned} a(\mathbf{u},\mathbf{v}) &= \textstyle\int_{\Omega_\varepsilon} \mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{v}^s \, d\Omega_\varepsilon, \\ l(\mathbf{v}) &= \textstyle\int_{\Omega_\varepsilon} \mathbf{b}\cdot\mathbf{v} \, d\Omega_\varepsilon + \int_{\Gamma_N} \mathbf{f}\cdot\mathbf{v} \, d\Gamma_N. \end{aligned} \tag{3.18}$$

Como indicado em (2.44), a forma fraca (3.17) também é satisfeita para qualquer perturbação τ do domínio do problema. Portanto, tem-se o seguinte problema

Encontrar $u_\tau \in U_\tau$, *tal que*

$$a_\tau(\mathbf{u}_\tau,\mathbf{w}_\tau) = l_\tau(\mathbf{w}_\tau), \qquad \forall\, \mathbf{w}_\tau \in V_\tau, \tag{3.19}$$

sendo $a_\tau(\mathbf{u}_\tau,\mathbf{w}_\tau)$ e $l_\tau(\mathbf{w}_\tau)$ escritos, respectivamente, como

$$a_\tau(\mathbf{u}_\tau,\mathbf{w}_\tau) = \int_{\Omega\tau} \mathbf{C}\nabla_\tau\mathbf{u}_\tau^s \cdot \nabla_\tau\mathbf{w}_\tau^s \, d\Omega_\tau, \tag{3.20}$$

$$l_\tau(\mathbf{w}_\tau) = \int_{\Omega\tau} \mathbf{b}_\tau\cdot\mathbf{w}_\tau \, d\Omega_\tau + \int_{\Gamma_N} \mathbf{f}_\tau\cdot\mathbf{w}_\tau \, d\Gamma_\tau, \tag{3.21}$$

e $\nabla_\tau(\cdot)$ é adotado para denotar $\nabla_\tau := \frac{\partial}{\partial x^\tau}(\cdot)$.

3.3.2 Cálculo da Derivada Topológica

Para se obter a expressão (3.7) da derivada topológica via análise de sensibilidade, necessita-se primeiramente calcular a derivada da função custo $\Psi(\tau, u_\tau)$, em relação ao parâmetro τ, para $\tau = 0$. A sensibilidade da função custo Ψ_τ pode ser obtida recorrendo-se ao método Lagrangeano, ou seja, aplicando-se a expressão (2.58).

Qualquer um dos funcionais de performance e sua respectiva sensibilidade, dados no Apêndice A, podem ser empregados. Da mesma maneira que em (Novotny et al., 2000), a energia de deformação será aqui utilizada como função objetivo. Esta função é particularmente interessante pois, além de simplificar os cálculos, ao não exigir a solução de um problema adjunto, tem uma interpretação física bem definida. A minimização da energia

interna leva a componentes de alta resistência mecânica, ou seja, a componentes bastante rígidos, o que é interessante em muitas aplicações da engenharia. Dessa forma, a função custo adotada é definida como

$$\Psi_\tau(\tau, \mathbf{u}_\tau) = \frac{1}{2}\int_{\Omega\tau} \mathbf{C}\nabla_\tau \mathbf{u}_\tau^s \cdot \nabla_\tau \mathbf{w}_\tau^s d\Omega_\tau = \frac{1}{2} a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau). \tag{3.22}$$

A sensibilidade à mudança de forma do funcional de energia de deformação é dado pela expressão (A.1) do Apêndice A. Essa mesma expressão, será deduzida a seguir empregando-se o método do Lagrangeano através da equação (2.58), conforme apresentado em (Novotny et al., 2000).

Uma vez caracterizada a função custo a ser minimizada, pode-se então partir para a obtenção da derivada do Lagrangeano (2.58). Sendo assim, a equação adjunta (2.57) para este caso particular fica

$$\begin{aligned} a_\tau(\lambda_\tau, \dot{\mathbf{u}}_\tau) &= -\left\langle \frac{\partial \Psi_\tau}{\partial \mathbf{u}_\tau}, \dot{\mathbf{u}}_\tau \right\rangle = a_\tau(\mathbf{u}_\tau, \dot{\mathbf{u}}_\tau) \qquad \forall\ \dot{\mathbf{u}}_\tau \in V_\tau, \\ &\Rightarrow \lambda_\tau = -\mathbf{u}_\tau. \end{aligned} \tag{3.23}$$

Portanto, para o funcional energia de deformação, as soluções das equações adjunta e de estado coincidem. Dessa maneira, não há a necesidade de se resolver um problema adjunto.

Substituindo a (3.23) na derivada do Lagrangeano (2.58), tem-se

$$\begin{aligned} \frac{\partial}{\partial \tau}\mathcal{L}_\tau(\tau, \mathbf{u}_\tau, \mathbf{u}_\tau) &= \frac{1}{2}\frac{\partial}{\partial \tau} a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau) - \frac{\partial}{\partial \tau} a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau) + \frac{\partial}{\partial \tau} l_\tau(\mathbf{u}_\tau) \\ &= -\frac{1}{2}\frac{\partial}{\partial \tau} a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau) + \frac{\partial}{\partial \tau} l_\tau(\mathbf{u}_\tau). \end{aligned} \tag{3.24}$$

Para se obter as derivadas na configuração original $\Omega_0 = \Omega_\varepsilon$, ou seja, em $\tau = 0$, pode-se utilizar o teorema do transporte de Reynolds (Gurtin, 1981). Assim, de maneira geral, a derivada da função custo é dada por

$$\frac{d}{d\tau}\Psi_\tau(\tau, \mathbf{u}_\tau)\bigg|_{\tau=0} = \frac{\partial}{\partial \tau}\int_{\Omega_\tau} \mathcal{G}(\tau, \mathbf{u}_\tau)\, d\Omega_\tau\bigg|_{\tau=0} = \frac{\partial}{\partial \tau}\int_{\Omega\varepsilon} \left(\frac{\partial \mathcal{G}}{\partial \tau}\bigg|_{\tau=0} + \mathcal{G}\ \mathrm{div}\mathbf{v} \right) d\Omega_\varepsilon. \tag{3.25}$$

Utilizando-se a equação (3.25) para derivar a forma bilinear $a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau)$ tem-se

$$\begin{aligned} \frac{d}{d\tau} a_\tau(\mathbf{u}_\tau, \mathbf{u}_\tau)\bigg|_{\tau=0} &= \frac{d}{d\tau}\int_{\Omega_\tau} \mathbf{C}\nabla_\tau \mathbf{u}_\tau^s \cdot \nabla_\tau \mathbf{u}_\tau^s\, d\Omega_\tau\bigg|_{\tau=0} \\ &= \int_{\Omega\varepsilon} \left[\frac{\partial}{\partial \tau}\mathbf{C}\nabla_\tau \mathbf{u}_\tau^s \cdot \nabla_\tau \mathbf{u}_\tau^s\bigg|_{\tau=0} + \mathbf{C}\nabla_\tau \mathbf{u}_\tau^s \cdot \nabla_\tau \mathbf{u}_\tau^s\ \mathrm{div}\mathbf{v} \right] d\Omega_\varepsilon. \end{aligned} \tag{3.26}$$

A derivada do gradiente simétrico de um campo vetorial é dada por (Silva, 1997)

$$\left.\frac{\partial}{\partial\tau}\left(\nabla_\tau \mathbf{u}_\tau^s\right)\right|_{\tau=0} = -\left(\nabla\mathbf{u}\nabla\mathbf{v}\right)^s. \tag{3.27}$$

Substituindo este último resultado na equação (3.26) vem

$$\begin{aligned}\left.\frac{d}{d\tau}a_\tau(\mathbf{u}_\tau,\mathbf{u}_\tau)\right|_{\tau=0} &= -\int_{\Omega\varepsilon}\left[\begin{array}{c}\mathbf{C}\left(\nabla\mathbf{u}\nabla\mathbf{v}\right)^s\cdot\nabla\mathbf{u}^s+\mathbf{C}\nabla\mathbf{u}^s\cdot\left(\nabla\mathbf{u}\nabla\mathbf{v}\right)^s\\ -\mathbf{C}\nabla\mathbf{u}^s\cdot\mathbf{u}^s\ \mathrm{div}\mathbf{v}\end{array}\right]d\Omega_\varepsilon\\ &= -\int_{\Omega\varepsilon}\left[2\nabla\mathbf{u}^T\cdot\mathbf{C}\nabla\mathbf{u}^s-\mathbf{C}\left(\nabla\mathbf{u}^s\cdot\nabla\mathbf{u}^s\right)\mathbf{I}\right]\ \nabla\mathbf{v}\ d\Omega_\varepsilon.\end{aligned} \tag{3.28}$$

Da mesma maneira, pode-se calcular a derivada do funcional $l_\tau(\mathbf{u}_\tau)$, considerando $\mathbf{b}$ constante, da seguinte forma

$$\begin{aligned}\left.\frac{d}{d\tau}l_\tau(\mathbf{u}_\tau)\right|_{\tau=0} &= -\int_{\Omega\varepsilon}(\mathbf{b}\ \mathrm{div}\mathbf{v})\cdot\mathbf{u}\ d\Omega_\varepsilon+\int_{\Gamma_N}(\mathbf{f}\ \mathrm{div}_\Gamma\mathbf{v})\cdot\mathbf{u}\ d\Gamma_\varepsilon\\ &= \int_{\Omega\varepsilon}(\mathbf{b}\cdot\mathbf{u})\mathbf{I}\cdot\nabla\mathbf{v}\ d\Omega_\varepsilon.\end{aligned} \tag{3.29}$$

O divergente superficial da velocidade é dado por $\mathrm{div}_\Gamma\ \mathbf{v}=(\mathbf{I}-\mathbf{n}\otimes\mathbf{n})\cdot\nabla\mathbf{v}$, sendo $\mathbf{v}=0$ sobre Γ_N.

Substituindo as equações (3.28) e (3.29) em (3.24), a derivada do Lagrangeano fica

$$\begin{aligned}\left.\frac{\partial\mathcal{L}_\tau}{\partial\tau}\right|_{\tau=0} &= \frac{1}{2}\int_{\Omega\varepsilon}\left[2\nabla\mathbf{u}^T\mathbf{C}\nabla\mathbf{u}^s-\mathbf{C}\left(\nabla\mathbf{u}^s\cdot\nabla\mathbf{u}^s\right)\mathbf{I}+2(\mathbf{b}\cdot\mathbf{u})\mathbf{I}\right]\cdot\ \nabla\mathbf{v}\ d\Omega_\varepsilon\\ &= \frac{1}{2}\sum\cdot\nabla\mathbf{v}\ d\Omega_\varepsilon,\end{aligned} \tag{3.30}$$

sendo o tensor $\sum$ dado por

$$\sum=\left[2\nabla\mathbf{u}^T\mathbf{C}\nabla\mathbf{u}^s-\mathbf{C}\left(\nabla\mathbf{u}^s\cdot\nabla\mathbf{u}^s\right)\mathbf{I}+2(\mathbf{b}\cdot\mathbf{u})\mathbf{I}\right]. \tag{3.31}$$

A partir da relação

$$\mathrm{div}\left(\sum{}^T\mathbf{v}\right)=\sum\cdot\nabla\mathbf{v}+\mathbf{v}\cdot\mathrm{div}\ \sum, \tag{3.32}$$

sendo $\mathrm{div}\sum=0$ (admitindo que o campo $\mathbf{u}$ satisfaz a equação de estado), e do teorema da divergência, pode-se mostrar que a derivada do Lagrangeano, na equação (3.30), torna-se (Novotny et al., 2000)

$$\left.\frac{d\Psi_\tau}{d\tau}\right|_{\tau=0}=\left.\frac{\partial\mathcal{L}_\tau}{\partial\tau}\right|_{\tau=0}=\int_{\Gamma\varepsilon}\sum\cdot\nabla\mathbf{v}\ d\Gamma_\varepsilon=\int_{\partial B\varepsilon}\sum\cdot\nabla\mathbf{v}\ d\partial B\varepsilon. \tag{3.33}$$

Considerando $\mathbf{v}=v_n\mathbf{n}$, com v_n constante, então, tem-se que

$$\left.\frac{d\Psi_\tau}{d\tau}\right|_{\tau=0}=v_n\int_{\partial B\varepsilon}\sum\mathbf{n}\cdot\mathbf{n}\ d\partial B\varepsilon, \tag{3.34}$$

sendo

$$\sum \mathbf{n} \cdot \mathbf{n} = (\mathbf{C}\nabla\mathbf{u}^s)\,\mathbf{n} \cdot (\nabla\mathbf{u})\mathbf{n} - \frac{1}{2}\mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{u}^s + \mathbf{b} \cdot \mathbf{u}. \tag{3.35}$$

Além do mais, como $(\mathbf{C}\nabla\mathbf{u}^s)\,\mathbf{n} = 0$ sobre ∂B_ε, equação (3.10), tem-se que

$$\int_{\partial B\varepsilon} \sum \mathbf{n} \cdot \mathbf{n}\; d\partial B\varepsilon = \int_{\partial B\varepsilon} -\left(\frac{1}{2}\mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{u}^s - \mathbf{b} \cdot \mathbf{u}\right)\; d\partial B\varepsilon. \tag{3.36}$$

Substituindo (3.36) em (3.35), tem-se

$$\left.\frac{d\Psi_\tau}{d\tau}\right|_{\tau=0} = v_n \int_{\partial B\varepsilon} g_T\; \partial B\varepsilon,$$

sendo

$$g_T(\mathbf{x}) = -\left(\frac{1}{2}\mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{u}^s - \mathbf{b} \cdot \mathbf{u}\right). \tag{3.37}$$

Finalmente, substituindo a equação (3.37) na definição da derivada topológica via análise de sensibilidade dada na equação (3.9) e adotando-se $f(\varepsilon) = meas(\Omega_\varepsilon) - meas(\Omega) = -meas(B_\varepsilon)$, tem-se $f'(\varepsilon) = -meas(\partial B_\varepsilon)$. Logo, a expressão final do gradiente topológico para problemas de elasticidade tridimensionais, cuja função custo é a energia de deformação, pode ser escrita, admitindo $g_T(\mathbf{x})$ contínuo ao longo de ∂B_ε, da seguinte maneira

$$G_T(\mathbf{x})_{3D} = \lim_{\varepsilon\to 0} \frac{1}{meas(\partial B\varepsilon)} \int_{\partial B_\varepsilon} g_T(\mathbf{x})\; d\partial B\varepsilon = -\left(\frac{1}{2}\mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{u}^s - \mathbf{b} \cdot \mathbf{u}\right)_{\hat{\mathbf{x}}}. \tag{3.38}$$

Note que este resultado pode ser facilmente particularizado para o caso de tensão plana, bastando para isso realizar a integração analítica ao longo da espessura do modelo, o que resulta em

$$G_T(\mathbf{x})_{2D} = -\rho\left(\frac{1}{2}\mathbf{C}\nabla\mathbf{u}^s \cdot \nabla\mathbf{u}^s - \mathbf{b} \cdot \mathbf{u}\right)_{\hat{\mathbf{x}}}, \tag{3.39}$$

sendo ρ a espessura da peça e os campos de tensão $(\mathbf{C}\nabla\mathbf{u}^s)$, de deformação $(\nabla\mathbf{u}^s)$ e deslocamentos $\mathbf{u}$ devem ser redefinidos de modo a satisfazerem as hipóteses de tensão plana.

Assim, observa-se que, para o caso do funcional de performance ser a energia de deformação, o cálculo do gradiente topológico é efetuado como um simples pós-processamento da solução de elementos finitos.

3.4 Algoritmo de Otimização Topológica

Como mencionado anteriormente, o conceito de derivada topológica representa a sensibilidade de uma função custo ψ à criação de um pequeno furo no domínio Ω. No entanto, deve-se colocar alguma restrição relacionada à medida do volume final de forma a evitar que os algoritmos de otimização topológica conduzam à solução trivial do problema, ou seja, $meas(\Omega) = 0$. Assim, minimiza-se a função custo ψ impondo um limite para a retirada de massa do domínio. Tem-se, então, o seguinte problema de minimização

$$\begin{cases} \text{Minimize: } \psi(\Omega) \\ \text{Sujeito a: } meas(\Omega) \geq meas(\bar{\Omega}) \end{cases}$$

Propõe-se então, o seguinte algoritmo geral de otimização topológica, baseado no conceito de derivada topológica e válido para qualquer tipo de funcional (Novotny et al., 2000; Cea et al., 1998).

Seja a seqüência domínios $\Omega_{k+1} = \{\Omega_k \mid meas(\Omega_k) \geq meas(\bar{\Omega})\}$, sendo k a k-ésima iteração, então

- Fornecer o domínio inicial Ω_0 e a restrição $meas(\bar{\Omega})$.
- Enquanto $meas(\Omega_k) \geq meas(\bar{\Omega})$ faça
  1. Encontrar a solução direta u_k e λ_k.
  2. Calcular $G_T(\hat{\mathbf{x}})_k$.
  3. Criar os "furos" nos pontos $\hat{\mathbf{x}}$ correspondentes a $\eta_k^{\text{inf}} \leq G_T(\hat{\mathbf{x}})_k \leq \eta_k^{\text{sup}}$ (η_k^{inf} e η_k^{sup} são calculados de forma proporcional ao volume de material a ser retirado em cada iteração), até que a quantidade de massa a ser retirada na iteração k seja atingida..
  4. Definir o novo domínio $k + 1$.
- Nesse ponto, espera-se ter a topologia ótima em mãos.

O algoritmo acima descreve como definir novos domínios iterativamente, minimizando a função custo adotada. A idéia é utilizar a informação obtida da derivada topológica, ou seja, a sensibilidade de cada nó à criação de um furo, para se remover massa do domínio. Os parâmetros η_k^{inf} e η_k^{sup} são calculados de forma proporcional ao volume de material a ser retirado a cada iteração. A partir da quantidade de massa a ser removida dada pelo usuário, define-se um valor inferior de derivada topológica que é atribuído ao parâmetro η_k^{inf}. O valor contido em η_k^{inf} tem a seguinte interpretação: se forem criados furos em todos os nós $\hat{\mathbf{x}}$ com $G_T(\hat{\mathbf{x}})_k \leq \eta_k^{\text{inf}}$, o problema terá sua massa reduzida à determinada pelo usuário na iteração. Dessa forma, η_k^{inf} é um parâmetro que indica, a partir de um valor de derivada topológica, a viabilidade da criação do furo. De maneira semelhante, porém com o objetivo oposto, pode-se determinar um η_k^{sup}. Determinando o parâmetro η_k^{sup}, poderia-se adotar alguma estratégia de reinserção de elementos. Nenhuma estratégia neste sentido foi adotada nessa dissertação.

A partir da definição de um η_k^{inf}, ou seja, a partir da identificação dos nós menos sensíveis à criação de furos, os "furos"[1] são criados. O procedimento adotado para que se chegasse o mais próximo possível da criação de um furo em um nó, foi remover todos os elementos que compartilham esse nó. Dessa maneira, em um domínio bidimensional são criadas regiões de vazio similares a um furo e, em um domínio tridimensional, são criadas regiões de vazio semelhantes a uma esfera. Essas regiões de vazio estãos centradas nos nós escolhidos. Obviamente, estas estratégias podem se tornar grosseiras em função do grau de refinamento da malha de elementos finitos e da forma do elemento utilizados.

Embora esse algoritmo seja bastante simples, permite a obtenção de resultados satisfatórios quando aplicado na otimização topológica, como mostrado na Seção 4. Nada impede que novos algoritmos sejam propostos para que se aproveitem melhor as informações contidas na derivada topológica. Deve-se ressaltar que nas iterações do algoritmo, da maneira como foi implementado, não existe a verificação de falha do componente. Dessa forma, o domínio será reduzido ao objetivo determinado $meas(\tilde{\Omega})$ mesmo que o componente saia de sua fase elástica, segundo algum critério, em uma das iterações.

[1] A palavra furos aparece entre aspas porque apesar do procedimento adotado assemelhar-se à criação de furos, o que está sendo efetivamente criado são vazios em torno dos nós.

3.4.1 Estrutura de Dados Necessária para Funcionamento do Algoritmo

Segue uma breve descrição da estrutura de dados necessária para se colocar o algoritmo em funcionamento. Não é objetivo nesta seção descrever aspectos da implementação do algoritmo, o que será feito mais tarde na Seção 5.4.

Para implementação do processo de otimização será necessária a seguinte estrutura de dados:

- uma variável para guardar a massa inicial do problema;
- uma variável para guardar a massa que se deseja obter ao fim da otimização;
- uma variável para guardar o número máximo de iterações;
- uma variável para guardar a quantidade de massa a ser removida por iteração;
- um vetor para guardar a quantidade de massa do problema em cada iteração;
- um vetor para guardar o número de elementos removidos em cada iteração;
- um vetor de tamanho igual ao número total de nós da discretização do domínio (malha do problema de elementos finitos) para armazenar os valores da derivada topológica e
- um vetor, de tamanho igual ao número total de nós da discretização do domínio, contendo os números dos nós em ordem crescente de sensibilidade à criação do furo.

3.4.2 Funcionamento do Algoritmo

A partir da estrutura de dados descrita acima, pode-se descrever o funcionamento do algoritmo. Primeiramente, os parâmetros de otimização são preenchidos a partir de valores dados pelo usuário. Esses parâmetros de otimização são as variáveis massa final do problema, número de iterações, quantidade de massa a ser removida por iteração e o número máximo de iterações.

Após a determinação desses parâmetros, o problema é resolvido pelo MEF e o vetor contendo a derivada topológica nos nós é preenchido. A partir dos valores contidos no vetor de derivada topológica, o vetor da ordem preferencial dos nós para criação dos furos é determinado. Os furos são criados, nessa seqüência determinada, até que a quantidade de massa a ser removida na iteração seja atingida. Um novo problema é construído, com menos elementos, e o processo é novamente executado.

Não se empregou o parâmetro η_k^{inf} de acordo com a definição dada na seção anterior, ou seja, não foram criados furos em todos os nós $\hat{\mathbf{x}}$ com $G_T(\hat{\mathbf{x}})_k \leq \eta_k^{\text{inf}}$. Adotou-se uma estratégia distinta para a criação dos furos. Nesse caso, removem-se todos os elementos que são compartilhados por um nó em torno do qual um furo deve ser criado. Essa estratégia implica pouco controle na quantidade de massa removida para cada furo criado. Isso ocorre devido ao fato de que nós são compartilhados por diferentes números de elementos, implicando a remoção de uma quantidade diferente de massa quando o furo é criado. Dessa forma, após a criação de cada furo, foi necessário conferir se a quantidade de massa a ser removida na iteração não foi ultrapassada.

Capítulo 4

Resultados da Aplicação da DT em Problemas de Elasticidade

Neste capítulo, mostram-se os resultados obtidos com a aplicação da derivada topológica em problemas $2D$ e $3D$ de otimização, considerando a energia de deformação como função objetivo.

Cada um dos exemplos tem um objetivo específico que será descrito em cada uma de suas introduções. As críticas aos resultados, bem como comentários sobre vantagens e desvantagens percebidas no algoritmo serão feitas, preferencialmente, em uma seção separada. Dessa forma, muitos dos comentários serão apenas observados no exemplo e depois descritos em detalhes na Seção 4.6. Para visualização das geometrias resultantes, bem como das distribuições de quaisquer quantidades ao longo do domínio foi usado o programa *ANSYS* (Ansys, 1995).

4.1 Componente em Estado de Tensão Plana

Esse exemplo tem o objetivo de ressaltar algumas características importantes do algoritmo implementado. A seguir, mostram-se características de convergência da solução. Faz-se uma comparação da solução obtida para o mesmo problema utilizando a otimização topológica

disponível no programa *ANSYS*.

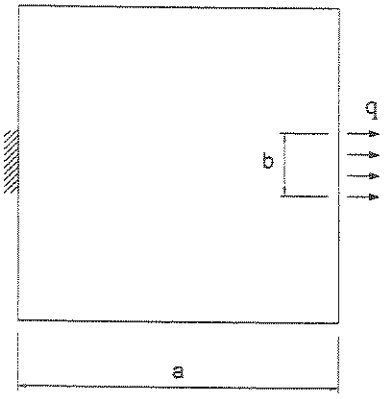


Figura 4.1: Definição do problema de estado plano.

Seja o problema de estado plano de tensão ilustrado na Figura 4.1. A forma inicial do problema é dada por um quadrado de lado $a = 100\ mm$, submetido a uma força distribuída $q = 2,5\ \frac{N}{mm}$ em uma linha de $b = 15\ mm$ de comprimento de sua aresta direita. Essa força é equilibrada pelas reações no engaste da região de sua aresta esquerda. A região engastada também tem comprimento b e está alinhada com a força distribuída.

O problema foi discretizado em uma malha de elementos finitos com 1390 elementos triangulares, ilustrada na Figura 4.2.

Uma primeira característica relevante do algoritmo a ser destacada são os decrementos lineares da massa nas iterações. A comparação da convergência do problema utilizando o programa baseado na derivada topológica com a convergência desse mesmo problema simulado no *ANSYS* pode ser encontrada na Figura 4.3. Conforme ilustrado na Figura 4.3a, a forma final do componente obtida pelo algoritmo de otimização do *ANSYS* tem uma área de aproximadamente 222 mm^2. Coerentemente, a forma final obtida pelo algoritmo de otimização baseado na derivada topológica tem uma área de aproximadamente 215 mm^2, conforme ilustrado na Figura 4.3b.

O critério utilizado para comparar os valores de convergência foi permitir uma variação

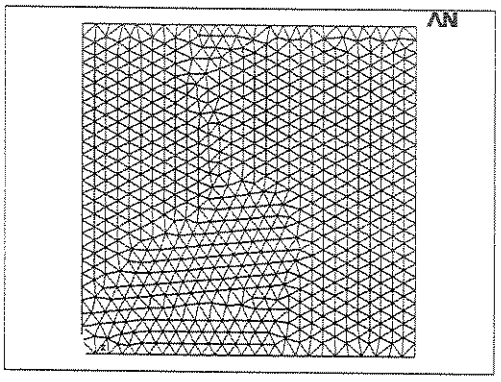
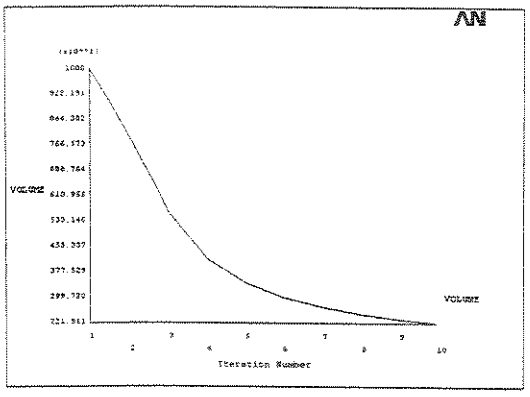


Figura 4.2: Malha utilizada no problema de estado plano.

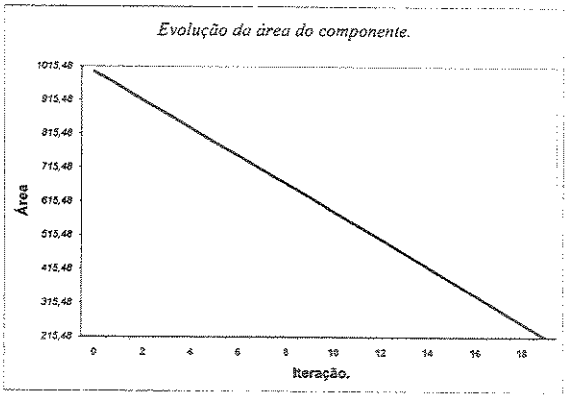
de aproximadamente 50% na máxima tensão equivalente de von Mises entre o problema original e o problema otimizado. Percebe-se que o *ANSYS* possui uma estratégia adaptável de evolução da otimização, com muita massa sendo retirada nas primeiras iterações e pouca nas últimas. No algoritmo da derivada topológica em todas as iterações é removida, aproximadamente, uma mesma quantidade de massa. Essa queda linear deve-se à estratégia adotada, já comentada na Seção 3.4.2, de remoção de massa. A partir de alterações na definição do parâmetro η_k^{inf}, pode-se obter resultados mais próximos aos obtidos pelo *ANSYS*. Isso explica a diferença no número de iterações encontrados nos dois casos.

Os resultados da otimização topológica realizada no *ANSYS* podem ser encontrados na Figura 4.4. Na Figura 4.4a, tem-se a forma final obtida para o componente determinada a partir uma distribuição de densidades ao longo do componente. Na Figura 4.4b, tem-se a distribuição da tensão equivalente de von Mises.

O resultado da otimização realizada através do algoritmo da derivada topológica pode ser encontrado na Figura 4.5a, a qual apresenta os valores nodais da tensão equivalente de von Mises. Nessa mesma figura, pode-se observar a forma final obtida para o componente. Na Figura 4.5b, encontra-se a distribuição da tensão de von Mises para o problema original. Os valores de tensão equivalente de von Mises encontrados para os problemas antes e depois da otimização (ver Figura 4.5) permitem confirmar uma variação de aproximadamente 43%,

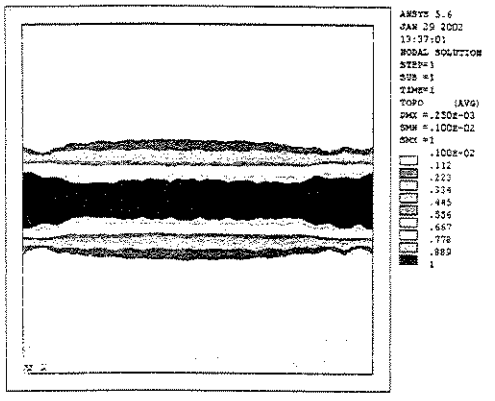


(a) Convergência da resposta (ANSYS).

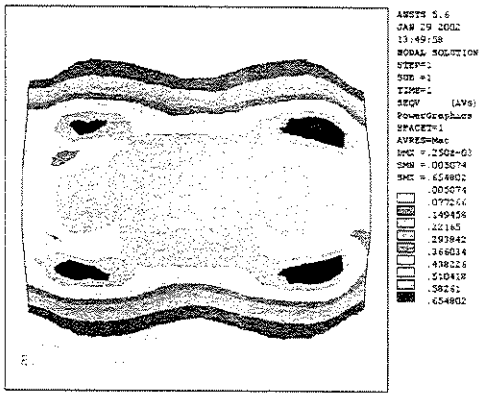


(b) Convergência da resposta (Derivada Topológica).

Figura 4.3: Comparativo da convergência dos problemas.



(a) Forma final obtida.

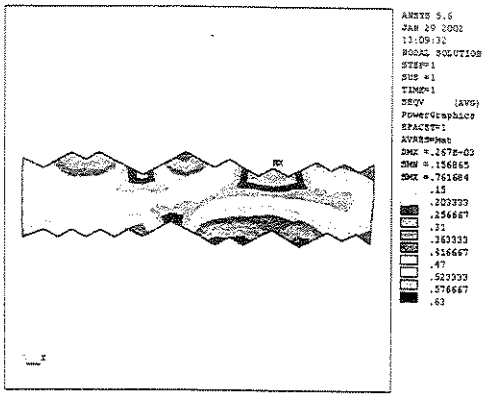


(b) Distribuição da tensão de Von Mises na forma final.

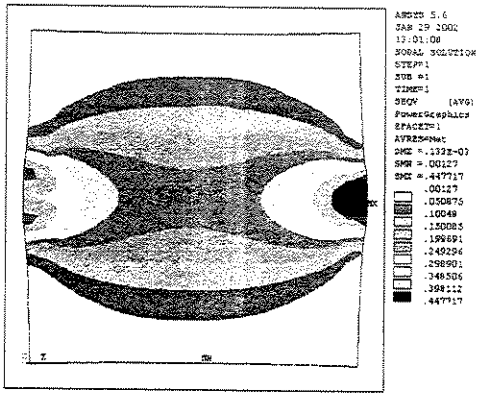
Figura 4.4: Resultados obtidos no Ansys.

próxima a meta de 50%.

Pode-se perceber comparando a Figura 4.5a com a Figura 4.4b, que a distribuição de tensão obtida nas duas otimizações são equivalentes. Na região alinhada com o carregamento e a restrição, pode-se perceber que essa tensão varia de $0,4\ MPa$ a $0,6\ MPa$. Pode-se perceber ainda que a distribuição apresentada na Figura 4.4b é mais homogênea que a apresentada na Figura 4.5a. Isso ocorre devido à geometria muito irregular resultante do processo da otimização (Figura 4.5a) na qual a distribuição de tensão é avaliada.



(a) Forma final obtida.



(b) Distribuição da tensão de Von Mises na forma final.

Figura 4.5: Resultados obtidos no algoritmo da derivada topológica.

4.2 Estrutura de Michell

Esse exemplo tem o objetivo de validar os resultados obtidos pelo método implementado. O problema de Michell, ilustrado na Figura 4.6, tem solução analítica que é amplamente discutida em (Hemp, 1973). Apresentam-se as soluções obtidas para a estrutura de Michell em dois casos. Em cada um dos casos as condições de contorno aplicadas à estrutura são diferentes. Para esses exemplos, as forças e dimensões ficarão parametrizadas por, respectivamente, P e a, pois seus valores numéricos são irrelevantes para o propósito determinado.

4.2.1 Primeiro Caso

Uma ilustração da definição do problema para o primeiro caso estudado é encontrada na Figura 4.6. Nesse caso, a parte do contorno sobre os apoios tem seus graus de liberdade de deslocamento na direção x e y travados.

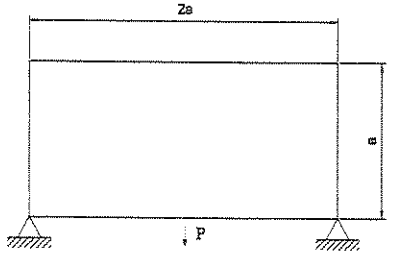


Figura 4.6: Definição do problema de Michell.

Para resolução do problema, foi utilizada uma malha de elementos finitos com 5000 elementos quadrados. A malha utilizada nos dois casos estudados foi a mesma e pode ser encontrada na Figura 4.7. No primeiro caso, foi especificada uma massa final objetivo de 40% da massa inicial, a ser obtida após 20 iterações.

Na Figura 4.8, são encontrados os resultados de algumas das iterações iniciais e da iteração final do algoritmo. Na Figura 4.8c, tem-se a forma final da estrutura, que é formada por um arco reforçado por tirantes. Essa é a solução teórica obtida para estrutura de Michell.

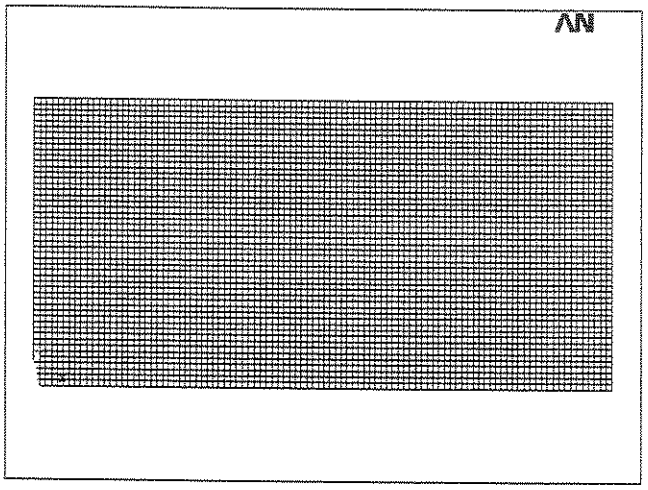
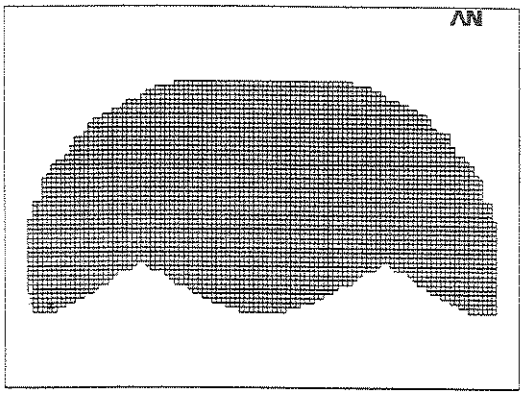
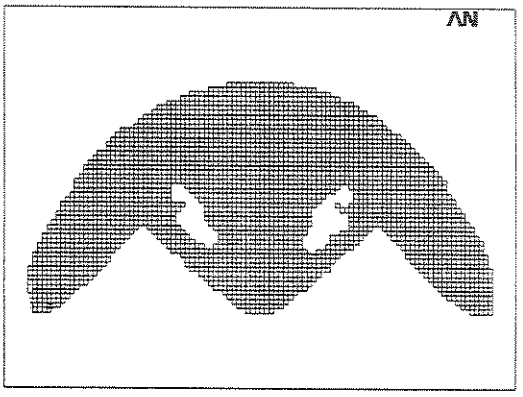


Figura 4.7: Malha utilizada para os problemas de Michell e Michell modificado.

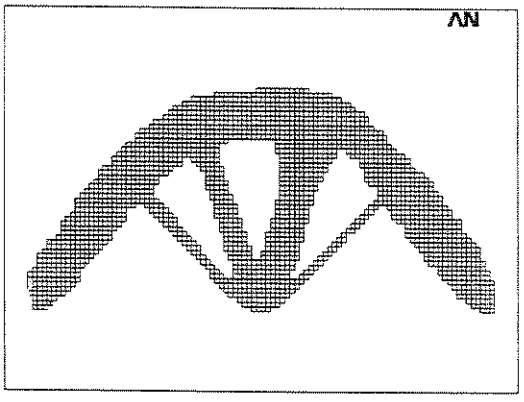
Percebe-se que os reforços estão coerentemente deslocados na direção do centro da estrutura onde a carga está aplicada. Pode-se perceber nesse exemplo uma pequena assimetria entre os dois lados da estrutura, considerando o ponto de aplicação da força como um eixo de simetria. Essas assimetrias não ocorrem devido a distorções da malha ou devido ao ponto onde o carregamento foi aplicado. Esse fato será comentado na Seção 4.6.



(a) Resultado da sétima iteração.



(b) Resultado da décima quarta iteração.



(c) Resultado da vigésima iteração.

Figura 4.8: Otimização do problema de Michell.

4.2.2 Segundo Caso

O segundo caso estudado é encontrado na Figura 4.9. Nesse caso, o contorno esquerdo tem todos os seus graus de liberdade de deslocamento, ou seja, deslocamentos na direção x e y, travados. Já o contorno direito, apenas os deslocamentos na direção y estão travados.

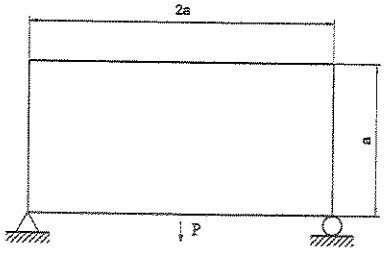
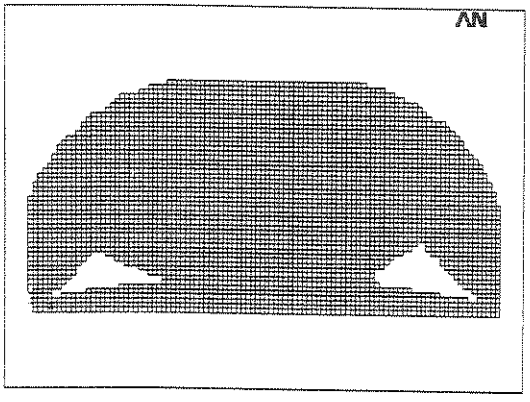
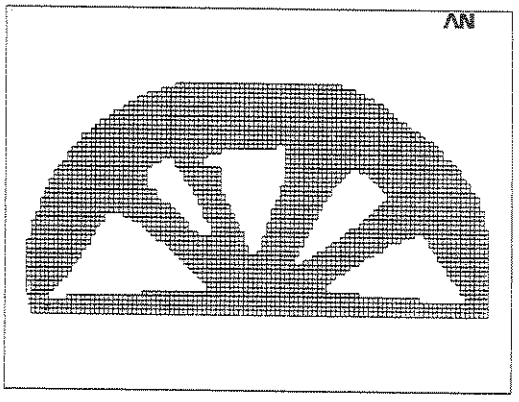


Figura 4.9: Definição do problema de Michell modificando o contorno.

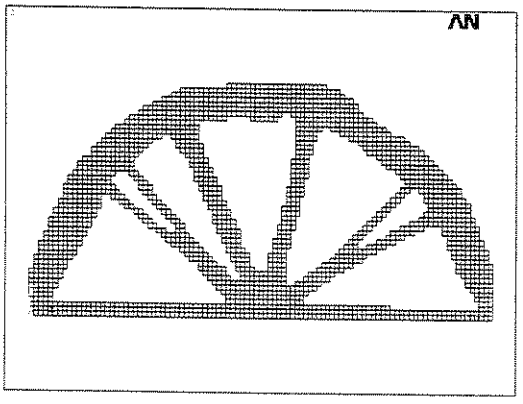
Nesse segundo caso, foi determinada uma massa final objetivo de 40% da massa inicial, a ser obtida após 25 iterações. Um número maior de iterações foi escolhido para que se tentasse minimizar as assimetrias encontradas no primeiro caso. Na Figura 4.10, tem-se os resultados de algumas iterações iniciais e da iteração final do problema. Na Figura 4.10c, tem-se a forma final de estrutura. Novamente, a estrutura final é formada por um arco reforçado por tirantes, porém com um reforço horizontal decorrente da tensão gerada pelo grau de liberdade extra de deslocamento na direção x. Dessa forma, tem-se uma estrutura ótima coerente, com reforços deslocados na direção do ponto de aplicação da carga e um reforço adicional para sustentar o esforço causado pela modifcação do contorno. Pode-se perceber que a assimetria acentuou-se nessa segunda tentativa. Os comentários sobre esse fato estão na Seção 4.6.



(a) Resultado da sétima iteração.



(b) Resultado da décima sexta iteração.



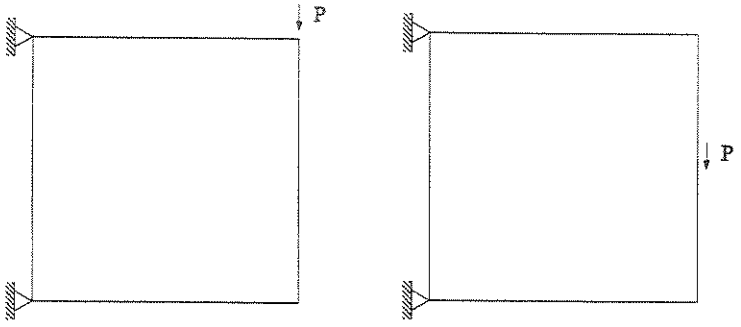
(c) Resultado da vigésima quinta iteração.

Figura 4.10: Otimização do problema de Michell modificado.

4.3 Problemas de Duas Barras e de Mão-francesa

Esse exemplo tem o objetivo de demonstrar a convergência da solução para problemas que, intuitivamente, tem-se uma idéia da forma final. São os problemas de duas barras e de mão-francesa. Nesses problemas, assim como no de Michell, a geometria e o carregamento estarão parametrizados, pois são irrelevantes para o propósito do exemplo.

Na Figura 4.11, tem-se a definição dos dois problemas propostos. Como se pode observar, as condições de contorno são idênticas, porém a carga está aplicada em pontos diferentes, originando dois resultados distintos.



(a) Definição do problema de mão francesa.

(b) Definição do problema de duas barras.

Figura 4.11: Definição de dois problemas para uma mesma malha.

Na Figura 4.12, encontra-se a ilustração da malha utilizada para resolução dos problemas propostos. Essa malha é composta por 5658 elementos finitos quadrados.

Para o problema de duas barras, foi atingida a solução intuitiva. O resultado final da otimização, que pode ser visto na Figura 4.13, é um sistema formado por duas barras que possuem um ângulo de aproximadamente 30° com a horizontal. Essa solução ótima pode ser obtida por um cálculo simples considerando a geometria do problema. Para que as forças de reação no suporte tenham sua decomposição no eixo y (eixo de aplicação da carga P) maximizadas, é necessário que o ângulo entre a horizontal e as barras sejam os maiores

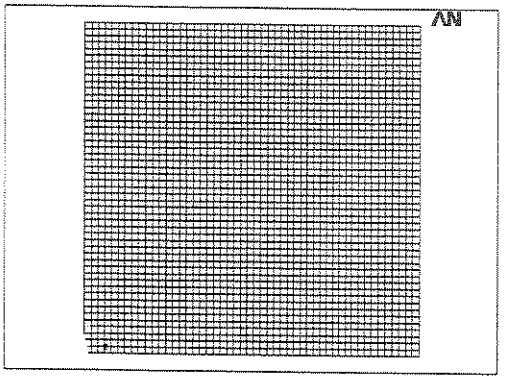


Figura 4.12: Malha utilizada nos problemas.

possíveis. Como a geometria é formada por um quadrado, considerando-se a dimensão da aresta do quadrado a, tem-se

$$\tan\ \theta = \frac{a/2}{a} = \frac{1}{2} \rightarrow \theta = 30°,$$

sendo θ o maior ângulo possível entre a barra e o eixo x. O tamanho do ângulo θ está limitado pelo ponto de aplicação da carga $\frac{a}{2}$.

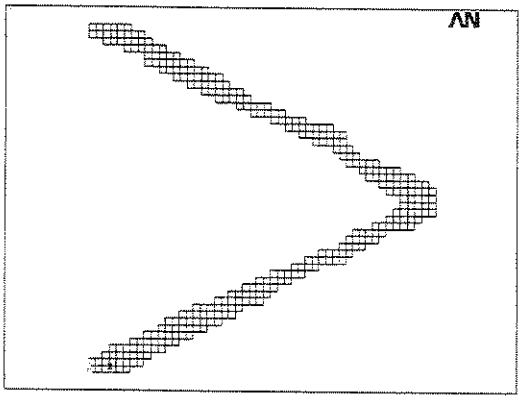


Figura 4.13: Resultado da otimização para o problema de duas barras.

Na Figura 4.14, tem-se o resultado final da otimização do problema de mão-francesa. Nesse caso, o resultado apresentado não foi o que se encontra usualmente no mercado. Porém, uma análise crítica leva a conclusão de que a forma ótima para esse tipo de componente, é a

apresentada na Figura 4.14.

A solução apresentada como resposta da otimização é formada por reforços partindo dos apoios que sustentam um reforço adicional da extremidade superior onde a carga está aplicada. Esse reforço é essencial para minimizar o momento fletor originado no ponto de fixação superior devido a carga P. Obviamente, por questões de fabricação, mãos-francesas, que geralmente são aplicadas em problemas de pequenos esforços, tem o formato de um L. Além disso, de certa forma, devido ao ponto de aplicação da carga (na extremidade da parte superior) tem-se neste exemplo, uma condição mais severa que as encontradas nas aplicações práticas.

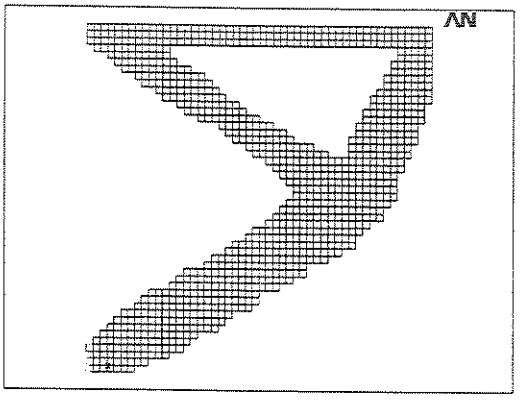


Figura 4.14: Resultado da otimização para o problema de mão-francesa.

4.4 Projeto de uma Bicicleta

Esse exemplo tem o objetivo de demonstrar a convergência da solução em uma aplicação com maior valor prático. Para tanto, parte-se de uma forma geométrica grosseira para se chegar em um projeto já consagrado, através de anos de evolução, do quadro de uma bicicleta.

Os pontos de aplicação dos esforços e suas intensidades, bem como as restrições utilizadas para se definir o problema, cuja ilustração encontra-se na Figura 4.15, são reais, retirados de (Novotny et al., 2000). Novamente, os esforços estão parametrizados porque a intenção do exemplo é estabelecer uma comparação do resultado da otimização com a forma já consagrada do projeto. Não é objetivo desse exemplo determinar o material adequado e fazer a análise de tensão do componente.

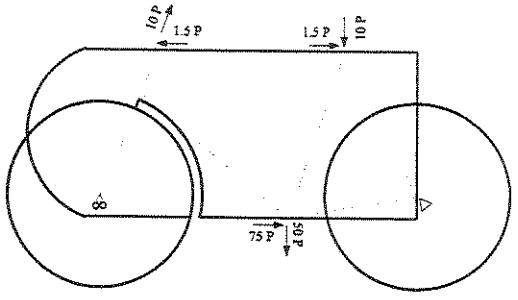


Figura 4.15: Projeto de uma bicicleta: definição do problema.

Na Figura 4.16, tem-se a forma inicial da geometria discretizada em uma malha de 2593 elementos finitos triangulares.

O resultado final, obtido em 20 iterações, está mostrado na Figura 4.17. Esse resultado assemelha-se muito ao desenho clássico da bicicleta, que pode ser visto nas linhas pontilhadas da Figura 4.15.

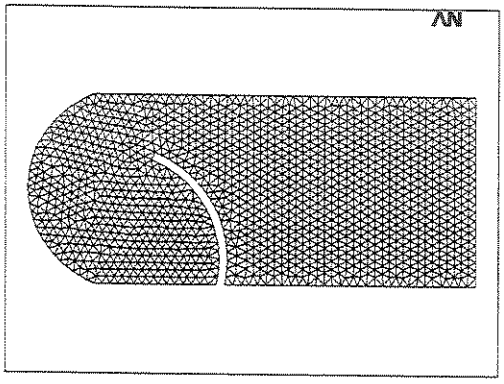


Figura 4.16: Malha utilizada para otimização.

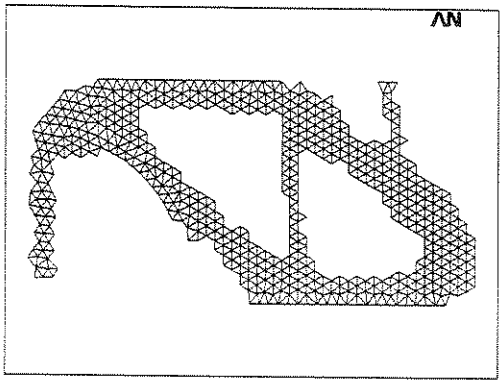


Figura 4.17: Resultado da otimização.

4.5 Problemas Tridimensionais

Esse exemplo tem o objetivo de validar os resultados de otimização topológica através da derivada topológica para problemas tridimensionais. Para tanto, dois problemas de geometria simples serão otimizados e alguns dos resultados obtidos serão comparados com aqueles em (Cea et al., 1998). A definição do problema está ilustrada na Figura 4.18. Essa geometria foi discretizada em 30000 elementos finitos hexaédricos e otimizada para dois casos de condições de contorno diferentes.

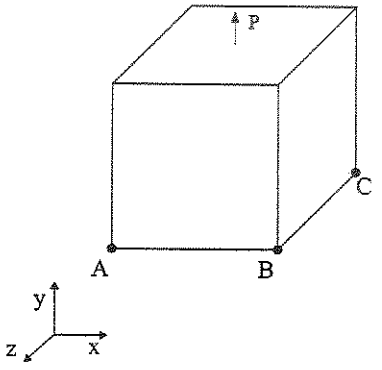
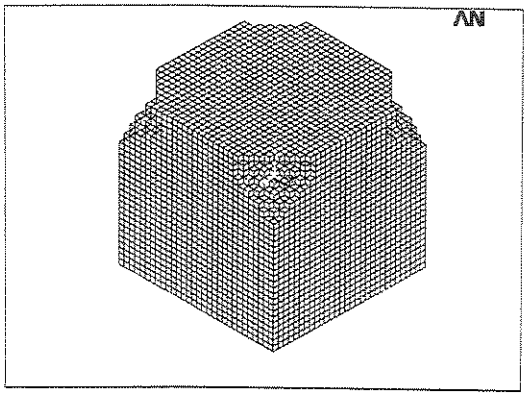


Figura 4.18: Definição do problema do cubo.

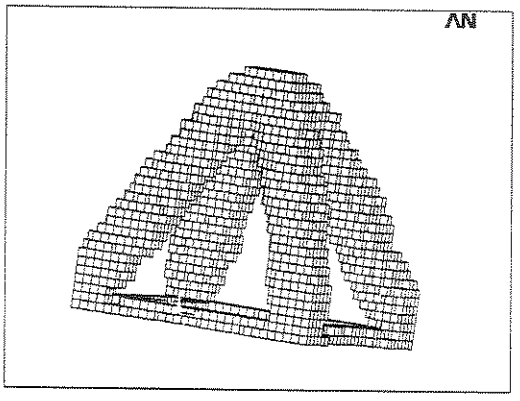
4.5.1 Primeiro Caso

Nesse caso, os vértices do cubo ilustrado na Figura 4.18, identificados pelas letras A, B, C e D, estão apenas com os graus de liberdade de deslocamento na direção de y travados. Ou seja, os deslocamentos na direção y estão impedidos para os vértices do cubo. O resultado da otimização, realizada em 30 iterações, com a massa final objetivo determinada para 5% da massa inicial, está mostrado em algumas iterações na Figura 4.19.

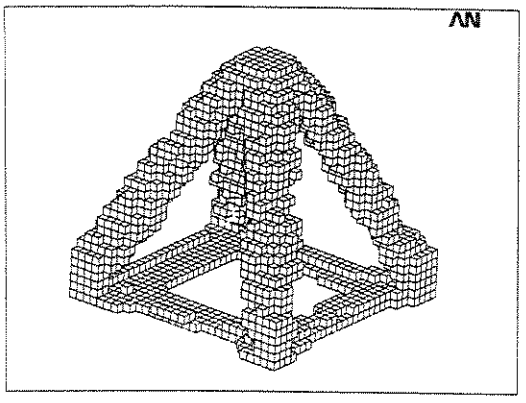
Os resultados obtidos para o mesmo problema em (Cea et al., 1998) estão ilustrados na Figura 4.20. Pode-se perceber que os problemas convergem para mesma solução. A qualidade da geometria final, apresentada na Figura 4.20b, é muito superior à apresentada



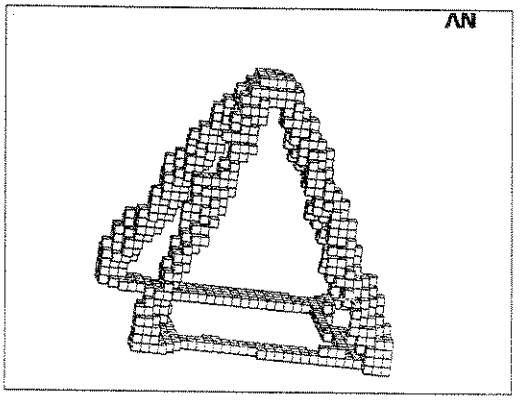
(a) Cubo na primeira iteração.



(b) Cubo na vigésima quinta iteração.



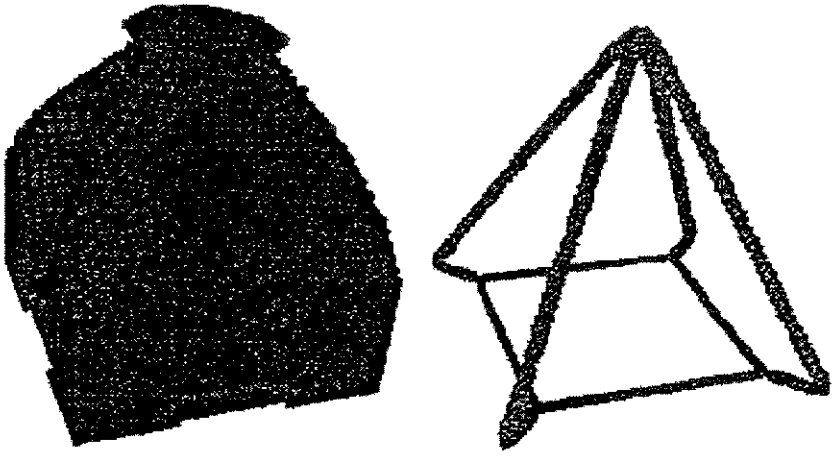
(c) Cubo na vigésima sétima iteração.



(d) Cubo na última iteração.

Figura 4.19: Otimização do cubo com a base livre.

na Figura 4.19d. Isso ocorre porque naquele, a geometria final foi determinada em 50 iterações reduzindo o volume final a 1% do volume inicial, com um processo de refinamento da malha na vigésima quinta iteração. Dessa forma, a maior qualidade da geometria pode ser atribuída a malha inicial mais fina e, principalmente, pelo refinamento dessa malha no meio do processo iterativo. O resultado da característica inicial da malha e da intervenção realizado no meio do processo iterativo foi uma geometria final de excelente qualidade. Porém, o importante para validação dos resultados é a forma final em si e não sua qualidade. Nesse sentido, pode-se perceber os resultados apresentados pela otimização topológica via derivada topológica apresenta resultados muito bons em problemas tridimensionais.



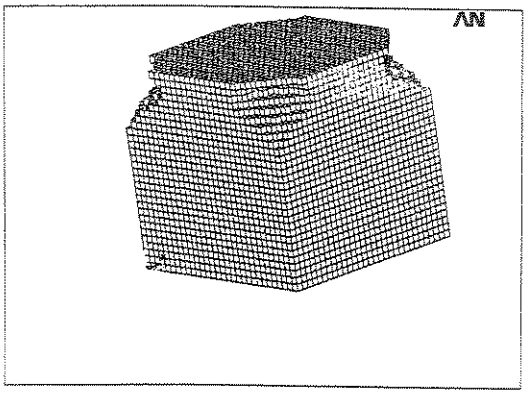
(a) Cubo na oitava iteração. (b) Cubo na última iteração.

Figura 4.20: Otimização do cubo encontrada em (Cea et al., 1998).

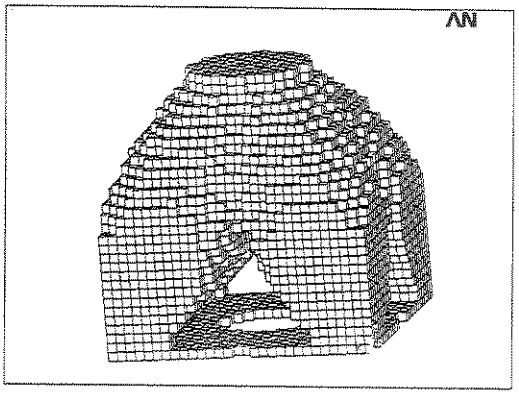
4.5.2 Segundo Caso

Nesse caso, os vértices do cubo ilustrado na Figura 4.18, identificados pelas letras A, B, C e D, estão com todos seus graus de liberdade travados, ou seja, os deslocamentos na direção de x, y e z estão impedidos. O resultado da otimização, realizada em 30 iterações, com a massa final objetivo determinada para 10% da massa inicial, está mostrado em algumas iterações

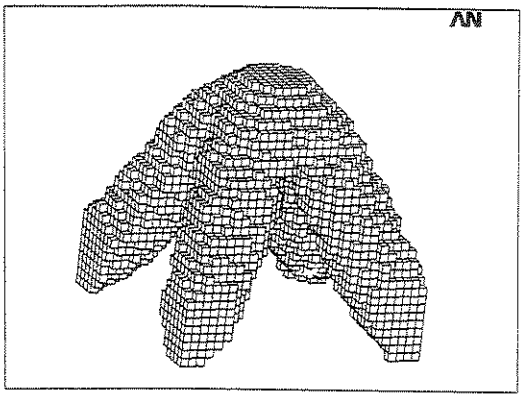
na Figura 4.21.



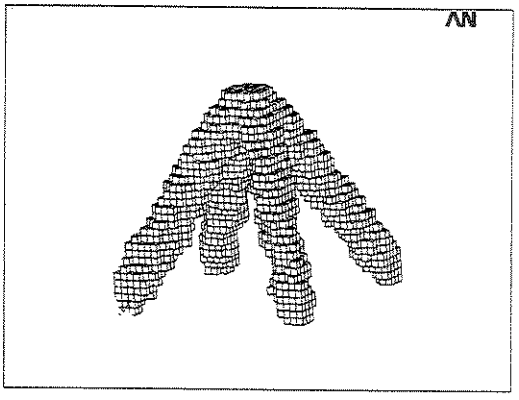
(a) Cubo na primeira iteração.



(b) Cubo na décima quinta iteração.



(c) Cubo na vigésima quinta iteração.

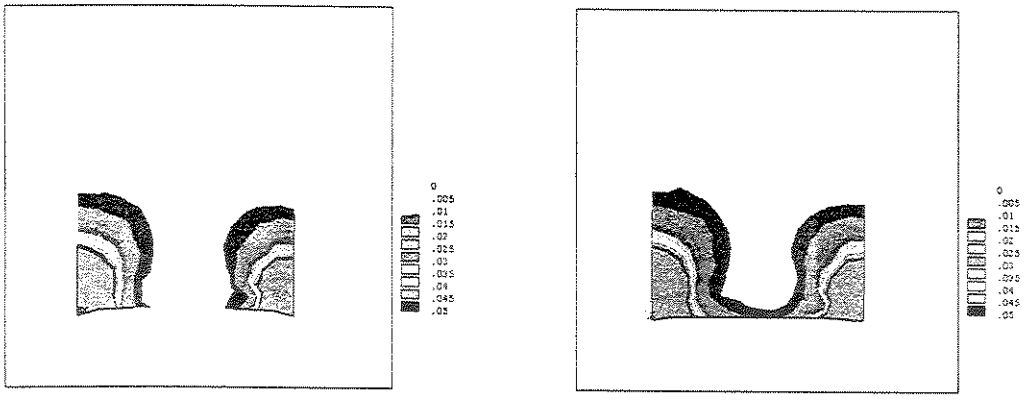


(d) Cubo na última iteração.

Figura 4.21: Otimização do cubo com a base engastada.

Para justificar as diferenças encontradas nas soluções para as diferentes condições de contorno do problema, uma análise da distribuição de tensão na geometria inicial dos dois cubos foi realizada. Esta análise está ilustrada na Figura 4.22, onde se encontra a justificativa dos reforços encontrados na base do cubo na solução do primeiro caso. Pode-se perceber, conforme a Figura 4.22a, que, quando todos os graus de liberdade estão travados, praticamente não se encontra distribuição de tensão na base do cubo. Já no caso dos graus de

liberdade de deslocamento x e z livres, percebe-se, claramente, conforme a Figura 4.22b, uma distribuição de tensão na base do cubo. Essa distribuição justifica o aparecimento dos reforços apresentados na forma final da otimização, ilustrado na Figura 4.19.



(a) Base engastada.

(b) Base livre.

Figura 4.22: Distribuição de tensão de von Mises para as diferentes condições de contorno.

4.6 Comentários sobre o Algoritmo Utilizado

Nesta seção, pretende-se destacar as vantagens e desvantagens do algoritmo utilizado, bem como criticar alguns resultados obtidos. Deve-se ressaltar que as críticas estão relacionadas ao algoritmo utilizado e não às informações obtidas da derivada topológica. Portanto, nada impede que essas informações sejam utilizadas de outra maneira levando a resultados mais satisfatórios.

Como vantagens do algoritmo, pode-se citar a simplicidade, eficiência e confiabilidade. O algoritmo utilizado é bastante simples não exigindo muito esforço computacional e levando a resultados bastante satisfatórios. Pode-se destacar também sua confiabilidade. O algoritmo convergiu em todos os testes mostrados na Seção 4, bem como em outros não apresentados.

Como uma desvantagem, pode-se citar a falta de um critério de parada por falha mecânica da estrutura. A otimização topológica deveria ser interrompida pela violação de algum critério de falha determinado pelo usuário, o que é comum nos problemas de otimização. Em geral, nos problemas de otimização existem restrições que não podem ser violadas para que o resultado de uma iteração seja aceito. Entretanto, a implementação de um critério de parada nesse algoritmo é bastante simples uma vez que o modelo de elementos finitos do problema é resolvido a cada iteração. Dessa forma, basta uma simples verificação do critério definido antes do começo da nova iteração. Outra desvantagem desse método é que nenhuma estratégia de verificação da topologia da malha é realizado durante as iterações. Nada nesse sentido foi implementado porque fugia do escopo do projeto.

Os problemas de otimização topológica do tipo "mata-forte" são muito sensíveis à qualidade da malha. Para o caso específico do algoritmo implementado, percebe-se que a partir de um grau de refinamento de malha, a convergência do problema se torna independente da malha. Porém, o problema sempre é sensível a quantidade de massa retirada por iteração. Nos problemas de Michell, puderam ser percebidas assimetrias nas formas finais dos processos de otimização. Como já comentado, essas assimetrias não são originadas de distorções da malha ou má aplicação do carregamento. Essas assimetrias têm origem na escolha da quantidade de massa a ser removida por iteração.

Na tentativa de determinar a origem das assimetrias, vários testes foram feitos como o ilustrado na Figura 4.23. Nessa tentativa, tentou-se resolver o problema de Michell modificado (ver Figura 4.9) com muitas iterações retirando um mínimo de massa por iteração. Como se pode perceber (ver Figura 4.23), a tentativa não solucionou o problema da assimetria que, pelo contrário, acentuou-se.

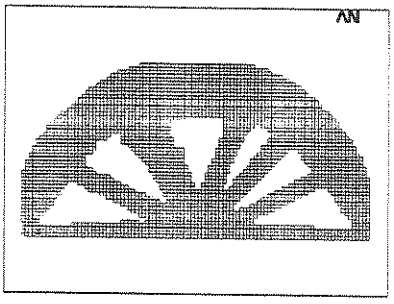


Figura 4.23: Erro na convergência da solução.

O caminho inverso, retirar muita massa em poucas iterações também foi testado, levando a resultados insatisfatórios. Esse problema pode ser atribuído a uma característica comum dos métodos do tipo "mata-forte", como já comentado na revisão bibliográfica. Como a distribuição de tensão muda sensivelmente de uma iteração para outra, alguns elementos removidos poderiam ser necessários na iteração subseqüente. Como nenhuma estratégia de reinserir elementos foi adotada, o problema pode ter sido acentuado.

A principal causa dessas assimetrias e de pequenas inconsistências na obtenção da forma final poderiam ser minimizadas por uma proposta aparentemente simples. Essa proposta seria adotar uma estratégia para minimizar a descaracterização do problema durante o processo iterativo. Essa descaracterização pode ser observada na Figura 4.23. Nesse caso, pode-se perceber que um problema relativamente diferente do original está sendo passado para uma iteração subseqüente. Relativamente diferente porque o lado direito do problema está mais rígido que o lado esquerdo, diferentemente da proposta inicial, onde (ver Figura 4.9) os dois lados possuíam a mesma rigidez.

A estratégia para inibir o aparecimento dessas descaracterizações seria eliminar, em

cada iteração, todos os elementos com valor de gradiente topológicos muito próximos. Dessa forma, a quantidade de massa removida a cada iteração não seria determinada somente pelo passo definido pelo usuário, mas também pelo estado atual dos valores da derivada topológica. Eliminando os elementos dessa maneira, poderia-se inibir o aparecimento desses problemas. Porém, para se utilizar essa estratégia, deveria ser proposto um novo algoritmo. Nesse novo algoritmo, uma estratégia para controlar a remoção de massa por iteração deveria ser repensada.

Capítulo 5

Desenvolvimento de Ferramentas de *Software* para Otimização

Neste capítulo, descrevem-se as ferramentas de engenharia de *software* utilizadas na implementação da otimização topológica e de um ambiente gráfico de geração de geometrias para análise de elementos finitos. O ambiente de geração de geometrias funcionará de maneira integrada com os outros ambientes (malha, definição do problema, análise, otimização e visualização dos resultados).

Este capítulo foi dividido em cinco seções. Na primeira seção, encontra-se a motivação para a implementação do ambiente de geometrias. Nas segunda e terceira seções, descrevem-se, respectivamente, os componentes utilizados no desenvolvimento do sistema e as metodologias de engenharia de *software* empregadas. Finalmente, nas duas últimas seções, tem-se a descrição da implementação, respectivamente, da otimização topológica e do ambiente de geometrias.

5.1 Ambientes Gráficos para Análise de Elementos Finitos

A motivação do aprofundamento nos conceitos e na implementação dos ambientes gráficos citados, o ambiente de geometria e os ambientes integrados, está na própria estruturação de uma análise de elementos finitos. Qualquer análise segue, obrigatoriamente, os seguintes passos: definição do domínio do problema, discretização do domínio, aplicação das condições de contorno e dos carregamentos, análise numérica e visualização dos resultados obtidos. Em alguns casos, existe ainda a necessidade de se definir um ambiente de otimização. Com isso, pode-se perceber que para a simulação de problemas através do MEF, a existência de um núcleo numérico eficaz e versátil é tão importante quanto à de um ambiente gráfico simples e de boa interface.

A evolução das ferramentas numéricas e algoritmos para solução de problemas tem permitido a análise de problemas cada vez mais complexos. Ao mesmo tempo, deve-se destacar a evolução das ferramentas gráficas, que tem permitido a definição de geometrias complexas e suas discretizações de maneira bastante precisa. Pode-se destacar ainda, como benefício da evolução das ferramentas gráficas, a visualização eficiente de resultados de análises. Uma análise que não permita uma visualização flexível e clara dos resultados obtidos, não somente para análise crítica do engenheiro como também para divulgação dos resultados, é de pouco valor prático.

5.2 Estudo de Componentes

Atualmente o desenvolvimento de *software* é basedo em componentes. Um componente é um conjunto de funcionalidades especializadas ou unidade de elementos de programação computacional (funções, classes, dentre outros), agrupados de maneira conveniente com um propósito definido. Esses componentes fazem parte de sistemas maiores ou produtos que tem a função de agregar e disponibilizar suas funcionalidades. O produto, que é um conjunto

de componentes ou sistemas combinados, pode ser distribuído (vendido ou disponiblizado gratuitamente) para o usuário final.

No desenvolvimento de aplicações para sistemas de engenharia CAD/CAM/CAE[1], os componentes tipicamente utilizados são:

- modeladores geométricos como o ACIS 3D *Geometric Modeler* (*Spatial Inc.*, www.spatial.com) ou o Parasolid *Kernel Modeler* (*Unigraphics Solutions*, www.ugs.com);
- sistemas gráficos para visualização, manipulação, impressão e interação com o usuário como o HOOPS/3d *Application Framework* e
- bibliotecas para o desenvolvimento de interfaces gráficas de usuários (GUI, *Graphical User Interface*) como MFC (*Microsof Foundation Classes*), Qt, Motif, dentre outras;
- outros componentes para aplicações específicas como, por exemplo, *solvers* para elementos finitos ou geradores de malha.

A seguir, tem-se uma breve descrição dos componentes que foram utilizados para o desenvolvimento do aplicativo de geração de geometrias. Os componentes que serão destacados são o modelador geométrico ACIS 3D *Geometric Modeler*, o sistema gráfico HOOPS/3d *Application Framework* e a biblioteca *Qt* para desenvolvimento de interfaces gráficas de usuários.

5.2.1 ACIS 3D *Geometric Modeler*

ACIS 3D *Geometric Modeler* (ACIS) é um conjunto de bibliotecas gráficas de componentes computacionais para modelagem geométrica tridimensional desenvolvidos pela *Spatial Corp.* (www.spatial.com). O ACIS está baseado em conceitos de orientação por objetos e na linguagem *C++*. Sua arquitetura permite o desenvolvimento de aplicativos envolvendo modelagem

[1]Respectivamente *Computer Aided Design*, *Computer Aided Maufacturing* e *Computer Aided Engineering*. Projeto, manufatura e engenharia auxiliados por computador.

geométrica e manipulação de *wireframes*[2], superfícies e sólidos (ACIS, 2001; Corney, 1997).

O ACIS consiste de um conjunto (hierarquia) de classes e funções que dão suporte à criação de curvas, superfícies e sólidos, bem como à integração de modelos geométricos definidos pelo usuário. O ACIS, que se encontra atualmente na versão 7.0, está presente em muitos aplicativos nas mais diversas áreas da computação gráfica, como jogos, filmes e programas de engenharia (CAD/CAM/CAE).

Conceitos Básicos

O ACIS considera separadamente a geometria (forma detalhada) da topologia (conectividade) dos objetos. Este conceito é conhecido como modelagem B-rep (*boundary representation*) que garante maior consistência na construção de modelos. Esta forma de modelagem permite distinguir entre pontos interiores, exteriores e sobre as superfícies (o que difere modeladores de sólidos de *wireframes*). O ACIS é naturalmente um modelador de sólidos, porém a representação em B-rep permite a coexistência consistente de sólidos e wireframes (ACIS, 2001).

O ACIS considera a geometria como sendo o conjunto de entidades físicas (pontos, curvas e superfícies) representadas pelo modelo, independentemente de suas relações de vizinhança (topologia) ou de espaço. A geometria no ACIS é subdividida em Geometria de Construção, que define a estrutura matemática para a definição das entidades e a Geometria de Modelo, que permite agregar funcionalidades às entidades criadas como *undo* e *redo*, salvamento em arquivos, tratamento de erros, gerenciamento de memória, entre outras. Normalmente, as geometrias de construção são temporárias dando lugar as de modelo que são permanentes e salvas com o modelo criado.

A topologia descreve como as entidades geométricas são relacionadas. Topologicamente uma linha reta e uma curva elíptica são consideradas iguais (arestas) bem como uma superície quadrilateral e uma hexaédrica (faces), apesar de seus conceitos geométricos serem diferentes.

O ACIS permite o salvamento e a recuperação de modelos em arquivos de formato

[2] *Wirefames* ou estruturas de arame consistem da representação de geometrias apenas através de suas arestas e vértices, ao contrário dos sólidos que contém também as faces do modelo.

próprio, podendo ser binários, SAB (*Standard ACIS Binary*, .sab), ou texto (ASCII), SAT (*Standard ACIS Text*, .sat). A convenção de escrita e leitura desses formatos de arquivo é um padrão aberto e a maioria dos aplicativos baseados em ACIS o utilizam. É possível fazer a conversão do formato ACIS para diferentes outros formatos (IGES, STEP, CATIA, PROE,VDA-FS). Essa possibilidade de se converter o formato do arquivo faz parte do escopo do módulo de geometrias, que será descrito adiante na Seção 5.5.

Arquitetura

Atualmente, o ACIS é composto por 53 componentes, destinados ao desenvolvimento de diversas funções. Os principais componentes são o *Kernel* (KERN) contendo classes para definição geometria, topologia e atributos para as entidades (salvamento e recuperação de modelos ou procedimentos (*undo* e *redo*) e classes matemáticas); o *Base* (BASE) responsável pelo gerenciamento de memória para estruturas ACIS e não-ACIS, além da definição de estruturas de dados básicas; o *Laws* (LAW) para representação e manipulação simbólica de equações e funções e o *AG Spline* para a criação e manipulação de curvas e superfícies do tipo *spline*.

Os outros componentes são responsáveis por funções como operações *booleanas* (união, subtração, intersecção), renderização, conversão de formatos de arquivos (IGES, STEP, etc), "cura" de modelos (*healing*), transformação de modelos (*offset*, rotação, *blending*), entre outras funcionalidades.

Os componentes do ACIS são compilados na forma de bibliotecas de objetos. Tais bibliotecas devem ser agregadas ao código dos programas e estão disponíveis na forma estática (.lib) ou dinâmica (.dll).

Interfaces de Programação

O acesso às funcionalidades do ACIS pode ser feito de diferentes maneiras. Através da linguagem C++, pela linguagem *Scheme* ou por uma combinação das duas.

O acesso através da linguagem C++ pode ser feito usando-se funções API (*Application*

Procedural Interface) ou funções de interface direta DI (*Direct Interface*). Esses métodos também podem ser estendidos ou redefinidos pelo usuário.

As funções API são a principal forma de acesso ao ACIS usando C++. Basicamente, são desenvolvidas de forma a permitir o acesso ao ACIS, associando tratamento de erros (argumentos e resultados), mecanismos de *undo* e *redo* e a consistência em função das diferentes versões do ACIS.

As DIs são funções ou métodos (membros de classes ou não) comuns de C++ que implementam as funcionalidades do ACIS e não permitem tratamento de erros ou as outras características oferecidas pelas funções API. A principal incoveniência do uso das DIs é o eventual problema de consistência entre as versões.

Há também um conjunto de classes e funções do ACIS derivadas do MFC denominadas ACIS MFC que permitem e facilitam a programação em plataforma Windows.

O *Scheme* é uma linguagem interpretada derivada da linguagem LISP, de domínio público, que proporciona um acesso rápido às funcionalidades do ACIS. Como é uma linguagem interpretada, não exige nenhum processo de compilação ou *link*. Pode-se utilizar o *Scheme* como ferramenta de treinamento ou teste das funcionalidades do ACIS para posterior implementação em C++.

5.2.2 HOOPS 3D *Application Framework*

O HOOPS 3D *Application Framework* (HOOPS/3dAF) é um conjunto de bibliotecas, baseadas em C++, para o desenvolvimento de aplicações gráficas 3D. É desenvolvido pela Tech Soft America (www.hoops3d.com) e encontra-se na versão 6.0 (Hoops, 2001; Merry e Leler, 1996).

O HOOPS/3dAF é definido como um *sistema gráfico* independente para o desenvolvimento de aplicações envolvendo visualização, manipulação, impressão, entre outras funcionalidades, em modelos tridimensionais. Juntamente com um modelador geométrico como o ACIS e uma biblioteca para o desenvolvimento de interfaces (MFC, Qt, Motif, etc) permite a confecção de aplicações gráficas.

O HOOPS/3dAF é constituído dos seguintes componentes:

- HOOPS/3dGS - HOOPS 3D *Graphics System.* É o núcleo do HOOPS/3dAF. Consiste de um conjunto de APIs para a criação, edição, armazenamento, manipulação, recuperação de dados (*querying*), renderização e impressão de todo tipo de informações gráficas 3D (textos, modelos geométricos, barras de cores, legendas, entre outros.);

- HOOPS *Internet Tools* - Ferramentas para compressão de dados e distribuição pela rede mundial de computadores (*Data Streaming, Collaboration* e *Web Publishing*). Através da definição de um formato de arquivo HSF (*Hoops Stream File*) permite a integração dos aplicativos desenvolvidos com base no HOOPS/3dAF com sistemas de Internet ou cliente-servidor. É acompanhado normalmente dos componentes HOOPS/ActiveX, HOOPS/Netscape e HOOPS/Net;

- HOOPS/GMB - *Geometric Modeler Bridges.* Faz a interface com os principais modeladores geométricos da atualidade, ou seja, ACIS e Parasolid. Permite uma integração rápida com os modelos gerados por esses componentes e o sistema gráfico do HOOPS/3dAF (mapeamento de entidades, conversão);

- HOOPS *Model/View/Operator* (MVO) *Class Library* . Um conjunto de classes desenvolvidas em C++ , a partir do HOOPS/3dAF, que permitem a implementação da maioria das funcionalidades básicas dos sistemas CAD/CAM/CAE. Inclui métodos para seleção, manipulação, impressão, salvamento de arquivos e visualização de informações gráficas. Possui a vantagem de ter o código-fonte aberto, o que possibilita a sua modificação/extensão;

- HOOPS/GUI - *Graphical User Interface Modules* . Assim como o ACIS MFC, deriva classes para facilitar a programação de interfaces gráficas e em diversos sistemas operacionais como, Windows (MFC), Qt, ActiveX, Netscape *plug-ins*, Java Swing e Motif. Também são de código-fonte aberto permitindo a modificação.

Conceitos Básicos

O sistema HOOPS/3dGS armazena as *cenas gráficas* como uma hierarquia de *segmentos*. Dessa forma, os *segmentos* constituem os elementos organizacionais mais básicos do HOOPS/3dGS.

De maneira geral, os segmentos representam conjuntos de geometrias, atributos e subsegmentos. A geometria representa qualquer elemento gráfico desenhado na tela, podendo ser elementos geométricos propriamemte ditos ou textos, barras de cores, imagens *bitmap*, etc. Os atributos representam as características do próprio segmento ou dos seus constituintes como forma, tamanho, cores, luzes, sombras, posições, etc.

Os tipos de geometria utilizados no HOOPS/3dGS são *Circles*, *Circular Arcs*, *Circular Wedge*, *Circular Chord*, *Cutting Planes*, *Cutting Lines*, *Ellipses*, *Elliptical Arcs*, *Grids*, *Images*, *Lines*, *Markers*, *Meshes*, *NURBS Curves*, *Polygons*, *Polylines*, *Shells* e *Text*.

O procedimento de programação através do HOOPS/3dGS é baseado primariamente na definição de segmentos, inserção de geometrias e/ou subsegmentos e definição de atributos.

5.2.3 Qt

Qt é uma biblioteca multi-plataforma para o desenvolvimento de interfaces gráficas de usuário, baseada em orientação por objetos e linguagem C++. É desenvolvida pela Trolltech AS. (www.trolltech.com) e atualmente encontra-se na versão 3.0.2 (Qt, 2001; Dalheimer, 1999).

Através da Qt é possível emular as interfaces gráficas nativas de diversos sistemas operacionais como:

- MS/Windows - 95, 98, NT, e 2000;
- Unix/X11 - Linux, Sun Solaris, HP-UX, Digital Unix, IBM AIX, SGI IRIX e vários outros e
- *Embedded* - plataformas Linux para dispositivos eletrônicos com saída gráfica como micros de mão (*palmtops*), telefones celulares, etc.

O ambiente gráfico KDE, comum nas distribuições dos sistemas operacionais *Linux*, foi originalmente desenvolvido com base em Qt.

Conceitos Básicos

Todos os objetos de interface criados a partir do Qt são denominados *widgets* e derivam de uma classe base denominada *QWidget*. Através da classe *QWidget* pode-se manipular qualquer tipo de evento gerado pelo usuário ou pelo sistema operacional.

Todo *widget* é um objeto gráfico retangular e está relacionado aos demais através de relações do tipo *pai-filho* (um *widget*-pai pode possuir vários *widgets*-filhos, etc.). *Widgets* de *alto-nível* são aqueles que estão no topo da hierarquia, não possuindo pais. Normalmente *widgets* de alto-nível são representados por janelas contendo uma moldura e uma barra de título, embora seja possível definir *widgets* com outra aparência. Em Qt, as classes *QMainWindow* e as várias subclasses *QDialog* são os tipos mais comuns de *widgets* de alto-nível.

Em Qt são usados diferentes métodos para processar os eventos gerados pelo interação entre o usuário e a interface gráfica. O primeiro define um conjunto de métodos virtuais para a classe *QWidget* capazes de manipular eventos de *baixo-nível* (eventos básicos ou padrão) como desenhar na tela (*"Paint"*), redefinir tamanhos (*"Resize"*), pressionar teclas (*"Key Press"*) e mover o *mouse* (*"Mouse Movement"*).

O segundo método é um paradigama inventado pelos desenvolvedores do Qt, denominado *"Signals and Slots"* para manipulação de eventos de *alto-nível* ou não-padrão. Nesse modelo o programador define diferentes tipos de "sinais" personalizados que podem ser emitidos pelos *widgets* quando são ativados pelos usuários, como por exemplo quando um botão de *menu* recebe um click de *mouse*. Os *"slots"* representam áreas distintas do código-fonte que implementam as ações a serem executadas quando os sinais forem emitidos.

De maneira geral, quando o usuário ativa um elemento qualquer da interface gráfica (*widget*), este gera um sinal. Em seguida, métodos padrão do Qt associam o sinal ao seu respectivo *slot* que executa a ação definida pelo programador.

5.3 Engenharia de *Software*

A necessidade de desenvolvimento de programas de computadores maiores, mais complexos e ao mesmo tempo mais robustos e confiáveis, favoreceu o surgimento de uma disciplina, na área das ciências da computação, denominada *Engenharia de Software* (Pressman, 1995).

A engenharia de software pode ser definida como um conjunto de métodos, ferramentas e procedimentos para viabilizar o desenvolvimento de programas de computadores de uma forma mais controlada e racional, bem como a sua qualificação e manutenção.

Os métodos envolvem um amplo conjunto de tarefas que incluem: planejamento e estimativa de projeto, análise de requisitos de software e de sistemas, projeto de estrutura de dados, especificação e codificação de programas, teste e manutenção.

As ferramentas, denominadas CASE (*Computer Aided Software Engineering*), ou engenharia de *software* auxiliada por computador, são programas (ou conjunto de programas) específicos, com o objetivo de automatizar as atividades associadas à engenharia de *software*.

Os procedimentos constituem o elo entre os métodos e as ferramentas para desenvolvimento do software.

Dentre as várias metodologias disponíveis para a aplicação dos conceitos da engenharia de software, pode-se citar o RUP, *Rational Unified Process*, que é baseado na orientação por objetos, utilizando uma linguagem de modelagem denominada UML ou *Unified Modeling Language* (Linguagem de Modelagem Unificada).

Existem várias ferramentas disponíveis que dão suporte à aplicação do RUP, da fase de concepção à fase de testes, documentação e distribuição, que são desenvolvidas pela empresa *Rational* (www.rational.com).

A seguir, tem-se um resumo dos conceitos da linguagem de modelagem unificada UML e do processo unificado de desenvolvimento RUP.

5.3.1 Linguagem de Modelagem Unificada (UML)

A *Unified Modeling Language* (UML) (Booch et al., 1999) é uma linguagem de modelagem para sistemas complexos que permite a visualização, especificação, construção e documentação do funcionamento do sistema durante a sua concepção e execução.

Inicialmente, a UML foi criada a partir das necessidades associadas ao processo de desenvolvimento de *software*. Entretanto, seus conceitos atingiram um caráter amplo e geral, o que permite a sua aplicação nos mais diversos sistemas, observando-se naturalmente as suas peculiariadades.

A UML trata da definição de modelos que representam o sistema, suas características principais e as suas relações com o meio ao qual estão inseridos. Os modelos representam simplificações da realidade e o seu nível de detalhamento é proporcional à complexidade do sistema. A UML é uma linguagem baseada nos conceitos da orientação por objetos.

Basicamente, a UML se utiliza da definição de abstrações ou conceitos de entidades responsáveis pela realização das "tarefas" ou procedimentos inerentes ao funcionamento do sistema que está sendo modelado. Define também as relações entre essas entidades e os diagramas que agrupam e representam estas relações.

As entidades principais da UML são as *entidades estruturais*, representando as partes estáticas do sistema (partes físicas ou conceituais); as *entidades comportamentais*, que representam as partes dinâmicas do sistema (comportamento ao longo do tempo); *entidades de agrupamento,* definindo a organização do sistema e o seu agrupamento em módulos com características comuns e *entidades de anotação*, são itens explanatórios (notas) que visam facilitar o melhor entendimento do sistema.

No que diz respeito às relações, são definidas as de *dependência*, *associação*, *generalização* e *realização*. As relações de dependência indicam que uma entidade necessita de uma ou mais outras para que suas atividades sejam realizadas. Já as relações de associação representam as diversas formas de conexão entre as entidades. As relações de generalização indicam a especialização de uma abstração em outras abstrações-filhas, representando uma hierarquia de entidades. Por último, as relações de realização representam os "contratos de

trabalho" entre entidades de subsistemas diferentes de forma a permitir a sua execução.

Os tipos de diagramas definidos pela UML são os de *classe*, *objetos*, *use-cases*, *sequência*, *colaboração*, *estado* (*statechart*), *atividade*, *componentes*, e *distribuição* (*deployment*). De uma forma geral, os diagramas representam as relações entre as entidades do sistema, de forma ilustrar o funcionamento em diversos ângulos. Basicamente os diagramas da UML podem ser descritos como se segue:

- diagramas de classe: ilustram o relacionamento entre as diversas classes do sistema;
- diagramas de objetos: representam o relacionamento entre os objetos (instâncias de classes) que devem ser criados para o funcionamento do sistema;
- diagramas de *use-cases*: *use-cases* são fluxos de enventos do sistema que fornecem algum resultado de valor para os usuários. Os diagramas de *use-cases* ilustram os diversos fluxos de evento do sistema;
- diagramas de sequência e colaboração: são duas variações de diagramas que ilustram a interação entre os objetos do sistema durante a execução dos fluxos de eventos. Os diagramas de sequência enfatizam a ordem temporal da troca de mensagem entre os objetos. Já os de colaboração enfatizam a organização estrutural entre os objetos que enviam e recebem mensagens;
- diagramas de estado: representam a evolução ou transição entre os estados do sistema. São importantes na compreensão do comportamento do sistema do ponto de vista das interfaces entre os diversos subsistemas;
- diagrama de atividade: é um tipo especial de diagrama de estado (fluxograma) que ilustra o comportamento do sistema do ponto de vista das atividades, na sua sequência de realização;
- diagramas de componentes: mostram a organização e dependência entre os conjuntos de componentes do sistema. Estes diagramas devem estar relacionados aos diagramas de classes, interfaces e colaborações;

- diagramas de distribuição: representam a configuração organizacional dos grupos de componentes (nós) em tempo de execução. Os diagramas de distribuição ilustram a arquitetura do sistema do seu ponto de vista estático.

5.3.2 Processo Unificado de Desenvolvimento (RUP)

O *Rational Unified Process* (RUP) (Jacobson et al., 1999) é um processo de desenvolvimento de software baseado em três características fundamentais.

A primeira delas é que se trata de um processo *iterativo*. Isso significa que, de acordo com a complexidade do sistema, o mesmo deve ser desenvolvido aos poucos, ou seja, à medida que suas principais funcionalidades básicas são implementadas com sucesso, outras novas funcionalidades mais complexas são inseridas, implementadas e testadas, gerando um processo de evolução cíclico.

A segunda característica é que o RUP se trata de um processo centrado na arquitetura do sistema. Parte-se do princípio que, desde que a arquitetura do sistema seja robusta o suficiente, é possível o desenvolvimento em etapas paralelas, minimizando o retrabalho, aumentado a reutilização de componentes e, consequentemente, a sua manutenção.

A terceira e última característica básica é que o RUP é um processo dirigido por *use-cases*. *Use-cases* são estudos de caso que visam capturar os fluxos de eventos principais definidos nos requerimentos do sistema e que levam a algum resultado de valor para os usuários. Um exemplo de *use-case* pode ser o conjunto de eventos associados ao envio de mensagens através de um sistema de correio eletrônico. Um sistema complexo pode ser constituído por vários *use-cases* e os mesmos podem ser "quebrados" em *use-cases* secundários, bem com se inter-relacionar e definir fluxos alternativos de eventos.

O RUP é baseado na UML e suporta o desenvolvimento utilizando técnicas de orientação por objetos. De maneira geral, uma primeira iteração do processo de desenvolvimento engloba as seguintes etapas

- Análise de Requerimentos: nessa fase são definidos os requisitos principais que o sistema

deve oferecer a todos os níveis de usuários. São analisadas as características básicas gerais para garantir o funcionamento do sistema, bem como critérios de qualidade, facilidade de uso, formas de acesso, tipos de interface, etc.;

- Definição dos *use-cases*: nessa segunda etapa, com base nos requerimentos que foram definidos, são extraídos os fluxos de eventos principais e os mesmos devem ser descritos, de acordo com a sequência de acontecimentos. É importante definir nessa fase também, os *use-cases* principais, ou essenciais, que serão detalhados e implementados na primeira iteração;

- Análise e Projeto de Classes do Sistema: aqui são definidas as entidades (classes, dados membro e métodos) principais para a realização das tarefas definidas pelos *use-cases* principais que serão implementados na primeira iteração. Devem ser analisados também, as formas de acesso (interfaces) entre os usuários e o sistema;

- Implementação: a implementação consiste da geração de códigos de programa propriamente dita. A partir das classes que foram definidas na etapa anterior, são gerados os códigos de programação na linguagem que foi adotada e os procedimentos são implementados;

- Testes e Distribuição: após todo o código ser implementado o sistema deve ser testado, com base nos requerimentos e *use-cases* que foram definidos e, após a sua aprovação, é disponibilizado para uso ou avaliação pelos usuários. Após essa fase, todo o processo se reinicia, a partir de novos requerimentos e *use-cases* que não foram implementados na iteração atual, enquanto durar o processo de desenvolvimento e evolução do sistema.

5.4 Derivada Topológica

O funcionamento e a estrutura principal de dados do algoritmo da derivada topológica foram descritos na Seção 3.4. Nesta seção, serão descritos alguns detalhes de implementação sem a preocupação com seu funcionamento. A derivada topológica foi implementada, utilizando-se

o processo e as ferramentas descritas neste capítulo. Agora, a partir das ferramentas de visualização gráfica utilizadas no desenvolvimento do projeto, pode-se mostrar as interações entre as partes do núcleo numérico de elementos finitos e a aplicação de otimização desenvolvida. A partir destas relações, será possível mostrar os passos envolvidos na construção de um aplicativo para otimização topológica.

5.4.1 Estrutura Geral das Classes de Elementos Finitos e Otimização

Na Figura 5.1, encontra-se uma ilustração da atual organização da base de programas empregada nesse trabalho. A partir dessa organização, será descrito onde e como a otimização topológica foi implementada.

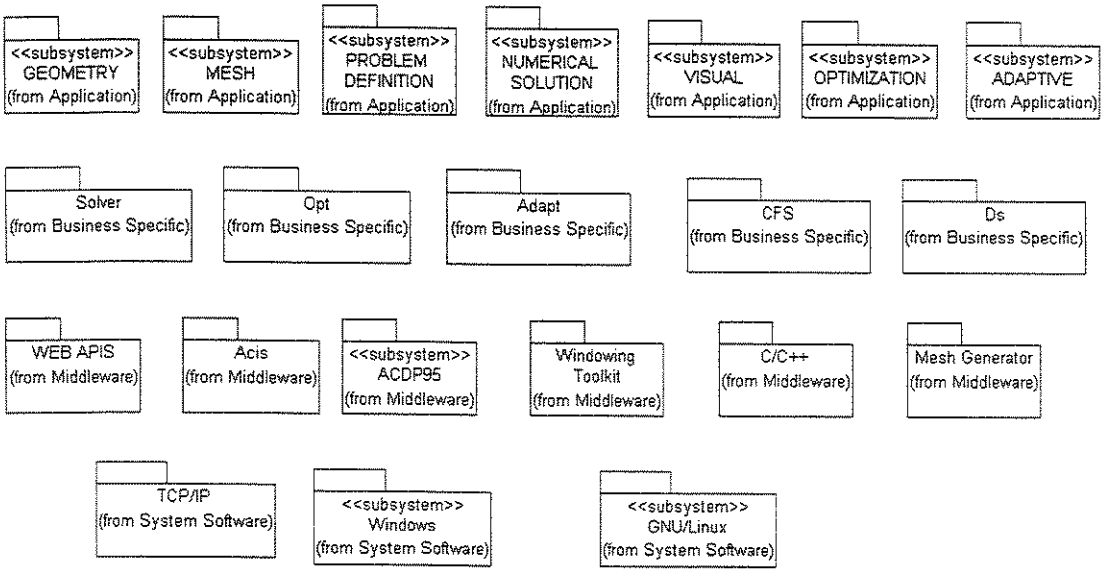


Figura 5.1: Estruturação das classes existentes para análises de elementos finitos e otimização.

Em uma distribuição de camadas, como aquela encontrada na Figura 5.1, as dependências sempre ocorrem das camadas superiores para as inferiores ou entre componentes (ou pacotes) de uma mesma camada. Dependências de camadas inferiores de superiores não

são permitidas. Dessa forma, a primeira camada, encontrada na base da Figura 5.1, referencia o sistema operacional utilizado ou o protocolo de comunicação *TCP/IP*, que é importante quando se tratam problemas cliente-servidor. A segunda camada, subindo na Figura 5.1, referencia os componentes que viabilizaram a construção dos programas. Componentes como o ACIS, já descrito na Seção 5.2.1, ou a própria linguagem *C/C++*.

Na terceira e quarta camadas, encontram-se os programas em desenvolvimento. Importantes programas como os núcleos numéricos para resolução de problemas de elementos finitos, dentro do pacote *Solver*, e para realização de otimização, dentro do pacote *Opt*, são encontrados na terceira camada. Esses dois pacotes são os de maior importância para implementação da derivada topológica. Pode-se perceber, na camada mais alta, os ambientes já citados no início desse capítulo, como os de geração de geometrias e malhas. As camadas e os pacotes encontrados na Figura 5.1 formam uma estrutura suficiente e eficaz para modelar problemas de elementos finitos.

5.4.2 Estrutura da Classe de Derivada Topológica

A classe de derivada topológica foi criada como uma classe de otimização pertencente ao pacote *Opt*, encontrado na Figura 5.1. Alterações no pacote *Solver* foram necessárias para adaptá-lo às novas situações impostas pela otimização topológica. O pacote não estava preparado para, por exemplo, suportar a remoção de elementos finitos de um modelo em uma análise.

A forma final da classe de derivada topológica é encontrada na Figura 5.2. Os principais atributos da classe são: *Model* que contém o modelo de elementos finitos, *NumSol* que contém o núcleo numérico capaz de manipular o modelo de elementos finitos e devolver sua solução e *DBManager* que contém um banco de dados com as informações da solução, geometria, malha e casos de carregamento. Esse último atributo é o objeto responsável por organizar as várias versões e soluções de um mesmo problema ao longo da otimização.

A classe encontrada na Figura 5.2 é um bom exemplo para se ressaltar as vantagens das ferramentas de engenharia de software utilizadas no desenvolvimento dos programas. Os

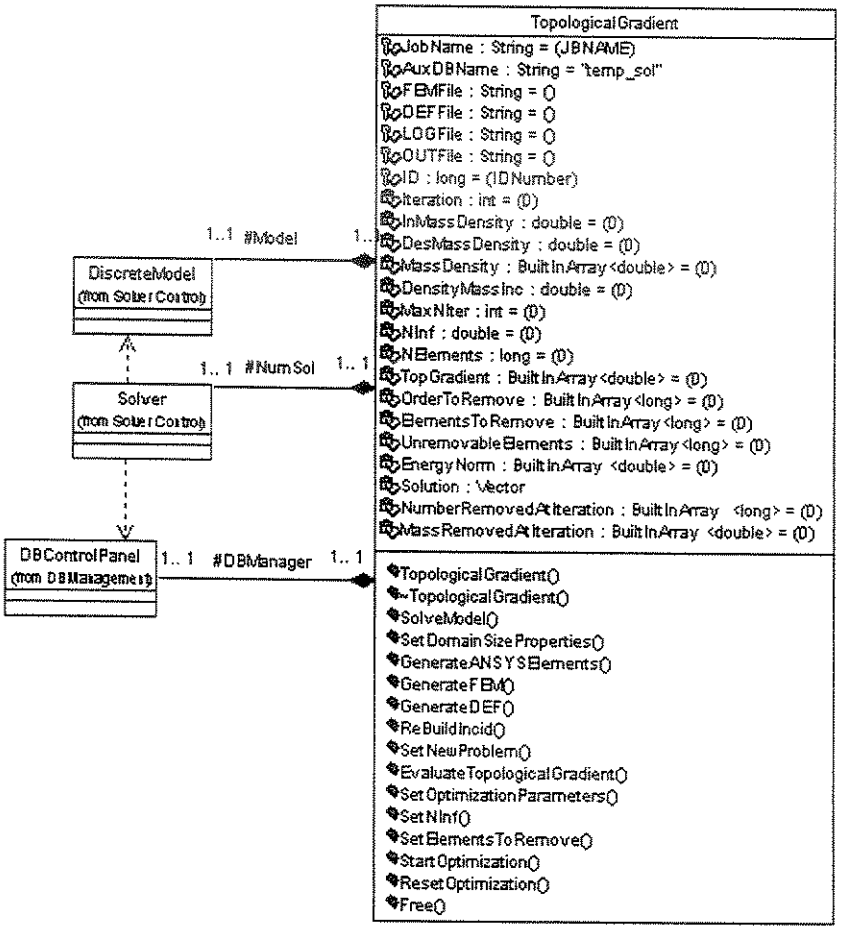


Figura 5.2: Classe de otimização topológica.

diagramas de classe e de seqüência, ver Seção 5.3.1, do pacote *Solver* permitiram uma rápida identificação e implementação das mudanças necessárias para o funcionamento da otimização topológica. Sem o auxílio destes diagramas, muitas dificuldades seriam encontradas na implementação das modificações. Isso ocorre, devido à dificuldade, natural em um programa extenso, de interagir com a grande quantidade de relações de estruturas complexas, como é o caso das estruturas das classes do pacote *Solver*. A maioria das modificações puderam ser propostas sem o estudo de linhas de código fonte das classes afetadas. Para desenvolvimento em grupo de *softwares* complexos, a adoção de um processo baseado em *UML* é essencial para se atingir o sucesso.

5.5 Módulo de Geometria

5.5.1 Descrição

O módulo de geometria é um ambiente computacional com interface gráfica que permite ao usuário criar e manipular domínios bidimensionais e tridimensionais. As ferramentas do módulo de geometria podem ser divididas em quatro grupos: importação e exportação de arquivos CAD, correção de geometrias importadas e criação e alteração de domínios.

A importação e exportação de domínios funcionam de maneira semelhante, transformando arquivos dos formatos mais comumente encontrados (*IGES, STEP, CATIA, PRO/E, VDAFS*) para o formato nativo da biblioteca *ACIS* e vice-versa. O referido formato nativo é o *SAT*, como já mencionado na Seção 5.2.1, encontrado em alguns softwares *CAD/CAM/CAE* como, por exemplo, no Autocad.

A correção de geometrias importadas é uma ferramenta muito importante na simulação computacional, pois permite corrigir imperfeições geométicas que inviabilizam a geração de malhas. A principal aplicação da correção de geometrias é como ferramenta de pós-processamento para importações de arquivos CAD. Isso ocorre porque, dificilmente, geometrias complicadas são importadas sem erros. A cura, ou seja, a correção da geometria, pode ser executada de maneira automática (correção automática dos erros encontrados) ou pode

fornecer a visualização dos erros para que possam ser corrigidos manualmente pelo usuário. Mesmo na correção automática o usuário pode interferir nos resultados. Para tanto, o usuário pode definir os valores das diferentes tolerâncias nas quais se baseia o processo de correção.

Na edição de domínios estão todas as ferramentas mais importantes de visualização, criação e modificação de desenhos. Este grupo foi cuidadosamente projetado para que o significativo número de ferramentas não torne seu uso complicado. Para tanto, foram selecionados os recursos mais essenciais de um bom modelador de geometrias através de avaliações de produtos já existentes.

5.5.2 Use Cases

Os *use cases* deste módulo foram divididos de acordo com as funcionalidades já descritas anteriormente. Cada uma dessas funcionalidades principais possui um *use case* próprio que gera outros de detalhamento. O diagrama de *use case* principal do módulo é encontrado na Figura 5.3.

Pela grande quantidade de ferramentas presente na edição de domínios, esse grupo foi reparticionado como pode ser visto na Figura 5.4. O reparticionamento foi feito gerando três grupos menores: manipulação de domínios bidimensionais, manipulação de domínios tridimensionais e ferramentas de modificação. Os grupos de manipulação de domínios bidimensionais e tridimensionais são, basicamente, descrições das entidades planas e sólidas que podem ser tratadas pelo programa. O grupo de ferramentas de modificação trata as principais funcionalidades exigidas para se conseguir manipular um domínio de maneira eficiente. Pode-se destacar ferramentas como as que permitem criação de raios de concordância, reflexões em eixos de simetria, livre movimentação de partes ao longo do plano de trabalho ou criação de chanfros.

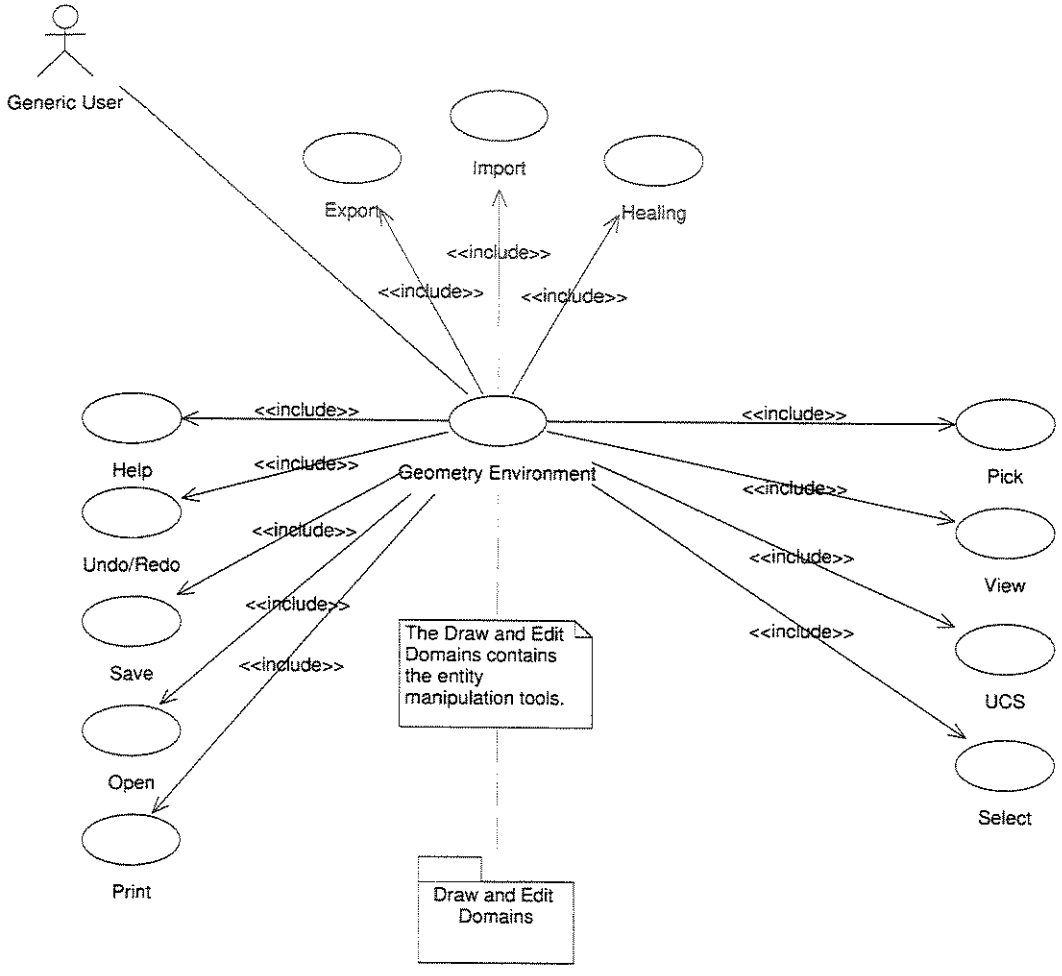


Figura 5.3: Diagrama principal de *Use Case*.

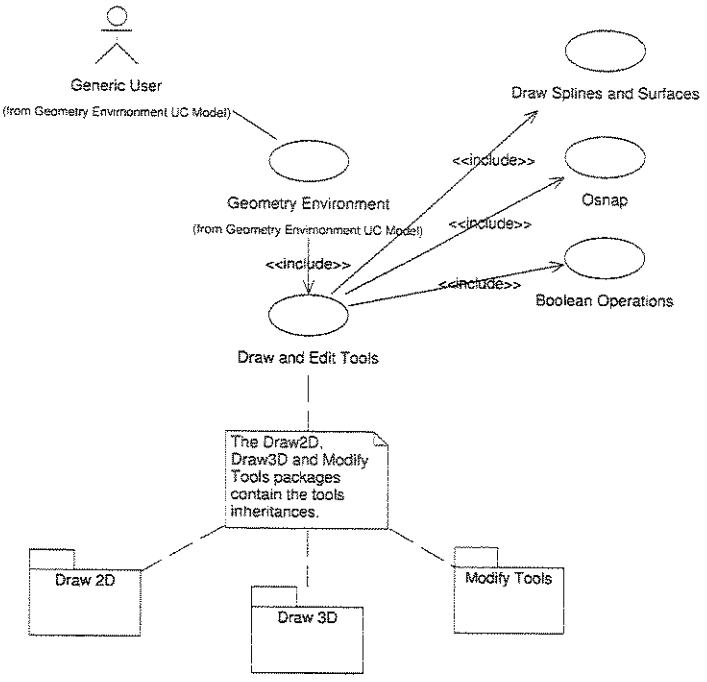


Figura 5.4: Diagrama de detalhamento das ferramentas de edição.

5.5.3 Implementação

O projeto das classes, orientado pelos *use cases,* resultou na abstração de quatro tipos diferentes de geometria: geometria importada, geometria exportada, geometria de correção e geometria a ser criada. Cada uma dessas geometrias é uma classe derivada da classe principal Geometry. A abstração pode ser vista na Figura 5.5.

Um resultado obtido da implementação pode ser visto no exemplo mostrado na Figura 5.6a. Neste exemplo, um domínio originalmente construído no programa *Solid Edge* foi importado para o formato *SAT*. Pode-se perceber que mesmo após o processo de cura, a geometria importada ainda contém defeitos. Essa importação resultou em um sólido com um buraco em sua geometria (ver Figura 5.6b).

Essa mesma geometria da Figura 5.6, agora importada de um arquivo no formato *IGES*, não apresentou nenhum defeito após o processo de cura. Pode-se perceber na Figura 5.7 que a geometria importada estava completamente danificada e na Figura 5.8 como ficou após a

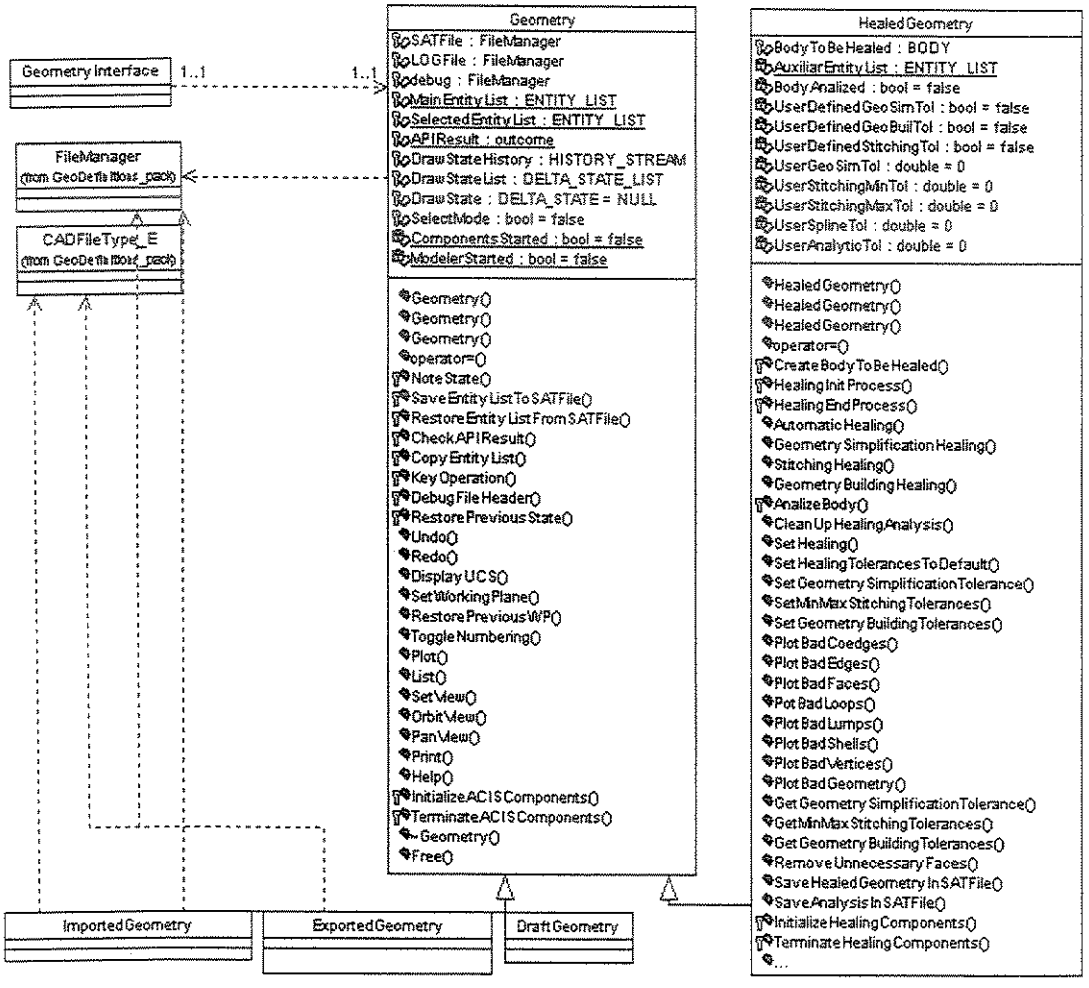
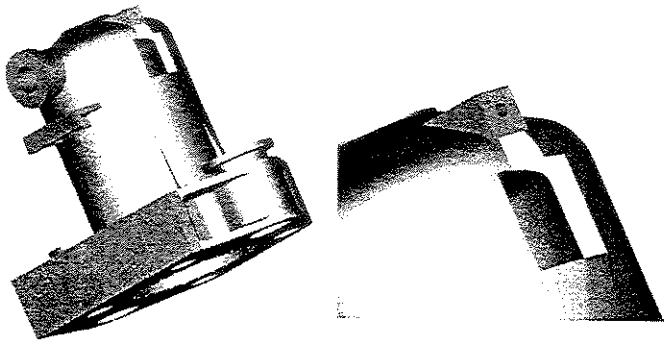


Figura 5.5: Diagrama principal de classes.



(a) Resultado (b) Detalhe do defeito

Figura 5.6: Geometria importada pelo programa.

cura. Essa geometria da Figura 5.8 foi exportada novamente para o formato *IGES* e pode ser aberta pelo programa *ANSYS* sem nenhuma restrição.

5.5.4 Etapas Futuras

Atualmente, o desenvolvimento do ambiente de manipulação de geometrias encontra-se em fase de implementação. A etapa inicial já cumprida, pode ser considerada a mais demorada, pois envolveu todo o processo de abstração de classes e suas funcionalidades. As classes de importação, exportação e cura de geometrias estão implementadas. Algumas correções e expansões ainda devem ser feitas, porém já cumprem suas propostas iniciais. A classe de construção de novas geometrias está em estágio inicial de implementação. Isso ocorre pela exigência de manipulação de grande quantidade de objetos gráficos. Os procedimentos para geração de geometrias a partir de pontos definidos pelo usuário já estão implementados, porém uma maior interatividididade, como por exemplo permitir ao usuário definir essas mesmas geometrias a partir de eventos do mouse, ainda está em fase de implementação.

Os próximos passos do desenvolvimento consiste em finalizar a implementação das classes e realizar mais exaustivamente testes no sistema. Fazer a integração com os outros módulos e disponibilizá-los para os usuários é o objetivo maior do projeto.

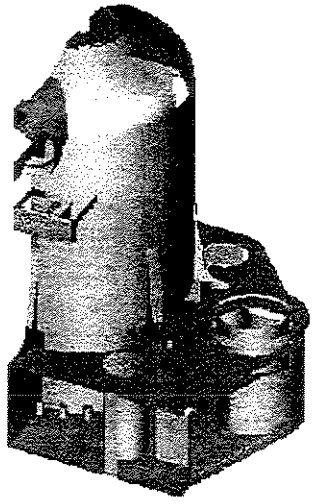


Figura 5.7: Geometria importada do arquivo IGES.

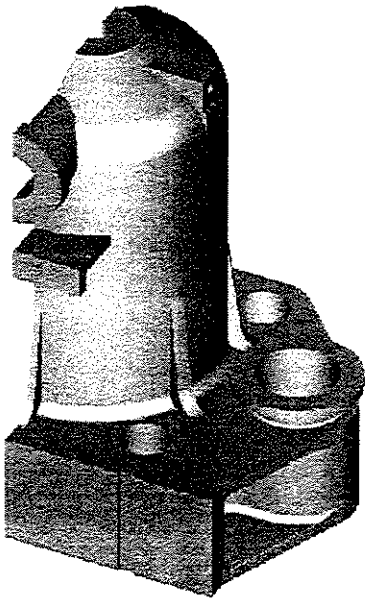


Figura 5.8: Geometria importada após processo de correção.

Capítulo 6

Conclusão

Nesse trabalho, foi mostrada a utilização do conceito de derivada topológica para realização de otimização topológica. Os resultados apresentados no Capítulo 4, mostram que, mesmo com a utilização de um algoritmo básico, os resultados obtidos são satisfatórios. É importante ressaltar que a estratégia de remoção de elementos, conforme descrito na Seção 3.4, foi apenas uma idéia para se utilizar as informações obtidas da derivada topológica. A derivada topológica, no entanto, é muito mais que uma ferramenta de otimização topológica, pois fornece a sensibilidade do problema à criação de um furo e não somente uma seqüência de elementos a serem removidos. Dessa forma, como já mencionado nos objetivos da tese no Capítulo 1, apesar da derivada topológica ter sido usada como base para criação de um método pertencente ao grupo *SERA*, seu conceito é bem mais amplo.

Diferentemente da maioria dos métodos pertencentes ao grupo *SERA*, a derivada topológica não está baseada na saturação de algum critério de tensão ou energia na estrutura em análise. Pode-se concluir que a derivada topológica é um conceito novo que deve ter, nos próximos anos, muitas extensões e novas aplicações. Os trabalhos realizados no *LNCC* (Laboratório Nacional de Computação Científica) muito referenciados nessa dissertação, (Novotny et al., 2000) e (Novotny et al., 2001a), abriram caminho, através da prova matemática da validade das ferramentas utilizadas, para esse novo conceito muito valioso a ser utilizado em problemas de otimização.

O conceito de derivada topológica pode ser aplicado em muitos outros campos, como por exemplo, na mecânica da fratura, corrosão, projeto de microestrutras, entre outros. Deve-se citar também a possibilidade da aplicação desse conceito em problemas de outras áreas da engenharia como, por exemplo, no eletromagnetismo.

O desenvolvimento dessa dissertação envolveu o aprendizado de uma grande quantidade de novos conceitos. Dentre esses novos conceitos pode-se citar o método dos elementos finitos, a análise de sensibilidade, a linguagem de programação *C++* e as ferramentas de engenharia de *software*. Mais especificamente na implementação dos programas, pode-se estabelecer contato com um grande aprendizado: o desenvolvimento em grupo e a expansão e manutenção de códigos, muito importantes para se atingir bons resultados. Embora nem todos esses conceitos tenham sido abordados com a profundidade necessária para uma total compreensão de seu alcance, formou-se uma base sólida. Uma contribuição à formação, foi a necessidade de inter-relacionar a derivada topológica à análise de sensibilidade à mudança de forma. Inter-relacionar conceitos, é uma capacidade cada vez mais exigida para obtenção de resultados significativos.

Como seguimento a esse trabalho, tem-se inúmeras sugestões. Um trabalho muito interessante seria a dedicação à criação de um algoritmo que melhor aproveitasse as informações contidas na derivada topológica. Trabalhos mais sofisticados poderiam ser feitos no sentido de integrar a otimização topológica com outras otimizações como, por exemplo, a otimização de forma, para que geometrias mais regulares, mais adaptadas à fabricação, pudessem ser obtidas. Finalmente, talvez a maior oportunidade que possa ser deixada como sugestão de continuidade desse trabalho seja buscar novas aplicações para a derivada topológica.

Bibliografia

ACIS. Spatial Corp. Online help for acis, version 6.3, 2001.

Ansys. Ansys Inc. Ansys release 5.1 – procedures manual, 1995.

Bathe, K. *Finite Element Procedures.* Prentice Hall, Englewood Cliffs, 1996.

Booch, G. *Object Oriented Design with Applications.* The Benjamin/Cummings Publishing Company, Inc., USA, 1991.

Booch, G., Rumbaugh, J. , Jacobson, I. *The Unified Modeling Language: User Guide.* Addison Wesley, Reading (MA), 1999.

Cea, J., Gareau, S., Guillaume, P. , Masmoudi, M. The shape and topological optimization connection. Relatório interno, UFR MIG, França, 1998.

Cheng, K., Olhoff, N. An investigation concerning optimal design of solid elastic plates. *International Journal Solids Structural*, v. 17p.305–323, 1981.

Choi, K., Twu, S. Equivalence of continuum and discrete methods of shape design sensitivity analysis. *AIAA Journal*, v. 27, n.10, p.1419–1424, 1989.

Cook, R., Malkus, D. , Plesha, M. *Concepts and Applications of Finite Element Analysis.* John Wiley & Sons, USA, third ed., 1991.

Corney, J. *3D Modeling with ACIS Kernel and Toolkit.* John Wiley and Sons, New York, 1997.

Dalheimer, M. *Programming with Qt.* OReilly & Associates, New York, 1999.

Eschenauer, H., Kobelev, V. , Schumacher, A. Bubble method for topology and shape optimization of structures. *Structural Optimization*, v. 8p.42–51, 1994.

Eshelby, J. The elastic energy-momentum tensor. *Journal of Elasticity*, v. 5p.321–335, 1975.

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.* COPPE/UFRJ, 1993., Tese (Doutorado).

Fancello, E., Guimarães, A., Feijóo, R. , Venere, M. Geração automática de malhas 2d em programação orientada a objetos. In: *Anais do XI Congresso Brasileiro de Engenharia Mecânica*, p. 635–638, São Paulo. ABCM, 1991.

Graham, N. *Learning C Plus Plus*. McGraw-Hill, Inc., USA, 1991.

Gurtin, M. *An Introduction to Continuum Mechanics*, v. 158 de *Mathematics in Science and Engineering*. Academic Press, 1981.

Haug, E., Choi, K. , Komkov, V. *Design sensitivity analysis of structural systems*, v. 177 de *Mathematics in Science and Engineering*. Academic Press, Orlando, 1986.

Hemp, W. *Optimum structures*. Clarendon Press, Oxford, 1973.

Hoops. Tech Soft America. Hoops/3daf online documentation, version 6.0, 2001.

Jacobson, I., Booch, G. , Rumbaugh, J. *The Unified Software Development Process*. Addison Wesley, Reading (MA), 1999.

Merry, J., Leler, W. *3D with Hoops*, 1996.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Análise de sensibilidade na otimização topológica. Relatório interno 33/2000, LNCC/MCT, Petrópolis-RJ, 2000.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Cálculo da derivada topológica via análise de sensibilidade à mudança de forma aplicada a otimização topológica no problema de poisson. Relatório interno 40/2000, LNCC/MCT, Petrópolis-RJ, 2001a.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Otimização topológica em elasticidade plana. *Anais do XII ENIEF, Bariloche-Argentina*, 2001b.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Derivada topológica via análise de sensibilidade à mudança de forma na otimização topológica. *A ser publicado na Revista Internacional sobre Métodos Numéricos para Cálculo y Diseño en Ingeniería*, 2002a.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. The topological derivative for the poisson's problem. *Submetido à Mathematical Models and Methods in Applied Sciences*, 2002b.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Topological optimization via shape sensitivity analysis applied to 2d elasticity. *A ser publicado no V-WCCM, Viena-Áustria*, 2002c.

Novotny, A., Feijóo, R., Padra, C. , Taroco, E. Topological sensitivity analysis. *Aceito na Computer Methods in Applied Mechanics and Engineering*, 2002d.

Oden, J., Carey, G. *Finite Elements Mathematical Aspects*, v. 4 de *Texas Finite Element Series*. Englewood Cliffis, New Jersey, Prentice-Hall, Inc., USA, 1983.

Oden, J., Reddy, J. *An Introduction to the Mathematical Theory of Finite Elements*. John Wiley & Sons, Co., New York, USA, 1976.

Prager, W., Rozvany, G. Optimal layout of grillages. *Journal Structural Mechanics*, v. 5p.1–18, 1977.

Pressman, R. S. *Engenharia de Software*. Makron Books, Brasil, 1995.

Qt. Troll Tech AS. Qt online reference documentation, version 2.3.0, 2001.

Rossow, M., Taylor, J. A finite element method for the optimal design of variable thickness sheets. *AIAA Journal*, v. 11p.1566–1569, 1973.

Rozvany, G. Grillages of maximum strength and maximum stiffness. *International Journal Mechanics Science*, v. 14p.651–666, 1972a.

Rozvany, G. Optimal load transmission by flexure. *Computational Methods Applied Mechanical Engineering*, v. 1p.253–263, 1972b.

Rozvany, G. Optimal layout theory: analytical solutions for elastic structures with several deflection constraints and load conditions. *Structural Optimization*, v. 4p.247–249, 1992.

Rozvany, G. Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics. *Structural and Multidisciplinary Optimization*, v. 21p.90–108, 2001.

Rozvany, G., Birker, T. On singular topologies in exact layout optimization. *Structural Optimization*, v. 8p.228–235, 1994.

Rozvany, G., Prager, W. A new class of of structural optimization problems: optimal archgrids. *Computational Methods in Applied Mechanical Engineering*, v. 19p.127–150, 1977.

Schumacher, A. *Topologieoptimierung von Bauteilstrukturen unter Verwendung von Lopchpositionierungkrieterien*. Universität Gesamthochschule-Siegen Berlin, 1995., Tese (Doutorado), Germany.

Silva, C. *Otimização Estrutural e Análise de Sensibilidade Orientadas Por Objetos*. DPM/FEM/UNICAMP, Campinas, 1997., Dissertação (Mestrado).

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.

Apêndice A

Funcionais de Performance para Análise de Sensibilidade

A seguir está apresentada a análise de sensibilidade de alguns funcionais de performance através da formulação contínua. Esses resultados foram extraídos de (Silva e Bittencourt, 1999).

A.1 Análise de Sensibilidade a Parâmetros Discretos

A seguir são desenvolvidas as expressões dos gradientes dos funcionais de massa, tensão e deslocamento para o problema de estado plano de tensão.

Funcional de Massa

O funcional de massa e a sua sensibilidade são dados por

$$\Psi_1 = \int_B \tilde{\rho}(\mathbf{x}) e(\mathbf{x})\, dA, \qquad \dot{\Psi}_1 = \int_B \tilde{\rho}(\mathbf{x}) \delta e\, dA.$$

Aproximando a espessura como $e(\mathbf{x}) \simeq p_k \phi_k$, sendo p_k as variáveis de projeto e ϕ_k funções de forma, pode-se escrever

$$\Psi_1(\mathbf{p}) = \sum_{k=1}^{n} p_k \int_{B_k} \tilde{\rho}(\mathbf{x})\, dA$$

$$\dot{\Psi}_1(\mathbf{p}) = \sum_{k=1}^{n} \delta p_k \int_{B_k} \tilde{\rho}(\mathbf{x})\, dA = \nabla_{\mathbf{p}} \Psi_1 \cdot \delta \mathbf{p} \rightarrow (\nabla_{\mathbf{p}} \Psi_1)_k = \frac{\delta \Psi_1}{\delta p_k} = \int_{B_k} \tilde{\rho}(\mathbf{x})\, dA.$$

Observa-se que não há a necessidade de formular um problema adjunto, pois este funcional é independente de $\mathbf{u}$.

Funcional de Tensão

Considere o funcional de tensão média na região $\mathcal{B}_\tau \subseteq \mathcal{B}$

$\Psi_2 = \int_{\mathcal{B}_\tau} \mathcal{G}(\mathbf{T}) m_\tau(\mathbf{x})\ dV = \frac{\int_{\mathcal{B}_\tau} \mathcal{G}(\mathbf{T})\ dV}{\int_{\mathcal{B}_\tau}\ dV}.$

Sua expressão de sensibilidade é dada por

$\dot{\Psi}_2 = \int_{\mathcal{B}} [\mathcal{G}_{,T}(u) \cdot \mathbf{T}(\dot{u})]\, m_T(\mathbf{x}) e(\mathbf{x})\ dA.$

Tem-se o seguinte problema adjunto do funcional Ψ_2: determinar o campo de deslocamentos adjuntos ς tal que

$a(\varsigma, v) = \int_{\mathcal{B}} [\mathcal{G}_{,T}(u) \cdot \mathbf{T}(v)]\, m_T(\mathbf{x})\ dA, \qquad \forall v \in \mathcal{V}.$

Pode-se, por exemplo, escrever o funcional de tensão Ψ_2 em relação à tensão equivalente de von Mises como,

$\Psi_2(\mathbf{p}) = \int_{\mathcal{B}} \sigma_{VM} m_\tau\ dA \rightarrow \mathcal{G}(\mathbf{T}(u)) = \sigma_{VM}.$

sendo que $\sigma_{VM} = \left\{\sigma_x^2 - \sigma_x\sigma_y + \sigma_y^2 + 3\tau_{xy}^2\right\}^{\frac{1}{2}}$ define a tensão equivalente no caso de tensão plana. Assim,

$$\mathcal{G}_{,T}(u) = \frac{1}{2\sigma_{VM}} \begin{bmatrix} 2\sigma_x - \sigma_y & 6\tau_{xy} \\ 6\tau_{xy} & 2\sigma_y - \sigma_x \end{bmatrix}.$$

Funcional de Deslocamento

Considere o funcional de deslocamento resultante nos pontos do domínio $\mathcal{G}(u) = |u(\mathbf{x})|$. Define-se o funcional de deslocamento resultante em um ponto como

$\Psi_3 \equiv |u(\mathbf{x}_k)| = \int_{\mathcal{B}} \mathcal{G}(u)\delta(\mathbf{x} - \mathbf{x}_k)\ dV$

Sua sensibilidade é descrita por

$\dot{\Psi}_3\,(\mathbf{p}) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot \dot{u})\delta(\mathbf{x} - \mathbf{x}_k)\ dA.$

Tem-se o seguinte problema adjunto do funcional Ψ_3: determinar o campo de deslocamentos adjunto ς tal que

$a(\varsigma, v) = \int_{\mathcal{B}} [\mathcal{G}_{,u}(u) \cdot v]\delta(\mathbf{x} - \mathbf{x}_k)\ dA, \qquad \forall v \in \mathcal{V}.$

Neste caso, $\mathcal{G}_{,u}(u) = \frac{1}{|u(\mathbf{x}_k)|}\{\ u_x(\mathbf{x}_k)\ \ u_y(\mathbf{x}_k)\ \}^T$ constitui-se no próprio carregamento adjunto do nó.

Se este funcional for definido em relação a um determinado grau de liberdade, então $\mathcal{G}_{,u}(u) = \{\ 1\ \ 0\ \}^T$ se $u_x(\mathbf{x}_k) > 0$; $\mathcal{G}_{,u}(u) = \{\ -1\ \ 0\ \}^T$ se $u_x(\mathbf{x}_k) < 0$; $\mathcal{G}_{,u}(u) = \{\ 0\ \ 1\ \}^T$ se $u_y(\mathbf{x}_k) > 0$; $\mathcal{G}_{,u}(u) = \{\ 0\ \ -1\ \}^T$ se $u_y(\mathbf{x}_k) < 0$.

A.2 Análise de Sensibilidade à Mudança de Forma

A seguir apresentam-se as expressões de análise de sensibilidade de alguns funcionais à mudança de forma.

Funcional de Massa

O seguinte funcional define a massa do corpo $\mathcal{B}_\tau$durante a variação de sua geometria

$$\Psi_1 = \int_{\mathcal{B}_\tau} \tilde{\rho}^\tau(\mathbf{x}^\tau)\ dV = \int_{\mathcal{B}} \tilde{\rho}^\tau(\mathbf{x})\ \det\mathbf{F} dV.$$

A sua sensibilidade em relação à variação do domínio é obtida por

$$\dot{\Psi}_1 = \int_{\mathcal{B}} [\mathbf{V}\cdot\nabla\tilde{\rho} + \tilde{\rho}\ \mathrm{Div}\mathbf{V}]\ dV \rightarrow \dot{\Psi}_1 = \int_{\mathcal{B}} \mathrm{Div}(\tilde{\rho}\mathbf{V})\ dV = \int_{\partial\mathrm{B}} \tilde{\rho}\mathbf{V}\cdot\mathbf{n}\ dA.$$

A componente k do vetor gradiente é dada por

$$(\nabla_{\mathbf{p}}\Psi_1)_k = \int_{\mathcal{B}} \mathrm{Div}\left(\tilde{\rho}\frac{\partial\mathbf{x}}{\partial p_k}\right)\ dV = \int_{\partial\mathrm{B}} \left(\tilde{\rho}\frac{\partial\mathbf{x}}{\partial p_k}\right)\cdot\mathbf{n}\ dA.$$

Funcionais de Tensão e Deslocamento

Estes funcionais depedem da solução da equação de estado do problema e sua sensibilidade pode ser obtida pelo método adjunto. Para um dado funcional, seu carregamento adjunto é o mesmo tanto para problemas de otimização de parâmetros como de forma. Assim, podem ser aplicadas as mesmas expressões de funcionais de tensão e deslocamento, com os seus respectivos carregamentos adjuntos, determinados para a sensibilidade a parâmetros.

Funcional de Energia e Deformação

A energia de deformação do corpo $\mathcal{B}\tau$ é definida pela expressão

$$\Psi_4 = \frac{1}{2}\int_{\mathcal{B}_\tau} C\left[(\text{grad } u^\tau)^S\right]\cdot(\text{grad } u^\tau)^S \; dV.$$

Por sua vez, a sensibilidade é determinada como

$$\begin{aligned}\dot{\Psi}_4 &= \int_{\mathcal{B}} C\left[\nabla\dot{u}^S\right]\cdot\nabla u^S \; dV - \int_{\mathcal{B}} C\left[(\nabla u\nabla\mathbf{V})^S\right]\cdot\nabla u^S \; dV + \\ &\quad \frac{1}{2}\int_{\mathcal{B}} C\left[\nabla u^S\right]\cdot\nabla u^S \; \text{Div}\mathbf{V} \; dV.\end{aligned}$$

A dependência da expressão anterior em relação a $\nabla\dot{u}$ pode ser eliminada. Desta maneira, evita-se a necessidade de resolver o problema adjunto para determinar a sensibilidade $\dot{\Psi}_4$. Logo,

$$\begin{aligned}\dot{\Psi}_4 &= \int_{\mathcal{B}}[(\nabla\mathbf{b})\mathbf{V}\cdot u + \mathbf{b}\cdot u \; \text{Div}\mathbf{V}] \; dV + \int_{-}[\Phi' + (\nabla\Phi)\mathbf{V}\cdot u + \Phi\cdot u(\mathbf{V}\cdot\mathbf{n}) \; \text{Div } \mathbf{n}] \; dA \\ &\quad + \int_{\mathcal{B}}\left\{C\left[(\nabla u\nabla\mathbf{V})^S\right]\cdot\nabla u^S - \frac{1}{2}C[\nabla u^S]\cdot\nabla u^S \; \text{Div}\mathbf{V}\right\} \; dV. \qquad \text{(A.1)}\end{aligned}$$

Apêndice B

Relação entre a DT e os Conceitos de ASMF

O cálculo da derivada topológica, através dos conceitos de análise de sensibilidade à mudança de forma foi demonstrado em (Novotny et al., 2000; Novotny et al., 2001a). Transcreve-se aqui o teorema e a sua prova.

Teorema 1 *Seja o gradiente topológico dado pela equação (3.2), tal que* $0 < G_T(\hat{\mathbf{x}}) < \infty$*, então o limite com* $\varepsilon \to 0$ *existe e pode ser escrito como*

$$G_T^*(\hat{\mathbf{x}}) = G_T(\hat{\mathbf{x}}) = \lim_{\varepsilon \to 0} \frac{1}{f'(\varepsilon)\,|v_n|} \left. \frac{d\psi(\Omega_\tau)}{d\tau} \right|_{\tau=0}. \tag{B.1}$$

Prova 1 *A prova deste teorema é dividida em duas partes*

Parte 1:

$$G_T(\hat{\mathbf{x}}) = \lim_{\substack{\varepsilon \to 0 \\ \delta\varepsilon \to 0}} \frac{\psi(\Omega_{\varepsilon+\delta\varepsilon}) - \psi(\Omega_\varepsilon)}{f(\varepsilon+\delta\varepsilon) - f(\varepsilon)} = G_T^*(\hat{\mathbf{x}}) \tag{B.2}$$

Parte 2:

$$G_T(\hat{\mathbf{x}}) = \lim_{\substack{\varepsilon \to 0 \\ \delta\varepsilon \to 0}} \frac{\psi(\Omega_{\varepsilon+\delta\varepsilon}) - \psi(\Omega_\varepsilon)}{f(\varepsilon+\delta\varepsilon) - f(\varepsilon)} = \lim_{\varepsilon \to 0} \frac{1}{f'(\varepsilon)\,|v_n|} \left. \frac{d\psi(\Omega_\tau)}{d\tau} \right|_{\tau=0} \tag{B.3}$$

Prova da Parte 1:

Do lado esquerdo da equação (B.2), tem-se

$$\lim_{\substack{\varepsilon \to 0 \\ \delta\varepsilon \to 0}} \frac{\psi(\Omega_{\varepsilon+\delta\varepsilon}) - \psi(\Omega_\varepsilon)}{f(\varepsilon+\delta\varepsilon) - f(\varepsilon)} = \lim_{\substack{\varepsilon \to 0 \\ \delta\varepsilon \to 0}} \frac{\psi(\Omega_{\varepsilon+\delta\varepsilon}) - \psi(\Omega_\varepsilon) + \psi(\Omega) - \psi(\Omega)}{f(\varepsilon+\delta\varepsilon) - f(\varepsilon)}. \tag{B.4}$$

Fatorando a equação (B.4), vem

$$\lim_{\substack{\varepsilon\to 0\\ \delta\varepsilon\to 0}}\left(\frac{\psi(\Omega_{\varepsilon+\delta\varepsilon})-\psi(\Omega)}{f(\varepsilon+\delta\varepsilon)}\frac{f(\varepsilon+\delta\varepsilon)}{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}-\frac{\psi(\Omega_{\varepsilon})-\psi(\Omega)}{f(\varepsilon)}\frac{f(\varepsilon)}{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}\right). \tag{B.5}$$

Observa-se que

$$\lim_{\varepsilon\to 0}\frac{\psi(\Omega_{\varepsilon+\delta\varepsilon})-\psi(\Omega)}{f(\varepsilon+\delta\varepsilon)}=\frac{\psi(\Omega_{\varepsilon})-\psi(\Omega)}{f(\varepsilon)}. \tag{B.6}$$

Então, substituindo a equação (B.6) na equação (B.4) e considerando a equação (3.2) tem-se, finalmente, que

$$\begin{aligned} G_T(\hat{\mathbf{x}}) &= \lim_{\varepsilon\to 0}\left[\frac{\psi(\Omega_{\varepsilon})-\psi(\Omega)}{f(\varepsilon)}\lim_{\delta\varepsilon\to 0}\left(\frac{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}\right)\right] \\ &= \lim_{\varepsilon\to 0}\frac{\psi(\Omega_{\varepsilon})-\psi(\Omega)}{f(\varepsilon)}=G_T^*(\hat{\mathbf{x}}) \end{aligned}$$

Prova da Parte 2:

Seja o lado esquerdo da equação (B.3), a qual pode ser escrita como

$$\lim_{\substack{\varepsilon\to 0\\ \delta\varepsilon\to 0}}\frac{\psi(\Omega_{\varepsilon+\delta\varepsilon})-\psi(\Omega_{\varepsilon})}{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}=\lim_{\substack{\varepsilon\to 0\\ \delta\varepsilon\to 0}}\frac{\psi(\Omega_{\varepsilon+\delta\varepsilon})-\psi(\Omega_{\varepsilon})}{\frac{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}{\delta\varepsilon}\delta\varepsilon}. \tag{B.7}$$

Observa-se que

$$\lim_{\delta\varepsilon\to 0}\frac{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}{\delta\varepsilon}=f'(\varepsilon). \tag{B.8}$$

Substituindo a equação (B.8) na (B.7) vem

$$\lim_{\varepsilon\to 0}\left(\frac{1}{f'(\varepsilon)}\lim_{\delta\varepsilon\to 0}\frac{\psi(\Omega_{\varepsilon+\delta\varepsilon})-\psi(\Omega_{\varepsilon})}{\delta\varepsilon}\right) \tag{B.9}$$

Considerando as equações (3.4) e (2.47), então a equação (B.9) pode ser reescrita, levando em conta que $\delta\varepsilon=\tau\,|v_n|$ equação (3.6), da seguinte maneira:

$$\lim_{\varepsilon\to 0}\frac{1}{f'(\varepsilon)}\lim_{\tau\to 0}\frac{\psi(\Omega_{\tau})-\psi(\Omega_0)}{\tau\,|v_n|}=\lim_{\varepsilon\to 0}\frac{1}{f'(\varepsilon)\,|v_n|}\left.\frac{d\psi(\Omega_{\tau})}{d\tau}\right|_{\tau=0}. \tag{B.10}$$

Finalmente, da equação (B.10) e do resultado da primeira parte da prova deste teorema (B.2), verifica-se que:

$$G_T^*(\hat{\mathbf{x}})=G_T(\hat{\mathbf{x}})=\lim_{\varepsilon\to 0}\frac{1}{f'(\varepsilon)\,|v_n|}\left.\frac{d\psi(\Omega_{\tau})}{d\tau}\right|_{\tau=0},\ \textit{onde}\ f'(\varepsilon)=\lim_{\delta\varepsilon\to 0}\frac{f(\varepsilon+\delta\varepsilon)-f(\varepsilon)}{f(\varepsilon)}$$

Com isto, o teorema está demonstrado.