UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA

OTIMIZAÇÃO ESTRUTURAL E ANÁLISE DE SENSIBILIDADE ORIENTADAS POR OBJETOS

Autor: **Cláudio Alessandro de Carvalho Silva**
Orientador: **Prof. Dr. Marco Lúcio Bittencourt**

20/97

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

OTIMIZAÇÃO ESTRUTURAL E ANÁLISE DE SENSIBILIDADE ORIENTADAS POR OBJETOS

Autor: **Cláudio Alessandro de Carvalho Silva**
Orientador: **Prof. Dr. Marco Lúcio Bittencourt**

Curso: Engenharia Mecânica
Área de Concentração: Mecânica dos Sólidos e Projeto Mecânico

Trabalho apresentado à Comissão de Pós-Graduação da Faculdade de Engenharia Mecânica, como parte dos requisitos para obtenção do título de Mestre em Engenharia Mecânica.

Campinas, 1997
S.P. - Brasil

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

Silva, Cláudio Alessandro de Carvalho
Si38o Otimização estrutural e análise de sensibilidade orientadas por objetos.--Campinas, SP: [s.n.], 1997.

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

1. Método dos elementos finitos. 2. Programação orientada a objetos (Computação). 3. C++ (Linguagem de programação por computador). 4. Programação não-linear. 5. Programação quadrática. I. Bittencourt, Marco Lúcio. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

DISSERTAÇÃO DE MESTRADO

Otimização Estrutural e Análise de Sensibilidade Orientadas por Objetos

Autor: **Cláudio Alessandro de Carvalho Silva**
Orientador: **Prof. Dr. Marco Lúcio Bittencourt**

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

Prof. Dr. Eduardo Alberto Fancello
DEM/UFSC

Prof. Dr. Ilmar Ferreira Santos
DPM/FEM/UNICAMP

Campinas, 10 de setembro de 1997.

A meus pais e avós.

Agradecimentos

Ao Prof. Dr. **Marco Lúcio Bittencourt** pela confiança, amizade, dedicação ao trabalho, incentivo e toda ajuda e orientação em mecânica dos sólidos e programação.

Ao Prof. Dr. **Euclides de Mesquita Neto** pelo orientação em trabalhos de iniciação científica, que foram, sem dúvida, o ínicio deste e de futuros trabalhos.

Ao Prof. Dr. **Ilmar Ferreira Santos** pelo exemplo de dedicação ao ensino e à pesquisa.

Ao Prof. Dr. **Mauro Jorge Atalla** pela amizade, conselhos e ajuda com programas e computadores.

Ao Prof. Dr. **Kamal Abel Radi Ismail** pela confiança e colaboração como coordenador da CPG/FEM.

Ao grandes amigos **Antônio Batista de Jesus** e **Cristina Minioli Saracho**, que me acompanham desde o primeiro dia nesta universidade, pelo inestimável apoio e incentivo (muito obrigado!).

Aos colegas e amigos **Fábio Russo**, **Rodrigo Nicoletti**, **Alexandre Scalabrin** e **Sabine Sirimarco Gomes** pelo companheirismo e ótimo ambiente de trabalho, e também, é claro, pelos churrascos, jogos, descontração, ...

A **Luciano**, **Tiago**, **Colla**, **Padovesi**, **Puls**, **Maurício**, **Thunder** e todo pessoal do DPM pelo incentivo.

Ao CNPq pelo indispensável apoio financeiro.

E, especialmente, a **Ellen** pela compreensão, carinho e dedicação em todos os momentos.

Toda nossa ciência, comparada com a realidade é primitiva e infantil – e, no entanto, é a coisa mais preciosa que temos.
Albert Einstein (1879 - 1955)

URBANA LEGIO OMNIA VINCIT.

Sumário

Lista de Figuras

Lista de Tabelas

Nomenclatura

Letras Latinas :

$a_{\mathbf{p}}(\cdot,\cdot)$ – Operador bilinear (função do vetor de projetos $\mathbf{p}$)

$\bar{b}_j$ – Perturbação na j-ésima restrição de desigualdade

$\bar{\mathbf{b}}$ – Vetor de perturbações nas restrições de desigualdade

$\mathbf{b}(\cdot)$ – Campo vetorial de forças de volume. Campo material

$\mathbf{b}^{\tau}(\cdot)$ – Campo vetorial de forças de volume em τ. Campo espacial

$\mathbf{c}_k$ – Vetor de parâmetros da função de penalização na k-ésima iteração do método de Herskovits

$\mathbf{d}$ – Direção de busca num método de otimização

$\mathbf{d}^{(k)}$ – Direção de busca na k-ésima iteração do algoritmo de minimização

$\mathbf{d}_0$ – Direção de busca obtida da solução do sistema de equações χ na k-ésima iteração (método de Herskovits)

$\mathbf{d}_1$ – Direção de deflexão da direção original $\mathbf{d}_0$ na k-ésima iteração (método de Herskovits)

$\mathbf{d}_d$ – Direção de decréscimo (métodos de programação quadrática recursiva)

$\mathbf{d}_c$ – Direção de correção (métodos de programação quadrática recursiva)

$e(\cdot)$ – Função escalar descrevendo a espessura em problemas de tensão plana

$\bar{e}_i$ – Perturbação na i-ésima restrição de desigualdade

$\bar{\mathbf{e}}$ – Vetor de perturbações nas restrições de desigualdade

$f(\cdot)$ – Função objetivo ou função custo do problema de otimização. Funcional

$\mathbf{f}$ – Vetor de carregamentos do problema estrutural. Vetor real de dimensão N

$\mathbf{f}_e^{adj}$ – Carregamento adjunto no elemento finito e

$g_i(\cdot)$ – Restrição de desigualdade do problema de otimização. Funcional

$\mathbf{g}(\cdot)$ – Vetor de restrições de desigualdade. Função vetorial de dimensão m

$h_j(\cdot)$ – Restrição de igualdade do problema de otimização. Funcional

$\mathbf{h}(\cdot)$ – Vetor de restrições de igualdade. Função vetorial de dimensão p

$l_{\mathbf{p}}(\cdot)$ – Operador linear (função do vetor de projetos $\mathbf{p}$)

$l_{\mathbf{p}}^{adj}(\cdot)$ – Operador linear associado ao método adjunto de análise de sensibilidade

m – Número de restrições de desigualdade num problema de otimização

$m_{\tau}(\cdot)$ – Função de média ou função de ponderação

n – Número de variáveis de projeto. Dimensão do vetor $\mathbf{p}$

$\mathbf{n}$ – Vetor unitário normal a uma superfície. Campo material

$\mathbf{n}^{\tau}$ – Vetor unitário normal a uma superfície em τ. Campo espacial

p – Número de restrições de igualdade num problema de otimização

$\mathbf{p}$ – Vetor de variáveis de projeto. Vetor real de dimensão n

$\mathbf{p}^*$ – Solução do problema de otimização. Ponto de mínimo, projeto ótimo

$\mathbf{p}^{(k)}$ – Vetor de variáveis de projeto na k-ésima iteração do algoritmo de minimização

q – Número de pontos de controle de uma curva *NURBS*

$\mathbf{s}^{(k)}$ – Vetor auxiliar do método BFGS

r – Parâmetro de uma curva *NURBS*

$\mathbf{r}$ – Resíduo da solução do problema estrutural

tr – Traço de um tensor

t – Variável de busca linear

t_k – Passo na direção de descrécimo da k-ésima iteração do algoritmo de minimização

t_0^I – Estimativa inicial de t_k estimada por linearização das restrições de desigualdade na k-ésima iteração (método de Herskovits)

t_0^E – Estimativa inicial de t_k estimada por linearização das restrições de igualdade na k-ésima iteração (método de Herskovits)

t_0 – Estimativa inicial de t_k. Menor valor entre t_0^I e t_0^E (método de Herskovits)

$u(\cdot,\cdot)$ – Campo de deslocamentos do problema estrutural. Campo material

$u^\tau(\cdot,\cdot)$ – Campo de deslocamentos do problema estrutural em τ. Campo espacial

$u^{(N)}$ – Aproximação de u num espaço de dimensão N (finita)

$\mathbf{u}$ – Solução discreta do campo de deslocamentos do problema estrutural. Vetor real de dimensão N

$v(\cdot)$ – Campo de deslocamentos virtuais do problema estrutural

$\mathbf{v}$ – Conjunto de variáveis do sistema de equações $\boldsymbol{\chi}$ (variáveis de projeto e multiplicadores de Lagrange)

$\mathbf{v}_0$ – Solução do sistema de equações $\boldsymbol{\chi}$ na k-ésima iteração

$\mathbf{v}_1$ – Solução auxiliar (defletida) do sistema de equações $\boldsymbol{\chi}$ na k-ésima iteração. (método de Herskovits)

$\mathbf{w}^{(k)}$ – Vetor auxiliar do método BFGS

$\mathbf{x}$ – Vetor em $\Re^3$. Ponto material

$\mathbf{x}^\tau$ – Posição ocupada pelo ponto $\mathbf{x}$ em τ. Ponto epacial

$\mathbf{y}^{(k)}$ – Vetor auxiliar do método BFGS

$\mathbf{z}^{(k)}$ – Vetor auxiliar do método BFGS

$\mathbf{A}(\cdot)$ – Gradiente da função vetorial $\mathbf{g}(\cdot)$. Dimensão $m \times n$

$\mathbf{B}$ – Matriz operador de deformação

$\mathbf{C}(\cdot)$ – Gradiente do funcional $f(\cdot)$

$\mathsf{C}[\cdot]$ – Operador tensor de elasticidade de Hooke

$\mathbf{D}$ – Tensor de elasticidade de Hooke

$\tilde{E}$ – Módulo de elasticidade longitudinal

$\mathbf{E}(\cdot)$ – Tensor de deformações associado ao campo de deslocamentos u. Campo material

$\mathbf{E}^\tau(\cdot)$ – Tensor de deformações associado ao campo de deslocamentos u^τ. Campo espacial

$\mathbf{F}(\cdot,\cdot)$ – Gradiente da função de movimento $\mathsf{X}^\tau(\cdot,\cdot)$

$\mathbf{G}(\cdot)$ – Matriz diagonal onde $G_{ii}(\cdot) = g_i(\cdot)$. Ordem m

$\mathbf{H}$ – Matriz hessiana do Lagrangeano L

I_k – Conjunto de índices das restrições potencialmente ativas na k-ésima iteração

$\mathbf{I}$ – Matriz identidade

$\mathbf{J}_e$ – Jacobiano do elemento finito e

L – Lagrangeano do problema de otimização

$\mathbf{L}(\cdot)$ – Gradiente da função vetorial $\mathbf{h}(\cdot)$. Dimensão $p \times n$

$\mathbf{K}$ – Matriz de rigidez do problema estrutural. Matriz real simétrica de ordem N

N – Dimensão da discretização em elementos finitos. Número natural

$\hat{N}_{j\chi}$ – j-ésima função da base de uma curva *NURBS* de ordem χ

$N_{j\chi}$ – j-ésima função da base de uma B-*spline* de ordem χ

$\mathbf{Q}^{(k)}$ – Aproximação numérica de $\mathbf{H}$ pelo método quase-Newton BFGS. Matriz simétrica positiva definida

$\mathbf{R}$ – Matriz do sistema χ na k-ésima iteração da solução iterativa (método de Herskovits)

$\mathbf{T}(\cdot)$ – Tensor de Cauchy associado ao campo de deslocamentos u. Campo material

$\mathbf{T}^{\tau}(\cdot)$ – Tensor de Cauchy associado ao campo de deslocamentos u^{τ}. Campo espacial

$\mathbf{V}_s(\cdot,\cdot)$ – Descrição espacial da velocidade de mapeamento de projeto

$\mathbf{V}(\cdot)$ – Campo de velocidades em $\tau = 0$

$\mathbf{X}_j$ – Coordenadas do j-ésimo ponto de controle de uma curva *NURBS*

$\mathsf{X}(\cdot,\cdot)$ – Inversa de $\mathsf{X}^{\tau}(\cdot,\cdot)$

$\mathsf{X}^{\tau}(\cdot,\cdot)$ – Função de movimento que descreve a mudança na geometria do domínio do problema estrutural

Letras Gregas :

$\alpha_i^{(N)}$ – coeficiente real (método de Galerkin)

α – Eficiência de decréscimo requirida no método de Herskovits. Escalar no intervalo $(0,1)$

β_i – i-ésimo peso de uma curva *NURBS*

γ – Parâmetro de controle de saturação das restrições no método de Herskovits. Escalar no intervalo $(0,1)$

δ – Parâmetro de controle do tipo de atualização dos multiplicadores de Lagrange no algoritmo de Herkovits. Escalar no intervalo $(0,1)$

$\delta(\cdot)$ – Delta de Dirac

ϵ – Parâmetro usado na atualização dos multiplicadores de Lagrange no algoritmo de Herkovits. Escalar positivo

ε – Precisão para restrições potencialmente ativas. Escalar no intervalo $(0,1)$

ζ_1 - Limite inferior para controle da norma da aproximação numérica da matriz hessiana. Escalar positivo

ζ_2 - Limite superior para controle da norma da aproximação numérica da matriz hessiana. Escalar positivo

η_k – Parâmetro dos métodos de programação quadrática recursiva

η – Parâmetro de decréscimo da regra de Armijo. Escalar no intervalo $(0, 1)$

$\bar{\eta}$ – Coordenada local do elemento triangular isoparamétrico

θ – Escalar auxiliar no método BFGS

θ_i – Parâmetro de deflexão da i-ésima restrição de desigualdade

ϑ_1 – Escalar auxiliar no método BFGS

ϑ_2 – Escalar auxiliar no método BFGS

ϑ_3 – Escalar auxiliar no método BFGS

$\boldsymbol{\kappa}$ – Vetor de nós de uma curva *NURBS*

$\tilde{\lambda}$ – Coeficiente de Lamé

$\boldsymbol{\lambda}$ – Vetor de multiplicadores de Lagrange das restrições de desigualdade

$\boldsymbol{\lambda}^*$ – Valor de $\boldsymbol{\lambda}$ na solução do problema de otimização

$\boldsymbol{\lambda}^{(k)}$ – Vetor de multiplicadores de Lagrange das restrições de desigualdade na k-ésima iteração do algoritmo de minimização

$\boldsymbol{\lambda}_0$ – Multiplicadores de Lagrange das restrições de desigualdade obtidos da solução do sistema de equações $\boldsymbol{\chi}$ na k-ésima iteração do algoritmo de otimização

$\bar{\boldsymbol{\lambda}}$ – Estimativa de $\boldsymbol{\lambda}$

λ_i – Multiplicador de Lagrange correspondente à i-ésima restrição de desigualdade

λ_I – Limite inferior dos multiplicadores de Lagrange de restrições de desigualdade. Escalar positivo

λ_S – Limite superior dos multiplicadores de Lagrange de restrições de desigualdade. Escalar positivo

$\tilde{\mu}$ – Coeficiente de Lamé

$\boldsymbol{\mu}$ – Vetor de multiplicadores de Lagrange das restrições de igualdade

$\boldsymbol{\mu}^*$ – Valor de $\boldsymbol{\mu}$ na solução do problema de otimização

$\boldsymbol{\mu}^{(k)}$ – Vetor de multiplicadores de Lagrange das restrições de igualdade na k-ésima iteração do algoritmo de minimização

μ_j – Multiplicador de Lagrange correspondente à j-ésima restrição de igualdade

$\tilde{\nu}$ – Coeficiente de Poisson

ν – Parâmetro da regra de Armijo para obtenção de estimativas de passo. Escalar no intervalo $(0, 1)$

ξ – Precisão exigida na solução numérica do problema de otimização. Escalar no intervalo $(0, 1)$

$\bar{\xi}$ – Coordenada local do elemento triangular isoparamétrico

$\tilde{\rho}(\cdot)$ – Campo escalar de densidade

ρ – Peso de deflexão da direção de busca no algoritmo de Herskovits. Escalar positivo

$\bar{\rho}$ – Valor de ρ quando a direção de deflexão $\mathbf{d}_1$ também é direção de descida (método de Herskovits)

σ_{VM} – Tensão equivalente de von Mises

σ_x, σ_y – Componentes do tensor de tensões

$\boldsymbol{\varsigma}$ – Solução adjunta da formulação discreta da análise de sensibilidade

τ – Parâmetro de variação do projeto

τ_{xy} – Componente do tensor de tensões

ϕ_i – Elemento de uma base de Schauder de um espaço $\mathcal{V}$

φ – Função de descréscimo de um algoritmo de minimização

ψ_1 – Funcional de massa

ψ_2 – Funcional de tensão

ψ_3 – Funcional de deslocamento

ψ_4 – Funcional de energia de deformação

$\boldsymbol{\chi}$ – Sistema de equações não-lineares obtido das condições de Karush-Kuhn-Tucker

χ – Ordem de uma curva *NURBS*

$\boldsymbol{\omega}^I$ – Peso aplicado às restrições de desigualdade no cálculo da deflexão $\mathbf{d}_1$. Vetor de termos positivos

$\boldsymbol{\omega}^E$ – Peso aplicado às restrições de igualdade no cálculo da deflexão $\mathbf{d}_1$. Vetor de termos positivos.

ω_1 – Limite inferior das componentes de $\boldsymbol{\omega}^I$. Escalar positivo

ω_2 – Limite superior das componentes de $\boldsymbol{\omega}^I$. Escalar positivo.

$\boldsymbol{\Lambda}^{(k)}$ – Matriz diagonal onde $\Lambda_{ii}^{(k)} = \lambda_i^{(k)}$

Γ^0, Γ^1, Γ^2 – Subconjuntos do contorno $\partial\mathcal{B}$

$\overset{\circ}{\Gamma}{}^0$, $\overset{\circ}{\Gamma}{}^1$, $\overset{\circ}{\Gamma}{}^2$ – Interiores de Γ^0, Γ^1, Γ^2

$\boldsymbol{\Phi}$ – Forças de superfície aplicadas em um corpo

$\boldsymbol{\Phi}^\tau$ – Forças de superfície aplicadas em um corpo em τ

Φ – Campo material

Φ_S – Descrição espacial de Φ

Ψ – Campo espacial

Ψ_m – Descrição material de Ψ

Ω – Região factível do problema de otimização

Ω_a – Subconjunto de Ω

Ω_a^0 – Interior de Ω_a

Símbolos :

$\mathcal{B}$ – Domínio do problema elástico. Subconjunto do espaço vetorial real tridimensional $\Re^3$

$\mathcal{B}_e$ – Subdomínio de $\mathcal{B}$

$\mathcal{B}_\tau$ – Subdomínio de $\mathcal{B}$

$\mathcal{B}_\tau$ – Região ocupada por $\mathcal{B}$ em τ

$\partial\mathcal{B}$ – Contorno da região $\mathcal{B}$

$\partial\mathcal{B}_\tau$ – Contorno da região $\mathcal{B}_\tau$

div – Divergente em relação ao ponto espacial

Div – Divergente em relação ao ponto material

grad – Gradiente em relação ao ponto espacial

$\mathcal{G}$ – Funcional

$\mathcal{P}$ – Função de penalização

$\mathcal{T}$ – Trajetória

$\mathcal{V}$ – Espaço dos deslocamentos cinematicamente admissíveis de um problema estrutural

$\mathcal{V}'$– Espaço dual de $\mathcal{V}$

$\mathcal{V}_N$ – Subespaço de $\mathcal{V}$ de dimensão N (finita)

dA – Diferencial de área

dV – Diferencial de volume

$Erro_{MCA}\%$ – Erro relativo da maior componente

$Erro_{PIN}$ – Erro no produto interno normalizado

PIN – Produto interno normalizado

$\nabla(\cdot)$ – Gradiente em relação ao ponto material

$\nabla(\cdot)^S$ – Parte simétrica do gradiente de uma função vetorial

$\nabla_{\mathbf{p}}(\cdot)$ – Gradiente em relação às variaveis de projeto

$\nabla_{\mathbf{p}}(\cdot)^{AS}$ – Gradiente determinado por análise de sensibilidade

$\nabla_{\mathbf{p}}(\cdot)^{MDF}$ – Gradiente determinado por diferenças finitas

$\Re$ – Conjunto dos números reais

$\Re^3$ – Espaço vetorial real tridimensional

span[·] – Espaço gerado por um conjunto de vetores

sup – Supremo

Resumo

SILVA, Cláudio Alessandro de Carvalho, *Otimização Estrutural e Análise de Sensibilidade Orientadas por Objetos,* Campinas, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1997, 120 p., Dissertação (Mestrado).

Neste trabalho, apresentam-se conceitos de otimização estrutural e análise de sensibilidade em elasticidade linear, tomando-se exemplos de problemas bidimensionais discretizados pelo método de elementos finitos.

A partir das características específicas dos problemas estruturais, identificam-se os requisitos necessários a um algoritmo de minimização eficiente para tais problemas. Foi implementado um algoritmo de programação quadrática recursiva de ponto interior para otimização, o qual apresenta taxa de convergência superlinear e utiliza busca linear imprecisa.

Utilizam-se a formulação contínua da análise de sensibilidade e o método adjunto no desenvolvimento das expressões de sensibilidade a parâmetros e forma, visando a determinação de gradientes de funcionais de performance estrutural. Na análise de sensibilidade a forma, a geometria do domínio é parametrizada em NURBS, definindo o campo de velocidades no contorno. Os gradientes obtidos pela análise de sensibilidade são aplicados na otimização de espessura e na obtenção de formas ótimas em problemas de estado plano de tensão.

Os procedimentos numéricos foram incorporados a uma base de programas já desenvolvida para análise pelo método de elementos finitos, empregando o paradigma por objetos em C++. Com isto foi possível aumentar a eficiência da interação entre os módulos de análise por elementos finitos, análise de sensibilidade e otimização.

Palavras chaves:

- Método dos elementos finitos
- Programa orientada a objetos (Computação)
- C++ (Linguagem de programação de computador)
- Programação Não-Linear
- Programação Quadrática

Abstract

SILVA, Cláudio Alessandro de Carvalho, *Object Oriented Structural Optimization and Design Sensitivity Analysis*, Campinas, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1997, 120 p., Master's thesis.

This work presents concepts of structural optimization and design sensitivity analysis in linear elasticity, considering two dimensional finite element problems.

The requirements of an efficient minimization algoritm for structural problems are identified, taking into account specific characteristics of this sort of problems. An interior point sequential quadratic programming algorithm, showing superlinear convergence and using inexact line search, was implemented.

Continuum approach of design sensitivity analysis and adjoint method were used to develop shape and size sensitivity expressions, aiming for computing performance functional gradients. In shape sensitivity analysis, the domain geometry was parametrized using NURBS, which defines the boundary velocity field. The gradients obtained from design sensitivity analysis were applied to thickness and shape optimization of plane stress problems.

The numerical procedures were linked to other programs previously developed for finite element analysis applying object-oriented concepts in C++. Indeed, it was possible to improve the efficiency of interaction among finite element analysis, design sensitivity analysis and optimization modules.

Key words:

- Finite element method
- Object oriented programming (Computing)
- C++ (Computer programming language)
- Non linear programming
- Quadratic programming

Capítulo 1

Introdução

1.1 Aspectos Gerais

Pode-se afirmar que é através da atividade de projeto que os recursos e ferramentas da engenharia são direcionados para a solução de problemas práticos. A qualidade destas soluções é julgada pela forma como os requisitos e exigências de cada problema são cumpridos e respeitados. Entretanto, apesar dos critérios de julgamento das soluções adotadas mudarem de acordo com contexto social e histórico, e de nem sempre serem claramente definidos ou compreendidos, é constante nesta atividade de engenharia o questionamento sobre a qualidade das alternativas possíveis e a busca daquelas que sejam de alguma forma *melhores*. Atualmente, a busca por metodologias mais eficientes e seguras de projeto faz parte do contexto de demanda por soluções combinando qualidade e menor preço, sendo também menos agressivas ao meio ambiente e às pessoas.

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

Num ambiente com pequena demanda por inovações é razoável supor que novos projetos sejam constituídos de pequenas alterações de trabalhos anteriores. Neste contexto, o questionamento se esta é a melhor linha de ação para a solução do problema torna-se uma preocupação secundária. Ainda assim dificuldades podem ser encontradas, principalmente devido à inexistência de critérios objetivos para comparar soluções equivalentes. Tais dificuldades tornam-se ainda mais evidentes quando é necessário analisar situações novas ou procurar por soluções originais, casos onde a experiência não fornece todas as respostas.

Por sua vez, em um ambiente de competição tecnológica agressiva e mercado exigente por novidades, essa abordagem se mostra inadequada pois as demandas por alta qualidade, economia de energia e matérias-primas, baixo impacto ambiental e pequeno tempo de desenvolvimento exigem a criação de novos produtos para os quais inexiste experiência anterior.

Uma alternativa proposta na década de 60 foi a metodologia de projeto baseada em otimização, consistindo numa abordagem sistemática de modelagem e busca da melhor solução possível direcionadas por critérios objetivamente definidos. Em outras palavras, essa metodologia busca a sistematização da atividade de projeto, onde conhecimento e experiência acumulada são aplicadas 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 se-

melhante.

O termo otimização está relacionado a atividades de obtenção de soluções mais eficientes e adequadas ou ainda a utilização mais racional de recursos. Matematicamente, o problema de otimização pode ser formulado como a determinação de máximos e mínimos de funções sujeitas a certas condições.

Na definição de um problema de otimização, deve-se, inicialmente, determinar um critério de performance para comparar as diversas soluções para um dado problema. Em alguns casos determina-se claramente um único critério de performance, denominado *função objetivo* ou *função custo*. Já para os problemas de otimização multiobjetivos, a formulação correta apenas é possível se for estabelecido um conjunto de funções custo. Uma vez determinadas as funções de performance do problema, devem ser identificadas suas variáveis, as quais assumem, em alguns casos, apenas valores inteiros, sendo em geral números reais.

Tais variáveis são chamadas *variáveis de projeto* pois, uma vez que assumam valores numéricos, um projeto é obtido. Por último, devem ser determinados os limites impostos à obtenção da melhor solução, ou seja, os limites de recursos e as exigências que a solução deve respeitar, sendo denominados *requisitos de projeto* no contexto de engenharia. Matematicamente são definidos como *funções de restrição*, podendo ser tanto de igualdade quanto de desigualdade.

Entretanto, dado um projeto, os valores dos funcionais somente distinguem entre projetos viáveis e não-viáveis (que não respeitam as restrições). Além disso, esta informação é relativa, permitindo apenas comparar um conjunto de soluções possíveis. Em outras palavras, os valores destes funcionais determinam apenas qual a melhor solução dentre um conjunto de opções, não sendo possível concluir *como* melhorar um dado projeto. Esta informação somente pode ser obtida a partir dos *gradientes* dos funcionais de performance, ou seja, da sensibilidade do modelo à variação de seus parâmetros.

Geralmente, em otimização estrutural, os funcionais são expressos de forma implícita nas variáveis de projeto. Desta maneira, não se conhece explicitamente a forma relacionando as variáveis ao valor dos funcionais do problema. Esta relação é obtida indiretamente e de modo numérico, a partir da solução de uma *equação de estado*. Neste caso, a avaliação dos gradientes também deve ser feita numericamente. Para isso, a primeira opção seria aplicar uma técnica de diferenças finitas, não exigindo modificações no programa de análise, pois se baseia numa aproximação discreta da derivada parcial: assume-se uma perturbação no valor atual da variável, calcula-se a variação correspondente na função e toma-se o quociente desses dois valores.

Porém, apesar da simplicidade, esse método não é recomendável para aplicações de otimização estrutural por dois motivos. Primeiro porque o ajuste do valor da perturbação a ser imposta a cada uma das variáveis exige uma interação excessiva com o usuário, pois o valor adequado varia pontualmente devido às não-linearidades do problema. Assumindo que as perturbações são ajustadas com certa precisão, existe a ineficiência que a técnica de diferenças finitas induz ao processo global de otimização. Em problemas com n variáveis de projeto, as técnicas de diferença para trás ou a frente exigem que a equação de estado, no caso estrutural um sistema linear de grandes dimensões, seja resolvida n vezes. Deve-se considerar, entretanto, que os melhores resultados são obtidos pela técnica de diferença central, demandando $2n$ soluções da equação de estado. Em outras palavras, o custo é demasiado alto mesmo para problemas de dimensão média.

A formulação de análise de sensibilidade fornece a base teórica e computacional para o cálculo dos gradientes de funcionais que dependem implicitamente das variáveis do problema, de maneira mais eficiente e sem a necessidade de especificar os valores de perturbação das variáveis durante os cálculos.

Em problemas estruturais, alguns dos funcionais relevantes são relacionados a grandezas de

massa, tensão, deformação, energia, freqüências e deslocamento. Por exemplo, pode-se formular um problema de projeto de um componente de menor peso respeitando suas especificações, onde a função custo é claramente a massa, sendo os demais funcionais (tensão, deslocamento, deformação, energia e exigências construtivas) restrições a serem cumpridas. Pode-se também buscar o melhor posicionamento ou geometria de detalhes construtivos tais como filetes, raios, furos em termos de menores concentrações de tensão ou menores deslocamentos em certos pontos.

Como quase totalidade dos componentes mecânicos são construídos a partir da combinação de elementos tais como barras, vigas membranas, placas e sólidos, podem ser identificadas cinco tipos de variáveis de projetos: (i) material, (ii) dimensões, (iii) forma, (iv) configuração ou (v) topologia. As variáveis de material, dimensão e forma podem ser definidas em cada componente para uma certa opção de projeto. Em contraste, variáveis de configuração regulam a combinação de componentes compondo a estrutura. Por sua vez, as variáveis de topologia controlam a adição ou remoção de componentes no sistema estrutural.

Nesse trabalho serão consideradas apenas variáveis de dimensões e forma. Do ponto de vista da análise de sensibilidade, estes tipos de variáveis são classificadas em dois grupos. Para o primeiro tipo, tem-se os problemas de análise de sensibilidade a parâmetros, envolvendo variáveis de dimensões que não alteram a forma do componente. No segundo caso, tem-se a análise de sensibilidade à forma, onde é possível estabelecer variáveis de projeto envolvendo a geometria do componente. Conseqüentemente, pode-se combinar a sensibilidade a parâmetros e forma, permitindo estabelecer problemas de otimização estrutural com estes dois tipos de variáveis.

O objetivo desse texto é formular e implementar o problema de otimização de parâmetros e forma para componentes estruturais bidimensionais sujeitos a carregamento estático. Tais problemas envolvem uma única função objetivo, variáveis de projeto assumindo valores reais, sendo a solução da equação de estado do problema estrutural feita pelo método de elementos finitos.

A seguir apresenta-se uma revisão histórica de otimização estrutural, destacando sua evolução e aspectos de programação matemática, otimização de forma, análise de sensibilidade e aspectos atuais. Posteriormente, aborda-se o problema de elasticidade linear e sua discretização, a definição do problema de otimização e funcionais de performance. Finalmente, consideram-se aspectos da implementação computacional através da programação por objetos.

1.2 Revisão Histórica

1.2.1 Origem e Desenvolvimento Inicial

Pode-se considerar o início da teoria clássica da otimização matemática no século XVII com os trabalhos de Euler e Lagrange, estabelecendo as condições necessárias de pontos estacionários de funções diferenciáveis sujeitas ou não a restrições de igualdade. Estas condições permitiram formular uma série de problemas teóricos e práticos, tais como a determinação de máximos e mínimos de funções.

No contexto estrutural, Galileu Galilei, ainda no século XVII, discutiu a forma ótima da viga e Lagrange formulou o problema da coluna elástica de menor peso. Podem ser citados, da mesma forma, trabalhos de estudiosos como Jacob Bernoulli (1655-1705), William Rowan Hamilton (1808-1865) e James Clerk Maxwell (1831-1879) também dedicados a formulações iniciais do problema de otimização estrutural, segundo o histórico de TIMOSHENKO[1].

Já no século XX, Karush, Kuhn e Tucker estabeleceram as condições de otimalidade para problemas também sujeitos a restrições de desigualdade, obtendo assim a forma matemática geral do problema de otimização. MICHELL[2], no início deste século, desenvolveu a teoria

básica para a configuração ótima de treliças submetidas a uma única carga e sujeitas apenas a restrições de tensão. Entretanto, tais trabalhos apresentam uma abordagem analítica de componentes altamente idealizados, dificilmente aplicáveis em situações práticas. Além disso, a falta de uma compreensão clara de como formular problemas estruturais genéricos em termos da otimização matemática, fez com que questões de otimização estrutural viessem a ser levantadas sistematicamente apenas em meados do século XX.

Durante a Segunda Guerra Mundial e início dos anos 50, o desenvolvimento de estruturas aeronáuticas seguiu duas abordagens: seleção prévia dos critérios de falha relevantes em bases intuitivas (cuja solução não representava necessariamente uma solução ótima do problema original), dando origem a um sistema de equações não-lineares; e a formulação de problemas de treliças planas dentro da filosofia de projeto para colapso plástico, obtendo-se problemas de programação linear (este tipo de otimização estrutural de sistemas, originado no contexto da engenharia civil, concentrou-se principalmente em treliças e pórticos de aço e não considera restrições de tensão, deflexão ou flambagem).

Em 1955, KLEIN[3] reconhece que os casos gerais de otimização estrutural podem ser vistos como problemas de programação matemática não-linear. Apesar de tratar casos relativamente simples, a grande inovação deste trabalho está na inclusão de restrições de desigualdade, formulando adequadamente o problema de otimização estrutural. Entretanto, o impacto desse trabalho foi reduzido pois foi aplicada a técnica clássica de converter restrições de desigualdade em restrições de igualdade através da introdução de variáveis de folga, aumentando a dimensão do problema e dificultando a sua solução.

Em 1960, SCHMIT[4] é o primeiro a compreender a viabilidade da aplicação sistemática de técnicas de programação matemática, já conhecidas pela comunidade de pesquisa operacional, para a solução de problemas gerais de otimização de projetos. Neste caso, aplica-se o conceito de *síntese estrutural sistemática* que, segundo o autor, pode ser definida como uma evolução racionalmente direcionada de uma configuração estrutural em termos de um critério eficientemente definido cumprindo um conjunto de propósitos funcionais. Esse conceito define, num enunciado simples, toda atividade de projeto como um técnica de otimização e, segundo VANDERPLAATS[5], representou uma mudança revolucionária na abordagem de projeto.

Em outras palavras, SCHIMT[4] é o primeiro a compreender o problema de projeto (síntese estrutural) como um técnica de alocação ótima de recursos escassos, respeitando um dado conjunto de restrições. Também introduz a idéia e indica a viabilidade de se acoplar a análise estrutural por elementos finitos com a programação matemática não-linear, obtendo uma ferramenta de projeto automatizado, aplicável a uma grande classe de problemas de interesse prático. De fato, não é mera coincidência que esse também seja o período de inicio da corrida espacial e do surgimento dos computadores digitais.

Durante toda a década de 60, imediatamente após a introdução desses conceitos, grande parte da atividade de pesquisa se dirigiu em tentativas de desenvolvimento de uma primeira geração de programas de síntese estrutural baseados em métodos de programação matemática. Entretanto, os primeiros resultados desse esforço pareciam mostrar que a otimização estrutural numérica, por exigir uma grande demanda de processamento e devido à limitação dos computadores da época, poderia dar poucos resultados práticos fora da esfera acadêmica. Assim, os esforços passaram a se concentrar, nessa década e na seguinte, no sentido de determinar formas mais eficientes de formular o problema e em aperfeiçoar os algoritmos de otimização.

Desenvolve-se então o conceito de critérios de otimalidade fisicamente motivados, reduzindo a dimensão dos problemas e permitindo determinar um mínimo aproximado de maneira satisfatória, mas que ainda não possuía uma base matemática sólida. Tais critérios eram utilizados na otimização de sistemas, em contraste com os métodos de programação matemática, ainda utilizados no projeto de componentes.

Também surge o conceito de aproximação com SCHIMT [6], que através de mudanças de coordenadas permite reduzir a dimensão do problema original. Ainda nesse período, desenvolve-se um base matemática para as técnicas de critérios de otimalidade fisicamente motivados através do conceito de dualidade, mostrando que tais técnicas são um caso especial da programação matemática clássica.

Apesar disso, no início da década de 70, a otimização estrutural ainda não estava sendo adotada pela comunidade profissional por vários motivos: o método de elementos finitos ainda não estava maduro como ferramenta de análise, a programação matemática não era conhecida por grande parte dos engenheiros e os aspectos numéricos da programação não-linear não estavam bem estabelecidos. Em meados dessa década, a grande barreira para a aceitação definitiva da otimização como técnica eficiente no projeto estrutural era a inexistência de programas de propósito geral difundindo sua aplicação prática, a exemplo de programas como Ansys e Nastran que popularizaram a análise por elementos finitos.

No início da década de 80, autores como VANDERPLAATS[5], SCHMIT[7] e TEMPLEMAN[8] afirmam que, após 20 anos de amadurecimento (a maioria dos algoritmos já era conhecida desde os anos 60, como mostram os anais editados por LAVI[9] em 1966), o estado da arte em programação matemática era tal que a existência de algoritmos de otimização não se constituía uma questão crítica.

Em outras palavras, a metodologia de otimização havia evoluído o suficiente para fornecer uma ferramenta prática de projeto. Porém, estes trabalhos citam como desafios o desenvolvimento de algoritmos eficientes para problemas altamente não-lineares (para reduzir a necessidade de aproximações), disponibilizar cálculos de gradientes nos programas de análise através de técnicas eficientes de análise de sensibilidade e determinação de modos eficientes de integrar os códigos de programação matemática e de análise numa ferramenta automatizada.

Livros

Como livros-texto desse período podem ser citados os trabalhos de LUENBERG[10], POLAK[11] e ARORA[12].

1.2.2 Consolidação das Técnicas de Otimização Estrutural

LEVY[13] apresenta a primeira metade da década de 80 como um período de consolidação das técnicas de otimização, verificando-se um crescimento considerável das aplicações se comparadas aos desenvolvimentos teóricos. Esse período também corresponde ao início da utilização da otimização estrutural por grandes corporações industriais fora da área aeroespacial, dedicando esforços substanciais na conversão da teoria de otimização em benefícios concretos para o projeto.

Entretanto, ainda havia a necessidade de tornar os algoritmos mais robustos, aumentando o espectro de sua utilização e tornando a otimização topológica e global, ainda áreas de pesquisa, em ferramentas práticas de engenharia. Contudo, a análise de sensibilidade em otimização permanecia como a maior tendência nos desenvolvimentos técnicos de meados da década de 80, bem como a aplicação cada vez maior de técnicas de otimização de forma em projetos de componentes.

Programação Matemática

A programação matemática passou por um grande amadurecimento nas décadas de 60 e 70, sendo todos os algoritmos básicos praticamente desenvolvidos nesse período. Entretanto, tais

algoritmos eram direcionados principalmente para aplicações em pesquisa operacional, causando certas dificuldades para torná-los ferramentas eficientes e confiáveis para o projeto estrutural [5]. Os trabalhos de programação matemática desse período passam a dar atenção especial às características específicas dos problemas estruturais.

Em primeiro lugar, muitos parâmetros, tais como propriedades de material, critérios de falha e condições de carregamento não são conhecidos com grande precisão, de forma que a busca de um ponto ótimo matematicamente preciso significa muito pouco do ponto de vista da engenharia.

Segundo, o custo de uma análise de projeto pode ser muito alto, envolvendo a solução de sistemas de grandes dimensões. Dessa forma, a taxa de convergência de um algoritmo de otimização é muito importante: um método que converge rapidamente para um ótimo aproximado é quase sempre mais útil que um método que converge lentamente para um ótimo preciso.

BELEGUNDU e ARORA[14, 15], em 1985, analisam os algoritmos aplicados na otimização estrutural do ponto de vista teórico e numérico. Mostrou-se que não é possível relacionar a performance de um algoritmo em programação matemática com o seu comportamento em problemas de otimização de projetos.

A razão é que problemas de programação matemática consistem de funções explícitas, de pequena dimensão e não possuem um grande número de mínimos locais, sendo em geral triviais em termos do tempo computacional de uma análise. Dessa forma, em tais problemas normalmente se utiliza uma combinação de tempo de CPU, tempo de preparação, facilidade de uso, etc., como pontos principais de um código. Conseqüentemente um programa pode parecer muito bom, mesmo exigindo centenas de avaliações da função objetivo e das restrições para resolver um problema de apenas três ou quatro variáveis. O custo da aplicação de tal código num problema complexo de análise é proibitivo.

Isto sugere que, em problemas práticos de projeto, somente dois critérios têm sentido: em primeiro lugar, se o algoritmo e sua implementação são confiáveis em atingir um mínimo aproximado a partir de um ponto inicial arbitrário; e segundo, se o programa realiza o menor número possível de avaliações das funções do problema e de seus gradientes. Nesse sentido, LIM e ARORA[16] apresentam um método de programação quadrática com estratégia de estratégias ativas, ou seja, apenas são determinados os gradientes de restrições próximas de se tornarem ativas.

Os resultados desse trabalho chamam atenção para as características de convergência dos métodos de programação quadrática seqüencial e para os métodos de ponto interior, que gerando uma seqüência de pontos viáveis (ou seja, respeitando todas as restrições do problema) sempre fornecem uma solução para o problema.

Nesse contexto, pode-se citar o trabalho de HERSKOVITS[17, 18, 19], propondo um algoritmo de pontos interiores para otimização não-linear baseado na solução iterativa das equações de Karush-Kuhn-Tucker, com convergência superlinear devido à inclusão de informações de segunda ordem do problema. Por essas características, este algoritmo tem sido aplicado com sucesso em problemas estruturais [19, 20, 21, 22].

Otimização de Forma

O crescente interesse no projeto da forma ótima de componentes reflete não apenas a compreensão da importância da forma na performance estrutural, mas também reflete a crescente sofisticação da análise (em áreas como a geração automática de malhas e análise adaptável) e otimização estrutural, permitindo a solução de problemas complexos de obtenção de formas e dimensões ótimas.

Até esse período, a solução de problemas de otimização de forma utilizava as coordenadas de nós do modelo de elementos finitos como variáveis de projeto [23]. Por essa descrição estar associada ao modelo discretizado havia várias desvantagens: grande número de variáveis, dificuldades de se garantir a compatibilidade e a suavidade do contorno e, principalmente, dificuldade de manter uma malha adequada durante o processo iterativo de otimização. Dessa maneira, os trabalhos se direcionaram na busca de uma abordagem mais integrada das etapas de projeto, eliminando a dependência em relação à discretização do modelo [24].

DING[25], analisando as etapas de síntese estrutural identificou os procedimentos que devem compor uma ferramenta automatizada: descrição do modelo (parametrização), geração do modelo de elementos finitos, análise, verificação da precisão dos resultados, iteração de otimização e verificação da convergência.

Nos casos em que a forma é independente das variáveis, a verificação do modelo de elementos finitos pode ser feita apenas uma vez já que a malha não será modificada durante a otimização. No caso da otimização de forma, o modelo sofre alterações de maneira automatizada dentro de um processo iterativo, devendo-se ter um critério também automático para verificação da precisão do modelo de elementos finitos. Isso torna necessário a inclusão de recursos de análise adaptável e refinamento automático de malhas na etapa de análise.

KIKUCHI *et al*[26] discutem métodos adaptáveis de elementos finitos para a otimização de estruturas elásticas lineares, em termos do refinamento da malha e reposicionamento de nós. Assim, os procedimentos são na prática: descrição do modelo, geração automática da malha, análise, verificação do erro e refinamento (análise adaptável), otimização e verificação da convergência.

Torna-se então inviável associar características do projeto à malha, pois esta é modificada a cada iteração. Por esse motivo devem ser incluídos recursos de geração e parametrização de geometrias. DING[25] discute as vantagens da descrição do contorno através de splines em termos de flexibilidade, simplicidade, generalidade e precisão: mantém-se a suavidade e convexidade do contorno, permitem obter resultados mais precisos na análise de sensibilidade e exigem um número relativamente reduzido de variáveis na parametrização.

BENNET e BOTKIN[24] enfatizaram ainda a necessidade de associar todas as características do modelo, como funções de performance do problema, com o modelo geométrico. No caso dos algoritmos tradicionais de otimização não-linear deve-se considerar que estes permitem grandes variações dos parâmetros do problema fora do domínio de solução, podendo dar origem a um projeto tão inviável que pode comprometer a convergência da otimização (a utilização de algoritmos de ponto interior evita essa desvantagem).

HAFTKA e GRANDHI[27] apresentam, em 1986, uma revisão dos desenvolvimentos no campo da otimização de forma até a metade dessa década. Os autores também observam que uma otimização de forma satisfatória exige a incorporação de recursos de análise de sensibilidade aos programas de análise.

Devido à sua complexidade, as formulações genéricas de sensibilidade à forma somente passaram a estar disponíveis para estruturas contínuas a partir da metade da década de 80, não sendo surpresa que somente a partir daí a otimização de forma passa a ter grande desenvolvimento. Observa-se que a maioria desses estudos estavam restritos a problemas bidimensionais devido a limitações nos recursos de geração automática e análise adaptável da época.

Análise de Sensibilidade

A grande maioria dos artigos publicados até a metade dos anos 80 estavam restritos à aplicações de treliças e pórticos, nos quais a matriz de rigidez pode ser expressa ou aproximada

por uma função linear das variáveis de projeto, ao passo que aplicações de estruturas contínuas eram ainda relativamente raras. Uma das razões é a relação implícita entre os parâmetros de resposta estrutural (deslocamento, tensão, freqüências naturais) e as variáveis de projeto. No caso de otimização de forma, as relações são ainda mais complexas. Isso justifica o extenso uso de técnicas de diferenças finitas até meados da década, apesar de ser compreendido desde então que isso representava uma barreira para a eficiência e precisão dos resultados.

CHOI e HAUG[28] apresentam, em 1983, um dos primeiros trabalhos em análise de sensibilidade de estruturas elásticas. Cita-se também o trabalho de WANG, SUN e GALLAGHER[23] de 1985, com uma formulação da análise de sensibilidade à variação da forma de estruturas contínuas, baseada na diferenciação das equações discretizadas do contínuo para problemas estáticos. Observa-se que esse trabalho ainda utiliza as coordenadas dos nós do modelo de elementos finitos como variáveis de projeto. YANG e CHOI[29] utilizam uma parametrização suave do contorno e o método de contorno, ou seja, utilizando integrais de contorno. Em 1986 surgem trabalhos dedicados a formulações mais gerais da análise de sensibilidade[30].

Em 1986, HAUG, CHOI e KOMKOV[31] reúnem a base matemática da análise de sensibilidade de sistemas estruturais desenvolvida pelos autores durante a primeira metade da década. Esse trabalho apresenta tanto a formulação discreta da análise de sensibilidade (diferenciação das equações do contínuo já discretizadas) para problemas de otimização de parâmetros, quanto sua forma contínua, baseada na derivada material [32] das equações do contínuo para análises estática, de autovalor e transiente.

Deve-se citar também o método semi-analítico, que a partir das expressões do método discreto avalia as derivadas da matriz de rigidez e do vetor de carregamentos por diferenças finitas. A partir daí, os trabalhos se concentraram na aplicação desses conceitos em implementações computacionais [33, 34, 35, 36, 37]. Paralelamente busca-se extender a teoria de análise de sensibilidade para outros tipos de análise, principalmente não-lineares [38].

Alguns autores [34, 39] apontam vantagens da formulação analítica sobre a discreta: a formulação discreta e semi-analítica exige o conhecimento do código do programa de análise, em geral complexos e não disponíveis; são obtidas expressões analítica sem nenhuma discretização ou passo de perturbação, sendo particularmente indicado para a otimização de forma, a qual é muito sensível a problemas de precisão no cálculo das derivadas. A desvantagem da formulação contínua é que no caso de sensibilidade à forma se obtém integrais de contorno, justamente onde o método de elementos finitos não fornece resultados precisos. Dessa forma, devem-se converter as integrais de contorno em integrais de domínio. A precisão obtida na formulação contínua é analisada por YANG[34] para diversos exemplos de otimização de forma, comparando a formulação discreta e as formulações contínuas de contorno e domínio.

O método de domínio exige a definição do campo de velocidades de forma em todo o domínio. SEONG e CHOI[33] desenvolvem o método de camada de contorno para aumentar a eficiência da avaliação das integrais do método de domínio, definindo o campo de velocidades não nulo apenas em regiões do domínio adjacentes ao contorno.

YAO e CHOI[35] apresentaram dois métodos de determinação do campo de velocidades: gerando uma nova malha na geometria perturbada, tomando a diferença entre esta e a malha original como o campo de velocidades; e resolvendo um novo problema elástico com campo de velocidades no contorno como deslocamentos impostos.

RAJAN e BELEGUNDU[36], para esta finalidade, também formularam um novo problema elástico, aplicando carregamentos fictícios em uma estrutura auxiliar. Estes dois últimos métodos geram campos de velocidades de alta qualidade, podendo-se inclusive utilizar esta informação para reposicionar os nós da malha original devido à alteração da forma, evitando-se a necessidade de gerar uma nova malha. A desvantagem está no custo, pois exigem a solução de um novo

sistema linear e a avaliação de integrais em todo o domínio. O método de camada de contorno apresenta-se como alternativa de compromisso entre precisão e eficiência.

A avaliação dos erros envolvidos na determinação de gradientes é discutida por TSENG e ARORA[37], que discutem uma série de critérios para a verificação numérica da precisão da análise de sensibilidade.

HAFTKA[39] também aponta como alternativa a utilização do método de elementos de contorno [40, 41] nos cálculos de sensibilidade e otimização [42] devido a precisão de seus resultados no contorno e a facilidade na geração de malhas.

Em resumo, pode-se afirmar que as técnicas de análise de sensibilidade já estavam bem definidas ao final da década de 80.

Programas

O período também é marcado pela introdução de recursos de análise de sensibilidade em pacotes comerciais de análise por elementos finitos. Por exemplo, o método semi-analítico foi implementado em programas como Nastran e Eal. Utilizando a formulação contínua da análise de sensibilidade como ferramenta de pós-processamento, uma série de implementações foram desenvolvidas utilizando programas comerciais como Ansys[35, 39].

Em termos de implementações de algoritmos de programação não-linear para otimização estrutural pode-se citar o desenvolvimento dos programas Ads[43] e Conlin[44].

Livros

Como livros-texto desse período podem ser citados os trabalhos de KIRSCH[45], GILL *et al.*[46], MANGASARIAN *et al.*[47], HAUG *et al.*[31], ARORA[48], LUENBERG[49] e VANDERPLAATS[50].

1.2.3 Algumas Características Atuais

Após trinta anos de amadurecimento do conceito de síntese estrutural, a otimização estrutural, no início dos anos 90, já havia deixado os círculos restritos da pesquisa aeroespacial e acadêmica e ocupava posição de destaque nos setores automobilístico, civil e naval. Dessa forma, programas comerciais de análise iniciam a década incorporando métodos mais sofisticados de otimização e análise de sensibilidade [51, 52, 53, 54].

Na década de 90, ocorre uma crescente sofisticação e generalização dos recursos de otimização juntamente com a sofisticação das técnicas de análise, aproximando-os das situações reais de projeto: otimização de forma com geração automática de malhas em domínios tridimensionais [55], técnicas de otimização topológica [56], análise de sensibilidade e otimização de sistemas estruturais compostos [57] e de problemas não-lineares [21, 22]. Entretanto, a utilização cada vez maior da otimização em projetos, juntamente com a expansão dos recursos computacionais, tem dado origem a uma demanda cada vez maior por eficiência e robustez dos programas, motivando a pesquisa em implementação computacional.

VANDERPLAATS[58] analisa a evolução técnica da implementação computacional até 1993. Em sua maioria, as práticas correntes de programação, nas quais os módulos de otimização são adicionados a programas de análise, são barreiras graves ao aumento de eficiência dos programas de otimização. De fato, a maior integração entre os módulos de otimização e análise pode ser obtida por uma implementação satisfatória do cálculo de derivadas que, na realidade, constitui a interface entre os dois módulos. A desvantagem é que essa abordagem

exige uma reimplementação de códigos extensos e complexos dos programas de análise no caso de práticas tradicionais de programação.

Como solução para essa demanda, vários trabalhos têm sido realizados aplicando conceitos de programação orientada por objetos [59] em soluções de engenharia [21, 60, 61, 62, 63, 64]. Tal metodologia facilita a incorporação de uma série de características como mantenabilidade e confiabilidade aos programas, aliada a eficiência em seu desenvolvimento e modificação. Tais características são altamente necessárias em sistemas computacionais de engenharia, os quais exigem modificações e extensões freqüentes devido ao acelerado desenvolvimento da teoria e da prática [65]. De fato, a abordagem de orientação por objetos tem sido bastante bem sucedida em outras áreas. Porém, a maioria dos grandes sistemas para engenharia foi criada no início da evolução da engenharia de programas e não exploram os seus novos conceitos, especialmente a programação orientada ao objeto. Entretanto, isso não deve ser uma barreira para a implementação de novos programas.

No campo da programação matemática, ARORA *et al*[66] discutiram as características atrativas dos métodos de transformação (especialmente métodos de Lagrangeano Aumentado, que são mais eficientes que os SUMT's[1]) especialmente na solução de problemas complexos e de grandes dimensões. De fato, tais métodos são globalmente convergentes[2] e de implementação bastante simples, permitindo criar facilmente rotinas eficientes para uma dada aplicação e sendo então muito recomendados para problemas de otimização envolvendo restrições complicadas altamente não-lineares. ELWAKEIL e ARORA[67], em 1995, descreveram métodos de determinação de pontos viáveis em otimização sujeita a restrições.

Também se mostra, atualmente, interesse nos métodos de otimização global, que ainda permanecem como um desafio matemático e computacional. O mínimo global (de maneira diferente dos mínimos locais) não é caracterizado por nenhuma condição matemática, mas pode ser calculado computacionalmente ainda que com esforço substancial. Segundo ARORA, ELWAKEIL e CHAHANDE[68] os métodos não são gerais e originalmente foram desenvolvidos para problemas não restritos, reiterando o interesse nos métodos de transformação citado anteriormente. Uma característica importante é que os métodos de otimização global sempre são baseados em computação massiva, explicando a atração por esses métodos, já que potência computacional é cada vez mais disponível atualmente. Nesse campo, também têm sido aplicados os algoritmos genéticos, principalmente em problemas altamente não-convexos e nos casos não-diferenciáveis, pois esta classe de algoritmos não exige a avaliação de gradientes. Segundo HASLINGER *et al.*[69], os algoritmos genéticos são uma opção interessante em problemas de otimização de forma com tais características pois, além de serem métodos de otimização global, são relativamente simples e permitem tratar problemas de grande escala.

Recentemente, mostra-se preocupação em estimar o erro da discretização na avaliação da análise de sensibilidade, como no trabalho de FUENMAYOR *et al.*[70] que extende a estimador de ZIENKIEWICZ e ZHU[71] para gerar malhas que sejam mais adequadas à avaliação da sensibilidade.

Livros

Como livros-texto desse período podem ser citados os trabalhos de BAZARAA *et al.*[72], PERESSINI *et al.*[73], KODIYALAM e SAXENA[74] e RAO[75].

[1]SUMT: *Sequential Unconstrained Minimization Technique*

[2]Um método *globalmente convergente* sempre converge para algum mínimo local a partir de qualquer estimativa inicial. Observe que esta característica não têm relação com otimização global.

1.2.4 Conclusões

Pode-se afirmar que uma das tendências atuais é a busca de implementações mais eficientes e integradas dos algoritmos disponíveis. Isso, de fato, significa:

1. As implementações devem incorporar metodologias de programação que permitam fácil e rápida atualização e integração, incorporando ainda os algoritmos mais adaptados a classe de problemas em questão.

2. A implementação da análise de sensibilidade se constitui num dos pontos centrais na eficiência e precisão dos programas de otimização estrutural.

3. No projeto de componentes, a otimização de parâmetros não representa mais um problema maior já que os algoritmos e programas disponíveis podem tratar com relativa facilidade problemas de otimização usuais. Para componentes, o interesse atual se concentra na obtenção da forma ótima. Esse problema representa uma série de dificuldades além da própria programação matemática, como a representação da geometria, geração automática de malhas e análise de sensibilidade. A literatura aponta diversas soluções para cada um dos problemas, mas a questão principal é integrá-las de maneira eficiente num ambiente de projeto visando a otimização.

4. A eficiência global de um programa de otimização não depende somente do algoritmo de programação não-linear mas também do programa de análise (que deve incorporar técnicas de análise de sensibilidade) e da forma de comunicação entre eles. A implementação deve visar uma integração máxima entre o otimizador e o programa de análise sem comprometer, entretanto, a modulação.

5. Em termos dos algoritmos de programação não-linear, a tendência é utilizar algoritmos mais genéricos. Dessa forma, o desafio é torná-los mais eficientes. É claro que não existe um algoritmo de minimização que seja ótimo para todos os tipos de problemas, mas é desejável que uma implementação seja eficiente para o maior número possível de casos de uma certa classe de problemas. De fato, como o problema de otimização é altamente geral, algoritmos muito especializados podem cair em desuso rapidamente.

1.3 Definições Matemáticas

1.3.1 Problema de Elasticidade Linear

Considere um corpo elástico ocupando a região $\mathcal{B} \subset \Re^3$. Convenciona-se identificar pontos do corpo com suas posições em $\mathcal{B}$, ou seja, $\mathbf{x} \in \mathcal{B}$ se e somente se $\mathbf{x}$ pertence ao corpo. Submetendo o corpo $\mathcal{B}$ a um conjunto de ações externas, podem-se produzir alterações em sua condição de equilíbrio e em sua geometria, definindo um problema geral de movimento de corpos deformáveis. No caso de corpos elásticos em equilíbrio estático, cujos gradientes dos campos de deslocamento sejam pequenos o suficiente para que a hipótese de linearização da equação constitutiva do material seja válida, tem-se problemas elastostáticos lineares. O sistema de equações de campo descrevendo o comportamento estático de um corpo elástico dentro dos limites da teoria linear consiste de três equações:

- a *relação deformação-deslocamento*

$$\mathbf{E} = \frac{1}{2}\left(\nabla u + \nabla^T u\right)$$

- a *relação tensão-deformação para pequenas deformações*

$$\mathbf{T} = \mathsf{C}[\mathbf{E}]$$

- e a *equação de equilíbrio*

$$\text{Div } \mathbf{T} + \mathbf{b} = \mathbf{0}$$

onde u é o campo de deslocamentos produzido pelas ações externas, $\mathbf{E}$ é o tensor de deformações, $\mathbf{T}$ é o tensor de tensões, C é o tensor de elasticidade e $\mathbf{b}$, o campo vetorial de forças de volume em $\mathcal{B}$.

Dessa forma, o problema misto de elastostática pode ser colocado como,

Determinar o campo de deslocamentos u tal que

$$\begin{cases} \mathbf{E} = \dfrac{1}{2}\left(\nabla u + \nabla^T u\right) = \nabla u^S \\ \mathbf{T} = \mathsf{C}[\mathbf{E}] \\ \text{Div } \mathbf{T} + \mathbf{b} = \mathbf{0} \end{cases} \tag{1.1}$$

com condições de contorno em $\partial\mathcal{B}$,

$$\begin{array}{ll} u \equiv \mathbf{0} & \mathbf{x} \in \Gamma^1 \\ \mathbf{Tn} \equiv \mathbf{\Phi} & \mathbf{x} \in \Gamma^2 \end{array} \tag{1.2}$$

sendo $\partial\mathcal{B} = \Gamma^0 \cup \Gamma^1 \cup \Gamma^2$ e $\overset{\circ}{\Gamma^0} \cap \overset{\circ}{\Gamma^1} \cap \overset{\circ}{\Gamma^2} = \emptyset$; Γ^0 é uma região do contorno $\partial\mathcal{B}$ sem condições especificadas e $\mathbf{n}$ o vetor unitário normal em cada ponto de $\partial\mathcal{B}$.

Neste caso, a relação constitutiva do material é dada pela lei de Hooke

$$\mathbf{T} = \mathsf{C}[\mathbf{E}] = 2\tilde{\mu}\mathbf{E} + \tilde{\lambda}\,(\text{tr } \mathbf{E})\,\mathbf{I}$$

sendo,

$$\tilde{E} = \frac{\tilde{\mu}\left(2\tilde{\lambda} + 3\tilde{\lambda}\right)}{\tilde{\mu} + \tilde{\lambda}} \qquad \tilde{\nu} = \frac{\tilde{\lambda}}{2\left(\tilde{\mu} + \tilde{\lambda}\right)}$$

respectivamente, o módulo de elasticidade e o coeficiente de Poisson do material do corpo em função dos coeficientes de Lamé $\tilde{\lambda}$ e $\tilde{\mu}$. O estado elástico $[u, \mathbf{E}, \mathbf{T}]$ satisfazendo as equações de contorno é uma solução do problema.

Devido à complexidade da equação diferencial do problema elastostático, a solução analítica é possível apenas para problemas com geometrias e condições de contorno bastante simples. No entanto, com o auxílio da formulação variacional, métodos numéricos podem ser introduzidos, possibilitando obter soluções aproximadas para uma grande variedade de problemas. Utilizando o Princípio dos Trabalhos Virtuais a partir de considerações físicas sobre o corpo ou uma técnica de resíduos ponderados aplicada à equação diferencial do problema, obtem-se o seguinte enunciado,

$$\int_{\mathcal{B}} \mathbf{T}(u) \cdot \mathbf{E}(v)\; dV = \int_{\partial\mathcal{B}} \mathbf{T}(u)\,\mathbf{n} \cdot v\; dA + \int_{\mathcal{B}} \mathbf{b} \cdot v\; dV$$

sendo $\mathbf{T}(u)$ o tensor de tensões associado ao campo de deslocamentos reais u; e $\mathbf{E}(v)$ o tensor de deformações infinitesimais associado ao campo de deslocamentos virtuais v. Juntamente com as condições de contorno (1.2), a equação anterior leva ao seguinte *problema variacional* cuja solução é também solução do problema original (1.1) sujeito às condições de contorno (1.2):

Determinar o campo de deslocamentos $u \in \mathcal{V}$ tal que

$$\int_{\mathcal{B}} \mathbf{T}(u) \cdot \mathbf{E}(v)\; dV = \int_{\Gamma^2} \mathbf{\Phi} \cdot v\; dA + \int_{\mathcal{B}} \mathbf{b} \cdot v\; dV \qquad \forall v \in \mathcal{V} \tag{1.3}$$

ou de forma resumida,

$$a_{\mathbf{p}}(u,v) = l_{\mathbf{p}}(v) \qquad \forall v \in \mathcal{V} \tag{1.4}$$

onde $a_{\mathbf{p}} : \mathcal{V} \times \mathcal{V} \to \Re$; $l_{\mathbf{p}} : \mathcal{V} \to \Re$, ou seja, $l_{\mathbf{p}} \in \mathcal{V}'$ dual de $\mathcal{V}$, *espaço dos deslocamentos cinematicamente admissíveis*. No contexto dos problemas estruturais, a equação acima é chamada de *equação de estado*. No contexto matemático, diz-se que esta equação é a *forma fraca* da equação diferencial do problema e neste caso o espaço de Hilbert $\mathcal{V}$ é o espaço de funções onde as condições do problema variacional e de contorno são respeitadas. De fato, os problemas de mecânica em geral, e não só os casos elastostáticos lineares, podem ser expressos na forma do problema variacional[3] (1.4).

Nos problemas de elasticidade linear, $a_{\mathbf{p}}(u,v)$ é um operador bilinear, limitado, coercivo e simétrico, e $l_{\mathbf{p}}(v)$ um operador linear limitado, segundo a norma do espaço $\mathcal{V}$. Desta maneira, respeitam-se então as hipóteses do teorema de Lax-Milgram, garantindo-se a existência e a unicidade da solução u. O subscrito $\mathbf{p}$ indica a dependência em relação a um vetor de parâmetros reais, que nada mais é que o vetor de variáveis de um projeto parametrizado: a cada valor de $\mathbf{p}$, tem-se um novo projeto, definindo um problema variacional diferente, de onde se conclui que a solução u é um campo vetorial tal que $u = u(\mathbf{x}, \mathbf{p})$.

No caso da otimização estrutural, é necessário definir funcionais de performance que dependem de $\mathbf{p}$ através de uma dependência geralmente implícita da solução u do problema variacional (1.4). Nestes problemas, além do valor dos funcionais, também se requer avaliar os gradientes dos funcionais. Entretanto, como em geral não se conhece a forma explícita destes funcionais e u somente é conhecido de maneira discreta, exige-se uma formulação que permita avaliar tais gradientes de forma numérica. A formulação de análise de sensibilidade fornece a base teórica e computacional para o cálculo dos gradientes de funcionais que dependem de u, de maneira mais eficiente que aplicar diretamente uma técnica de diferenças finitas.

1.3.2 Discretização do Problema de Elasticidade Linear

O teorema de Lax-Milgram apenas garante a existência e a unicidade da solução u do problema variacional (1.4), não fornecendo, entretanto, um procedimento para a sua obtenção. Aplica-se então o método de Galerkin para resolver problemas deste tipo.

Método de Galerkin

Seja $\mathcal{V}$ um espaço de Hilbert real separável. Sejam $a_{\mathbf{p}} : \mathcal{V} \times \mathcal{V} \to \Re$ e $l_{\mathbf{p}} \in \mathcal{V}'$ como antes; o conjunto $\{\phi_i\}_{i=1}^{\infty}$ formando uma base de Schauder para $\mathcal{V}$ e $\mathcal{V}_N = \text{span}\,[\phi_1, \ldots, \phi_N]$. Considere $a_{\mathbf{p}} : \mathcal{V}_N \times \mathcal{V}_N \to \Re$ ($a_{\mathbf{p}}$ restrita a $\mathcal{V}_N$) forma bilinear, limitada e coerciva e $l_{\mathbf{p}} : \mathcal{V}_N \to \Re$ ($l_{\mathbf{p}}$ restrita a $\mathcal{V}_N$) linear e contínua.

Pode-se então considerar o problema aproximado de determinar $u^{(N)} \in \mathcal{V}_N$ solução do problema

$$a_{\mathbf{p}}\left(u^{(N)}, v\right) = l_{\mathbf{p}}(v) \qquad \forall v \in \mathcal{V}_N \tag{1.5}$$

que, da mesma maneira que (1.4), também satisfaz as condições do teorema de Lax-Milgram, existindo, portanto, um único $u^{(N)}$.

Como $u^{(N)} \in \mathcal{V}_N$ então,

$$u^{(N)} = \sum_{i=1}^{N} u_i^{(N)} \phi_i \tag{1.6}$$

sendo necessário apenas determinar os coeficientes (escalares) $u_i^{(N)}$, $i = 1, \ldots, N$.

[3] Utilizando o enunciado mais geral do Princípio das Potências Virtuais.

Além disso, observa-se que (1.5) é equivalente a

$$a_{\mathbf{p}}\left(u^{(N)}, \phi_j\right) = l_{\mathbf{p}}\left(\phi_j\right) \qquad \forall j = 1, \ldots, N \tag{1.7}$$

Substituindo (1.6) em (1.7), obtem-se que,

$$\sum_{i=1}^{N} u_i^{(N)} a_{\mathbf{p}}\left(\phi_i, \phi_j\right) = l_{\mathbf{p}}\left(\phi_j\right) \qquad \forall j = 1, \ldots, N \tag{1.8}$$

Tem-se então um sistema linear $N \times N$ com a seguinte forma,

$$\begin{bmatrix} a_{\mathbf{p}}(\phi_1,\phi_1) & a_{\mathbf{p}}(\phi_2,\phi_1) & \cdots & a_{\mathbf{p}}(\phi_N,\phi_1) \\ a_{\mathbf{p}}(\phi_1,\phi_2) & a_{\mathbf{p}}(\phi_2,\phi_2) & \cdots & a_{\mathbf{p}}(\phi_N,\phi_2) \\ \vdots & \vdots & \ddots & \vdots \\ a_{\mathbf{p}}(\phi_1,\phi_N) & a_{\mathbf{p}}(\phi_2,\phi_N) & \cdots & a_{\mathbf{p}}(\phi_N,\phi_N) \end{bmatrix} \begin{Bmatrix} u_1^{(N)} \\ u_2^{(N)} \\ \vdots \\ u_N^{(N)} \end{Bmatrix} = \begin{Bmatrix} l_{\mathbf{p}}(\phi_1) \\ l_{\mathbf{p}}(\phi_2) \\ \vdots \\ l_{\mathbf{p}}(\phi_N) \end{Bmatrix} \tag{1.9}$$

ou, em forma reduzida, $\mathbf{Ku} = \mathbf{f}$.

A solução deste sistema de equações (única de acordo com o teorema de Lax-Milgram) fornece o vetor de coeficientes $\mathbf{u} = \begin{bmatrix} u_1^{(N)} & u_2^{(N)} & \cdots & u_N^{(N)} \end{bmatrix}^T$ e conseqüentemente $u^{(N)}$ a partir de (1.6).

Resta analisar a qualidade da solução aproximada obtida. Observa-se que dados,

$$a_{\mathbf{p}}(u, v) = l_{\mathbf{p}}(v) \qquad \forall v \in \mathcal{V}$$

$$a_{\mathbf{p}}\left(u^{(N)}, v\right) = l_{\mathbf{p}}(v) \qquad \forall v \in \mathcal{V}_N$$

obtem-se pela subtração que,

$$a_{\mathbf{p}}\left(u - u^{(N)}, v\right) = 0 \qquad \forall v \in \mathcal{V}_N$$

Portanto, no caso em que $a_{\mathbf{p}}$ é simétrico[4], vê-se que $u^{(N)}$ é exatamente a projeção ortogonal de u sobre $\mathcal{V}_N$ com respeito à norma[5] definida por $a_{\mathbf{p}}(\cdot,\cdot)$. Assim, $u^{(N)}$ é a melhor aproximação possível em $\mathcal{V}_N$ medida nesta norma, ou seja

$$\left\|u - u^{(N)}\right\|_{a_{\mathbf{p}}} \leq \|u - v\|_{a_{\mathbf{p}}} \qquad \forall v \in \mathcal{V}_N \tag{1.10}$$

Além disso, como $\{\phi_i\}_{i=1}^{\infty}$ forma uma base de Schauder, então existem $\left\{\alpha_i^{(N)}\right\}_{i=1}^{\infty}$ tais que $\sum_{i=1}^{N} \alpha_i^{(N)} \phi_i \to u$ quando $N \to \infty$. Assim, a partir da relação (1.10) tem-se,

$$\left\|u - u^{(N)}\right\|_{a_{\mathbf{p}}} \leq \left\|u - \sum_{i=1}^{N} \alpha_i^{(N)} \phi_i\right\|_{a_{\mathbf{p}}} \to 0 \qquad \text{quando } N \to \infty$$

implicando que $u^{(N)} \to u$ quando $N \to \infty$.

É claro que a forma da matriz $\mathbf{K}$ e do vetor $\mathbf{f}$ depende da escolha da base $\{\phi_i\}_{i=1}^{\infty}$ para o espaço $\mathcal{V}$. Diferentes escolhas são possíveis, conduzindo a diferentes métodos numéricos. No caso do Método de Elementos Finitos, adotado neste trabalho para a solução do problema elastostático, o domínio $\mathcal{B}$ é divido em subdomínios elementares $\mathcal{B}_e$ e as funções da base são polinômios por partes.

[4]Uma forma bilinear coerciva e simétrica é um *produto interno* do espaço $\mathcal{V}$. Assim, sendo um produto interno, se $a(w, v) = 0$, $\forall v$, então $w = 0$.

[5]Uma forma bilinear com estas características define a norma $\|u\|_{a_{\mathbf{p}}} = \sqrt{a_{\mathbf{p}}(u, u)}$.

1.3.3 Problema de Otimização

Modelo Matemático Geral

Apesar da forma bastante geral em que se pode apresentar um problema de otimização, define-se um modelo matemático geral englobando um grande número de possibilidades, da seguinte forma,

$$\begin{aligned} &\min\ f(\mathbf{p}) \\ &\text{sujeita à} \\ &\qquad g_i(\mathbf{p}) \leq 0 \qquad i = 1, 2, \ldots, m \\ &\qquad h_j(\mathbf{p}) = 0 \qquad j = 1, 2, \ldots, p \end{aligned} \tag{1.11}$$

sendo $\mathbf{p} \in \Re^n$, $f : \Re^n \longrightarrow \Re$, $g_i : \Re^n \longrightarrow \Re$ e $h_j : \Re^n \longrightarrow \Re$ $\forall i, j \in \aleph$. Em geral, o interesse está em situações onde uma ou mais dessas funções são não-lineares.

Observa-se que $f(\mathbf{p})$ é a *função objetivo* do problema. As inequações $g_i(\mathbf{p}) \leq 0$ são chamadas *restrições de desigualdade.* Pode-se ter um número m qualquer de restrições de desigualdade, sendo simples observar que qualquer desigualdade pode ser colocada nessa forma. Assumindo que o problema tenha alguma solução $\mathbf{p}^* \in \Re^n$, as restrições nesse ponto podem ser tanto satisfeitas na igualdade (restrições *ativas*) quanto na desigualdade (restrições *inativas*). As equações $h_j(\mathbf{p}) = 0$ são denominadas *restrições de igualdade.* Num problema de otimização, deve-se ter sempre $p < n$. No caso de $p = n$, a única solução possível é aquela obtida do sistema de n equações e n incógnitas. Se $p > n$ o problema é sobredeterminado, podendo existir equações redundantes ou inexistir solução.

A região Ω do domínio onde todas as restrições são satisfeitas simultaneamente é chamada *região factível* ou *de viabilidade*, ou seja,

$$\Omega = \{\mathbf{p} \in \Re^n | g_i(\mathbf{p}) \leq 0,\ i = 1, 2, \ldots, m \text{ e } h_j(\mathbf{p}) = 0,\ j = 1, 2, \ldots, p\} \tag{1.12}$$

Dessa forma, resolver um problema de otimização significa determinar o ponto $\mathbf{p}^* \in \Omega$ no qual $f(\mathbf{p})$ tenha valor mínimo. Se $\mathbf{p}^*$ existir, denomina-se o mesmo como *ponto de mínimo*, *ponto ótimo* ou *projeto ótimo* no contexto de engenharia.

É importante observar que uma vez reduzidos à forma padrão, todos os problemas de otimização se tornam semelhantes. Dessa forma, os mais variados problemas podem ser analisados de maneira unificada.

Funcionais de Performance Estrutural

Nos problemas estruturais deste trabalho, os funcionais f, g_i e h_j são escolhidos dentre os seguintes:

Massa O funcional definindo a massa do corpo é dado por,

$$\psi_1 = \int_{\mathcal{B}} \tilde{\rho}(\mathbf{x})\ dV \tag{1.13}$$

onde $\tilde{\rho}(\mathbf{x})$ é o campo escalar de densidade do corpo.

Tensão Define-se a função de média $m_r(\mathbf{x})$ como uma função não nula apenas em uma região $\mathcal{B}_r \subseteq \mathcal{B}$ com integral unitária sobre o domínio da seguinte forma,

$$m_r(\mathbf{x}) = \begin{cases} 1/\int_{\mathcal{B}_r} dV & \mathbf{x} \in \mathcal{B}_r \\ 0 & \mathbf{x} \notin \mathcal{B}_r \end{cases}$$

Considere um funcional de tensão $\mathcal{G}(\mathbf{T})$, $\mathbf{T} = \mathbf{T}(u(\mathbf{x}))$, envolvendo o critério de falha relevante do problema, como tensões principais, von Mises, Tresca, etc. Em geral, utiliza-se o seguinte funcional de tensão média na região $\mathcal{B}_r \subseteq \mathcal{B}$,

$$\psi_2 = \int_{\mathcal{B}} \mathcal{G}(\mathbf{T})\, m_r(\mathbf{x})\ dV = \frac{\int_{\mathcal{B}_r} \mathcal{G}(\mathbf{T})\ dV}{\int_{\mathcal{B}_r} dV} \tag{1.14}$$

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

$$\psi_3 \equiv |u(\mathbf{x}_k)| = \int_{\mathcal{B}} \mathcal{G}(u)\, \delta(\mathbf{x} - \mathbf{x}_k)\ dV \tag{1.15}$$

sendo $\mathbf{x}_k$ as coordenadas do nó onde o deslocamento precise ser controlado ($\mathbf{x}_k$ poderia ser, por exemplo, as coordenadas do nó de deslocamento resultante máximo). Também se pode definir o funcional de deslocamento de um determinado grau de liberdade,

$$\psi_3 \equiv |u_x(\mathbf{x}_k)| = \int_{\mathcal{B}} |u_x(\mathbf{x})|\, \delta(\mathbf{x} - \mathbf{x}_k)\ dV \qquad \mathcal{G}(u) = |u_x(\mathbf{x})|$$

Energia de Deformação A energia de deformação de $\mathcal{B}$ sujeito ao estado elástico $[u, \mathbf{T}, \mathbf{E}]$ é definida pelo seguinte funcional

$$\psi_4 = \frac{1}{2}\int_{\mathcal{B}} \mathbf{T}(u) \cdot \mathbf{E}(u)\ dV = \frac{1}{2}\int_{\mathcal{B}} \mathsf{C}\left[\nabla u^S\right] \cdot \nabla u^S\ dV \tag{1.16}$$

1.4 Implementação

Apesar da metodologia de Programação Orientada por Objetos ser indicada para o tratamento de programas extensos, ainda há relativamente poucas aplicações em engenharia. Isto se deve parcialmente ao fato das primeiras implementações de linguagens orientadas por objetos comprometerem seriamente a eficiência. Entretanto, linguagens como C++ têm performance igual às linguagens procedurais tradicionais. Esse trabalho visa apresentar os conceitos e os resultados obtidos na aplicação de orientação por objetos no desenvolvimento de um ambiente de otimização estrutural em problemas lineares.

O desenvolvimento de programas em engenharia exige a necessidade de combinar o trabalho de um razoável número de pessoas e a incorporação freqüente de novas técnicas. Na programação tradicional, existe grande dificuldade de conciliar produtividade e qualidade no desenvolvimento de programas, pois a estrutura de dados subjacente do programa é manipulada diretamente. As estruturas de dados exigidas para o armazenamento e manipulação de dados de maneira eficiente são geralmente complexas e sua mínima alteração exige a revisão de todo o código na procura de dependências. A programação orientada por objetos introduz a característica de encapsulamento de informações no desenvolvimento de tipos abstratos de dados, realizado em C++ através da construção de classes. O encapsulamento permite controlar a visibilidade dos dados, impedindo o acesso direto à estrutura de dados subjacente. Neste caso, acesso e manipulação dos dados somente são feitas por envio de mensagens, através da interface pública formada por funções associadas aos dados. Essa característica permite que sejam implementadas estruturas de dados eficientes mantendo uma interface de utilização simples.

Um ambiente de otimização estrutural geralmente é desenvolvido através da incorporação de módulos de análise de sensibilidade e de otimização a um programa de análise estrutural

já existente. A eficiência desse ambiente depende não só da implementação de cada um dos módulos, mas também da forma como tais módulos trocam informações durante o processo de otimização. Através da formulação contínua da análise de sensibilidade, obtém-se um algoritmo de cálculo de gradientes, o qual não requer detalhes da implementação do programa de análise estrutural. Tal algoritmo é então implementado como uma classe, sendo o modelo de elementos finitos do problema um dos dados dessa classe. Assim, tem-se acesso as características do modelo e da análise mantidos na memória, ao invés da solução usual de trocar informações através de arquivos de dados. O algoritmo de otimização é implementado através da derivação da classe de análise de sensibilidade. Cada tipo de funcional de performance é implementado através de herança de uma classe genérica de funcionais. Assim, o acesso às funções de avaliação é feita de maneira única, independente do tipo de funcional acessado. Isso também permite a incorporação de novos tipos de funcionais sem exigir alterações.

1.5 Objetivos

O objetivo principal deste trabalho foi desenvolver uma base de programas para a otimização de parâmetros e de forma em problemas lineares, considerando exemplos de elasticidade linear. Para isso, considerou-se como algoritmo de minimização o procedimento de pontos interiores de Herskovits[17, 18, 19]. Para o cálculo dos gradientes de funcionais envolvidos, partiu-se da análise de sensibilidade contínua desenvolvida em [31]. Foram empregados os conceitos de programação por objetos através da linguagem C++ para o desenvolvimento dos programas, permitindo obter ganhos consideráveis em termos de eficiência, modulação e portabilidade do código.

O texto a seguir apresenta os principais pontos do trabalho. No Capítulo 2, considera-se um revisão do problema de otimização e procedimentos numéricos para a solução do problema de mínimo [48], destacando o algoritmo de pontos interiores de Herskovits. Nos Capítulos 3 e 4, abordam-se, respectivamente, as formulações contínua da análise de sensibilidade à parâmetros e à forma. Finalmente, no Capítulo 5 tem-se aspectos da implementação de classes para os algoritmos descritos. Os resultados da aplicação dos procedimentos são apresentados ao final de cada capítulo.

Capítulo 2

Métodos Numéricos para Otimização

Uma vez estabelecidos os funcionais de performance e as variáveis de projeto definindo o problema de otimização, devem-se determinar as estratégias que permitam determinar a solução. Assumindo que os funcionais do problema possuem derivadas contínuas, as condições necessárias de Karush-Kuhn-Tucker estabelecem os requisitos a serem satisfeitos em pontos estacionários de funcionais sujeitos a um conjunto de restrições. Como os problemas práticos são geralmente não-lineares, de grandes dimensões e implícitos das variáveis de projeto, é necessária a aplicação de métodos numéricos de solução. Entretanto, como os problemas de otimização estrutural têm características que os tornam diferentes dos problemas provenientes da programação matemática, devem ser considerados critérios específicos para a escolha dos algoritmos.

Assim, o objetivo desse capítulo é estabelecer as condições de otimalidade, analisar as características de alguns métodos numéricos a partir de sua formulação, apresentando um algoritmo com um comportamento razoável na solução de problemas estruturais e implementar um módulo de otimização de projetos. Deve-se deixar claro, entretanto, que a função do módulo de otimização é apenas gerar um novo projeto a partir dos valores dos funcionais e gradientes no projeto corrente. Já os cálculos destes valores são atribuições respectivamente dos módulos de análise e de análise de sensibilidade, que serão tratados posteriormente.

2.1 Condições de Otimalidade

2.1.1 Condições Necessárias de Karush-Kuhn-Tucker

Após formular o problema geral de otimização (1.11), deve-se analisar a existência ou não de soluções. As condições que devem ser satisfeitas no ponto ótimo são chamadas *condições necessárias de otimalidade*. Em outras palavras, qualquer ponto que não satisfaça as condições de otimalidade não pode ser um ponto ótimo. Entretanto, as condições necessárias fornecem apenas pontos candidatos a solução. As *condições suficientes*, por sua vez, fornecem subsídios para julgar se os pontos candidatos são realmente soluções do problema. Os pontos satisfazendo simultaneamente as condições necessárias e suficientes serão possíveis soluções do problema de otimização[1]. Entretanto, as condições de otimalidade permitem analisar apenas a existência de *mínimos locais*. A determinação de uma solução global é mais complexa, se baseando apenas em estratégias computacionais pois não existe caracterização matemática geral de pontos de mínimo global.

As condições de otimalidade podem ser aplicadas de duas formas:

[1]Entretanto, se as condições suficientes não forem satisfeitas ou não puderem ser aplicadas não será possível concluir se o ponto é ou não ótimo.

1. Dado um ponto, as condições de otimalidade podem ser usadas para verificar se este é ou não um candidato a ótimo.

2. As condições de otimalidade podem ser resolvidas diretamente para determinar os pontos candidatos a mínimo.

A seguinte definição é usada no enunciado das condições de otimalidade.

Definição 2.1 ***Ponto Regular.*** *Um ponto* $\mathbf{p}^* \in \Omega$ (equação (1.12)) *é dito regular se os vetores gradientes das restrições de igualdade e de desigualdade ativas forem linearmente independentes em* $\mathbf{p}^*$.

Teorema 2.1 ***Condições necessárias de Karush-Kuhn-Tucker (primeira ordem).*** *Seja* $\mathbf{p}^* \in \Omega$ (1.12) *um ponto regular que é ponto mínimo de um problema de otimização na forma geral* (1.11)

$$\begin{aligned} &\min\ f(\mathbf{p}) \\ &\textit{sujeita à} \\ &\qquad g_i(\mathbf{p}) \leq 0 \qquad i = 1, 2, \ldots, m \\ &\qquad h_j(\mathbf{p}) = 0 \qquad j = 1, 2, \ldots, p \end{aligned}$$

Logo devem existir multiplicadores de Lagrange $\boldsymbol{\lambda}^* \in \Re^m$ *e* $\boldsymbol{\mu}^* \in \Re^p$ *tais que o* ***Lagrangeano*** *do problema, definido por*

$$L(\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{p}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{p}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{p}) = f(\mathbf{p}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{p}) + \boldsymbol{\mu}^T \mathbf{h}(\mathbf{p}) \tag{2.1}$$

seja estacionário ($\nabla_{\mathbf{p}} L = \mathbf{0}$) *com respeito a* $\mathbf{p}^*$, $\boldsymbol{\lambda}^*$ *e* $\boldsymbol{\mu}^*$, *ou seja,*

$$\frac{\partial L}{\partial p_j} \equiv \frac{\partial f(\mathbf{p}^*)}{\partial p_j} + \sum_{i=1}^{m} \lambda_i^* \frac{\partial g_i(\mathbf{p}^*)}{\partial p_j} + \sum_{i=1}^{p} \mu_i^* \frac{\partial h_i(\mathbf{p}^*)}{\partial p_j} = 0 \qquad j = 1, 2, \ldots, n \tag{2.2}$$

$$g_i(\mathbf{p}^*) \leq 0 \qquad i = 1, 2, \ldots, m \tag{2.3}$$

$$\lambda_i^* g_i(\mathbf{p}^*) = 0 \qquad i = 1, 2, \ldots, m \tag{2.4}$$

$$h_i(\mathbf{p}^*) = 0 \qquad i = 1, 2, \ldots, p \tag{2.5}$$

e

$$\lambda_i^* \geq 0 \qquad i = 1, 2, \ldots, m \tag{2.6}$$

As equações (2.2), (2.3), (2.4) e (2.5) definem um sistema de $n + 2m + p$ equações e $n + 2m + p$ incógnitas: $\mathbf{p}^* \in \Re^n$, $\boldsymbol{\lambda}^* \in \Re^m$, $\boldsymbol{\mu}^* \in \Re^p$e os m valores $g_i(\mathbf{p}^*)$ que também são incógnitas de (2.4). As condições de Karush-Kuhn-Tucker são resolvidas da seguinte forma:

1. Primeiro solucionam-se as equações (2.4), denominadas *condições de ativação.* Tais condições são facilmente resolvidas anulando ora os multiplicadores λ_i, ora as funções g_i. Este procedimento fornece os diversos casos de solução (2^m casos *normais* mais o caso trivial onde $\lambda_i = 0$ e $g_i = 0$) nos quais as equações (2.2), (2.3) e (2.5) serão resolvidas. As condições de ativação simplesmente fornecem todas as combinações possíveis de restrições de desigualdade ativas e inativas[2].

[2]De fato, estas condições apenas enunciam a conclusão de que somente as restrições ativas são relevantes para a solução do problema. Contudo, durante a formulação de um problema geral de otimização, nunca se sabe *a priori* quais restrições de desigualdade serão ou não ativas, sendo portanto introduzidas mais restrições do que necessário. Em problemas de menor escala, isso pode não ser um problema grave, mas no caso de problemas de grandes dimensões (tratados numericamente), de um total de várias centenas de restrições, apenas algunas dezenas serão ativas na solução.

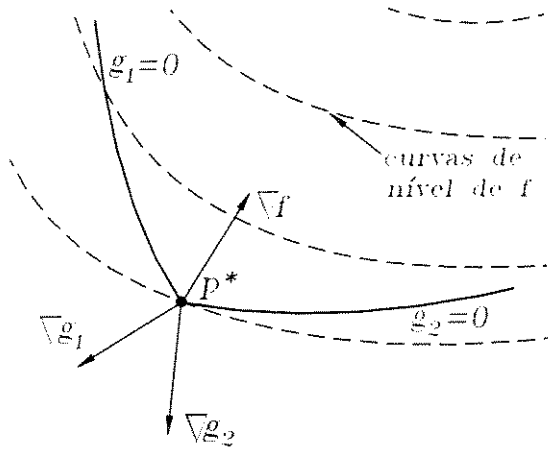


Figura 2.1: Significado geométrico das condições de Karush-Kuhn-Tucker

2. Resolve-se o sistema formado por (2.2), (2.3) e (2.5) para cada caso de solução obtido no passo 1. Deve-se observar que o sistema pode ou não ter solução nestes casos.

3. Verifica-se a condição (2.6) para o conjunto de soluções calculado no passo 2. Soluções com todos $\lambda_i^* \geq 0$ não podem se máximos locais.

As condições de Karush-Kuhn-Tucker têm um importante significado geométrico, como ilustrado na Figura 2.1. Reescrevendo (2.2), tem-se,

$$-\nabla_{\mathbf{p}} f(\mathbf{p}^*) = \sum_{i=1}^{m} \lambda_i^* \nabla_{\mathbf{p}} g_i(\mathbf{p}^*) + \sum_{i=1}^{p} \mu_i^* \nabla_{\mathbf{p}} h_i(\mathbf{p}^*) \tag{2.7}$$

A equação (2.7) mostra que no ponto estacionário, o oposto do gradiente da função objetivo (direção de decréscimo máximo) é uma combinação linear dos gradientes das restrições ativas, sendo os multiplicadores de Lagrange os coeficientes dessa combinação. Assim, pode-se observar que qualquer ação no sentido de minimizar ainda mais a função f violaria necessariamente alguma restrição. Dessa forma, $f(\mathbf{p}^*)$ é o menor valor possível da função $f(\mathbf{p})$ sujeita às restrições impostas.

As seguintes considerações devem ser feitas acerca das condições necessárias de Karush-Kuhn-Tucker de primeira ordem [48]:

1. Os pontos satisfazendo as condições (2.2) a (2.5) podem ser restritos ou irrestritos. Os pontos são irrestritos se não houver restrições de igualdade e as restrições de igualdade forem todas inativas. Tais pontos podem ser mínimos locais, máximos locais ou pontos de inflexão.

2. Se houver restrições de igualdade e as restrições de desigualdade forem todas inativas, então os pontos satisfazendo as condições (2.2) a (2.5) são apenas estacionários. Podem ser mínimos locais, máximos locais ou pontos de inflexão.

3. Se alguma restrição de desigualdade é nula e seu multiplicador é positivo de acordo com (2.6), então os pontos satisfazendo as condições (2.2) a (2.5) não podem ser máximos da função objetivo. Tais pontos, entretanto, também podem não ser mínimos do problema. Isso vai depender das condições suficientes.

2.1.2 Condições Suficientes de Otimalidade

A solução das condições necessárias (2.2) a (2.5) fornece um conjunto de pontos candidatos que devem ser analisados em busca dos pontos de mínimo. Nesse sentido se coloca o teorema de Weierstrass, que garante a existência de um mínimo global para a função $f(\mathbf{p})$, se esta é contínua e a região factível Ω é fechada e limitada. Como vários problemas distintos podem ser formulados através de um modelo geral de otimização não se pode garantir a continuidade da função $f(\mathbf{p})$ para todas as situações.

Entretanto, no contexto de problemas estruturais, onde as hipóteses de continuidade do material sejam válidas, funções envolvendo relações de grandezas como volume, deslocamento, tensão, freqüência natural, propriedades de material e carregamento, apresentam continuidade em alguma região do domínio $\Re^n$. Tal região deve estar matematicamente bem definida por restrições convenientes em um problema bem colocado. Além disso, ter um domínio Ω fechado e limitado não representa uma dificuldade muito grande. Primeiro porque as restrições são da forma $g_i(\mathbf{p}) \leq 0$ ou $h_i(\mathbf{p}) = 0$ permitindo definir uma região fechada e limitada. É ainda muito comum incluir no conjunto de restrições um intervalo real fechado de valores que cada variável de projeto pode assumir[3], ou seja, $p_{i\,\min} \leq p_i \leq p_{i\,\max}$.

A partir daí, pode-se afirmar que um problema de otimização bem definido respeita as hipóteses do teorema de Weierstrass e assim, algumas das soluções são mínimos (locais ou globais) do problema. Se o problema é de *programação convexa*, ou seja, todas as funções envolvidas são convexas e definidas numa região de viabilidade convexa, *todo mínimo local é também um mínimo global do problema.* Mas como a hipótese de convexidade é muito severa, o teorema seguinte fornece condições suficientes menos exigentes para caracterizar um ponto de mínimo.

Teorema 2.2 ***Condições suficientes para problemas gerais de otimização.*** *Seja* $\mathbf{p}^*$ *um ponto satisfazendo as condições necessárias de primeira ordem de Karush-Kuhn-Tucker para um problema geral de otimização. Define-se a matriz hessiana do Lagrangeano do problema como*

$$\nabla^2_{\mathbf{p}} L(\mathbf{p}^*) = \nabla^2_{\mathbf{p}} f(\mathbf{p}^*) + \sum_{i=1}^{m} \lambda_i^* \nabla^2_{\mathbf{p}} g_i(\mathbf{p}^*) + \sum_{i=1}^{p} \mu_i^* \nabla^2_{\mathbf{p}} h_i(\mathbf{p}^*) \tag{2.8}$$

Definem-se ainda direções viáveis não-nulas $\mathbf{d}$, $\mathbf{d} \neq \mathbf{0}$, *como as soluções do sistema formado pelas equações* (2.9) *e* (2.10).

$$\nabla_{\mathbf{p}} h_i^T \mathbf{d} = 0 \qquad i = 1, 2, \ldots, p \tag{2.9}$$

$$\nabla_{\mathbf{p}} g_i^T \mathbf{d} = 0 \qquad \forall i, 1 \leq i \leq m \quad | \quad \lambda_i > 0 \tag{2.10}$$

$$\nabla_{\mathbf{p}} g_i^T \mathbf{d} \leq 0 \qquad \forall i, 1 \leq i \leq m \quad | \quad \lambda_i = 0 \tag{2.11}$$

Se

$$\mathbf{d}^T \nabla^2_{\mathbf{p}} L(\mathbf{p}^*)\, \mathbf{d} > 0 \tag{2.12}$$

então $\mathbf{p}^*$ *é um ponto de mínimo local isolado.*

É importante notar que se a matriz $\nabla^2_{\mathbf{p}} L(\mathbf{p}^*)$ é positiva-definida, ou seja, a condição (2.12) é respeitada para qualquer $\mathbf{d}$ não-nulo, então $\mathbf{p}^*$ satisfaz a condição suficiente para um ponto de mínimo local isolado, não sendo necessária nenhuma verificação adicional, segundo o teorema seguinte.

Teorema 2.3 ***Condição suficiente forte.*** *Seja* $\mathbf{p}^*$ *um ponto satisfazendo as condições necessárias de primeira ordem de Karush-Kuhn-Tucker para um problema geral de otimização.*

[3]De fato, uma das preocupações durante a formulação de problemas reais de otimização é impor limites explícitos à evolução do processo de solução, com o objetivo de prevenir uma possível divergência devido a problemas numéricos.

Logo, se a matriz hessiana do Lagrangeano do problema em $\mathbf{p}^*$, $\nabla^2_{\mathbf{p}} L(\mathbf{p}^*)$ *definida por* (2.8), *for positiva-definida, então* $\mathbf{p}^*$ *é um ponto de mínimo isolado.*

Entretanto, se $\nabla^2_{\mathbf{p}} L(\mathbf{p}^*)$ não é positiva-definida, então não se pode obter a mesma conclusão. Neste caso, deve-se calcular $\mathbf{d}$ através de (2.9), (2.10) e (2.11), avaliando-se condição (2.12) em seguida.

Um caso freqüente em algumas aplicações (como por exemplo em programação linear) ocorre quando o número total de restrições ativas (com pelo menos uma desigualdade) num ponto candidato $\mathbf{p}^*$ é igual ao número n de variáveis de projeto. Como $\mathbf{p}^*$ satisfaz as condições (2.2) a (2.5), os gradientes das restrições ativas são linearmente independentes. Dessa forma, a única solução para o sistema (2.9) e (2.10) é a trivial $\mathbf{d} = \mathbf{0}$, onde o Teorema 2.3 não pode ser aplicado. Entretanto, como $\mathbf{d} = \mathbf{0}$ é a única solução possível, não existem direções viáveis na vizinhança de $\mathbf{p}^*$ nas quais a função objetivo possa ser reduzida ainda mais. Assim, o ponto $\mathbf{p}^*$ é um mínimo local do problema.

2.1.3 Significado dos Multiplicadores de Lagrange

A solução do sistema de equações (2.2) a (2.5) fornece, além dos pontos candidatos a mínimo, os respectivos multiplicadores de Lagrange das restrições em cada um desses pontos.

Os multiplicadores de Lagrange são importantes na *análise de pós-otimalidade*, ou seja, na análise da solução ótima e suas relações com a variação de alguns parâmetros da formulação do problema. Os multiplicadores de Lagrange fornecem informação sobre a sensibilidade da função objetivo à variação dos limites das restrições. Se o problema foi resolvido na sua forma geral, com $g_i(\mathbf{p}) \leq 0$ e $h_i(\mathbf{p}) = 0$, isto é com os limites das restrições iguais a zero, os multiplicadores podem ser usados para estudar o benefício de se relaxar uma restrição (adotar um limite maior que zero) ou a penalização em torná-la mais severa (assumir um limite menor que zero). Neste casos, tem-se, respectivamente, uma ampliação ou redução da região factível Ω, facilitando ou dificultando a obtenção do mínimo.

Considere a seguinte modificação do problema de otimização original (1.11):

$$\begin{aligned} &\min\ f(\mathbf{p}) \\ &\text{sujeita à} \\ &\qquad g_i(\mathbf{p}) \leq \bar{e}_i \qquad i = 1,2,\ldots,m \\ &\qquad h_j(\mathbf{p}) = \bar{b}_j \qquad j = 1,2,\ldots,p \end{aligned} \tag{2.13}$$

onde $\bar{e}_i$ e $\bar{b}_j$ são pequenas variações na vizinhança de zero. Naturalmente, a solução ótima $\mathbf{p}^*$ e a função custo do problema perturbado dependem dos vetores $\bar{\mathbf{e}}$ e $\bar{\mathbf{b}}$, ou seja, $\mathbf{p}^* = \mathbf{p}^*(\bar{\mathbf{e}}, \bar{\mathbf{b}})$ e $f = f(\bar{\mathbf{e}}, \bar{\mathbf{b}})$. No entanto, a relação explícita de f em função de $\bar{\mathbf{e}}$ e $\bar{\mathbf{b}}$ não é conhecida. Contudo, o seguinte teorema permite calcular implicitamente as derivadas parciais $\partial f/\partial \bar{e}_i$ e $\partial f/\partial \bar{b}_j$.

Teorema 2.4 ***Sensibilidade à variação das restrições.*** *Seja* $g_i(\mathbf{p})$, $i = 1,2,\ldots,m$ *e* $h_j(\mathbf{p})$, $j = 1,2,\ldots,p$, *com duas derivadas contínuas. Seja* $\mathbf{p}^*$ *um ponto regular que juntamente com os multiplicadores* λ_i^* *e* μ_i^*, *satisfaz tanto as condições necessárias de Karush-Kuhn-Tucker* (2.2) *a* (2.6) *quanto as condições suficientes do* Teorema 2.2. *Se para cada* $g_i(\mathbf{p}) = 0$, *é verdade que* $\lambda_i^* > 0$, *então a solução* $\mathbf{p}^*(\bar{\mathbf{e}}, \bar{\mathbf{b}})$ *do problema modifidado* (2.13) *é uma função continuamente diferenciável de* $\bar{\mathbf{b}}$ *e* $\bar{\mathbf{e}}$ *em alguma vizinhança de* $\bar{\mathbf{b}} = \mathbf{0}$ *e* $\bar{\mathbf{e}} = \mathbf{0}$. *E ainda,*

$$\frac{\partial f(\mathbf{p}^*(\mathbf{0},\mathbf{0}))}{\partial \bar{e}_i} = -\lambda_i^* \qquad i = 1,2,\ldots,m \tag{2.14}$$

$$\frac{\partial f(\mathbf{p}^*(\mathbf{0},\mathbf{0}))}{\partial \bar{b}_j} = -\mu_j^* \qquad j = 1,2,\ldots,p \tag{2.15}$$

Este teorema fornece os valores das derivadas implícitas da função objetivo com respeito

aos parâmetros do lado direito das restrições para o problema escrito na forma padrão (1.11). Uma vez conhecidos estes valores, pode-se obter uma expansão em série de Taylor da função objetivo em termos dos $\bar{e}_i$ e $\bar{b}_j$ e estimar suas variações se os limites das restrições forem alterados.

Empregando (2.14) e (2.15), a expansão em série de Taylor de primeira ordem da função objetivo em torno de $(\bar{\mathbf{e}}, \bar{\mathbf{b}}) = (\mathbf{0}, \mathbf{0})$ é, , dada por,

$$\begin{aligned} f(\bar{\mathbf{e}}, \bar{\mathbf{b}}) &= f(\mathbf{0},\mathbf{0}) + \sum_i \frac{\partial f(\mathbf{0},\mathbf{0})}{\partial \bar{e}_i}\bar{e}_i + \sum_j \frac{\partial f(\mathbf{0},\mathbf{0})}{\partial \bar{b}_j}\bar{b}_j \\ &= f(\mathbf{0},\mathbf{0}) - \sum_i \lambda_i^* \bar{e}_i - \sum_j \mu_j^* \bar{b}_j \end{aligned} \tag{2.16}$$

Logo,

$$f(\bar{\mathbf{e}}, \bar{\mathbf{b}}) - f(\mathbf{0},\mathbf{0}) = \Delta f = -\sum_i \lambda_i^* \bar{e}_i - \sum_j \mu_j^* \bar{b}_j \tag{2.17}$$

A partir da equação (2.16), pode-se entender o significado da expressão (2.6). Se uma determinada restrição de desigualdade está ativa na solução, significa que a função objetivo não pode assumir valores ainda menores sem violar esta restrição. Assim, se tal restrição for relaxada tomando-se $\bar{e}_i > 0$, expande a região viável, permitindo obter mais pontos candidato a mínimo, com o possível decrescimento da função custo. Assim, a função objetivo deve ter uma variação negativa em (2.17). Observa-se ainda em (2.17) que se $\lambda_i^* < 0$ então a relaxação da restrição com $\bar{e}_i > 0$ resulta num aumento no valor de f, o que é uma contradição implicando numa penalidade ao se relaxar o conjunto de restrições. Logo é necessário que $\lambda_i^* > 0$. No caso de restrições de igualdade, entretanto, não se tem nenhuma relação. Por isso não há nenhuma exigência quanto ao sinal dos multiplicadores de Lagrange das restrições de igualdade.

Por fim, quanto maior o valor absoluto de um multiplicador de Lagrange, mais influente é a modificação da respectiva restrição na variação do resultado obtido. Além disso, o sinal do multiplicador de uma restrição de igualdade informa o sentido de tal variação.

2.2 Métodos Numéricos – Conceitos Gerais

Apesar de fornecerem a base teórica para a determinação de pontos de mínimo em problemas genéricos, as condições de Karush-Kuhn-Tucker não fornecem um procedimento geral aplicável a uma grande variedade de problemas práticos. Nesse sentido, a utilização de ferramentas numéricas da programação matemática se torna necessário.

ARORA[14, 15] e VANDERPLAATS[5], no início da década de 80, demonstraram a necessidade de dispensar atenção especial às características específicas dos problemas de otimização estrutural. De fato, não é possível relacionar a performance de um algoritmo em um problema de programação matemática com o seu comportamento em problemas de otimização de projetos. A razão é que problemas de programação matemática consistem de funções explícitas, de pequena dimensão e pequeno número de mínimos locais. Assim, os problemas de programação matemática são em geral triviais em termos do tempo computacional de uma análise. Dessa forma, em tais problemas normalmente se utiliza uma combinação de tempo de CPU, tempo de preparação, facilidade de uso, etc., como aspectos principais de um código. Conseqüentemente um programa pode parecer muito bom mesmo exigindo centenas de avaliações da função objetivo e das restrições para resolver um problema de apenas três ou quatro variáveis. O custo da aplicação de tal código em um problema complexo de análise é proibitivo.

Isto sugere que, em problemas práticos de projeto, somente dois critérios são significativos. Em primeiro lugar, se o algoritmo e sua implementação são confiáveis em atingir um mínimo aproximado a partir de um ponto inicial arbitrário. E segundo, se o programa realiza o menor

número possível de avaliações das funções do problema e de seus gradientes.

2.2.1 Forma Geral do Problema de Otimização Estrutural

Do ponto de vista de aplicação, um código de otimização é utilizado juntamente com um programa de análise. No campo da engenharia estrutural, a técnica de análise mais utilizada é sem dúvida o método dos elementos finitos, tanto por sua precisão quanto pela sua versatilidade. Dessa forma, é interessante analisar as características de um problema de otimização cuja etapa de análise é feita via elementos finitos.

Pode-se reescrever o problema geral de otimização (1.11) para o contexto estrutural como,

$$\begin{aligned} &\min\ f(\mathbf{p}, \mathbf{u}(\mathbf{p})) \\ &\text{sujeita à} \\ &\qquad g_i(\mathbf{p}, \mathbf{u}(\mathbf{p})) \leq 0 \qquad i = 1, 2, \ldots, m \\ &\qquad h_j(\mathbf{p}, \mathbf{u}(\mathbf{p})) = 0 \qquad j = 1, 2, \ldots, p \end{aligned} \tag{2.18}$$

com $\mathbf{u}$ satisfazendo a equação de equilíbrio de elementos finitos $\mathbf{K}(\mathbf{p})\,\mathbf{u} = \mathbf{f}(\mathbf{p})$. Observa-se que $\mathbf{p}$ é o vetor das variáveis de projeto; $\mathbf{u}$ é um vetor de N deslocamentos nodais, sendo N o número de graus de liberdade da discretização em elementos finitos da estrutura; $\mathbf{K}(\mathbf{p})$ é a matriz de rigidez da estrutura; e $\mathbf{f}$ é o vetor de carregamentos aplicados. Observe que a formulação (2.18) é exatamente a mesma indicada em (1.11), sendo apenas enfatizada a dependência implícita em relação a $\mathbf{u}$.

O problema (2.18) tem as seguintes características:

1. O problema é geral, isto é, nenhuma hipótese é assumida em relação a forma das funções objetivo e de restrição. Apenas se exige que a primeira derivada exista em todos os pontos de Ω. Esta não é uma restrição severa pois quantidades, tais como tensões e deslocamentos, consideradas em problemas de otimização são geralmente funções continuamente diferenciáveis das variáveis de projeto.

2. Como se pode observar em (2.18), as funções f, g_i e h_j são geralmente implícitas nas variáveis de projeto. Logo, $f(\mathbf{p}, \mathbf{u})$ depende do vetor de deslocamentos $\mathbf{u}$ que por sua vez depende de $\mathbf{p}$ através da equação $\mathbf{K}(\mathbf{p})\,\mathbf{u} = \mathbf{f}(\mathbf{p})$. Dessa forma, a avaliação das funções e de seus gradientes é muito cara devido à essa dependência implícita.

3. O problema de projeto ótimo é altamente não-linear e em geral não-convexo. Assim, podem existir muitos mínimos locais.

4. O problema é tipicamente de grandes dimensões, ou seja, envolve geralmente grande número de variáveis e restrições.

Em resumo, o custo computacional de análise do projeto pode ser muito alta.

2.2.2 Convergência de um Algoritmo

Um algoritmo é classificado como *globalmente convergente* se alcança um mínimo local a partir de um ponto inicial arbitrário. Fica evidente a partir dessa definição que a convergência global é uma indicação de robustez e confiabilidade, mas não de eficiência. Este critério é usado como um ponto básico no estudo dos vários métodos e não deve ser confundido com o problema de determinação de mínimo global.

A convergência global é uma exigência importante porque permite utilizar o recurso da otimização com confiança. A implementação numérica dos algoritmos devem também exibir propriedades de convergência global.

De acordo como o teorema de convergência global de Zangwill, apresentado em [49], as três principais exigências de um algoritmo globalmente convergente são:

1. A seqüência de pontos $\mathbf{p}^{(k)}$ gerada pelo algoritmo deve estar contida em um conjunto compacto.
2. Deve haver uma função de decréscimo para o algoritmo.
3. O algoritmo deve ser fechado fora do conjunto de soluções.

A exigência 1 garante a existência de uma solução se o conjunto Ω for fechado e limitado[4]. Isso é obtido se todas as funções do problema forem contínuas. A exigência 2 significa que a cada ponto gerado, o valor da função de decréscimo diminui. Um progresso adequado de um algoritmo em direção a um mínimo pode ser monitorado com a identificação de uma função de decréscimo. No caso de problemas de minimização irrestrita, a função objetivo é obviamente a função de decréscimo do problema, mas em problemas sujeitos a restrições, diversas funções de decréscimo têm sido usadas. A exigência 3 implica que o algoritmo deve ser tal que o vetor direção de busca seja uma função contínua das variáveis de projeto. Isto significa que uma direção de busca conveniente pode ser determinada a cada iteração, ou seja, um decréscimo em direção ao mínimo pode ser mantido. Esta exigência também evita oscilações na função de decréscimo.

As exigências anteriores são bastante razoáveis para muitos problemas de mecânica estrutural.

2.2.3 Estado das Restrições em um Ponto

Considere as seguintes situações para as funções de restrição de um problema de otimização:

- **Restrição Ativa.** Uma restrição está ativa em um ponto $\mathbf{p}^{(k)}$ se tal restrição é satisfeita na igualdade, i.e, $g_i\left(\mathbf{p}^{(k)}\right) = 0$ ou $h_j\left(\mathbf{p}^{(k)}\right) = 0$.
- **Restrição Inativa.** Uma restrição de desigualdade está inativa em um ponto $\mathbf{p}^{(k)}$ se tal restrição é satisfeita na desigualdade, i.e, $g_i\left(\mathbf{p}^{(k)}\right) < 0$.
- **Restrição Violada.** Uma restrição de desigualdade está violada em um ponto $\mathbf{p}^{(k)}$ se tem valor positivo nesse ponto, $g_i\left(\mathbf{p}^{(k)}\right) > 0$. Uma restrição de igualdade está violada se tem o valor diferente de zero, $h_j\left(\mathbf{p}^{(k)}\right) \neq 0$. Observe que por estas definições, uma restrição de igualdade somente pode estar ativa ou violada em um determinado ponto $\mathbf{p}^{(k)}$.
- **Restrição ε-Ativa.** Qualquer inequação $g_i\left(\mathbf{p}^{(k)}\right) \leq 0$ é dita estar ε-ativa em um ponto $\mathbf{p}^{(k)}$ se $g_i\left(\mathbf{p}^{(k)}\right) < 0$, mas $g_i\left(\mathbf{p}^{(k)}\right) + \varepsilon \geq 0$, onde $\varepsilon > 0$ é um número pequeno arbitrário. Uma restrição ε-ativa para um ponto $\mathbf{p}^{(k)}$ significa simplesmente que $\mathbf{p}^{(k)} \in \Omega$, mas está arbitrariamente próximo do contorno de Ω.

O conjunto formado pelas restrições ativas, ε-ativas e violadas é chamado de *conjunto potencialmente ativo* ou simplesmente de *conjunto ativo.*

[4]Em espaços de dimensão finita, um conjunto fechado e limitado é um conjunto compacto.

É importante que num dado algoritmo, o conjunto potencialmente ativo do problema sofra pouca ou nenhuma oscilação. Em outras palavras, um algoritmo deve ser estável o suficiente para identificar rapidamente as restrições relevantes ao problema. Mais ainda, é muito interessante que este algoritmo, após um número pequeno de iterações iniciais, tenha a capacidade de apenas utilizar o conjunto potencialmente ativo da iteração corrente em seus cálculos.

Qualquer técnica que utilize apenas o conjunto potencialmente ativo nos cálculos da iteração (sendo portanto necessário avaliar apenas os gradientes desse conjunto de funcões) é chamada de *estratégia de restrições potenciais* ou *estratégia de conjunto ativo.*

Como afirmado anteriormente, em problemas de otimização estrutural a parcela mais significativa do custo computacional reside na avaliação dos funcionais e de seus gradientes. Dessa forma, a eficiência de um método de otimização para problemas estruturais depende diretamente de sua capacidade de incorporar alguma estratégia de restrições potenciais [14, 48].

2.2.4 Normalização das Restrições

As restrições potencialmente ativas são utilizadas para calcular tanto a direção de busca quanto o tamanho de passo a cada iteração. Entretanto, diferentes funções de restrição podem ter ordens de grandeza bastante distintas. Por exemplo, num problema de mecânica estrutural, restrições de tensão tem ordem de grandeza muito superior daquelas de deslocamento. Se tais restrições fossem violadas, seria difícil avaliar comparativamente a severidade de tais violações.

Ao contrário, se as restrições não forem normalizadas, o mesmo valor ε pode ser convenientemente aplicado a todas as restrições, evitando-se também problema de condicionamento numérico ao se fazer operações entre valores de ordens de grandeza muito diferentes.

Entretanto pode ser bastante difícil normalizar algumas restrições. Por exemplo, o limite inferior de algumas restrições pode ser zero impedindo a normalização com respeito a esse limite, já que a divisão por zero é impossível. Além disso, o processo de normalização pode transformar restrições lineares em não-lineares, o que é bastante inconveniente. Nestes casos, diferentes formas de normalização devem ser aplicadas ou mesmo manter as restrições na forma original. Daqui em diante todas as restrições serão consideradas normalizadas.

2.3 Métodos Numéricos – Algoritmos

Devido ao grande número de algoritmos de otimização, torna-se importante considerá-los comparativamente, de modo a determinar quais os métodos mais eficientes, assim como as características mais interessantes que um algoritmo deve possuir. Os métodos discutidos a seguir não se baseiam em nenhuma estrutura especial do problema, como convexidade, podendo ser aplicados a problemas genéricos de programação não-linear. Dessa forma, métodos tais como programação geométrica, critérios físicos de otimalidade e métodos que exigem interação com o usuário durante o processo iterativo não serão considerados. Apesar de tais métodos serem eficientes em situações específicas, não são de propósito geral o bastante para serem combinados com programas de análise de engenharia, em particular de elementos finitos.

Pode-se dividir os métodos de programação matemática em duas categorias [14]:

1. *Métodos Primais.*
2. *Métodos de Transformação.*

Os métodos primais tratam diretamente o problema em sua forma original. Serão discutidos brevemente os métodos de programação quadrática seqüencial, gradiente projetado, direções

viáveis, critério de otimalidade, programação linear seqüencial e métodos de projeção. Os métodos de transformação convertem o problema original sujeito a restrições em uma seqüência de problemas irrestritos.

A maioria dos métodos resolve o problema (1.11) gerando uma seqüência $\left\{\mathbf{p}^{(k)}\right\}$ dada por

$$\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + t_k \mathbf{d}^{(k)} \tag{2.19}$$

onde o escalar t_k é o *passo* na iteração k e $\mathbf{d}^{(k)}$ é uma *direção de decréscimo* da *função de descida* φ, tal que,

$$\varphi\left(\mathbf{p}^{(k+1)}\right) < \varphi\left(\mathbf{p}^{(k)}\right) \qquad \text{ou} \qquad \varphi\left(\mathbf{p}^{(k)} + t_k \mathbf{d}^{(k)}\right) < \varphi\left(\mathbf{p}^{(k)}\right) \tag{2.20}$$

sendo t_k calculado de modo a impor um decréscimo adequado a φ. Daí pode-se observar que a ausência de uma função de descida impede a determinação conveniente do tamanho de passo t_k, não podendo ser garantido o decréscimo da função de custo a cada iteração.

2.3.1 Métodos Primais

Métodos de Programação Quadrática Recursiva

Os métodos de programação quadrática recursiva [72] empregam o método de Newton ou métodos quasi-Newton para resolver diretamente as condições necessárias de otimalidade. Tais métodos se tornaram populares devido às suas características de forte convergência. Assim, podem determinar qualquer solução das equações (2.2) a (2.5), mesmo um ponto de máximo local.

Por isso, ao invés de procurar qualquer solução do problema original, emprega-se uma pequena variante desse método, originando um subproblema quadrático de minimização, cujas condições de otimalidade são as mesmas do método original, mas que tende a dirigir o processo iterativo para soluções melhoradas em relação a estimativa inicial. Dessa maneira, obtem-se o subproblema de minimização de uma aproximação quadrática do Lagrangeano do problema (1.11), sujeito a uma aproximação linear das restrições a cada iteração k. Assim, este tipo de procedimento é também conhecido como método do *Lagrangeano projetado* ou *Lagrange-Newton*. Este método determina tanto a solução primal quanto a dual (multiplicadores de Lagrange).

Um algoritmo genérico de SQP (*Sequential Quadratic Programming*) pode ser estabelecido como,

Passo 1. Ajuste $k = 0$; estime um projeto inicial $\mathbf{p}^{(0)}$; os escalares $\varepsilon \in (0,1)$ e $\eta_0 > 0$; e uma matriz positiva definida $\mathbf{Q}^{(0)}$.

Passo 2. Resolva o subproblema quadrático em $\mathbf{p}^{(k)}$:

$$\begin{aligned}
&\min\ \tfrac{1}{2}\mathbf{d}^T\mathbf{Q}^{(k)}\mathbf{d} + \eta_k \nabla f\left(\mathbf{p}^{(k)}\right)^T + f\left(\mathbf{p}^{(k)}\right) \\
&\text{sujeita à} \\
&g_i\left(\mathbf{p}^{(k)}\right) + \nabla g_i\left(\mathbf{p}^{(k)}\right)^T \leq 0 \qquad\qquad i \in I_k \\
&h_j\left(\mathbf{p}^{(k)}\right) + \nabla h_j\left(\mathbf{p}^{(k)}\right)^T = 0 \qquad\qquad j = 1, 2, \ldots, p
\end{aligned}$$

onde I_k é um subconjunto de $I_m = \{1, 2, \ldots, m\}$.
A solução é $\mathbf{d}^{(k)}$, $\boldsymbol{\lambda}^{(k)}$, $\boldsymbol{\mu}^{(k)}$, onde $\boldsymbol{\lambda}^{(k)}$e $\boldsymbol{\mu}^{(k)}$é o conjunto dos multiplicadores de Lagrange associados às restrições do subproblema quadrático. Se $\left\|\mathbf{d}^{(k)}\right\| < \xi$, então pare. Senão, continue.

Passo 3. Determine o tamanho de passo t_k implicando num decréscimo da função de descida e obtenha um novo passo a partir de (2.19).

Passo 4. Atualize η_{k+1}, $\mathbf{Q}^{(k+1)}$; ajuste $k = k + 1$ e volte ao passo 2.

Vários métodos podem ser obtidos com diferentes escolhas da função de decréscimo, estratégia de busca linear e a incorporação (ou não) de alguma estratégia de conjunto ativo, de modo que I_k seja apenas o conjunto das restrições de desigualdade potencialmente ativas (as restrições de igualdade sempre são potencialmente ativas). Em geral, adota-se para a matriz $\mathbf{Q}^{(k)}$ uma atualização quasi-Newton da matriz Hessiana do Lagrangeano $\nabla^2 L\left(\mathbf{p}^{(k)}\right)$ para obter uma convergência superlinear.

A solução do problema de programação quadrática anterior pode ser expressa como $\mathbf{d} = -\eta\mathbf{d}_d + \mathbf{d}_c$, onde $-\mathbf{d}_d$ é um vetor obtido pela projeção (com respeito a $\mathbf{Q}$) do gradiente da função objetivo no hiperplano tangente às restrições ativas e $\mathbf{d}_c$ é um vetor normal a esse plano. Em outras palavras, como $-\mathbf{d}_d$ é um passo de redução e $\mathbf{d}_c$ é um passo de correção, a cada iteração se busca uma redução tanto na função de custo quanto na violação às restrições. Nos algoritmos, adota-se $\eta = 1$.

Dessa classe de algoritmos pode-se citar o *algoritmo de Han*, não utilizando nenhuma estratégia de conjunto ativo, e o *algoritmo de Pschenichny's*, incorporando essa estratégia e sendo satisfatoriamente aplicado a problemas de otimização estrutural [14, 15, 16].

Métodos de Gradiente Projetado

O método de gradiente projetado de Rosen [49, 72] é um método de direções viáveis. Os métodos de direções viáveis envolvem, em geral, a resolução de um subproblem quadrático ou linear, necessários para determinar quais as restrições do subproblema linearizado pertencem ao conjunto ativo. A idéia básica de Rosen, entretanto, é obter o conjunto de restrições ativas a partir de regras simples, determinando $\mathbf{d}$ de forma fechada, sem a necessidade de resolver subproblemas quadráticos ou lineares. Isto significa que, ao invés de buscar a melhor direção de busca considerando as restrições, determina-se uma direção que não é tão boa, mas é mais fácil de ser calculada.

Entretanto, como a avaliação das funções e dos gradientes de um problema estrutural é bastante cara (sendo portanto a etapa de maior custo em termos de tempo), é melhor aplicar um método de determinação da direção de busca baseado na solução de um subproblema linear ou quadrático, obtendo a melhor direção de decréscimo possível, ao invés de implementar a filosofia de gradiente projetado.

Considerando a decomposição $\mathbf{d} = -\mathbf{d}_d + \mathbf{d}_c$, tem-se que $-\mathbf{d}_d$ é aplicado a partir de um projeto viável $\mathbf{p}^{(k)}$. Se t_k é um tamanho de passo nessa direção, $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} - t_k\mathbf{d}_d$ é em geral inviável e deve ser sucedida por um seqüência de passos de correção $\mathbf{d}_c$ para se obter um próximo passo viável. Isso causa dificuldades de manter o decréscimo da função objetivo pois ao retornar à região viável, pode ser que $f\left(\mathbf{p}^{(k+1)}\right) > f\left(\mathbf{p}^{(k)}\right)$. E ainda, se as restrições forem altamente não-lineares, tais passos de correção podem ser muito caros.

Além disso, devido à mudanças repentidas no conjunto ativo, não se pode mostrar que o método de gradiente projetado é globalmente convergente. Entretanto, modificar o método para torná-lo globalmente convergente pode eliminar sua simplicidade.

Métodos de Gradiente Reduzido

Os métodos de gradiente reduzido [49, 72] são baseados em uma técnica de eliminação de variáveis desenvolvida por Wolfe para resolver problemas de programação não-linear com restrições lineares. Sua generalização para tratar o caso de restrições não-lineares é conhecido como método de gradiente reduzido generalizado. A idéia básica é utilizar as restrições ativas para fazer uma eliminação de variáveis. O problema então se torna irrestrito nas variáveis restantes. Tal método pode ser aplicado de duas formas: convertendo as desigualdades em igualdades introduzindo variáveis de folga não-negativas ou aplicando uma estratégia de conjunto ativo.

Este método se mostra essencialmente o mesmo do método do gradiente projetado se for utilizada uma estratégia de conjunto ativo. Se o método for aplicado convertendo as restrições de desigualdade em restrições de igualdade, este método se mostra muito mais eficiente que o método do gradiente projetado e globalmente convergente. Entretanto, convertendo todas as restrições em igualdades faz com que todas estejam ativas em cada iteração, exigindo a avaliação de toda as funções e de seus gradientes. Isto torna sua aplicação inviável para problemas de otimização estrutural de média e grande escala. Dessa forma, segundo BELEGUNDU e ARORA[14], para resolver problemas de otimização estrutural não é necessário fazer diferença entre os métodos de gradiente projetado e gradiente reduzido.

Métodos de Direções Viáveis ou Pontos Interiores

A idéia básica dos métodos de direções viáveis ou pontos interiores é, a partir de um projeto viável, se mover para um projeto viável melhorado sem iteragir fora da região Ω. Apesar da maioria dos autores de programação matemática [11, 49, 72] classificarem os métodos anteriores como sendo de direções viáveis, pode-se observar que tais procedimentos incluem, na verdade, etapas de correção para eliminar ou minimizar a violação às restrições em cada iteração. De fato, no caso da programação quadrática seqüencial, não há nenhuma exigência de viabilidade no algoritmo (ou seja, são permitidas iterações inviáveis), sendo a função de descida responsável por avaliar a severidade das restrições. Os métodos de gradiente projetado e reduzido incluem etapas de correção de projetos inviáveis em cada iteração. Isso permite que haja iterações fora da região viável Ω (1.12).

Um algoritmo genérico de direções viáveis é o seguinte:

Passo 1. Estimar um projeto viável inicial $\mathbf{p}^{(0)}$. Ajuste $k = 0$.

Passo 2. Determinar uma direção de busca conveniente $\mathbf{d}^{(k)}$ tal que $\nabla f\left(\mathbf{p}^{(k)}\right)^T \mathbf{d}^{(k)} < 0$ é mínimo, respeitando $\nabla g_i\left(\mathbf{p}^{(k)}\right)^T \mathbf{d}^{(k)} \leq 0$ para as restrições de desigualdade ativas e $\nabla h_j\left(\mathbf{p}^{(k)}\right)^T \mathbf{d}^{(k)} = 0$ para $j = 1, \ldots, p$. Se $\left\|\mathbf{d}^{(k)}\right\| < \varepsilon$, pare; $\mathbf{p}^{(k)}$ é uma solução.

Passo 3. Determinar t_k tal que $f\left(\mathbf{p}^{(k)} + t_k \mathbf{d}^{(k)}\right) < f\left(\mathbf{p}^{(k)}\right)$, $g_i\left(\mathbf{p}^{(k)} + t_k \mathbf{d}^{(k)}\right) \leq 0$ e $h_j\left(\mathbf{p}^{(k)} + t_k \mathbf{d}^{(k)}\right) = 0$. O novo projeto é dado por (2.19); retorne ao passo 2.

Diferentes algoritmos de busca linear podem ser utilizados no passo 3, sendo a regra de Armijo[11, 49] mais comumente empregada. No caso de estarem presentes apenas restrições de desigualdade, é usual ser utilizado um termo de *deflexão* da direção de busca para dentro de Ω na forma $\nabla g_i\left(\mathbf{p}^{(k)}\right)^T \mathbf{d} \leq \theta_i$, onde $\theta_i < 0$. Isto impede que se obtenha uma direção de busca quase tangente ao contorno de Ω, exigindo um passo t muito pequeno, retardando a

convergência. Existem entretanto dificuldades computacionais relacionadas principalmente ao cálculo dos fatores de deflexão a cada iteração e à realização de uma busca linear restrita.

Os métodos de direções viáveis possuem duas vantagens significativas. Em primeiro lugar, como todos os pontos da seqüência gerada pelo método são viáveis, se o processo se interromper antes de atingir a solução (o que é muito provável nas aplicações práticas), o projeto final obtido é sempre uma melhoria da estimativa inicial, podendo representar uma solução aceitável para o problema prático original. Segundo, em geral pode-se garantir que, se o método gera uma seqüência convergente, o projeto final é, ao menos, um mínimo local restrito do problema.

2.3.2 Métodos de Transformação

De maneira diferente dos métodos primais, os quais tratam o problema de otimização em sua forma original, uma outra estratégia para abordar o problema de otimização com restrições é converter o problema sujeito a restrições numa seqüência de problemas irrestritos. Esta classe inclui os métodos de penalidades, de barreiras e os métodos de Lagrangeano aumentado ou de multiplicadores.

Os métodos de transformação resolvem, a cada passo, a função transformada irrestrita

$$\varphi(\mathbf{p}, \mathbf{c}_k) = f(\mathbf{p}) + \mathcal{P}(\mathbf{g}(\mathbf{p}), \mathbf{h}(\mathbf{p}), \mathbf{c}_k) \tag{2.21}$$

onde $\mathbf{c}_k$ é um vetor de parâmetros de controle da função de penalização $\mathcal{P}$ no passo k de penalização. A solução $\mathbf{p}_k$ do problema irrestrito do passo k é o ponto inicial do passo $k+1$. A evolução da seqüência de vetores $\{\mathbf{c}_k\}$ controla a seqüência de problemas irrestritos.

SUMT's

As *Técnicas de Minimização Seqüencial Irrestrita* (ou *Sequential Uncostrained Minimization Techniques — SUMT's*) são dividas em métodos de penalidades ou barreiras. Os métodos de penalidades iteragem na região irrestrita, aumentando a penalização devido a violação das restrições a cada passo. No limite, se atinge a região viável. Os métodos de barreiras, ao contrário, constroem uma "barreira " interior ao contorno da região viável, impedindo que a seqüência de projetos se torne inviável. A cada passo, a "barreira " se torna mais próxima ao contorno da região viável, e dessa maneira pode-se saturar alguma restrição. Entretanto, os *SUMT's* possuem uma série de características indesejáveis, principalmente a falta de robustez quando o valor das componentes de $\mathbf{c}_k$ se torna muito grande. Neste caso, as funções de penalidade se tornam mal-comportadas e ainda, a matriz Hessiana da função transformada tende a se tornar mal-condicionada quando $\mathbf{c}_k \to \infty$. Além disso, existe a dificuldade de selecionar a seqüência $\{\mathbf{c}_k\}$ mais adequada, a qual depende do problema a ser tratado.

Métodos de Lagrangeano Aumentado

Nos *SUMT's* existe a necessidade de fazer, no limite, o parâmetro de penalização infinitamente grande para se obter a solução ótima do problema original. E isto pode causar dificuldades numéricas e efeitos de mal-condicionamento. Os *métodos de Lagrangeano Aumentado* foram desenvolvidos como uma tentativa de se eliminar estes problemas. Nestes métodos, não há a necessidade de se controlar a seqüência de parâmetros de penalização tendendo ao infinito. Dessa forma, introduzem-se funções de penalização que recuperam a solução ótima para parâmetros de valor finito. Tais funções são chamadas de *funções de penalização exatas*. Mais ainda, assim como os *SUMT's* são globalmente convergentes e ainda pode-se mostrar que possuem taxas de convergência mais rápidas.

Tanto os *SUMT's* quando os *métodos de Lagrangeano aumentado* usam estratégias de conjunto ativo, pois somente as restrições potencialmente ativas precisam ser consideradas para o cálculo de φ e $\nabla\varphi$. Contudo, a convergência global dos métodos de transformação somente pode ser provada com a exigência de que cada subproblema irrestrito seja resolvido exatamente e globalmente. Como na prática os problemas somente podem ser resolvidos aproximadamente e, em geral, em termos de mínimos locais devido ao custo computacional, pode-se questionar a convergência global de tais métodos. No caso do Lagrangeano Aumentado, entretanto, se alguma regra de atualização dos multiplicadores for adotada, a minimização exata do subproblema irrestrito pode ser substituída por um único passo de minimização. Além disso, apesar da eficiência no cálculo de φ e $\nabla\varphi$, deve-se tem em mente que é necessário resolver uma *seqüência* de problemas de minimização irrestrita a cada iteração do método, o que pode ser muito caro computacionalmente.

2.3.3 Comentários

Baseado na discussão anterior, as características mais importantes de um algoritmo do ponto de vista da otimização estrutural são *convergência global*, *eficiência* e *generalidade*.

1. Como citado na Seção 2.2.2, um algoritmo globalmente convergente é *confiável*, ou seja, garante-se que uma solução será obtida a partir de qualquer ponto inicial.

2. Um algoritmo *eficiente* tem alta taxa de convergência e faz o menor número possível de avaliação das funções objetivo, de restrição e seus gradientes a cada iteração. Isso inclui a incorporação de estratégias de conjunto ativo, de modo que somente os gradientes das funções potencialmente ativas precisem ser calculados no passo de determinação da direção de busca, além de um método de busca linear com uma precisão satisfatória, aliada a um número mínimo de avaliações das funções, isto é *busca linear inexata*.

3. Um algoritmo de otimização *geral* deve ser capaz de tratar qualquer problema que possa ser colocado na forma padrão (1.11), sem impor qualquer restrição à forma das funções do problema, além da sua continuidade.

Seria interessante também que o método fosse *simples* de ser usado. Isso significa não exigir o ajuste de um grande número de parâmetros, o que muitas vezes não somente exige o conhecimento da estrutura matemática do algoritmo, mas também das características do problema. É claro, entretanto, que para um usuário aplicar de maneira eficiente e confiável ferramentas gerais de análise e projeto em engenharia, um razoável conhecimento do problema e das ferramentas matemáticas sempre será necessário.

2.4 Método de Pontos Interiores de Herskovits

O método de Pontos Interiores de Herskovits [17] consiste de um algoritmo de pontos interiores para otimização não-linear sujeita a restrições de igualdade e desigualdade. Devido à sua característica de convergência global com taxa superlinear, demonstrada em [17, 18], e por gerar uma seqüência de pontos viáveis, tem sido aplicado com sucesso em problemas estruturais [19, 20, 21]. O método exige apenas a solução de dois sistemas lineares de mesma matriz (e dimensão $n + m + p$) para determinar a direção de busca. Além disso, como a seqüência dos multiplicadores de Lagrange das restrições de desigualdade é estritamente positiva, as direções de busca são sempre direções de decréscimo da função objetivo [17]. No caso de haver apenas restrições de desigualdade, não há necessidade de nenhuma função de penalização.

A versão básica do algoritmo também não exige nenhuma estratégia de restrições ativas para eliminar variáveis. Entretanto, a partir de suas características na solução de problemas pode-se propor uma forma de incorporar uma estratégia de restrições ativas com a finalidade de aumentar sua eficiência.

2.4.1 Idéias Básicas

Para obter a direção de busca a cada iteração, este algoritmo de pontos interiores emprega uma técnica iterativa de ponto fixo na solução direta do sistema de equações, nas variáveis primais e duais, obtido das condições necessárias de otimalidade de Karush-Kuhn-Tucker (2.2), (2.4) e (2.5). O algoritmo é tal que as inequações (2.3) e (2.6) são satisfeitas a cada iteração através de uma busca linear imprecisa, garantido a convergência para pontos necessários.

A versão básica do algoritmo será apresentada para um problema sujeito apenas a restrições de desigualdade, ou seja, o sistema não-linear a ser resolvido é composto apenas por (2.2) e (2.4). Neste caso, a busca linear também é realizada exigindo que haja decréscimo da função objetivo a cada passo. Deve-se observar que, como a seqüência de cada multiplicador de Lagrange gerada pelo algoritmo é estritamente positiva, o problema não pode convergir para qualquer ponto necessário, mas apenas para pontos de mínimo da função objetivo. Posteriormente, a inclusão de restrições de igualdade será apresentada como uma alteração desse algoritmo original.

As seguintes hipóteses são assumidas para o problema (1.11):

Hipótese 2.1 *Deve existir um número real a tal que o conjunto $\Omega_a \equiv \{\mathbf{p} \in \Omega \mid f(\mathbf{p}) \leq a\}$ é compacto e tem um interior Ω_a^0.*

Hipótese 2.2 *Cada $\mathbf{p} \in \Omega_a^0$ satisfaz (2.4).*

Hipótese 2.3 *As funções f e g_i são continuamente diferenciáveis em Ω_a e suas derivadas satisfazem a condição de Lipschitz.*

Hipótese 2.4 ***Condição de Regularidade*** *Para todo $\mathbf{p} \in \Omega_a^0$, os vetores gradiente de restrições ativas são linearmente independentes.*

Considerando a seguinte notação,

$$\begin{array}{ll} f(\mathbf{p}) : \Re^n \longrightarrow \Re & \text{Função objetivo} \\ \mathbf{g}(\mathbf{p}) : \Re^n \longrightarrow \Re^m & \text{Vetor de restrições de desigualdade} \\ \mathbf{G}(\mathbf{p}) : \Re^n \longrightarrow \Re^{m \times m} & \text{Matriz diagonal onde } G_{ii}(\mathbf{p}) = g_i(\mathbf{p}) \\ \mathbf{C}(\mathbf{p}) : \Re^n \longrightarrow \Re^n & \text{Gradiente de } f(\mathbf{p}), \text{ ou seja, } C_i(\mathbf{p}) = \frac{\partial f}{\partial x_i}(\mathbf{p}) \\ \mathbf{A}(\mathbf{p}) : \Re^n \longrightarrow \Re^{m \times n} & \text{Gradiente de } \mathbf{g}(\mathbf{p}), \text{ ou seja, } A_{ij}(\mathbf{p}) = \frac{\partial g_i}{\partial x_j}(\mathbf{p}) \end{array} \tag{2.22}$$

as condições necessárias de Karush-Kuhn-Tucker dadas por (2.2), (2.3), (2.4) e (2.6) podem ser reescritas como,

$$\mathbf{C}(\mathbf{p}) + \mathbf{A}^T(\mathbf{p})\boldsymbol{\lambda} = \mathbf{0} \tag{2.23}$$

$$\mathbf{G}(\mathbf{p})\boldsymbol{\lambda} = \mathbf{0} \tag{2.24}$$

$$g_i(\mathbf{p}) \leq 0 \qquad i = 1, \ldots, m \tag{2.25}$$

$$\lambda_j \geq 0 \qquad j = 1, \ldots, m \tag{2.26}$$

Determinação da Direção de Busca

As equações (2.23) e (2.24) formam um sistema não-linear de $n+m$ equações e incógnitas da seguinte forma,

$$\chi(\mathbf{v}^*) = \mathbf{0} \tag{2.27}$$

onde

$$\mathbf{v} = \left\{ \begin{array}{c} \mathbf{p} \\ \boldsymbol{\lambda} \end{array} \right\} \qquad \boldsymbol{\chi}(\mathbf{v}) = \boldsymbol{\chi}(\mathbf{p}, \boldsymbol{\lambda}) = \left\{ \begin{array}{c} \mathbf{C}(\mathbf{p}) + \mathbf{A}(\mathbf{p})^T \boldsymbol{\lambda} \\ \mathbf{G}(\mathbf{p})\boldsymbol{\lambda} \end{array} \right\} \tag{2.28}$$

Aplicando o método de Newton para determinar as raízes de $\boldsymbol{\chi}$, obtém-se a fórmula iterativa de ponto fixo,

$$\boldsymbol{\chi}'\left(\mathbf{v}^{(k)}\right)\left\{\mathbf{v}^{(k+1)} - \mathbf{v}^{(k)}\right\} = -\boldsymbol{\chi}\left(\mathbf{v}^{(k)}\right)$$

ou seja,

$$\left[\begin{array}{cc} \mathbf{H}\left(\mathbf{p}^{(k)}, \boldsymbol{\lambda}^{(k)}\right) & \mathbf{A}^T\left(\mathbf{p}^{(k)}\right) \\ \boldsymbol{\Lambda}^{(k)}\mathbf{A}\left(\mathbf{p}^{(k)}\right) & \mathbf{G}\left(\mathbf{p}^{(k)}\right) \end{array} \right] \left\{ \begin{array}{c} \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \\ \boldsymbol{\lambda}^{(k+1)} - \boldsymbol{\lambda}^{(k)} \end{array} \right\} = - \left\{ \begin{array}{c} \mathbf{C}\left(\mathbf{p}^{(k)}\right) + \mathbf{A}\left(\mathbf{p}^{(k)}\right)^T \boldsymbol{\lambda}^{(k)} \\ \mathbf{G}\left(\mathbf{p}^{(k)}\right)\boldsymbol{\lambda}^{(k)} \end{array} \right\} \tag{2.29}$$

tal que,

- $\boldsymbol{\Lambda}^{(k)}$: matriz diagonal $\boldsymbol{\Lambda}_{ii}^{(k)} = \lambda_i^{(k)}$.
- $\mathbf{H}\left(\mathbf{p}^{(k)}, \boldsymbol{\lambda}^{(k)}\right)$: matriz Hessiana do Lagrangeano.

Introduzindo uma aproximação quasi-Newton para $\mathbf{H}$, ou seja $\mathbf{Q} \simeq \mathbf{H}$, e manipulando (2.29) obtém-se, para $\boldsymbol{\Delta}\mathbf{p}^{(k)} = \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)}$

$$\left\{ \begin{array}{l} \mathbf{Q}^{(k)}\boldsymbol{\Delta}\mathbf{p}^{(k)} + \mathbf{A}^T\left(\mathbf{p}^{(k)}\right)\boldsymbol{\lambda}^{(k+1)} = -\mathbf{C}\left(\mathbf{p}^{(k)}\right) \\ \boldsymbol{\Lambda}^{(k)}\mathbf{A}\left(\mathbf{p}^{(k)}\right)\boldsymbol{\Delta}\mathbf{p}^{(k)} + \mathbf{G}\left(\mathbf{p}^{(k)}\right)\boldsymbol{\lambda}^{(k+1)} = \mathbf{0} \end{array} \right. \tag{2.30}$$

Por simplificação, o sistema (2.30) será reescrito como,

$$\left\{ \begin{array}{l} \mathbf{Q}\mathbf{d}_0 + \mathbf{A}^T\boldsymbol{\lambda}_0 = -\mathbf{C} \\ \boldsymbol{\Lambda}\mathbf{A}\mathbf{d}_0 + \mathbf{G}\boldsymbol{\lambda}_0 = \mathbf{0} \end{array} \right. \tag{2.31}$$

Como este é um método de direções viáveis, deve-se evitar que alguma restrição se torne ativa antes de se atingir a solução. Para um ponto no contorno da região Ω, qualquer direção de busca viável deve ser necessariamente tangente às restrições ativas, exigindo um tamanho de passo tendendo a zero, o que reduz muito a taxa de convergência. Dessa forma, HERSKOVITS[17] propôs defletir $\mathbf{d}_0$ para o interior da região viável modificando (2.31) da seguinte maneira,

$$\left\{ \begin{array}{l} \mathbf{Q}\mathbf{d} + \mathbf{A}^T\bar{\boldsymbol{\lambda}} = -\mathbf{C} \\ \boldsymbol{\Lambda}\mathbf{A}\mathbf{d} + \mathbf{G}\bar{\boldsymbol{\lambda}} = -\rho\boldsymbol{\Lambda}\boldsymbol{\omega}^I \end{array} \right. \tag{2.32}$$

onde $\boldsymbol{\omega}^I \in \Re^m$ é um vetor de termos positivos, ρ é um escalar também positivo e $\bar{\boldsymbol{\lambda}}$ é a nova estimativa de $\boldsymbol{\lambda}$. A idéia é alterar (2.24) para,

$$\mathbf{G}(\mathbf{p})\bar{\boldsymbol{\lambda}} + \rho\boldsymbol{\Lambda}\boldsymbol{\omega}^I = \mathbf{0}$$

significando impor uma diminuição da região viável.

Existem várias formas de atualizar $\boldsymbol{\Lambda}\boldsymbol{\omega}^I$ a cada iteração, mas a regra geral deve ser tal que $\lambda_i\omega_i^I$ cresça sempre que a restrição i se aproxime de zero. Por sua vez, ρ deve ser definido de forma que se anule na solução. Supondo que se conheça $\boldsymbol{\omega}^I$, uma maneira eficiente de calcular a direção de busca defletida $\mathbf{d}$ é introduzir o sistema,

$$\left\{ \begin{array}{l} \mathbf{Q}\mathbf{d}_1 + \mathbf{A}^T\boldsymbol{\lambda}_1 = \mathbf{0} \\ \boldsymbol{\Lambda}\mathbf{A}\mathbf{d}_1 + \mathbf{G}\boldsymbol{\lambda}_1 = -\boldsymbol{\Lambda}\boldsymbol{\omega}^I \end{array} \right. \tag{2.33}$$

Pode-se mostrar que o vetor direção de busca $\mathbf{d}$ é dado pela soma [20],

$$\mathbf{d} = \mathbf{d}_0 + \rho\mathbf{d}_1 \tag{2.34}$$

como ilustrado na Figura 2.2. Observa-se que $\mathbf{d}_0$ é um vetor indicando a direção de decréscimo de $f(\mathbf{p})$, enquanto que ρ é o peso que controla a contribuição do vetor de deflexão $\mathbf{d}_1$ na direção final $\mathbf{d}$.

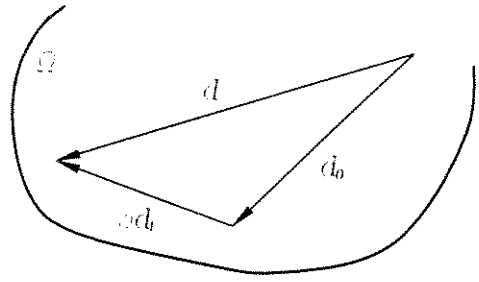


Figura 2.2: Direção de busca original $\mathbf{d}_0$, deflexão $\rho\mathbf{d}_1$ e direção de busca resultante $\mathbf{d}$.

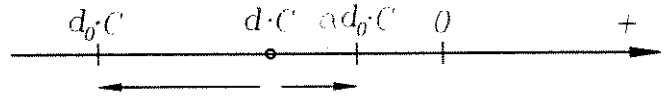


Figura 2.3: Relação entre $\mathbf{d}_0 \cdot \mathbf{C}$ e $\mathbf{d} \cdot \mathbf{C}$.

Como a deflexão é proporcional a ρ, deve-se estabelecer algum limite para seu cálculo, de tal forma que $\mathbf{d}$ ainda seja uma direção de descida para $f(\mathbf{p})$. Como, em geral, o decréscimo de $f(\mathbf{p})$ na direção defletida $\mathbf{d}$ é menor que na direção $\mathbf{d}_0$, ou seja,

$$\mathbf{d} \cdot \mathbf{C} \geq \mathbf{d}_0 \cdot \mathbf{C}$$

introduz-se então a constante $\alpha \in (0,1)$ de forma que

$$\mathbf{d} \cdot \mathbf{C} \leq \alpha \mathbf{d}_0 \cdot \mathbf{C} \qquad \alpha \in (0,1) \tag{2.35}$$

Como $\mathbf{d}_0 \cdot \mathbf{C} \leq 0$, então também $\mathbf{d} \cdot \mathbf{C} \leq 0$, conforme a Figura 2.3. Substituindo (2.34) em (2.35) obtém-se uma expressão para ρ,

$$\mathbf{d}_0 \cdot \mathbf{C} + \rho \mathbf{d}_1 \cdot \mathbf{C} \leq \alpha \mathbf{d}_0 \cdot \mathbf{C}$$

$$\Rightarrow \quad \rho \leq \frac{(\alpha - 1)\, \mathbf{d}_0 \cdot \mathbf{C}}{\mathbf{d}_1 \cdot \mathbf{C}} \qquad \text{se} \qquad \mathbf{d}_1 \cdot \mathbf{C} > 0 \tag{2.36}$$

Na prática, adota-se

$$\rho = \frac{(\alpha - 1)\, \mathbf{d}_0 \cdot \mathbf{C}}{\mathbf{d}_1 \cdot \mathbf{C}} \qquad \text{se} \qquad \mathbf{d}_1 \cdot \mathbf{C} > 0 \tag{2.37}$$

Para $\alpha \to 0$, a deflexão é muito grande, o decréscimo é pequeno e a convergência muito lenta. Entretanto, essa é a alternativa para problemas cujo contorno da região viável tenha curvatura acentuada. Se a não-linearidade do contorno for pequena, pode-se então adotar um α próximo a 1. Em outras palavras, α controla a ineficiência máxima permitida na deflexão da direção de busca original $\mathbf{d}_0$. Como $\|\mathbf{d}_0\| \to 0$ na solução[5], também $\rho \to 0$.

Em resumo, para calcular a direção de busca $\mathbf{d}$ a cada iteração, resolvem-se os sistemas lineares de mesma matriz (2.31) e (2.33), e em seguida (2.36) e (2.34). Demonstra-se que a direção $\mathbf{d}$ calculada dessa maneira é uma direção de decréscimo de $f(\mathbf{p})$ [17].

[5]Num método de ponto fixo, $\mathbf{p}^{(k)} \to \mathbf{p}^{(k+1)}$ e dessa forma $\|\mathbf{d}_0\| = \left\|\mathbf{p}^{(k)} - \mathbf{p}^{(k+1)}\right\| \to 0$. Assim, num processo numérico de solução é razoável adotar $\|\mathbf{d}_0\| < \xi$, $\xi \in (0,1)$ como um critério de convergência, sendo ξ a precisão exigida.

Busca Linear

Uma vez obtida a direção de busca $\mathbf{d}$, deve-se determinar o tamanho de passo t tal que o novo projeto seja viável e $f\left(\mathbf{p}^{(k+1)}\right) < f\left(\mathbf{p}^{(k)}\right)$. A convergência global do método é demonstrada mesmo para busca linear imprecisa, ou seja, pode-se aplicar um procedimento de maior eficiência global sem que haja problemas de convergência. Na busca linear imprecisa, exige-se apenas a viabilidade de $\mathbf{p}$ aliada a algum decréscimo de $f(\mathbf{p})$, realizando, no entanto, um número mínimo de avaliações das funções e de suas derivadas. Um método de busca imprecisa utilizado com sucesso é a regra de Armijo [49].

Em problemas sem restrições, a regra de Armijo é utilizada para impor apenas um determinado decréscimo à função objetivo, ao invés de determinar o passo que proporcione o maior decréscimo possível na direção de busca escolhida. Assim, dados $\nu, \eta \in (0,1)$, toma-se t_k como o primeiro elemento da seqüência

$$\left(1, \nu, \nu^2, \nu^3, \ldots\right)$$

tal que

$$f\left(\mathbf{p}^{(k)}+t_k\mathbf{d}\right) \leq f\left(\mathbf{p}^{(k)}\right) + \eta \mathbf{d}^T \mathbf{C}^{(k)} t_k \tag{2.38}$$

Para problemas com restrições deve-se considerar também a exigência de viabilidade, ou seja, t_k também deve ser tal que,

$$g_i\left(\mathbf{p}^{(k)}+t_k\mathbf{d}\right) \leq 0 \qquad i = 1, \ldots, m \tag{2.39}$$

Esta regra de busca linear imprecisa é comprovadamente eficiente para problemas sujeitos a restrições. Entretanto, se o valor de t_k for muito menor que a unidade, podem ser necessários vários passos da regra de Armijo, implicando em várias avaliações das funções do problema. Como discutido anteriormente, isso é inviável para problemas estruturais, sendo necessário adotar alguma modificação para aumentar a eficiência da busca linear. EVSUKOFF[20] propôs aplicar linearizações das restrições seguidas de interpolações quadráticas dos funcionais do problema para essa finalidade.

As soluções de problemas de otimização de estruturas geralmente se localizam no contorno da região viável, com algumas restrições ativas. Assim, uma estimativa razoável para t_k pode ser feita linearizando-se as restrições na direção $\mathbf{d}^{(k)}$ em torno do ponto $\mathbf{p}^{(k)}$ e adotando o valor que anula esta estimativa linear, como mostrado nas Figuras 2.4(a) e 2.5(a). Define-se então o seguinte conjunto de retas,

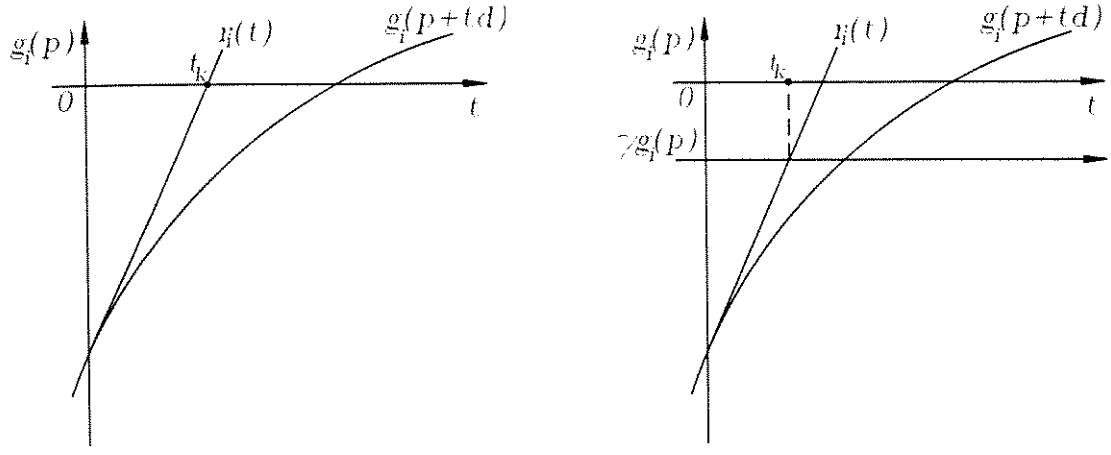
$$r_i(t) = \mathbf{d}^T \nabla g_i\left(\mathbf{p}^{(k)}\right) t + g_i\left(\mathbf{p}^{(k)}\right) \qquad i = 1, 2, \ldots, m \tag{2.40}$$

Para evitar que se venha a saturar alguma restrição precocemente (antes que $\|\mathbf{d}_0\|$ seja suficientemente pequeno), diminuindo a taxa de convergência, na prática não se toma t_k como o valor tal que $r_i(t) = 0$. Ao invés disso, desloca-se o eixo $g_i\left(\mathbf{p}^{(k)}+t\mathbf{d}\right) = 0$ para $g_i\left(\mathbf{p}^{(k)}+t\mathbf{d}\right) = \gamma g_i\left(\mathbf{p}^{(k)}\right)$, $\gamma \in (0,1)$, de acordo com as Figuras 2.4(b) e 2.5(b). Logo, toma-se t tal que $r_i(t) = \gamma g_i\left(\mathbf{p}^{(k)}\right)$. Considerando então todo conjunto de restrições do problema, tem-se então a expressão da estimativa inicial t_0 para t_k,

$$t_0 = t_0^I = \min\left(\frac{(\gamma - 1)\, g_i\left(\mathbf{p}^{(k)}\right)}{\mathbf{d}^T \nabla g_i\left(\mathbf{p}^{(k)}\right)}\right) \qquad i = 1, 2, \ldots, m \tag{2.41}$$

Observa-se que o parâmetro γ deve ser atualizado a cada passo de maneira que permita a saturação de restrições na solução. Assim a seqüência desse parâmetro deve ser controlada para que se anule apenas próximo à solução, ou seja, quando $\|\mathbf{d}_0\| \simeq \xi$, sendo ξ a precisão exigida.

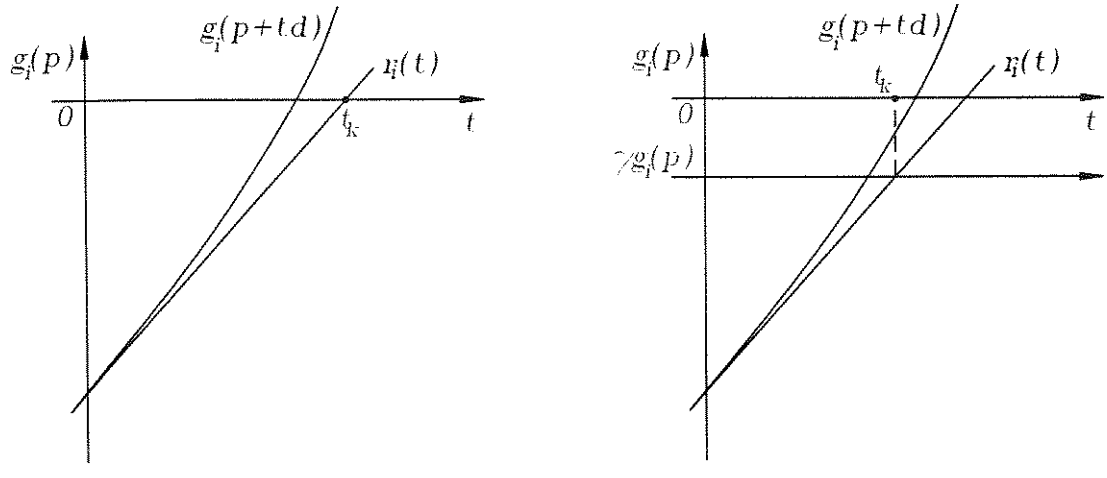
Deve-se então submeter a estimativa $\mathbf{p}^{(k)}+t_0\mathbf{d}$ a (2.38) e (2.39), respectivamente condição de decréscimo de $f(\mathbf{p})$ e condição de viabilidade. Avaliando as funções do problema em $\mathbf{p}^{(k)}+t_0\mathbf{d}$,



(a) Estimativa com eixo original.

(b) Estimativa com eixo deslocado.

Figura 2.4: Estimativa do tamanho de passo pela linearização das restrições. Função com crescimento menor que a estimativa linear.



(a) Estimativa com eixo original.

(b) Estimativa com eixo deslocado.

Figura 2.5: Estimativa do tamanho de passo pela linearização das restrições. Função com crescimento maior que a estimativa linear.

então pode ocorrer uma das seguintes situações:

1. o ponto respeita (2.38) e (2.39);
2. o ponto viola a condição de decréscimo (2.38);
3. o ponto viola (2.39), ou seja, $\mathbf{p}^{(k)}+t_0\mathbf{d}$ viola algumas ou todas as restrições;
4. o ponto viola (2.38) e (2.39).

Na situação 1, a estimativa é aceita, ou seja, $t_k = t_0$ e $\mathbf{p}^{(k+1)}= \mathbf{p}^{(k)}+t_k\mathbf{d}$; em conseqüência a avaliação de f e das restrições g_i é válida para o próximo passo. Na situação 2, usa-se a regra de Armijo para obter a nova estimativa de passo $t_1 = \nu t_0$, avaliando f em $\mathbf{p}^{(k)}+t_1\mathbf{d}$; toma-se t_k como o valor de t que minimiza a interpolação quadrática obtida de $f\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T\mathbf{C}\left(\mathbf{p}^{(k)}\right)$ e $f\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$. Na situação 3, avaliam-se as restrições violadas em $\mathbf{p}^{(k)}+t_1\mathbf{d}$, com $t_1 = \nu t_0$, e toma-se t_k como a menor raiz positiva das interpolações quadráticas obtidas de $g_i\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T\nabla g_i\left(\mathbf{p}^{(k)}\right)$ e $g_i\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$ para estas restrições. Na situação 4, toma-se t_k como o menor valor obtido dos dois procedimentos anteriores. Nas situações 2, 3 e 4, faz-se $\mathbf{p}^{(k+1)}= \mathbf{p}^{(k)}+t_k\mathbf{d}$ e avaliam-se f e as restrições g_i no novo ponto. Verificam-se novamente os critérios de aceitação do passo, repetindo-se as interpolações se necessário.

Atualizações

Com o novo projeto $\mathbf{p}^{(k+1)}$ já determinado, pode-se retornar ao programa de análise e avaliar os gradientes $\mathbf{C}\left(\mathbf{p}^{(k+1)}\right)$ e $\mathbf{A}\left(\mathbf{p}^{(k+1)}\right)$ para o próximo passo.

Resta estabelecer regras para atualizar $\mathbf{Q}$, $\boldsymbol{\lambda}$ e $\boldsymbol{\omega}^I$. Diferentes algoritmos podem ser obtidos de acordo com a maneira adotada para realizar esta atualização. Deve-se porém, respeitar as seguintes hipóteses:

Hipótese 2.5 *Devem existir números positivos λ_I, λ_S e ε tais que $0 \leq \lambda_i \leq \lambda_S$ para todo i e $\lambda_i \geq \lambda_I$ para qualquer i tal que $g_i(\mathbf{x}) + \varepsilon \geq 0$. Assim, todos os multiplicadores têm limite superior mas somente os multiplicadores das retrições potencialemte ativas têm limite inferior não-nulo.*

Hipótese 2.6 *Devem existir números positivos ζ_1 e ζ_2 tais que*

$$\zeta_1 |\mathbf{d}|^2 \leq \mathbf{d}^T\mathbf{Q}\mathbf{d} \leq \zeta_2 |\mathbf{d}|^2 \qquad \forall \mathbf{d} \in \Re^n$$

Hipótese 2.7 *Devem existir números positivos ω_1 e ω_2 tais que $\omega_1 \leq \omega_j^I \leq \omega_2$, $j = 1, 2, \ldots, m$.*

A matriz $\mathbf{Q}$ é uma matriz simétrica positiva-definida de ordem n e satisfazendo a Hipótese 2.6. Pode-se tomar $\mathbf{Q}$ igual a identidade ou adotar uma regra para estimar a matriz Hessiana $\mathbf{H}(\mathbf{p}, \boldsymbol{\lambda})$. Uma estimativa quasi-Newton seguindo os mesmos procedimentos de algoritmos *SQP* pode ser usado. Um exemplo é a regra BFGS com a modificação de Powell para garantir que as aproximações sejam positivas-definidas [48].

No caso das variáveis duais λ_i, a seguinte regra de atualização pode ser estabelecida,

$$\lambda_i = \sup\left[\lambda_{0i}; \epsilon \|\mathbf{d}_0\|^2\right] \qquad i = 1, 2, \ldots, m \qquad \epsilon > 0 \tag{2.42}$$

A regra (2.42) garante o limite inferior das variáveis duais da Hipótese 2.5. Ocorre que após um número finito de iterações, os multiplicadores das restrições ativas se igualam a λ_{0i}. E ainda, se as Hipóteses 2.6 e 2.7 forem satisfeitas, os λ_i obtidos pela regra (2.42) respeitam a Hipótese 2.5 e portanto também têm um limite superior.

Enquanto que a regra anterior funciona muito bem próxima da solução, longe desta é interessante que a direção de busca não aponte diretamente para nenhuma restrição, mas que se desvie do contorno de Ω. Isto pode ser obtido aumentando-se o multiplicador das restrições que estão próximas de zero nos passos iniciais. Então define-se o parâmetro $\delta \in (0,1)$ e $\delta > \xi$ tal que se $\|\mathbf{d}_0\|^2 \geq \delta$,

$$\lambda_i = -\frac{1}{g_i(\mathbf{p})} \qquad i = 1, 2, \ldots, m \tag{2.43}$$

De acordo com (2.32), a deflexão de $\mathbf{d}_0$ relativa a i-ésima restrição é proporcional a ω_i. Seria interessante incluir alguma informação de segunda ordem (curvatura) sobre as restrições para evitar que estas sejam saturadas antes da solução. Porém, em aplicações práticas de otimização é muito difícil ter esta informação disponível. Nestes casos, pode-se tentar algum tipo de estimativa sendo entretanto difícil se estabelecer uma regra geral. As implementações apresentadas em [20, 19] obtém resultados satisfatórios adotando $\omega_i^I \equiv 1$ $(i = 1, \ldots, m)$, constantes, em todas as iterações. Dessa forma, a deflexão relativa a i-ésima restrição é proporcional apenas a λ_i. De acordo com a regra de atualização de $\boldsymbol{\lambda}$, o valor dos multiplicadores de Lagrange de restrições próximas de estarem ativas tendem a ter valores maiores que os demais e portanto, mantém-se o efeito desejado.

2.4.2 Algoritmo Básico

A partir das considerações da seção anterior, pode-se apresentar um algoritmo básico de pontos interiores para problemas de otimização sujeitos apenas a restrições de desigualdade.

Passo 1. Inicialização.

Assumir valores convenientes para os parâmetros α, ξ, ν, η, $\gamma \in (0,1)$ e $\bar{\rho}$, $\epsilon > 0$. Escolher um projeto inicial viável, ou seja, $\mathbf{p}_0 \in \Omega$ e um conjunto de multiplicadores $\lambda_i > 0$ $(i = 1, \ldots, m)$ por exemplo $\lambda_i = -\frac{1}{g_i(\mathbf{p})}$. Escolher uma matriz real positiva definida $\mathbf{Q}$ de ordem $n \times n$ como estimativa da matriz Hessiana do Lagrangeano. Na ausência de maior informação, pode-se assumir $\mathbf{Q} \equiv \mathbf{I}$ como estimativa inicial.

Passo 2. Cálculo da Direção de Busca.

1. Contruir a matriz $\mathbf{R}$,

$$\mathbf{R} = \begin{bmatrix} \mathbf{Q} & \mathbf{A}^T \\ \boldsymbol{\Lambda}\mathbf{A} & \mathbf{G} \end{bmatrix}$$

e resolver os sistemas

$$\mathbf{R}\mathbf{v}_0 = \mathbf{b}_0 \qquad \text{e} \qquad \mathbf{R}\mathbf{v}_1 = \mathbf{b}_1$$

sendo

$$\mathbf{v}_0 = \begin{Bmatrix} \mathbf{d}_0 \\ \boldsymbol{\lambda}_0 \end{Bmatrix} \qquad \mathbf{b}_0 = \begin{Bmatrix} -\mathbf{C} \\ \mathbf{0} \end{Bmatrix} \qquad \mathbf{v}_1 = \begin{Bmatrix} \mathbf{d}_1 \\ \boldsymbol{\lambda}_1 \end{Bmatrix} \qquad \mathbf{b}_1 = \begin{Bmatrix} \mathbf{0} \\ -\boldsymbol{\lambda} \end{Bmatrix}$$

2. Se $\mathbf{d}_1 \cdot \mathbf{C} > 0$,

$$\rho = \frac{(\alpha - 1)\,\mathbf{d}_0 \cdot \mathbf{C}}{\mathbf{d}_1 \cdot \mathbf{C}}$$

caso contrário

$$\rho = \bar{\rho}$$

3. Calcular a direção de busca pela expressão,

$$\mathbf{d} = \mathbf{d}_0 + \rho \mathbf{d}_1$$

e ainda

$$\bar{\boldsymbol{\lambda}} = \boldsymbol{\lambda}_0 + \rho \boldsymbol{\lambda}_1$$

Passo 3. Busca Linear.

1. Cálculo da estimativa de tamanho de passo t_0

$$t_0 = \min \left(\frac{(\gamma - 1)\, g_i\left(\mathbf{p}^{(k)}\right)}{\mathbf{d}^T \nabla g_i\left(\mathbf{p}^{(k)}\right)} \right) \qquad i = 1, 2, \ldots, m$$

A partir do sistema (2.32),

$$\mathbf{d}^T \nabla g_i\left(\mathbf{p}^{(k)}\right) = \frac{-\rho \lambda_i \omega_i - g_i\left(\mathbf{p}^{(k)}\right) \bar{\lambda}_i}{\lambda_i} = -\rho \omega_i - g_i\left(\mathbf{p}^{(k)}\right) \frac{\bar{\lambda}_i}{\lambda_i}$$

evitando o cálculo de m produtos escalares.

2. Testar as condições de decréscimo e a viabilidade de $\mathbf{p}^{(k)} + t_0 \mathbf{d}$,

$$f\left(\mathbf{p}^{(k)} + t_0 \mathbf{d}\right) \leq f\left(\mathbf{p}^{(k)}\right) + \eta \mathbf{d}^T \mathbf{C}^{(k)} t_0$$

$$g_i\left(\mathbf{p}^{(k)} + t_0 \mathbf{d}\right) \leq 0 \qquad i = 1, \ldots, m$$

3. Se as duas condições forem respeitadas então assumir $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + t_0 \mathbf{d}$; ir para o passo 4 e utilizar as avaliações $f\left(\mathbf{p}^{(k)} + t_0 \mathbf{d}\right)$ e $g_i\left(\mathbf{p}^{(k)} + t_0 \mathbf{d}\right)$ para os cálculos da iteração seguinte.
Se, entretanto, alguma condição for violada nesse ponto utiliza-se a regra de Armijo para obter uma nova estimativa, ou seja, $t_1 = \nu t_0$; se a violação for na condição de decréscimo, toma-se t_k como o ponto mínimo da interpolação quadrática de f, obtida a partir de $f\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T \mathbf{C}^{(k)}$ e $f\left(\mathbf{p}^{(k)} + t_1 \mathbf{d}\right)$; se a violação for na condição de viabilidade, toma-se t_k como a menor raiz positiva das interpolações quadráticas obtidas dos valores $g_i\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T \nabla g_i\left(\mathbf{p}^{(k)}\right)$ e $g_i\left(\mathbf{p}^{(k)} + t_1 \mathbf{d}\right)$ das restrições violadas considerando o eixo deslocado para $(1 - \gamma)\, g_i\left(\mathbf{p}^{(k)}\right)$; se a interpolação quadrática conduzir a um ponto inviável toma-se t_k como a raiz da interpolação linear entre $g_i\left(\mathbf{p}^{(k)}\right)$ e $g_i\left(\mathbf{p}^{(k)} + t_1 \mathbf{d}\right)$ para o eixo deslocado $(1 - \gamma)\, g_i\left(\mathbf{p}^{(k)}\right)$. Se ambas as condições de decréscimo e viabilidade são violadas, toma-se t_k como o menor valor entre as duas estimativas anteriores. Verificam-se as condições de aceitação do passo, repetindo-se as interpolações até que sejam satisfeitas. Calcular $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + t_k \mathbf{d}$, avaliar $f\left(\mathbf{p}^{(k+1)}\right)$ e $g_i\left(\mathbf{p}^{(k+1)}\right)$ para os cálculos da iteração seguinte e ir para o passo 4.

Passo 4. Teste de Convergência.

Se $\|\mathbf{d}_0\|^2 > \xi$ e $\left\| \dfrac{f\left(\mathbf{p}^{(k+1)}\right) - f\left(\mathbf{p}^{(k)}\right)}{f\left(\mathbf{p}^{(k)}\right)} \right\| > \xi$ ir para o passo 5. Senão encerre o processo, sendo $\mathbf{p}^{(k+1)}$ a solução ótima obtida pelo algoritmo para o problema.

Passo 5. Atualizações.

1. Calcular os gradientes $\mathbf{C}\left(\mathbf{p}^{(k+1)}\right)$ e $\mathbf{A}\left(\mathbf{p}^{(k+1)}\right)$ no novo ponto.

2. Atualizar a matriz $\mathbf{Q}$ pela regra BFGS.
3. Se $\|\mathbf{d}_0\|^2 \geq \delta$ então

$$\lambda_i = -\frac{1}{g_i(\mathbf{p})} \qquad i = 1, 2, \ldots, m$$

senão

$$\lambda_i = \sup\left[\lambda_{0i}; \epsilon \|\mathbf{d}_0\|^2\right] \qquad i = 1, 2, \ldots, m$$

4. Se $\|\mathbf{d}_0\|^2 < \gamma$ então $\gamma = \|\mathbf{d}_0\|^2$.
5. $k = k + 1$.
6. Ir para o passo 2.

2.4.3 Inclusão de Restrições de Igualdade

É possível extender o algoritmo anterior para tratar o problema de otimização geral (1.11) onde estão presentes também restrições de igualdade. As modificações no algoritmo básico são baseadas em duas idéias: modificação do problema original convertendo as restrições de igualdade em restrições de desigualdades e a aplicação de uma função de penalização baseada nos valores destas restrições convertidas, forçando que estejam ativas na solução do problema modificado e assim recuperando a solução do problema original.

Em adição à notação (2.22) considere,

$$\begin{array}{ll} \mathbf{h}(\mathbf{p}) : \Re^n \longrightarrow \Re^p & \text{Vetor de restrições de igualdade} \\ \mathbf{L}(\mathbf{p}) : \Re^n \longrightarrow \Re^{p \times n} & \text{Gradiente de } \mathbf{h}(\mathbf{p}), \text{ ou seja, } L_{ij}(\mathbf{p}) = \frac{\partial h_i}{\partial x_j}(\mathbf{p}) \end{array} \tag{2.44}$$

Assim, as condições necessárias de otimalidade dadas por (2.2) a (2.6) podem ser reescritas como,

$$\mathbf{C}(\mathbf{p}) + \mathbf{A}^T(\mathbf{p})\boldsymbol{\lambda} + \mathbf{L}^T(\mathbf{p})\boldsymbol{\mu} = \mathbf{0} \tag{2.45}$$

$$\mathbf{G}(\mathbf{p})\boldsymbol{\lambda} = \mathbf{0} \tag{2.46}$$

$$\mathbf{h}(\mathbf{p}) = \mathbf{0} \tag{2.47}$$

$$g_i(\mathbf{p}) \leq 0 \qquad i = 1, \ldots, m \tag{2.48}$$

$$\lambda_j \geq 0 \qquad j = 1, \ldots, m \tag{2.49}$$

O algoritmo geral emprega uma técnica iterativa de ponto fixo de Newton na solução direta do sistema de equações não-lineares formado por (2.45), (2.46) e (2.47) para obter a direção de busca a cada iteração. Assim como no algoritmo básico, a busca linear deve ser tal que as inequações (2.48) e (2.49) sejam respeitadas a cada iteração. Observa-se que as condições necessárias não impõem nenhuma restrição ao sinal dos multiplicadores μ_j das restrições de igualdade.

Determinação da Direção de Busca

Aplicando o método de Newton no sistema formado por (2.45), (2.46) e (2.47) tem-se a seguinte fórmula iterativa, com $\boldsymbol{\Delta}\mathbf{p}^{(k)} = \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)}$,

$$\begin{cases} \mathbf{Q}^{(k)}\boldsymbol{\Delta}\mathbf{p}^{(k)} + \mathbf{A}^T\left(\mathbf{p}^{(k)}\right)\boldsymbol{\lambda}^{(k+1)} + \mathbf{L}^T\left(\mathbf{p}^{(k)}\right)\boldsymbol{\mu}^{(k+1)} = -\mathbf{C}\left(\mathbf{p}^{(k)}\right) \\ \boldsymbol{\Lambda}^{(k)}\mathbf{A}\left(\mathbf{p}^{(k)}\right)\boldsymbol{\Delta}\mathbf{p}^{(k)} + \mathbf{G}\left(\mathbf{p}^{(k)}\right)\boldsymbol{\lambda}^{(k+1)} = \mathbf{0} \\ \mathbf{L}\left(\mathbf{x}^{(k)}\right)\boldsymbol{\Delta}\mathbf{p}^{(k)} = -\mathbf{h}\left(\mathbf{x}^{(k)}\right) \end{cases} \tag{2.50}$$

De forma simplificada,

$$\begin{cases} \mathbf{Q}\mathbf{d}_0 + \mathbf{A}^T\boldsymbol{\lambda}_0 + \mathbf{L}^T\boldsymbol{\mu}_0 = -\mathbf{C} \\ \boldsymbol{\Lambda}\mathbf{A}\mathbf{d}_0 + \mathbf{G}\boldsymbol{\lambda}_0 = \mathbf{0} \\ \mathbf{L}\mathbf{d}_0 = -\mathbf{h} \end{cases} \tag{2.51}$$

Como no algoritmo anterior, introduz-se o sistema

$$\begin{cases} \mathbf{Q}\mathbf{d}_1 + \mathbf{A}^T\boldsymbol{\lambda}_1 + \mathbf{L}^T\boldsymbol{\mu}_1 = \mathbf{0} \\ \boldsymbol{\Lambda}\mathbf{A}\mathbf{d}_1 + \mathbf{G}\boldsymbol{\lambda}_1 = -\boldsymbol{\Lambda}\boldsymbol{\omega}^I \\ \mathbf{L}\mathbf{d}_1 = -\boldsymbol{\omega}^E \end{cases} \tag{2.52}$$

e assim

$$\mathbf{d} = \mathbf{d}_0 + \rho\mathbf{d}_1$$

Considere agora a seguinte função objetivo penalizada,

$$\varphi(\mathbf{p}) = f(\mathbf{p}) + \sum_{j=1}^{p} c_j \left|h_j(\mathbf{p})\right| \tag{2.53}$$

onde $c_j > 0 \quad (j = 1, \ldots, p)$. Impõe-se uma penalização sempre que as funções $h_j(\mathbf{p})$ sejam diferentes de zero, ou seja, para valores convenientes de c_j, exigir decréscimo de $\varphi(\mathbf{p})$ a cada passo força que as restrições de igualdade sejam saturadas na solução. Dessa forma, $\varphi(\mathbf{p})$ passa a ser a nova função objetivo do problema.

Pode-se mostrar que (2.53) é uma função de penalização exata para valores c_j suficientemente grandes. Em outras palavras, existe um conjunto de c_j finitos tal que o mínimo de $\varphi(\mathbf{p})$ submetido apenas a restrições de desigualdade é a solução do problema original (1.11). Numericamente é interessante utilizar este tipo de função de penalização, já que não é necessário utilizar seqüências de parâmetros de penalização indo para infinito. Entretanto, (2.53) tem a desvantagem de não ser suave quando as restrições de igualdade se tornam ativas, sendo novamente necessário evitar a saturação das restrições.

Dessa forma, serão considerados como "pontos interiores", todos aqueles que pertencerem à região viável extendida $\tilde{\Omega}$ definida como,

$$\tilde{\Omega} = \{\mathbf{p} \in \Re^n | g_i(\mathbf{p}) \leq 0,\ i = 1, 2, \ldots, m \text{ e } h_j(\mathbf{p}) \leq 0,\ j = 1, 2, \ldots, p\} \tag{2.54}$$

Demonstra-se que a direção de busca $\mathbf{d}_0$ obtida de (2.51) é uma direção de decréscimo de (2.53) desde que $c_j > |\mu_{0j}|$ para todo j [17]. Logo, aplicando (2.36) juntamente com (2.53) tem-se,

$$\rho = \frac{(\alpha - 1)\,\mathbf{d}_0 \cdot \nabla\varphi}{\mathbf{d}_1 \cdot \nabla\varphi} \qquad \text{se} \qquad \mathbf{d}_1 \cdot \nabla\varphi > 0 \tag{2.55}$$

e assim $\mathbf{d}$ também é uma direção de decréscimo de $\varphi(\mathbf{p})$.

Nas implementações do algoritmo, aplica-se a seguinte regra para a determinação dos parâmetros de penalização:

$$\text{Se } c_j < 1.2\,|\mu_{0j}| \qquad \text{então } c_j = 2.0\,|\mu_{0j}| \qquad j = 1, \ldots, m \tag{2.56}$$

E ainda, como não se impõe nenhuma restrição ao sinal de μ_{0j} então o peso $\boldsymbol{\omega}^E$ do sistema (2.52), que deve ser estritamente positivo, é implementado como,

$$\omega_j^E = |\mu_{0j}| \qquad j = 1, \ldots, m \tag{2.57}$$

Busca Linear

Aplica-se o mesmo procedimento de busca linear do algoritmo básico introduzindo as modificações necessárias, ou seja, deve-se determinar um tamanho de passo que garanta o decréscimo da função penalizada $\varphi(\mathbf{p})$ e que os pontos gerados pelo algoritmo pertençam a $\tilde{\Omega}$.

Atualizações

As estratégias de atualização são mantidas inalteradas. Isso significa também considerar as restrições de igualdade na atualização quasi-Newton de $\mathbf{Q}$.

Como os sistemas (2.51) e (2.52) são independentes de $\mu^{(k)}$ não é necessário estabelecer nenhuma regra de atualização para os multiplicadores de Lagrange das restrições de igualdade para o funcionamento do algoritmo. Assim, na solução, deve-se considerar $\mu^{(k)} = \mu_0$.

2.4.4 Algoritmo Geral

A partir das considerações da seção anterior pode-se apresentar um algoritmo básico de pontos interiores para problemas de otimização genéricos, sujeitos à restrições de igualdade e desigualdade:

Passo 1. Inicialização.
Assumir valores convenientes para os parâmetros α, ξ, ν, η, $\gamma \in (0,1)$ e $\bar{\rho}$, $\epsilon > 0$. Escolher um projeto inicial viável, ou seja, $\mathbf{p}_0 \in \Omega$ e um conjunto de multiplicadores $\lambda_i > 0$ $(i = 1, \ldots, m)$, por exemplo, $\lambda_i = -\frac{1}{g_i(\mathbf{p})}$. Escolher uma matriz real positiva definida $\mathbf{Q}$ de ordem $n \times n$ como estimativa da matriz Hessiana do Lagrangeano. Na ausência de maior informação, pode-se assumir $\mathbf{Q} \equiv \mathbf{I}$ como estimativa inicial.

Passo 2. Cálculo da Direção de Busca.

1. Contruir a matriz $\mathbf{R}$,

$$\mathbf{R} = \begin{bmatrix} \mathbf{Q} & \mathbf{A}^T & \mathbf{L}^T \\ \mathbf{\Lambda A} & \mathbf{G} & \mathbf{0} \\ \mathbf{L} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

e resolver os sistemas

$$\mathbf{R}\mathbf{v}_0 = \mathbf{b}_0 \quad \text{e} \quad \mathbf{R}\mathbf{v}_1 = \mathbf{b}_1$$

sendo

$$\mathbf{v}_0 = \begin{Bmatrix} \mathbf{d}_0 \\ \boldsymbol{\lambda}_0 \\ \boldsymbol{\mu}_0 \end{Bmatrix} \quad \mathbf{b}_0 = \begin{Bmatrix} -\mathbf{C} \\ \mathbf{0} \\ -\mathbf{h} \end{Bmatrix} \quad \mathbf{v}_1 = \begin{Bmatrix} \mathbf{d}_1 \\ \boldsymbol{\lambda}_1 \\ \boldsymbol{\mu}_1 \end{Bmatrix} \quad \mathbf{b}_1 = \begin{Bmatrix} \mathbf{0} \\ -\boldsymbol{\lambda} \\ -\boldsymbol{\mu} \end{Bmatrix}$$

2. Se $\mathbf{d}_1 \cdot \nabla\varphi > 0$,

$$\rho = \frac{(\alpha - 1)\, \mathbf{d}_0 \cdot \nabla\varphi}{\mathbf{d}_1 \cdot \nabla\varphi}$$

caso contrário

$$\rho = \bar{\rho}$$

3. Calcular a direção de busca pela expressão,

$$\mathbf{d} = \mathbf{d}_0 + \rho \mathbf{d}_1$$

e ainda

$$\bar{\lambda} = \lambda_0 + \rho \lambda_1$$

Passo 3. Busca Linear.

1. Cálculo da estimativa de tamanho de passo t_0

$$t_0^I = \min\left(\frac{(\gamma-1)\,g_i\left(\mathbf{p}^{(k)}\right)}{\mathbf{d}^T\nabla g_i\left(\mathbf{p}^{(k)}\right)}\right) \qquad i = 1,2,\ldots,m$$

$$t_0^E = \min\left(\frac{(\gamma-1)\,h_j\left(\mathbf{p}^{(k)}\right)}{\mathbf{d}^T\nabla h_j\left(\mathbf{p}^{(k)}\right)}\right) \qquad j = 1,2,\ldots,p$$

$$t_0 = \min\left\{t_0^I, t_0^E\right\}$$

onde, a partir do sistema (2.51),

$$\mathbf{d}^T\nabla g_i\left(\mathbf{p}^{(k)}\right) = \frac{-\rho\lambda_i\omega_i^I - g_i\left(\mathbf{p}^{(k)}\right)\bar{\lambda}_i}{\lambda_i} = -\rho - g_i\left(\mathbf{p}^{(k)}\right)\frac{\bar{\lambda}_i}{\lambda_i}$$

$$\mathbf{d}^T\nabla h_j\left(\mathbf{p}^{(k)}\right) = -\rho\omega_j^E = -\rho\,|\mu_{0j}|$$

evitando o cálculo de $m+p$ produtos escalares.

2. Testar a condição de decréscimo e a viabilidade de $\mathbf{p}^{(k)}+t_0\mathbf{d}$,

$$\varphi\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right) \le \varphi\left(\mathbf{p}^{(k)}\right) + \eta\mathbf{d}^T\nabla\varphi\left(\mathbf{p}^{(k)}\right)t_0$$

$$g_i\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right) \le 0 \qquad i = 1,\ldots,m$$

$$h_j\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right) \le 0 \qquad j = 1,\ldots,p$$

3. Se as duas condições forem respeitadas então assumir $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)}+t_0\mathbf{d}$; ir para o passo 4 e utilizar as avaliações $f\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right)$, $g_i\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right)$ e $h_j\left(\mathbf{p}^{(k)}+t_0\mathbf{d}\right)$ cálculos da iteração seguinte.
Se, entretanto, alguma condição for violada nesse ponto, utiliza-se a regra de Armijo para obter uma nova estimativa, ou seja, $t_1 = \nu t_0$; se a violação for na condição de decréscimo, toma-se t_k como o ponto mínimo da interpolação quadrática de φ, obtida a partir de $\varphi\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T\nabla\varphi\left(\mathbf{p}^{(k)}\right)$ e $\varphi\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$; se a violação for na condição de viabilidade, toma-se t_k como a menor raiz positiva das interpolações quadráticas obtidas a partir dos valores $g_i\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T\nabla g_i\left(\mathbf{p}^{(k)}\right)$ e $g_i\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$ considerando o eixo deslocado para $(1-\gamma)\,g_i\left(\mathbf{p}^{(k)}\right)$ para restrições de desigualdade violadas ou $h_j\left(\mathbf{p}^{(k)}\right)$, $\mathbf{d}^T\nabla h_j\left(\mathbf{p}^{(k)}\right)$ e $h_j\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$ considerando o eixo deslocado para $(1-\gamma)\,h_j\left(\mathbf{p}^{(k)}\right)$ para restrições de igualdade "violadas"[6]; se a interpolação quadrática conduzir a um ponto inviável toma-se t_k como menor a raiz da interpolação linear entre $g_i\left(\mathbf{p}^{(k)}\right)$ e $g_i\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$ para o eixo deslocado $(1-\gamma)\,g_i\left(\mathbf{p}^{(k)}\right)$ ou $h_j\left(\mathbf{p}^{(k)}\right)$ e $h_j\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right)$ para o eixo deslocado $(1-\gamma)\,h_j\left(\mathbf{p}^{(k)}\right)$. Se ambas as condições de decréscimo e viabilidade são violadas, toma-se t_k como o menor valor entre as duas estimativas anteriores. Verificam-se as condições de aceitação do passo, repetindo-se as interpolações até que sejam satisfeitas. Calcular $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)}+t_k\mathbf{d}$, avaliar $f\left(\mathbf{p}^{(k+1)}\right)$, $g_i\left(\mathbf{p}^{(k+1)}\right)$e $h_j\left(\mathbf{p}^{(k+1)}\right)$ para os cálculos da iteração seguinte e ir para o passo 4.

[6]O sentido de violação para a busca linear significa que o ponto está fora da região viável estendida $\tilde{\Omega}$.

Passo 4. Teste de Convergência.

Se $\|\mathbf{d}_0\|^2 > \xi$ e $\left\| \dfrac{f\left(\mathbf{p}^{(k+1)}\right) - f\left(\mathbf{p}^{(k)}\right)}{f\left(\mathbf{p}^{(k)}\right)} \right\| > \xi$ ir para o passo 5. Senão encerre o processo, sendo $\mathbf{p}^{(k+1)}$ a solução ótima obtida pelo algoritmo para o problema.

Passo 5. Atualizações.

1. Calcular os gradientes $\mathbf{C}\left(\mathbf{p}^{(k+1)}\right)$, $\mathbf{A}\left(\mathbf{p}^{(k+1)}\right)$ e $\mathbf{L}\left(\mathbf{p}^{(k+1)}\right)$ no novo ponto.
2. Atualizar a matriz $\mathbf{Q}$ pela regra BFGS.
3. Se $\|\mathbf{d}_0\|^2 \geq \delta$ então

$$\lambda_i = -\frac{1}{g_i(\mathbf{p})} \qquad i = 1, 2, \ldots, m$$

senão

$$\lambda_i = \sup\left[\lambda_{0i}; \epsilon \|\mathbf{d}_0\|^2\right] \qquad i = 1, 2, \ldots, m$$

4. Se $\|\mathbf{d}_0\|^2 < \gamma$ então $\gamma = \|\mathbf{d}_0\|^2$.
5. $k = k + 1$.
6. Ir para o passo 2.

2.4.5 Parâmetros do Algoritmo de Herskovits

A execução do algoritmo de Herskovits depende do ajuste de vários parâmetros. Apesar do valor ótimo desses parâmetros depender muito do problema tratado, é possível estabelecer algumas orientações baseando-se no significado de cada constante e nos resultados de alguns testes:

- **α :** O parâmetro $\alpha \in (0, 1)$ controla a ineficiência permitida na deflexão da direção de busca original $\mathbf{d}_0$ quando a direção de deflexão $\mathbf{d}_1$ não for uma direção de decréscimo. Dessa forma, quanto menor o valor desse parâmetro menor será o valor da derivada direcional $\mathbf{d} \cdot \mathbf{C}$ ou $(\mathbf{d} \cdot \nabla\varphi)$; quanto maior, maior a declividade da direção de busca. Como citado anteriormente, deve ser ajustado de acordo com o grau de não-linearidade do contorno da região viável, evitando que as estimativas de passo t_0 obtidas por linearização das restrições não sejam seguidamente rejeitadas pelo critério de viabilidade. Isso resulta em passos pequenos e na necessidade de um maior número de avaliações das funções de restrição, reduzindo a eficiência global do processo. Assim, um valor menor de α pode resultar na viabilidade de passos maiores.

- **ξ :** O parâmetro $\xi > 0$ indica a precisão exigida para a solução do problema, ou seja, é usado como critério de parada para o processo iterativo. Seu valor é comparado com $\|\mathbf{d}_0\|^2$ e com a diferença percentual entre dois valores sucessivos da função objetivo. Dessa forma, deve-se avaliar o significado físico da precisão exigida, levando-se em conta as unidades utilizadas.

- **$\bar{\rho}$:** A constante $\bar{\rho} > 0$ é utilizada quando a direção $\mathbf{d}_1$ também for uma direção de decréscimo da função objetivo. Essa situação pode ocorrer nas primeiras iterações. Adota-se, em geral, $\bar{\rho} = 1$.

- ν : O parâmetro $\nu \in (0,1)$ controla a diminuição das estimativas de passo na regra de Armijo. Na busca linear dos algoritmos das seções 2.4.2 e 2.4.4 este parâmetro é utilizado apenas para fornecer um novo ponto para a interpolação quadrática da função objetivo ou das restrições violadas. Dessa forma, este parâmetro tem pouca influência na evolução do processo iterativo. Valores de 0.8 ou 0.7 são geralmente utilizados.

- η : A constante $\eta \in (0,1)$ é utilizada na regra de Armijo para determinar o decréscimo mínimo aceitável da função objetivo na busca linear inexata. Se $\eta = 0$ então nenhum decréscimo é exigido, ao passo que se $\eta = 1$, exige-se um nível de decréscimo que somente pode ser atingido se a função objetivo for linear. Em problemas altamente não-lineares, em geral se utiliza valores de η próximos de 0 já que a condição de viabilidade costuma ser muito restritiva.

- γ : O parâmetro $\gamma \in (0,1)$ é utilizado para evitar a saturação das restrições na etapa de interpolação linear. Seu valor é importante apenas nas iterações iniciais pois a certa altura passa a ser atualizada pelo valor de $\|\mathbf{d}_0\|$. Geralmente, toma-se próximo de 0 favorecendo passos maiores.

- ϵ : O parâmetro $\epsilon > 0$ é utilizado na regra de atualização das variáveis duais λ_i para forçar seus valores negativos para zero mais rapidamente. Entretanto, como não há um procedimento fechado para essa atualização, não há uma regra para se estimar ϵ. Na implementação de [20] é utilizado $\epsilon = 0.1$, deixando claro que valores como 0.01 e 0.001 podem ser utilizados, dependendo mais uma vez das unidades utilizadas.

- δ : O parâmetro $\delta > 0$ é um dos mais influentes na convergência do algoritmo pois controla o tipo de atualização das variáveis duais das restrições de desigualdade. Em outras palavras, esse parâmetro julga se o ponto atual está suficientemente próximo do contorno da região viável. Seu valor depende muito do problema e das unidades, sendo necessário avaliar como $\|\mathbf{d}_0\|^2$ evolui ao longo do processo e a precisão exigida para a solução.

2.5 Estudo de Casos

A seguir, apresentam-se alguns exemplos da aplicação do algoritmo de minimização de Herskovits.

2.5.1 Minimização de Volume de uma Viga — Restrições de Desigualdade

Busca-se, neste exemplo, a minimização do volume de uma viga de seção transversal constante (base b e altura h, como mostrado na Figura 2.6) submetida a um momento fletor $M = 4 \times 10^4$ kN·mm e a um esforço cortante $V = 75$ kN, mantendo seu comprimento fixo [48]. As condições de contorno da viga são tais que as tensões máximas de flexão e cisalhamento são, respectivamente, $6M/bh^2$ e $3V/bh$. Exige-se que as tensões não ultrapassem 10 MPa na flexão e 2 MPa no cisalhamento (restrições g_1 e g_2). Além disso, a altura h não deve exceder o dobro da largura (restrição g_3). Impõem-se ainda os limites 10 mm $\leq b, h \leq$ 1000 mm (restrições g_4 a g_7).

Considerando densidade e comprimento da viga constantes, deve-se minimizar a área da seção transversal. As variáveis de projeto são claramente b e h e a função objetivo do problema é a fórmula que define a área em função das variáveis b e h, ou seja, $f(b,h) = bh$. As restrições

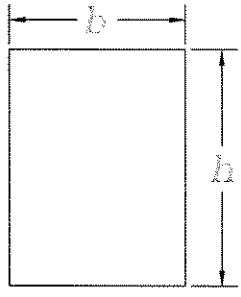


Figura 2.6: Dimensões b e h da seção transversal retangular de viga.

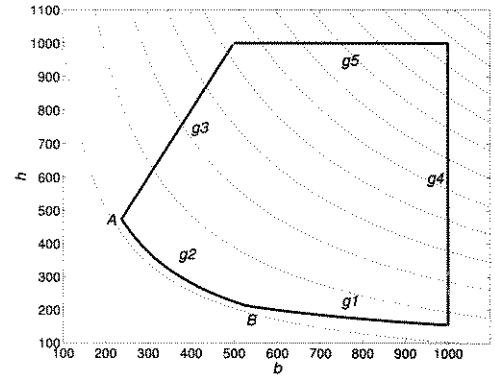


Figura 2.7: Curvas de nível de $f(b,h) = bh$ e região viável Ω delimitada pelas funções g_i.

g_1 a g_7 têm a seguinte forma,

$$g_1(b,h) = 6\left(4 \times 10^4\right)/bh^2 \leq 0.01 \qquad g_4(b,h) = b \leq 1000.0$$
$$g_2(b,h) = 3(75)/bh \leq 0.002 \qquad g_5(b,h) = h \leq 1000.0$$
$$g_3(b,h) = h \leq 2b \qquad g_6(b,h) = b \geq 10.0$$
$$g_7(b,h) = h \geq 10.0$$

Escrevendo o problema na forma geral (1.11) com as restrições normalizadas tem-se,

$$\min f(b,h) = bh$$
$$\text{sujeita à} \quad g_1(b,h) = \frac{2.4\times10^7}{bh^2} - 1.0 \leq 0 \qquad g_4(b,h) = \frac{b}{1000.0} - 1.0 \leq 0$$
$$g_2(b,h) = \frac{1.125\times10^5}{bh} - 1.0 \leq 0 \qquad g_5(b,h) = \frac{h}{1000.0} - 1.0 \leq 0$$
$$g_3(b,h) = \frac{h}{2b} - 1.0 \leq 0 \qquad g_6(b,h) = 1.0 - \frac{b}{10.0} \leq 0$$
$$g_7(b,h) = 1.0 - \frac{h}{10.0} \leq 0$$

como citado anteriormente, a normalização das restrições é um meio de facilitar a comparação numérica entre as restrições (observe que todas as restrições são comparadas com o valor 1.0) eliminando as diferenças entre ordens de grandezas provenientes das unidades utilizadas para cada característica física que precise ser controlada. Neste exemplo também pode ser observada uma dificuldade introduzida pela normalização das restrições: a restrição g_3, antes linear, ao ser normalizada torna-se uma restrição não-linear.

A solução do problema é qualquer ponto viável com a restrição g_2 ativa (linha $A - B$ da Figura 2.7), onde $f(b,h) = 112500.000$ mm^2. Aplicando o algoritmo de Herskovits foram obtidos os seguintes resultados numéricos.

Solução:

- Parâmetros: $\alpha = 0.5$, $\xi = 10^{-5}$, $\bar{\rho} = 1.0$, $\nu = 0.8$, $\eta = 0.2$, $\gamma = 0.1$, $\epsilon = 0.1$, $\delta = 0.1$.
- Ponto inicial: $b = 500.0$ mm, $h = 900.0$ mm.
- Valor da função objetivo no ponto inicial: 450000.000 mm^3.
- Resultados:

  Ponto ótimo: $b = 349.33386599$ mm, $h = 322.04149397$ mm.

  Valor da função objetivo no ponto ótimo: 112500.00009745 mm^3.

  Número de iterações: 10.

  Número de total avaliações da função objetivo: 13.

Tabela 2.1: Valor final das restrições e multiplicadores Lagrange (restrições de desigualdade).

Restr.	valor final	λ	valor final
$g_1 =$	-3.375595e-01	$\lambda_1 =$	2.980719e-04
$g_2 =$	-8.662604e-10	$\lambda_2 =$	1.125047e+05
$g_3 =$	-5.390634e-01	$\lambda_3 =$	9.495071e-05
$g_4 =$	-6.506661e-01	$\lambda_4 =$	1.000000e+00
$g_5 =$	-6.779585e-01	$\lambda_5 =$	1.000000e+00
$g_6 =$	-3.393339e+01	$\lambda_6 =$	1.000000e+00
$g_7 =$	-3.120415e+01	$\lambda_7 =$	1.000000e+00

Número de avaliações das funções de restrição: 16.

Número de avaliações dos gradientes das funções: 10.

A Tabela 2.1 mostra o valor final das restrições e dos multiplicadores. As Figuras 2.8(a), 2.8(b), 2.8(c) e 2.8(d) mostram a seqüência de projetos gerados pelo algoritmo na região Ω, a evolução da função objetivo e das variáveis b e d durante as iterações.

A Figura 2.8(a) exemplifica como a deflexão da direção de busca, evitando saturar restrições não-relevantes, aumenta a eficiência global da otimização. Observa-se que ao se desviar da restrição g_3 na iteração 4, o passo na direção de busca pode ser grande o suficiente para se aproximar da restrição g_2, fazendo com que o maior decréscimo da função objetivo durante o processo ocorresse nessa iteração. Além disso, pode-se observar, nos demais gráficos deste exemplo, a convergência da solução a partir da iteração 6.

2.5.2 Minimização do Volume de uma Viga — Restrições de Igualdade e Desigualdade

Pode-se modificar o problema apresentado na Seção 2.5.1 convertendo a restrição $g_3 = h/2b - 1.0 \leq 0$ numa restrição de igualdade (h_1). Colocando este novo problema na forma geral (1.11) tem-se,

$$\min f(b,h) = bh$$

sujeita à

$$g_1(b,h) = \frac{2.4\times10^7}{bh^2} - 1.0 \leq 0 \qquad g_4(b,h) = \frac{h}{1000.0} - 1.0 \leq 0$$

$$g_2(b,h) = \frac{1.125\times10^5}{bh} - 1.0 \leq 0 \qquad g_5(b,h) = 1.0 - \frac{b}{10.0} \leq 0$$

$$g_3(b,h) = \frac{b}{1000.0} - 1.0 \leq 0 \qquad g_6(b,h) = 1.0 - \frac{h}{10.0} \leq 0$$

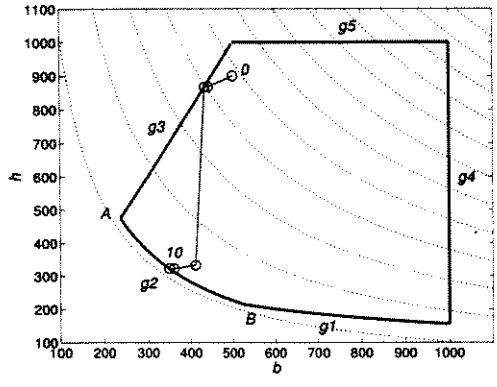
$$h_1(b,h) = \frac{h}{2b} - 1.0 = 0$$

A solução desse problema é o ponto A, intersecção entre g_2 e h_1 (que são ativas na solução), mostrado na Figura 2.7, ou seja, $b = 237.17082451$ mm e $h = 474.34164903$ mm. A dificuldade deste problema consiste em atingir o ponto A sendo que nos pontos da linha $A - B$ a função objetivo tem mesmo valor. A seguinte solução numérica foi obtida.

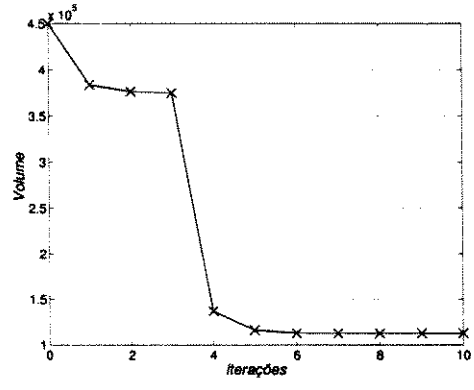
Solução:

- Parâmetros: $\alpha = 0.05$, $\xi = 10^{-7}$, $\bar{\rho} = 1.0$, $\nu = 0.8$, $\eta = 0.2$, $\gamma = 0.1$, $\epsilon = 0.1$, $\delta = 0.1$.
- Ponto inicial: $b = 900.0$ mm, $h = 450.0$ mm.
- Valor da função objetivo no ponto inicial: 405000.000 mm^3.
- Resultados:

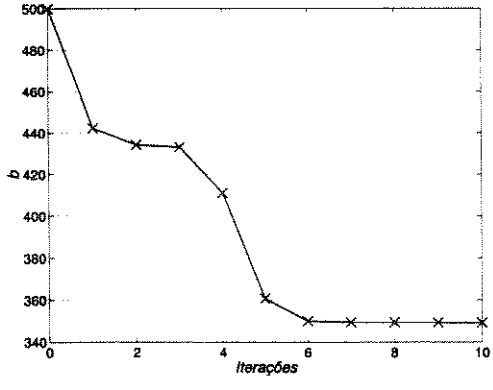
Ponto ótimo: $b = 237.17108439$ mm, $h = 474.34215707$ mm.



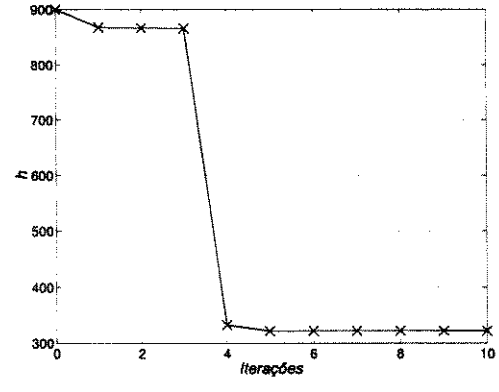
(a) Seqüência de projetos na região viável.



(b) Valor da função objetivo.



(c) Variável b.



(d) Variável h.

Figura 2.8: Processo de otimização do volume da viga (restrições de desigualdade).

Tabela 2.2: Valor final das restrições e multiplicadores Lagrange (restrições de igualdade e desigualdade).

Restr.	valor final	λ	valor final
$g_1 =$	-5.502553e-01	$\lambda_1 =$	1.000000e+00
$g_2 =$	-2.166805e-06	$\lambda_2 =$	1.308915e-07
$g_3 =$	-7.628289e-01	$\lambda_3 =$	1.000000e+00
$g_4 =$	-5.256578e-01	$\lambda_4 =$	1.000000e+00
$g_5 =$	-2.271711e+01	$\lambda_5 =$	7.457126e-07
$g_6 =$	-4.643422e+01	$\lambda_6 =$	7.281173e-07
$h_1 =$	-2.468669e-08	$\mu_1 =$	-4.564688e+04

Valor da função objetivo no ponto ótimo: 112500.244 mm^3.

Número de iterações: 35.

Número de total avaliações da função objetivo: 99.

Número de avaliações das funções de restrição: 93.

Número de avaliações dos gradientes das funções: 35.

A Tabela 2.2 mostra o valor final das restrições e dos multiplicadores. A Figuras 2.9(a), 2.9(b), 2.9(c) e 2.9(d) mostram a seqüência de projetos gerados pelo algoritmo na região Ω, a evolução da função objetivo e das variáveis b e d durante as iterações.

Como se observa nas Figuras 2.9(a) a 2.9(d), a existência de restrições de igualdade faz com que o processo de solução apresente oscilações nas iterações iniciais, antes de apresentar um progresso direcionado para a solução. Isto ocorre pois são impostas condições mais severas a solução deste tipo de problemas se comparados a problemas com restrições de desigualdade apenas. Além disso, como a função de decréscimo não é apenas a função objetivo mas a penalização desta em relação à violação das restrições de igualdade, o valor da função objetivo pode sofrer acréscimo.

2.5.3 Minimização do Volume de uma Viga Engastada

Uma viga engastada de seção transversal retangular e comprimento $l = 2$ m está submetida ao carregamento concentrado $F = 100$ kN em sua extremidade, como mostrado na Figura 2.10. As propriedades do material da viga são o módulo de elasticidade $\tilde{E} = 200$ GPa e a tensão normal admissível de $\sigma_a = 250$ MPa (g_1). Devido às condições de contorno, a tensão máxima normal na viga é $6Fl/bh^2$. As variáveis de projeto são a base b e a altura h da seção transversal que estão restritas aos seguintes intervalos: 0.04 m $\leq b \leq$ 0.2 m e $h \leq 0.2$ m (g_2 a g_4). Mantendo o comprimento fixo, deve-se determinar a viga de menor peso que respeite as restrições de projeto.

As restrições do problema são,

$g_1(b,h) = 6(2)(100 \times 10^3)/bh^2 \leq 250 \times 10^6$ $\quad$ $g_3(b,h) = b \leq 0.2$

$g_2(b,h) = h \leq 0.2$ $\quad$ $g_4(b,h) = b \geq 0.04$

Colocando o problema na forma geral (1.11) com restrições normalizadas tem-se,

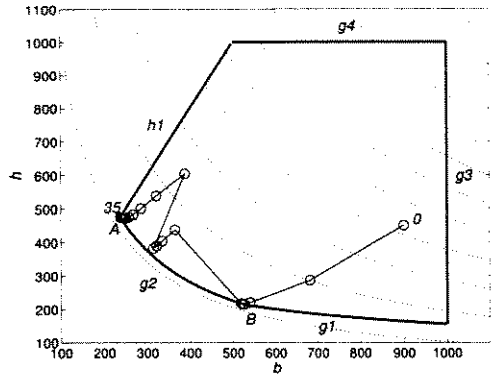
$\min f(b,h) = 2bh$

sujeita à $\quad$ $g_1(b,h) = \frac{0.0048}{bh^2} - 1.0 \leq 0$ $\quad$ $g_3(b,h) = 5b - 1.0 \leq 0$

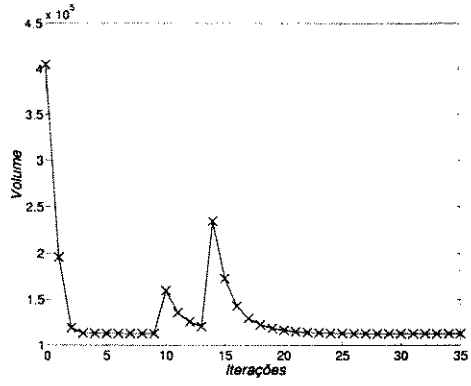
$g_2(b,h) = 5h - 1.0 \leq 0$ $\quad$ $g_4(b,h) = 1.0 - 25b \leq 0$

A solução desse problema é o ponto $b = 0.120$ m e $h = 0.200$ m. Este ponto é a intersecção entre g_1 e g_3, ativas na solução, como ilustrado na Figura 2.11(a). A seguinte solução numérica foi obtida com o algoritmo de Herskovits.

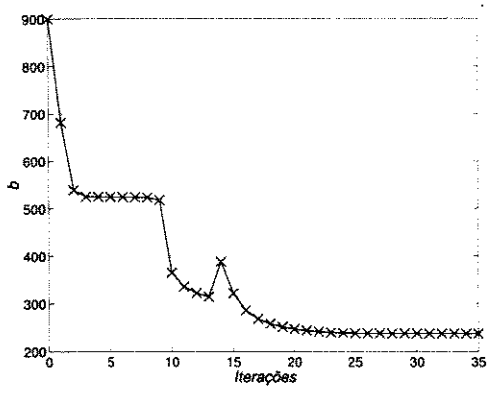
Solução:



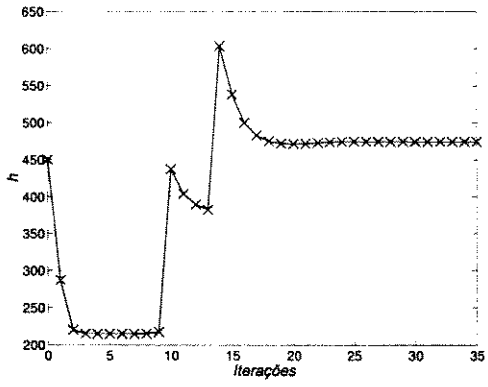
(a) Seqüência de projetos na região viável.



(b) Valor da função objetivo.



(c) Variável b.



(d) Variável h.

Figura 2.9: Processo de otimização do volume da viga (restrições de igualdade e desigualdade).

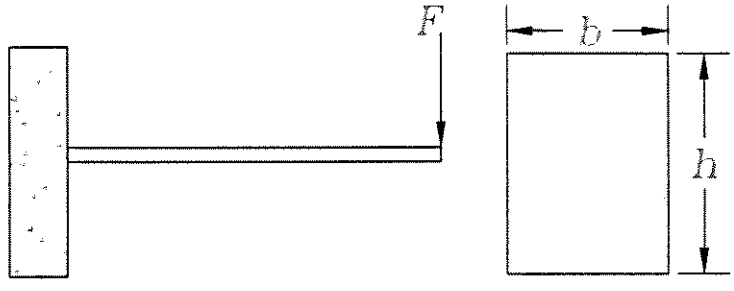


Figura 2.10: Carregamento F e dimensões b e h da seção transversal retangular de viga.

Tabela 2.3: Valor final das restrições e multiplicadores Lagrange.

Restr.	valor final	λ	valor final
$g_1 =$	-3.815654e-05	$\lambda_1 =$	2.543094e-02
$g_2 =$	-3.998159e-01	$\lambda_2 =$	1.000000e+00
$g_3 =$	-1.343418e-04	$\lambda_3 =$	1.000000e+00
$g_4 =$	-2.000921e+00	$\lambda_4 =$	1.588382e-07

- Parâmetros: $\alpha = 0.95$, $\xi = 10^{-9}$, $\bar{\rho} = 1.0$, $\nu = 0.8$, $\eta = 0.2$, $\gamma = 0.1$, $\epsilon = 0.1$, $\delta = 10^{-8}$.
- Ponto inicial: $b = 0.190$ m, $h = 0.170$ m.
- Valor da função objetivo no ponto inicial: 0.06460 m^3.
- Resultados:

  Ponto ótimo: $b = 0.12003683$ mm, $h = 0.19997313$ mm.

  Valor da função objetivo no ponto ótimo: 0.04800 m^3.

  Número de iterações: 5.

  Número de total avaliações da função objetivo: 8.

  Número de avaliações das funções de restrição: 11.

  Número de avaliações dos gradientes das funções: 5.

A Tabela 2.3 mostra o valor final das restrições e dos multiplicadores. A Figuras 2.11(a), 2.11(b), 2.11(c) e 2.11(d) mostram a seqüência de projetos gerados pelo algoritmo na região Ω, a evolução da função objetivo e das variáveis b e d durante as iterações.

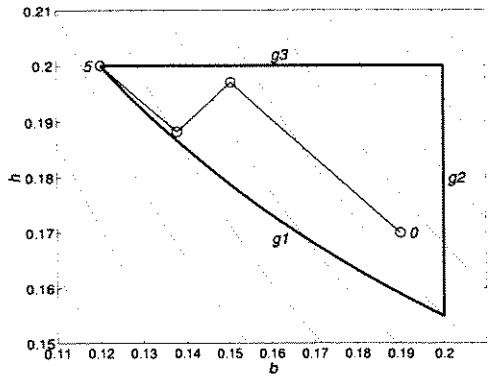
2.5.4 Projeto Ótimo de uma Mola

O problema consiste em projetar a mola de massa mínima que suporte os carregamentos de tração em compressão sem falha do material (no critério da tensão de cisalhamento), satisfazendo, ao mesmo tempo, outras exigências de performance como deflexão mínima, mínima freqüência das ondas transversais e limites das variáveis. As variáveis de projeto do problema são o diâmetro médio da espira D, o diâmetro do arame d e o número de espiras ativas n_a, como mostrado na Figura 2.12 [48].

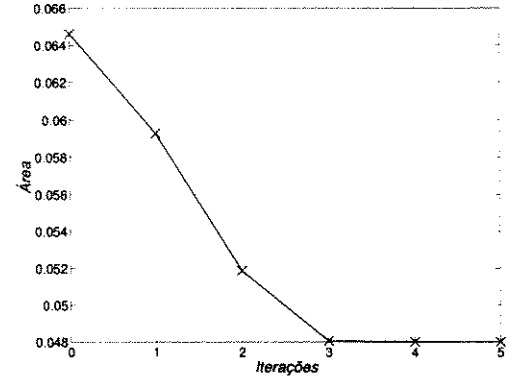
Para formular o problema de projeto de molas helicoidais, os seguintes dados e notações são definidos:

Número de espiras inativas	$n_i = 2$
Carregamento	$P = 10$ lbs
Módulo de cisalhamento	$\tilde{\mu} = 1.15 \times 10^7$ lb/in^2
Deflexão da mola	Δl
Deflexão mínima	$\Delta l_{\min} = 0.5$ in
Densidade do material	$\tilde{\rho} = 7.38342 \times 10^{-4}$ lb-sec^2/in^4
Tensão admissível de cisalhamento	$\tau_a = 8.0 \times 10^4$ lb/in^2
Limite inferior da freqüência transversal	$\omega_{\min} = 100$ Hz
Limite do diâmetro externo da mola	$\bar{D} = 1.5$ in

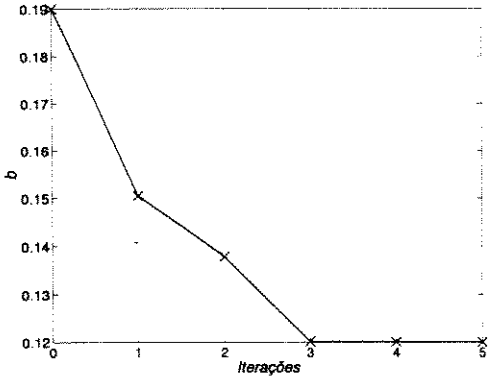
Uma mola sob tensão ou compressão experimenta torção. Dessa forma, impõe-se uma restrição na tensão de cisalhamento. Tem-se as seguintes expressões para a mola:



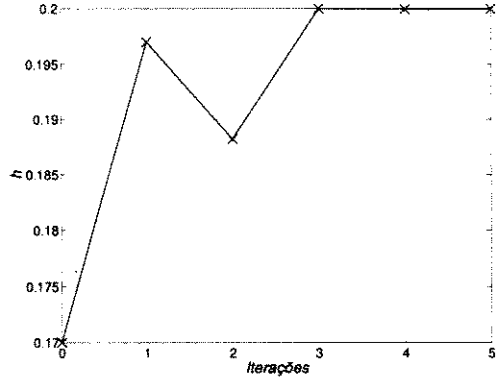
(a) Seqüência de projetos na região viável.



(b) Valor da função objetivo.



(c) Variável b.



(d) Variável h.

Figura 2.11: Processo de otimização do volume da viga engastada.

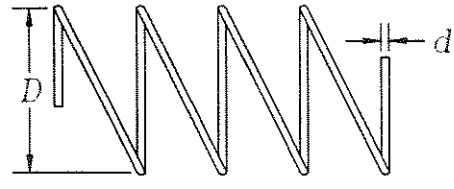


Figura 2.12: Dimensões D e d da mola.

Relação carregamento/deflexão	$P = \left(\frac{d^4\tilde{\mu}}{8D^3 n_a}\right)\Delta l$
Tensão de cisalhamento	$\tau = \frac{8k_c PD}{\pi d^3}$
Fator de concentração de tensão[7]	$k_c = \frac{(4D-d)}{(4D-4d)} + \frac{0.615d}{D}$
Freqüência das ondas transversais	$\omega = \frac{d}{2\pi D^2 n_a}\sqrt{\frac{\tilde{\mu}}{2\tilde{\rho}}}$

Estas expressões são usadas para determinar os funcionais do problema:

Função Objetivo O problema consiste em minimizar a massa da mola (volume × densidade). Como a densidade é constante, o problema se resume a minimizar o volume da mola

$$f(D, d, n_a) = (n_a + 2)\, Dd^2$$

Restrição de Deflexão É comum exigir que a deflexão devida ao carregamento P seja no mínimo $\Delta l_{\min}$. Este tipo de restrição é comum no projeto de molas, pois em muitas aplicações, a função da mola é fornecer uma força de restauração modesta enquanto outros componentes realização grandes deslocamentos devido a funções cinmáticas. Assim, $\Delta l \geq \Delta l_{\min}$, ou seja,

$$\frac{8D^3 n_a}{d^4\tilde{\mu}} \geq \Delta l_{\min}$$

Restrição de Tensão Esta restrição previne a falha do material no critério da tensão de cisalhamanto. Assim, deve-se ter $\tau \leq \tau_a$, ou seja,

$$\frac{8k_c PD}{\pi d^3}\left[\frac{(4D-d)}{(4D-4d)} + \frac{0.615d}{D}\right] \leq \tau_a$$

Restrição de Freqüência Deseja-se também, evitar ressonância em aplicações dinâmicas. Ressonância pode ser evitada tornando a freqüência das ondas transversais a maior possível, ou seja, $\omega \geq \omega_{\min}$. A restrição é então,

$$\frac{d}{2\pi D^2 n_a}\sqrt{\frac{\tilde{\mu}}{2\tilde{\rho}}} \geq \omega_{\min}$$

Restrição no Diâmetro Externo Devido a exigências construtivas, o diâmetro externo da mola é limitado. Logo,

$$D + d \leq \bar{D}$$

Limites das Variáveis Devido à exigências práticas as variáveis podem assumir valores somente dentro de uma faixa limitada, ou seja,

$$0.05 \leq d \leq 0.20 \qquad 0.25 \leq D \leq 1.30 \qquad 2 \leq n_a \leq 21$$

[7]Expressão de concentração de tensão de Wahl: relação determinda experimentalmente para considerar altas tensões em certos pontos de molas.

Tabela 2.4: Valor final das restrições e multiplicadores Lagrange.

Restr.	valor final	λ	valor final
$g_1 =$	-1.450270e-03	$\lambda_1 =$	6.879232e+02
$g_2 =$	-3.087149e-05	$\lambda_2 =$	2.477375e+04
$g_3 =$	-3.956557e+00	$\lambda_3 =$	2.527492e-01
$g_4 =$	-7.545542e-01	$\lambda_4 =$	1.325287e+00
$g_5 =$	-6.434158e-04	$\lambda_5 =$	1.544348e+03
$g_6 =$	-6.262732e+01	$\lambda_6 =$	1.596743e-02
$g_7 =$	-1.300760e+01	$\lambda_7 =$	7.687710e-02

A partir destes dados , pode-se expressar o problema na forma geral (1.11) com restrições normnalizadas,

$$\min f(D,d,n_a) = (n_a+2)\,Dd^2$$

sujeita à

$$g_1(b,h) = 1 - \frac{D^3 n_a}{71875d^4} \leq 0 \qquad g_4(b,h) = \frac{D+d}{1.5} - 1 \leq 0$$

$$g_2(b,h) = \frac{4D^2 - dD}{12566\,(Dd^3 - d^4)} + \frac{1}{5108d^2} - 1 \leq 0 \qquad g_5(b,h) = 1 - \frac{d}{0.05} \leq 0$$

$$g_3(b,h) = 1 - \frac{140.45d}{D^2 n_a} \leq 0 \qquad g_6(b,h) = 1.0 - \frac{D}{0.005} \leq 0$$

$$g_7(b,h) = 1.0 - 0.5n_a \leq 0$$

Este é um problema com vários mínimos locais, sendo o mínimo global correspondente a um valor de 0.01270 para a função objetivo do problema, com as restrições g_1 e g_2 ativas.

Solução:

- Parâmetros: $\alpha = 0.7$, $\xi = 10^{-10}$, $\bar{\rho} = 1.0$, $\nu = 0.8$, $\eta = 0.2$, $\gamma = 0.1$, $\epsilon = 0.1$,$\delta = 10^{-9}$.
- Ponto inicial: $D = 0.4$ in, $d = 0.06$ in e $n_a = 20.0$.
- Valor da função objetivo no ponto ótimo: 0.03168
- Resultados:

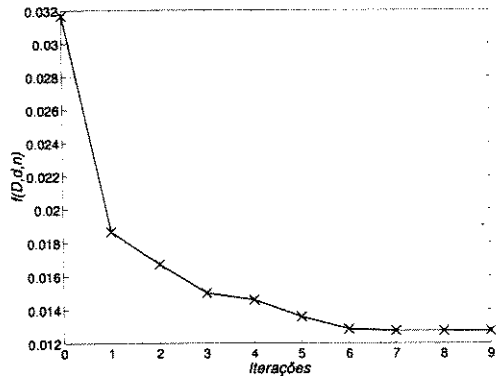
  Ponto ótimo: $D = 0.31813658$, $d = 0.05003217$, $n_a = 14.00760211$.
  Valor da função objetivo no ponto ótimo: 0.01274790.
  Número de iterações: 9.
  Número de total avaliações da função objetivo: 16.
  Número de avaliações das funções de restrição: 21.
  Número de avaliações dos gradientes das funções: 9.

A Tabela 2.4 mostra o valor final das restrições. As Figuras 2.13(a), 2.13(b), 2.13(c) e 2.13(d) mostram a evolução da função objetivo e das variáveis D, d e n_a durante as iterações.

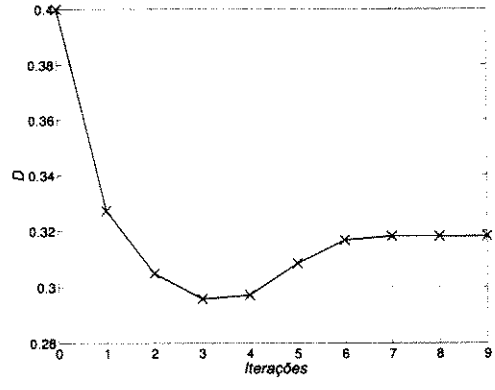
O resultado atingido corresponde a um mínimo local do problema, com a restrição g_2 ativa. Determinar o mínimo global com este método requer a realização de várias análises, alterando-se o ponto inicial, selecinando-se a solução onde a função objetivo tenha menor valor.

2.5.5 Comentários

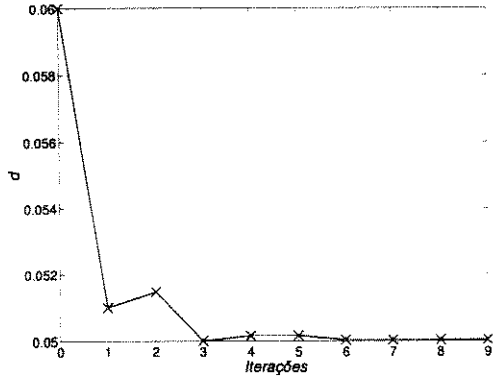
Acerca do comportamento do método nesse problemas os seguintes comentários podem ser feitos:



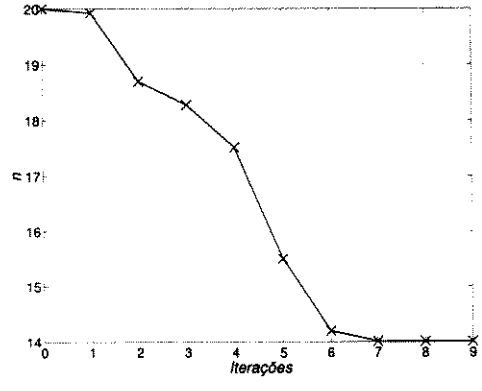
(a) Valor da função objetivo.



(b) Variável D.



(c) Variável d.



(d) Variável n_a.

Figura 2.13: Processo de otimização do volume da mola.

- O algoritmo atingiu a solução ótima dos diversos problemas com sucesso.
- Apesar do método exigir o ajuste de um grande número de parâmetros, apenas os parâmetros α, δ e ξ têm impacto significativo na eficiência do processo de solução. Como citado anteriormente, a escolha dos parâmetros δ e ξ dependem diretamente das unidades das variáveis de projeto, pois tanto o critério de parada do algoritmo quanto a atualização das variáveis de projeto são comparados com o valor $\|\mathbf{d}_0\|$. Esse foi o caso do exemplo na Seção 2.5.3, onde foi necessário adotar valores muito pequenos para δ e ξ.
- O parâmetro δ deve ser ajustado para que não seja maior que $\|\mathbf{d}_0\|$ nas primeiras iterações, permitindo que a seqüência de projetos se desvie das restrições que não serão ativas na solução. Um valor grande de ξ também pode fazer com que o algoritmo se encerre precocemente, muito longe da solução do problema. Como α controla o peso da deflexão da direção de busca original, é interessante adotar valores pequenos quando o ponto inicial está distante da solução evitando que a seqüência de projetos se aproxime de restrições não relevantes, reduzindo a convergência. Isso também é necessário em problemas com restrições de igualdade pois a função de penalização pode forçar uma saturação das restrições longe da solução;
- O resultado mais importante para avaliar a eficiência computacional de um algoritmo é o máximo entre o número de avaliações da função objetivo e o número de avaliações das funções de restrição. Isto porque as funções mais relevantes do problema geralmente são implícitas exigindo a solução de um sistema linear de grande dimensões, e esse número indica quantas vezes foi necessário resolver este sistema linear.
- Enquanto que em problemas envolvendo apenas restrições de desigualdade há decréscimo da função objetivo a cada passo, quando estão presentes restrições de igualdade é exigido decréscimo da *função objetivo penalizada*, podendo haver então acréscimo do valor da função objetivo original como pode ser observado na Figura 2.9(b).
- Os gradientes são avaliados uma vez a cada iteração.

2.6 Inclusão de Estratégia de Restrições Ativas

Com o objetivo de aumentar a eficiência do algoritmo de Herskovits, pode-se incorporar uma estratégia de restrições ativas semelhante a desenvolvida por LIM e ARORA[16]. A inclusão de tal técnica requer pequenas alterações no algoritmo básico na etapa de atualização da matriz hessiana, determinação da direção de busca e busca linear.

Nessa estratégia, determina-se, a cada iteração, o conjunto de restrições $\varepsilon-$ativas para uma dado $\varepsilon > 0$. São utilizados somente os funcionais desse conjunto na atualização da matriz hessiana, não sendo necessário determinar os gradientes dos demais funcionais. Em seguida, somente os funcionais ativos são utilizados na determinação da direção de busca. No algoritmo de Herskovits, diminui-se a dimensão da matriz $\mathbf{R}$, diminuindo o custo da determinação da direção de busca. Observa-se que as restrições correspondentes aos limites das variáveis, cujos gradientes não envolvem cálculos adicionais para serem determinados, podem ser sempre mantidas no conjunto ativo, fazendo com que todas as estimativas de projeto sejam mantidas dentro de um conjunto fechado e limitado conhecido. Na etapa de busca linear, seria necessário apenas modificar a interpolação quadrática das restrições pois, se alguma restrição não pertencente ao conjunto ativo for violada, a interpolação quadrática deve então ser feita a partir de três valores do funcional: o valor do funcional na iteração anterior, a estimativa por linearização e a estimativa pela regra de Armijo.

Como os casos tratados neste trabalho têm moderado custo computacional, a implementação utilizada não incorpora nenhuma estratégia de restrições ativas.

Capítulo 3

Análise de Sensibilidade em Elasticidade Linear

Uma das considerações mais importantes na análise ou projeto é a determinação dos efeitos da alteração de algumas de suas características (variáveis de projeto), ou seja, a determinação da *sensibilidade* à variação de seus parâmetros. Nesse capítulo se considera a análise de sensibilidade em problemas cujas variáveis não influenciam a forma do domínio do problema estrutural como, por exemplo, problemas envolvendo propriedades geométricas de elementos estruturais simples (área e momentos de inércia), propriedades de material ou espessuras ótimas em problemas de tensão plana.

Duas abordagens são possíveis: desenvolver as expressões dos gradientes em relação às equações de sistemas já discretizados ou obtê-las diretamente das equações do meio contínuo, as quais são então discretizadas para efeito de cálculo.

Nos problemas de elasticidade linear estas duas abordagens são equivalentes tanto teórica quanto numericamente. Porém, a segunda opção apresenta como vantagens a obtenção de um enunciado analítico das derivadas dos funcionais do problema, sem nenhuma discretização envolvida, além de uma certa independência em relação ao código de análise. Observa-se ainda que os problemas onde a forma do componente é considerada uma variável (análise de sensibilidade à variação da forma) são formulados de maneira mais conveniente como problemas de domínio contínuo [31]. Logo, a formulação contínua permite que problemas de sensibilidade a parâmetros e a forma possam ser tratados e implementados de maneira unificada.

Uma vez implementados os procedimentos de determinação de gradientes, a técnica de minimização do Capítulo 2 pode ser aplicada na determinação de espessuras ótimas em problemas de tensão plana.

3.1 Formulação Discreta da Análise de Sensibilidade

Considere estruturas com número finito de graus de liberdade, comportamento elástico do material e pequenas deformações, ou seja, respeitando as hipóteses da elasticidade infinitesimal. Tais estruturas podem ser treliças, pórticos, estados planos e sólidos tridimensionais discretizados pelo Método de Elementos Finitos.

De acordo com a seção anterior, a solução do problema discretizado envolve a solução de um sistema linear de equações da forma (1.9), descrito de forma resumida como,

$$\mathbf{K}(\mathbf{p})\,\mathbf{u} = \mathbf{f}(\mathbf{p}) \tag{3.1}$$

sendo $\mathbf{u} = \mathbf{u}(\mathbf{p})$ e $\mathbf{p}$ o vetor das variáveis de projeto

Os funcionais usualmente utilizados em problemas de otimização estrutural podem ser colocados na seguinte forma geral,

$$\psi(\mathbf{p}) = \mathcal{G}(\mathbf{p}, \mathbf{u}(\mathbf{p})) \tag{3.2}$$

Este funcional depende tanto de forma explícita das variáveis de projeto quanto de forma implícita, através da solução $\mathbf{u}$ da equação de estado (3.1). O objetivo da análise de sensibilidade é determinar a dependência total deste tipo de funcional em relação às variáveis de um projeto parametrizado.

Uma variação do funcional (3.2) é a seguinte,

$$\begin{aligned} \dot{\psi}(\mathbf{p}) &= \nabla_{\mathbf{p}}\psi \cdot d\mathbf{p} = \frac{\partial \mathcal{G}}{\partial \mathbf{p}} \cdot d\mathbf{p} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot d\mathbf{u} = \frac{\partial \mathcal{G}}{\partial \mathbf{p}} \cdot d\mathbf{p} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot (\nabla_{\mathbf{p}}\mathbf{u})\, d\mathbf{p} \\ &= \left[\frac{\partial \mathcal{G}}{\partial \mathbf{p}} + (\nabla_{\mathbf{p}}\mathbf{u})^T \frac{\partial \mathcal{G}}{\partial \mathbf{u}}\right] \cdot d\mathbf{p} \Longrightarrow \nabla\psi = \left[\frac{\partial \mathcal{G}}{\partial \mathbf{p}} + (\nabla_{\mathbf{p}}\mathbf{u})^T \frac{\partial \mathcal{G}}{\partial \mathbf{u}}\right] \end{aligned} \tag{3.3}$$

Na equação anterior, os termos $\dfrac{\partial \mathcal{G}}{\partial \mathbf{p}}$ e $\dfrac{\partial \mathcal{G}}{\partial \mathbf{u}}$ podem ser determinados, já que se conhece a forma funcional de $\mathcal{G}$. O problema reside em determinar $\nabla_{\mathbf{p}}\mathbf{u}$.

Como $\mathbf{u}$ é definido de forma implícita pela solução do sistema (3.1), sua derivabilidade em relação a $\mathbf{p}$ não é imediata. Admite-se no momento a existência desta derivada e adiante serão discutidos alguns aspectos envolvidos nessa operação. Dois métodos, direto e adjunto, podem ser empregados para calcular $\nabla_{\mathbf{p}}\mathbf{u}$, conforme descrito a seguir.

3.1.1 Método Direto

Considerando a existência de $\nabla_{\mathbf{p}}\mathbf{u}$, define-se o vetor $\mathbf{r}$, resíduo de (3.1), como,

$$\mathbf{r} = \mathbf{K}\mathbf{u} - \mathbf{f} = \mathbf{0}$$

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

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

Logo,

$$\mathbf{K}\frac{\partial \mathbf{u}}{\partial p_k} = \frac{\partial \mathbf{f}}{\partial p_k} - \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} \qquad k = 1, \ldots, n \tag{3.4}$$

Resolvendo os n sistemas lineares de (3.4), obtem-se o tensor $\nabla_{\mathbf{p}}\mathbf{u}$, pois cada solução de (3.4) fornece uma coluna de $\nabla_{\mathbf{p}}\mathbf{u}$. Observa-se que uma vez fatorada a matriz $\mathbf{K}$, obtém-se as derivadas $\frac{\partial \mathbf{u}}{\partial p_k}$ por substituição. Após determinar $\nabla_{\mathbf{p}}\mathbf{u}$, basta utilizá-lo na expressão (3.3) para calcular a sensibilidade de ψ em relação às variáveis de projeto.

Em resumo, o cálculo do gradiente do funcional ψ pelo método direto envolve os seguintes passos:

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

2. Calcular $\nabla_{\mathbf{p}}\mathbf{u}$ pela equação (3.4), através da solução de n sistemas lineares com mesma matriz $\mathbf{K}$,

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

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

$$\frac{\partial \psi}{\partial p_k} = \frac{\partial \mathcal{G}}{\partial p_k} + \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \cdot \frac{\partial \mathbf{u}}{\partial p_k} \qquad k = 1, \ldots, n \tag{3.5}$$

3.1.2 Método Adjunto

Seja o vetor $\varsigma \in \Re^N$ e considere o produto escalar de ς por (3.4),

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

Portanto,

$$\mathbf{K}^T\varsigma \cdot \frac{\partial \mathbf{u}}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} \tag{3.6}$$

Comparando (3.6) com (3.5), pode-se calcular ς pela solução do sistema,

$$\mathbf{K}^T\varsigma = \frac{\partial \mathcal{G}}{\partial \mathbf{u}} \tag{3.7}$$

Desta forma, a sensibilidade de ψ com respeito a $\mathbf{p}$ pode ser avaliada pela expressão,

$$\frac{\partial \psi}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} + \frac{\partial \mathcal{G}}{\partial p_k} \qquad k = 1, \ldots, n \tag{3.8}$$

sendo necessário resolver apenas um sistema linear, independente do número de variáveis de projeto.

Em resumo, o cálculo do gradiente do funcional ψ pelo método adjunto envolve os seguintes passos:

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

2. Resolver o problema adjunto (3.7)

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

3. Calcular as componentes de $\nabla_{\mathbf{p}}\psi$ através de (3.8),

$$\frac{\partial \psi}{\partial p_k} = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k} - \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u} + \frac{\partial \mathcal{G}}{\partial p_k} \qquad k = 1, \ldots, n$$

3.1.3 Comparação entre os Métodos Direto e Adjunto

A precisão dos resultados obtidos pelos métodos direto e adjunto é equivalente [31]. Porém, o número de operações realizadas em cada caso pode diferir bastante. Para determinar qual dos métodos anteriores é o mais eficiente basta comparar o número de sistemas lineares a serem resolvidos, assim como e o número de vetores a serem armazenados e operados para a solução do problema.

No método direto, calcula-se todo o tensor $\nabla_{\mathbf{p}}\mathbf{u}$, exigindo a solução de n (número de variáveis de projeto) sistemas lineares (de mesma matriz) de dimensão N, independente do número de funcionais do problema. O método adjunto, por sua vez, resolve um sistema linear de dimensão N por funcional (também de mesma matriz). Como em otimização o número de funcionais ativos[1] é sempre menor ou igual a n para um projeto viável, o método adjunto será de forma geral mais eficiente. Principalmente se o algoritmo utilizado for capaz de reconhecer o conjunto de restrições potencialmente ativas.

Além disso, em muitos casos o número de variáveis de projeto é maior que o número de funcionais cujos gradientes exigem procedimentos de análise de sensibilidade para serem determinados.

[1]Do ponto de vista da otimização, apenas os funcionais ativos e violados são relevantes. Portanto, somente é necessário calcular a sensibilidade destes funcionais.

3.2 Formulação Contínua da Análise de Sensibilidade

No caso da formulação contínua, consideram-se as equações na sua forma original, deduzindo-se analiticamente as expressões de sensibilidade, as quais são resolvidas numericamente. Observa-se que no caso discreto, determinam-se as derivadas tomando-se as expressões na sua forma discreta. Ambas as formulações são equivalentes para problemas lineares.

Considere, portanto, problemas elastostáticos lineares colocados na forma do problema variacional (1.4).

Demonstra-se em [31] que as formas bilinear $a_{\mathbf{p}}(u,v)$ e linear $l_{\mathbf{p}}(v)$ são diferenciáveis com respeito ao projeto $\mathbf{p}$, mantendo u fixo. Da mesma maneira, u e ∇u são diferenciáveis com relação a $\mathbf{p}$. Assim, sendo $\delta\mathbf{p}$ uma variação do projeto atual $\mathbf{p}$, tem-se,

$$\dot{a}_{\mathbf{p}}(u,v) = Da_{\mathbf{p}}(u,v)[\delta\mathbf{p}] = \lim_{\tau\to 0}\frac{a_{\mathbf{p}+\tau\delta\mathbf{p}}(u,v) - a_{\mathbf{p}}(u,v)}{\tau} = \nabla_{\mathbf{p}} a_{\mathbf{p}}(u,v)\cdot\delta\mathbf{p} \tag{3.9}$$

$$\dot{l}_{\mathbf{p}}(v) = Dl_{\mathbf{p}}(v)[\delta\mathbf{p}] = \lim_{\tau\to 0}\frac{l_{\mathbf{p}+\tau\delta\mathbf{p}}(v) - l_{\mathbf{p}}(v)}{\tau} = \nabla_{\mathbf{p}} l_{\mathbf{p}}(v)\cdot\delta\mathbf{p} \tag{3.10}$$

$$\dot{u}(\mathbf{x},\mathbf{p}) = Du(\mathbf{x},\mathbf{p})[\delta\mathbf{p}] = \lim_{\tau\to 0}\frac{u(\mathbf{x},\mathbf{p}+\tau\delta\mathbf{p}) - u(\mathbf{x},\mathbf{p})}{\tau} = \nabla_{\mathbf{p}} u(\mathbf{x},\mathbf{p})\ \delta\mathbf{p} \tag{3.11}$$

As expressões anteriores representam as derivadas direcionais na direção $\delta\mathbf{p}$ ou ainda a primeira variação do cálculo de variações. Como $\dot{a}_{\mathbf{p}}(u,v)$ é contínua e $\dot{a}_{\mathbf{p}}(u,v)$, $\dot{l}_{\mathbf{p}}(v)$ e $\dot{u}(\mathbf{x},\mathbf{p})$ são lineares em $\delta\mathbf{p}$, as equações (3.9) a (3.11) são denominadas derivadas de Fréchet.

Observe que a geometria de $\mathcal{B}$ não depende dos parâmetros. Portanto, considerando apenas condições de contorno homogêneas, o espaço $\mathcal{V}$ de soluções cinematicamente possíveis é mantido inalterado, ou seja, $\dot{v} = 0$.

A maioria dos funcionais de performance estrutural pode ser colocada na seguinte forma geral,

$$\psi(\mathbf{p}) = \int_{\mathcal{B}} \mathcal{G}(u,\nabla u,\mathbf{p})\ dV \qquad u = u(\mathbf{x},\mathbf{p}) \qquad \mathbf{x}\in\mathcal{B} \tag{3.12}$$

onde $\mathcal{G}$ é um funcional continuamente diferenciável com relação aos seus argumentos, podendo representar, por exemplo, o volume ou um critério de falha estrutural numa região de $\mathcal{B}$. A derivada direcional de (3.12) na direção da reta $\mathbf{p}+t\delta\mathbf{p}$ fica então definida por,

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

Como $\mathcal{B}$ não depende das variáveis de projeto, a diferenciação pode ser tomada dentro do integrando. Logo, empregando a regra da cadeia, tem-se,

$$\dot{\psi}(\mathbf{p}) = \int_{\mathcal{B}} D\{\mathcal{G}(u(\mathbf{x},\mathbf{p}),\nabla u(\mathbf{x},\mathbf{p}),\mathbf{p})\}[\delta\mathbf{p}]\ dV$$

$$\dot{\psi}(\mathbf{p}) = \int_{\mathcal{B}} (\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u} + \mathcal{G}_{,\mathbf{p}}\cdot\delta\mathbf{p})\ dV \tag{3.13}$$

A seguir discutem-se os métodos direto e adjunto da formulação contínua de análise de sensibilidade.

3.2.1 Método Direto

Na expressão (3.13), observa-se que o cálculo de derivadas de funcionais dependentes da solução u, exigem o conhecimento de $\dot{u}$ que, por sua vez, pode ser determinado a partir da derivada total da equação (1.4),

$$[a_{\mathbf{p}}(u,v)]^{\cdot} = [l_{\mathbf{p}}(v)]^{\cdot} \Rightarrow \dot{a}_{\mathbf{p}}(u,v) + a_{\mathbf{p}}(\dot{u},v) = \dot{l}_{\mathbf{p}}(v)$$

$$a_{\mathbf{p}}(\dot{u},v) = \dot{l}_{\mathbf{p}}(v) - \dot{a}_{\mathbf{p}}(u,v) \tag{3.14}$$

A equação (3.14) dá origem a um novo problema variacional de mesma forma bilinear que (1.4), cuja solução é justamente a derivada do campo de deslocamentos $\dot{u}$. A partir dessa solução determina-se $\nabla\dot{u}$, obtendo-se $\dot{\psi}(\mathbf{p})$ a partir de (3.13). No entanto, os termos do lado direito de (3.14) dependem da direção $\delta\mathbf{p}$. Desta maneira, este procedimento não é de interesse, pois deseja-se obter expressões explícitas para as derivadas em função de $\delta\mathbf{p}$.

3.2.2 Método Adjunto

Deve-se observar ainda que o problema colocado na forma (3.14) é de utilidade prática limitada pois requer a solução analítica de $\dot{u}$ para que se possa calcular o gradiente $\nabla\dot{u}$. Como na maioria dos casos $\dot{u}$ somente é conhecido em pontos discretos, introduz-se, a exemplo da formulação discreta, um *problema adjunto*, possuindo as mesmas restrições em deslocamentos do problema original, mas sujeito a um caso de carregamento dependente do funcional $\psi(\mathbf{p})$, denominado *carregamento adjunto*. O objetivo é obter uma expressão explícita para $\dot{\psi}$ em termos de $\delta\mathbf{p}$; para isso utilizam-se os dois primeiros termos da integral do lado direito de (3.13).

Observa-se que os dois primeiros termos da integral do lado direito de (3.13) constituem uma forma linear em $\dot{u}$ limitada,

$$l_{\mathbf{p}}^{adj}(\dot{u}) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot \dot{u} + \mathcal{G}_{,\nabla u} \cdot \nabla\dot{u})\ dV$$

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

$$a_{\mathbf{p}}(\varsigma, \dot{u}) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot \dot{u} + \mathcal{G}_{,\nabla u} \cdot \nabla\dot{u})\ dV \tag{3.15}$$

Como $a_{\mathbf{p}}$ é simétrico, tem-se a partir de (3.14) e (3.15) ,

$a_{\mathbf{p}}(\dot{u}, \varsigma) = a_{\mathbf{p}}(\varsigma, \dot{u}) = \dot{l}_{\mathbf{p}}(\varsigma) - \dot{a}_{\mathbf{p}}(u, \varsigma)$

de onde se conclui que,

$$l_{\mathbf{p}}^{adj}(\dot{u}) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot \dot{u} + \mathcal{G}_{,\nabla u} \cdot \nabla\dot{u})\ dV = \dot{l}_{\mathbf{p}}(\varsigma) - \dot{a}_{\mathbf{p}}(u, \varsigma)$$

Substituindo a equação anterior em (3.13) tem-se a expressão da derivada do funcional $\psi(\mathbf{p})$,

$$\dot{\psi}(\mathbf{p}) = \dot{l}_{\mathbf{p}}(\varsigma) - \dot{a}_{\mathbf{p}}(u, \varsigma) + \int_{\mathcal{B}} \mathcal{G}_{,\mathbf{p}} \cdot \delta\mathbf{p}\ dV \tag{3.16}$$

ou tomando-se (3.9) e (3.10),

$$\dot{\psi}(\mathbf{p}) = \nabla\psi \cdot \delta\mathbf{p} = \left\{ \nabla l_{\mathbf{p}}(\varsigma) - \nabla a_{\mathbf{p}}(u, \varsigma) + \int_{\mathcal{B}} \mathcal{G}_{,\mathbf{p}}\ dV \right\} \cdot \delta\mathbf{p}$$

Assim, a determinação de $\dot{\psi}(\mathbf{p})$ depende apenas do estado adjunto ς. Para isso, define-se o seguinte problema variacional,

Determinar o campo de deslocamentos ς tal que

$$a_{\mathbf{p}}(\varsigma, v) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot v + \mathcal{G}_{,\nabla u} \cdot \nabla v)\ dV \qquad \forall v \in \mathcal{V} \tag{3.17}$$

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 [31].

Nesse caso, não é mais necessário avaliar $\nabla\dot{u}$, devendo-se apenas resolver o problema adjunto (3.17) determinando ς. Observa-se, entretanto, que é necessário formular um problema adjunto por funcional dependente de u.

Em problemas analíticos, $\mathbf{p}$ é um vetor cujas componentes podem ser escalares ou funções.

Neste último caso os produtos internos devem ser tomados dentro dos integrandos. Na prática, porém, as funções correspondendo a parâmetros do modelo são aproximadas por um subespaço de dimensão finita, sendo então utilizado um conjunto finito de funções de forma. Isso significa que as componentes de $\mathbf{p}$ são, na maioria dos casos de interesse, sempre *escalares*, ou seja, $\mathbf{p} \in \Re^n$.

Considerando isto, o algoritmo para o cálculo de (3.16) é o seguinte:

1. Contrói-se o termo independente de (3.17).
2. Resolve-se o problema adjunto (3.17) determinando ς.
3. Determina-se o gradiente de ψ da seguinte forma,

$$(\nabla_{\mathbf{p}}\psi)_k = \frac{\partial \psi}{\partial p_k} = \frac{\partial l_{\mathbf{p}}}{\partial p_k}(\varsigma) - \frac{\partial a_{\mathbf{p}}}{\partial p_k}(u,\varsigma) + \int_{\mathcal{B}} \frac{\partial \mathcal{G}}{\partial p_k}\, dV \qquad k = 1,\ldots,n \tag{3.18}$$

3.3 Estado Plano de Tensão

A formulação contínua da análise de sensibilidade será aplicada ao problema de tensão plana. Considere um corpo que está nas hipóteses de tensão plana, ou seja, despreza-se a tensão normal ao plano contendo o domínio $\mathcal{B}$, a espessura do corpo é pequena se comparada às demais dimensões e os carregamentos não dependem da coordenada normal à $\mathcal{B}$. Logo, a partir de (1.4), as formas bilinear $a_{\mathbf{p}}(\cdot,\cdot)$ e linear $l_{\mathbf{p}}(\cdot)$ estarão dadas por,

$$a_{\mathbf{p}}(u,v) = \int_{\mathcal{B}} e(\mathbf{x})\left[\mathbf{T}(u)\cdot\mathbf{E}(v)\right]\, dA \tag{3.19}$$

$$l_{\mathbf{p}}(v) = \int_{\mathcal{B}} e(\mathbf{x})\left[\mathbf{b}(\mathbf{x})\cdot v\right]\, dA + \int_{\Gamma^2} e(\mathbf{x})\left[\mathbf{\Phi}\cdot v\right]\, dL \tag{3.20}$$

onde:

- $e(\mathbf{x})$, $\mathbf{x} \in \mathcal{B} \subset \Re^2$, é uma função escalar descrevendo a espessura do corpo.
- Os deslocamentos dos pontos do corpo são funções de suas coordenadas no plano, ou seja, $u(\mathbf{x}) = \left\{ \begin{array}{cc} u_x(x,y) & u_y(x,y) \end{array} \right\}^T$.
- $\mathbf{b}$ é o campo vetorial de forças de volume e $\mathbf{\Phi}$ o campo de forças de superfície.

Na prática, a espessura $e(\mathbf{x})$ é aproximada por uma combinação linear de funções de forma, sendo as constantes dessa combinação as variáveis de projeto a serem otimizadas, ou seja,

$$e(\mathbf{x}) \simeq \sum_{k=1}^{n} p_k \phi_k \tag{3.21}$$

Sejam, então, as funções ϕ_i definidas como,

$$\phi_k(\mathbf{x}) = \left\{ \begin{array}{lll} 1 & \text{se} & \mathbf{x} \in \mathcal{B}_k \\ 0 & \text{se} & \mathbf{x} \notin \mathcal{B}_k \end{array} \right. \qquad \text{onde} \qquad \bigcup_{k=1}^{n} \mathcal{B}_k = \mathcal{B} \qquad \bigcup_{k=1}^{n} \partial\mathcal{B}_k \cap \partial\mathcal{B} = \partial\mathcal{B} \tag{3.22}$$

Tais funções dividem o corpo em subregiões $\mathcal{B}_k$ de espessura constante. Assim, os termos (3.19) e (3.20) podem ser reescritos como,

$$a_{\mathbf{p}}(u,v) = \sum_{k=1}^{n} p_k \int_{\mathcal{B}_k} \mathbf{T}(u)\cdot\mathbf{E}(v)\, dA$$

$$l_{\mathbf{p}}(v) = \sum_{k=1}^{n} p_k \left\{ \int_{\mathcal{B}_k} \mathbf{b}(\mathbf{x}) \cdot v \, dA + \int_{\Gamma^2 \cap \partial\mathcal{B}_k} \mathbf{\Phi} \cdot v \, dL \right\}$$

De onde se conclui que,

$$\frac{\partial a_{\mathbf{p}}}{\partial p_k}(u,v) = \int_{\mathcal{B}_k} \mathbf{T}(u) \cdot \mathbf{E}(v) \, dA \tag{3.23}$$

$$\frac{\partial l_{\mathbf{p}}}{\partial p_k}(v) = \left\{ \int_{\mathcal{B}_k} \mathbf{b}(\mathbf{x}) \cdot v \, dA + \int_{\Gamma^2 \cap \partial\mathcal{B}_k} \mathbf{\Phi} \cdot v \, dL \right\} \tag{3.24}$$

Dessa forma, a equação (3.18) pode ser reescrita como,

$$(\nabla_{\mathbf{p}} \psi)_k = \int_{\mathcal{B}_k} \mathbf{b}(\mathbf{x}) \cdot \varsigma \, dA + \int_{\Gamma_2 \cap \partial\mathcal{B}_k} \mathbf{\Phi} \cdot \varsigma \, dL - \int_{\mathcal{B}_k} \mathbf{T}(u) \cdot \mathbf{E}(\varsigma) \, dA + \int_{\mathcal{B}} \frac{\partial \mathcal{G}}{\partial p_k} \, dV \tag{3.25}$$

onde ς é a solução do problema adjunto.

Observa-se que numa discretização por elementos finitos, $a_{\mathbf{p}}(u,v) = \mathbf{v} \cdot \mathbf{K}\mathbf{u}$ e $l_{\mathbf{p}}(v) = \mathbf{f} \cdot \mathbf{v}$ e conseqüentemente[2], $\frac{\partial a_{\mathbf{p}}}{\partial p_k}(u,\varsigma) = \varsigma \cdot \frac{\partial \mathbf{K}}{\partial p_k}\mathbf{u}$ e $\frac{\partial l_{\mathbf{p}}}{\partial p_k}(\varsigma) = \varsigma \cdot \frac{\partial \mathbf{f}}{\partial p_k}$. Verifica-se que $\frac{\partial \mathbf{K}}{\partial p_k}$ é a matriz $N \times N$ formada pela inclusão das matrizes dos elementos da região $\mathcal{B}_k$, divididas pelo valor da espessura p_k, na matriz de rigidez global. Por sua vez, $\frac{\partial \mathbf{f}}{\partial p_k}$ é o vetor $N \times 1$ formado pela inclusão dos carregamentos dependentes da espessura impostos na região $\mathcal{B}_k$, divididos pelo valor da espessura p_k, no vetor de carregamentos global. Nestes dois termos consideram-se que as condições de contorno homogêneas foram aplicadas.

3.3.1 Determinação de Gradientes

A seguir são desenvolvidas as expressões dos gradientes dos funcionais definidos na seção 1.3.3 para o problema de estado plano de tensão.

Funcional de Massa

O funcional definindo a massa do corpo é, de acordo com (1.13),

$$\psi_1 = \int_{\mathcal{B}} \tilde{\rho}(\mathbf{x}) \, e(\mathbf{x}) \, dA$$

Logo,

$$\dot{\psi}_1 = \int_{\mathcal{B}} \tilde{\rho}(\mathbf{x}) \, \delta e \, dA \tag{3.26}$$

Portanto,

$$\psi_1(\mathbf{p}) = \sum_{k=1}^{n} p_k \int_{\mathcal{B}_k} \tilde{\rho}(\mathbf{x}) \, dA$$

$$\dot{\psi}_1(\mathbf{p}) = \sum_{k=1}^{n} \delta p_k \int_{\mathcal{B}_k} \tilde{\rho}(\mathbf{x}) \, dA = \nabla_{\mathbf{p}} \psi_1 \cdot \delta \mathbf{p}$$

$$(\nabla_{\mathbf{p}} \psi_1)_k = \frac{\partial \psi_1}{\partial p_k} = \int_{\mathcal{B}_k} \tilde{\rho}(\mathbf{x}) \, dA \tag{3.27}$$

[2]De fato, para problemas de lineares, a discretização das expressões da formulação contínua leva às mesmas expressões da formulação discreta.

Discretizando o domínio em elementos triangulares com sistema local de referência $\bar{\xi} - \bar{\eta}$, por exemplo, tem-se,

$$(\nabla_{\mathbf{p}}\psi_1)_k = \frac{\partial \psi_1}{\partial p_k} = \sum_{e_k} \tilde{\rho}_{e_k} \int_0^1 \int_0^{1-\bar{\eta}} \det \mathbf{J}_{e_k} \, d\bar{\xi} d\bar{\eta}$$

sendo $\tilde{\rho}_{e_k}$ e $\mathbf{J}_{e_k}$, respectivamente, a densidade do material e o Jacobiano do elemento.

Observa-se que não há a necessidade de formular um problema adjunto pois este funcional é independente de u.

Funcional de Tensão

Considerando o funcional de tensão (1.14), sua expressão de sensibilidade é dada por,

$$\dot{\psi}_2 = \int_{\mathcal{B}} \left[\mathcal{G}_{,T}(u) \cdot \mathbf{T}(\dot{u})\right] m_r(\mathbf{x}) \, e(\mathbf{x}) \; dA \tag{3.28}$$

Neste caso, aplicando (3.17), tem-se o seguinte problema adjunto do funcional ψ_2,

Determinar o campo de deslocamentos adjunto ς *tal que*

$$a_{\mathbf{p}}(\varsigma, v) = \int_{\mathcal{B}} \left[\mathcal{G}_{,T}(u) \cdot \mathbf{T}(v)\right] m_r(\mathbf{x}) \; dA \qquad \forall v \in \mathcal{V} \tag{3.29}$$

Pode-se, por exemplo, escrever o funcional de tensão ψ_2 em relação à tensão equivalente de von Mises como,

$$\psi_2(\mathbf{p}) = \int_{\mathcal{B}} \sigma_{VM} \; m_r \; dA \quad \longrightarrow \quad \mathcal{G}(\mathbf{T}(u)) = \sigma_{VM}$$

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

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

Se $\mathcal{B}_r$ coincide com o elemento e de maior tensão de von Mises ponderada pela área, tem-se que, para elementos triangulares no sistema local de referência $\bar{\xi} - \bar{\eta}$,

$$\psi_2(\mathbf{p}) = \frac{\int_{\mathcal{B}_r} \sigma_{VM} \, e(\mathbf{x}) \; dA}{\int_{\mathcal{B}_r} e(\mathbf{x}) \; dA} = \frac{1}{A_e} \int_0^1 \int_0^{1-\bar{\eta}} \sigma_{VM} \; \det \mathbf{J}_e \, d\bar{\xi} d\bar{\eta} \tag{3.30}$$

sendo $A_e = \int_0^1 \int_0^{1-\bar{\eta}} \det \mathbf{J}_e \, d\bar{\xi} d\bar{\eta}$ a área deste elemento.

Considerando uma discretização em elementos triangulares, o carregamento adjunto é então obtido da seguinte expressão,

$$\begin{aligned} \int_{\mathcal{B}} \left[\mathcal{G}_{,T}(u) \cdot \mathbf{T}(v)\right] m_r(\mathbf{x}) \; dA &= \frac{1}{A_e} \int_0^1 \int_0^{1-\bar{\eta}} \mathbf{g}_{adj} \cdot \mathbf{DBv} \; \det \mathbf{J}_e \, d\bar{\xi} d\bar{\eta} \\ &= \frac{1}{A_e} \int_0^1 \int_0^{1-\bar{\eta}} \mathbf{B}^T \mathbf{D}^T \mathbf{g}_{adj} \; \det \mathbf{J}_e \, d\bar{\xi} d\bar{\eta} \cdot \mathbf{v} \end{aligned}$$

sendo $\mathbf{g}_{adj} = \dfrac{1}{\sigma_{VM}} \begin{bmatrix} 2\sigma_x - \sigma_y & 2\sigma_y - \sigma_x & 12\tau_{xy} \end{bmatrix}^T$; $\mathbf{B}$ é a matriz operador de deformação; e $\mathbf{D}$ o tensor de elasticidade de Hooke. Em resumo, o carregamento adjunto nos nós do elemento e é fornecido pelo vetor

$$\mathbf{f}_e^{adj} = \frac{1}{A_e} \int_0^1 \int_0^{1-\bar{\eta}} \mathbf{B}^T \mathbf{D}^T \mathbf{g}_{adj} \; \det \mathbf{J}_e \, d\bar{\xi} d\bar{\eta} \tag{3.31}$$

Deve-se observar que quanto mais refinada a malha, o valor da tensão média no elemento e se aproxima do valor de tensão máxima do modelo. Se este funcional for uma restrição da otimização de uma estrutura, pode-se então tomar o valor de tensão admissível próximo da tensão admissível do material em questão. Em contrapartida, em malhas onde o elemeno e tenha área

relativamente grande, deve-se compensar o efeito de média no valor da tensão admissível. Isto pode ser observado nos exemplos do final desse capítulo.

Apesar da influência do tamanho do elemento no valor do funcional e no valor máximo admissível (mesmo que o resultado da análise de elementos finitos seja satisfatório), a escolha deste funcional em termos de tensão média se justifica, pois a tensão nodal máxima pode ocorrer em nós engastados, onde a aplicação de carregamentos adjuntos não forneceria nenhuma informação para o cálculo de gradientes. Já o elemento de maior tensão média apresenta deformação, e portanto, pelo menos algum grau de liberdade livre onde aplicar o carregamento adjunto.

Funcional de Deslocamento

O funcional (1.15) tem a sensibilidade descrita por,

$$\dot{\psi}_3(\mathbf{p}) = \int_{\mathcal{B}} (\mathcal{G}_{,u} \cdot \dot{u})\, \delta(\mathbf{x} - \mathbf{x}_k)\ dA \tag{3.32}$$

Aplicando (3.17) tem-se o seguinte problema adjunto do funcional ψ_3,

Determinar o campo de deslocamentos adjunto ς *tal que*

$$a_{\mathbf{p}}(\varsigma, v) = \int_{\mathcal{B}} [\mathcal{G}_{,u}(u) \cdot v]\, m_r(\mathbf{x})\ dA \qquad \forall v \in \mathcal{V} \tag{3.33}$$

Neste caso,

$$\mathcal{G}_{,u}(u) = \frac{1}{|u(\mathbf{x}_k)|} \begin{Bmatrix} u_x(\mathbf{x}_k) \\ u_y(\mathbf{x}_k) \end{Bmatrix}$$

que é o próprio carregamento adjunto no nó.

Se este funcional for definido em relação a um determinado grau de liberdade, então

$$\mathcal{G}_{,u}(u) = \begin{Bmatrix} 1 \\ 0 \end{Bmatrix} \text{ se } u_x(\mathbf{x}_k) > 0 \qquad \text{ou } \mathcal{G}_{,u}(u) = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix} \text{ se } u_y(\mathbf{x}_k) > 0$$

$$\begin{Bmatrix} -1 \\ 0 \end{Bmatrix} \text{ se } u_x(\mathbf{x}_k) < 0 \qquad \begin{Bmatrix} 0 \\ -1 \end{Bmatrix} \text{ se } u_y(\mathbf{x}_k) < 0$$

3.4 Derivabilidade da Solução da Equação de Estado

Nas seções anteriores admitiu-se a existência da derivada de u com respeito às variáveis de projeto $\mathbf{p}$. Serão consideradas agora as condições para esta existência.

Para um dado projeto $\mathbf{p}$, assumem-se como hipóteses que a forma bilinear $a_{\mathbf{p}}(u, v) = \mathbf{v} \cdot \mathbf{K}\mathbf{u}$ seja contínua, coerciva e limitada e a forma linear $l_{\mathbf{p}}(v) = \mathbf{f} \cdot \mathbf{v}$ seja contínua. Dessa forma respeitam-se as condições do teorema de Lax-Milgram, garantindo-se a existência e a unicidade da solução do problema variacional (1.4). A proposição seguinte estabelece as condições para a existência da derivada da solução da equação de estado.

Proposição 3.1 *Se* $a_{\mathbf{p}}$ *e* $l_{\mathbf{p}}$ *satisfazem as hipóteses anteriores e as aplicações* $\mathbf{p} \mapsto a_{\mathbf{p}}$ *e* $\mathbf{p} \mapsto l_{\mathbf{p}}$ *são continuamente diferenciáveis em* $\mathbf{p}$, *logo* $\mathbf{p} \mapsto u(\mathbf{p})$, *solução de* (1.4) *é continuamente diferenciável, sendo* $\dot{u}$ *dada por*

$$a_{\mathbf{p}}(\dot{u}, v) = \dot{l}_{\mathbf{p}}(v) - \dot{a}_{\mathbf{p}}(u, v)$$

Esta proposição pode ser enunciada de maneira análoga para o caso discreto.

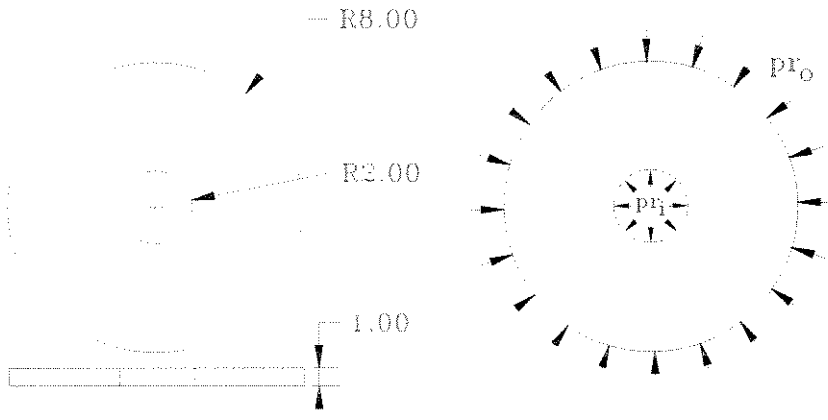


Figura 3.1: Dimensões e carregamento do disco.

3.5 Estudo de Casos

3.5.1 Disco Sujeito a Pressões — Sensibilidade à Variação da Espessura

Considere o disco de pequena espessura sujeito a pressões nas superfícies interna e externa, de acordo com a Figura 3.1. A solução analítica deste problema em termos do deslocamento radial $u(r)$ tensões radial σ_r e tangencial σ_t, é dada pelas seguintes expressões [76],

$$u(r) = \left(\frac{1-\tilde{\nu}}{\tilde{E}}\right) C_1 r + \left(\frac{1+\tilde{\nu}}{\tilde{E}}\right) C_2/r$$

$$\sigma_r = C_1 - C_2/r^2 \qquad \sigma_t = C_1 + C_2/r^2$$

onde $\tilde{E}$ e $\tilde{\nu}$ são, respectivamente, o módulo de elasticidade e o coeficiente de Poisson do material. Por sua vez, as constantes C_1 e C_2 são definidas como,

$$C_1 = \frac{1}{p_1}\left[\frac{f_i r_i^2 - f_o r_o^2}{r_o^2 - r_i^2}\right] \qquad C_2 = \frac{1}{p_1}\left[\frac{(f_i - f_o)\, r_i^2 r_o^2}{r_o^2 - r_i^2}\right]$$

sendo p_1, r_i e r_o a espessura e os raios interno e externo do disco; f_i e f_o os carregamentos distribuídos nos perímetros interno e externo do disco. Assim $pr_i = f_i/p_1$ e $pr_o = f_o/p_1$, onde pr_i é a pressão interna e pr_o a pressão externa. Os seguintes valores numéricos foram adotados,

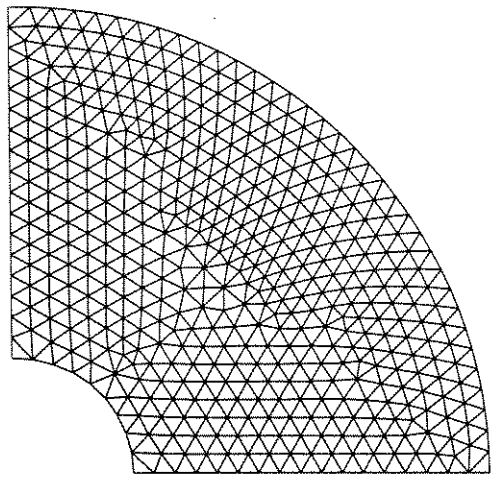
$\tilde{E} = 2.1 \times 10^6$ kN/cm^2	$r_i = 2$ cm	$f_i = 100$ kN/cm
$\tilde{\nu} = 0.3$	$r_o = 8$ cm	$f_o = 10$ kN/cm
	$p_1 = 1$ cm	

considerando ainda $u_a = 0.0005$ cm e $\sigma_a = 1500$ kN/cm^2 como valores máximos admissíveis em deslocamento e tensão equivalente.

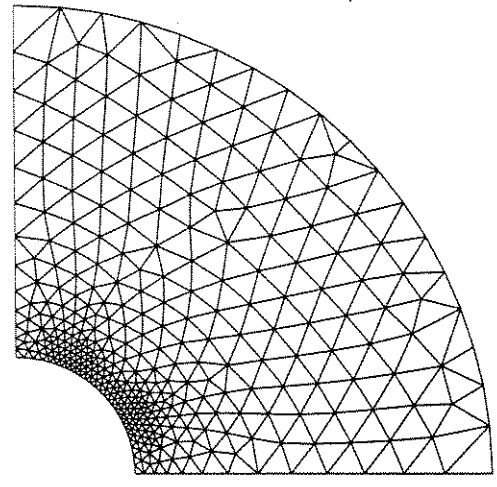
O problema foi discretizado utilizando elementos lineares (malhas 1 e 2 — Figura 3.2) e quadráticos (malhas 3 e 4 — Figura 3.3) de tamanho médio constante e com refinamento adaptável.

Para o funcional ψ_3 de deslocamento resultante máximo (1.15), os valores obtidos analítica e numericamente são mostrados na Tabela 3.1. Pode-se observar nesses resultados que quanto mais adequada for a malha para a análise de elementos finitos, mais precisos são os valores obtidos para o funcional de deslocamento ψ_3 e sua derivada.

A Tabela 3.2 mostra os resultados obtidos para o funcional de tensão equivalente de von Mises. As duas primeiras colunas fazem a comparação entre o valor *pontual* máximo analítico dessa tensão e os valores máximos de tensão *nodal* em cada uma das malhas. Como esperado,

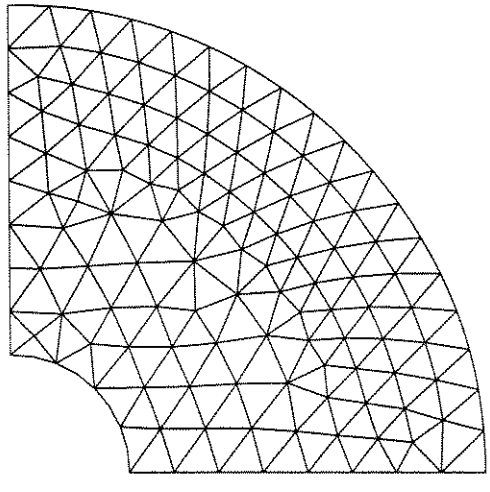


(a) Malha 1: Elementos lineares com tamanho médio constante.

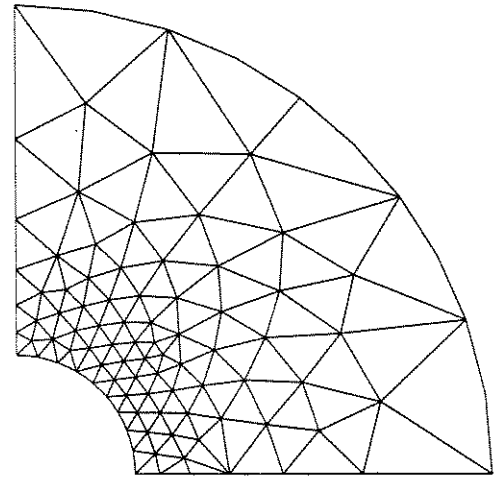


(b) Malha 2: Elementos lineares com refinamento por análise adaptável.

Figura 3.2: Malhas de elementos lineares.



(a) Malha 3: Elementos quadráticos com tamanho médio constante.



(b) Malha 4: Elementos quadráticos com refinamento por análise adaptável.

Figura 3.3: Malhas de elementos quadráticos.

Tabela 3.1: Solução analítica e resultados numéricos do disco para o funcional $\psi_3(p_1)$ e sua derivada em $p_1 = 1$.

	$\psi_3(p_1)/u_a - 1$	Erro%	$(d\psi_3/dp_1)/u_a$	Erro%
Analítico	−0.7676190	—	−0.2323810	—
Malha 1	−0.7684886	0.1132854	−0.2315119	0.3739979
Malha 2	−0.7680394	0.0547668	−0.2319730	0.1755737
Malha 3	−0.7679410	0.0419479	−0.2320576	0.1391680
Malha 4	−0.7675882	0.0040124	−0.2324100	0.0124795

Tabela 3.2: Solução analítica e resultados numéricos do disco para tensão nodal máxima e para o funcional ψ_2 e sua derivada em $p_1 = 1$.

	$\sigma_{MAX}/\sigma_a - 1$	Erro%	$\psi_2(p_1)/\sigma_a - 1$	Erro%	$(d\psi_2/dp_1)/\sigma_a$	Erro%
Analítico	−0.8891167	—	−0.8891167	—	−0.1108832	—
Malha 1	−0.8949121	0.6518154	−0.8949121	0.6518154	−0.1050603	5.2513816
Malha 2	−0.8926842	0.4012409	−0.8926842	0.4012409	−0.1073265	3.2076094
Malha 3	−0.8905253	0.1584481	−0.8998823	1.2108197	−0.1001086	9.7170717
Malha 4	−0.8895874	0.0529384	−0.8981857	1.0200011	−0.1017961	8.1952000

os resultados mais precisos são obtidos com a utilização de técnicas de refinamento e elementos de maior grau. As colunas seguintes mostram o efeito de se utilizar o funcional ψ_2 (1.14), de máxima tensão de von Mises ponderada no domínio do elemento, como estimativa da máxima tensão nodal de von Mises. Observa-se que em algumas situações (como nas malhas 1 e 2), o funcional ψ_2 é uma boa estimativa da tensão nodal máxima. Nestes dois casos, a tensão nodal máxima coincide com o valor do funcional ψ_2 devido às características do modelo, do elemento linear (tensão constante) e ao pequeno tamanho dos elementos. Entretanto, nas malhas 3 e 4, onde os resultados da técnica de elementos finitos são mais precisos, ocorrem os maiores erros dos valores de ψ_2 e sua derivada em comparação com a solução analítica de tensão pontual máxima. Isto ocorre pois as malhas de elementos quadráticos apresentam elementos de maior tamanho maiores, se comparadas a malhas de elementos lineares, para atingir o mesmo nível de erro no deslocamento. Além disso, elementos quadráticos apresentam variação linear da tensão, reduzindo o valor da tensão ponderada no domínio do elemento.

Esse resultado mostra que ao utilizar o funcional ψ_2 como restrição na otimização da espessura do disco, o valor máximo admitido para a tensão ponderada no elemento deve ser ajustado de forma a manter a tensão nodal máxima abaixo do valor admissível do material. Assim, em um problema de otimização, que envolve uma seqüência de soluções do modelo de elementos finitos, a malha de elementos quadráticos com refinamento mostrada na Figura 3.3b) conduzirá ao resultado mais preciso, desde que o valor σ_a seja ajustado corretamente (observar os resultados das Tabelas 3.4 e 3.5).

3.5.2 Otimização da Espessura do Disco

Deslocamento Absoluto como Restrição

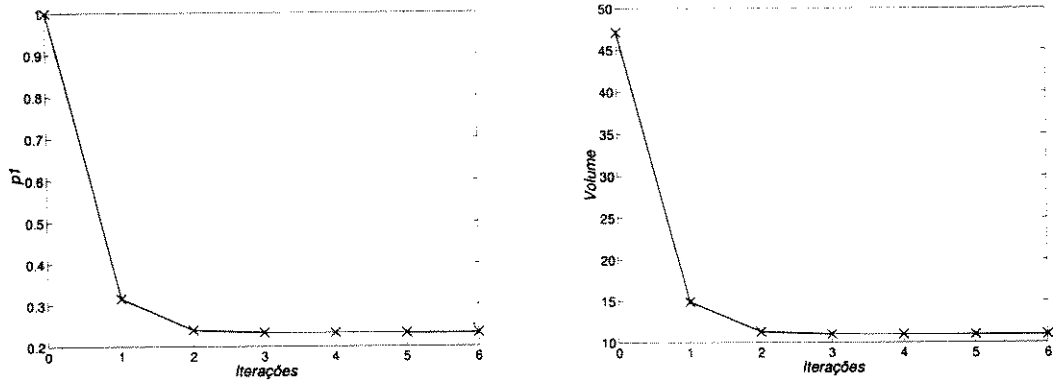
Considere o problema de minimizar a massa do disco do exemplo com a restrição funcional de se manter o deslocamento radial máximo menor ou igual a $u_a = 0.0005$ cm. Foi aplicada o método de pontos interiores de Herskovits descrito no Capítulo 2 com as constantes $\alpha = 0.8$, $\eta = 0.2$, $\varepsilon = 0.1$, $\xi = 10^{-7}$, $\delta = 0.1$, $\bar{\rho} = 1.0$, $\nu = 0.8$, $\gamma = 0.1$ e espessura inicial $p_1 = 1.0$ cm. A Tabela 3.3 apresenta os resultados para cada uma das malhas utilizadas no exemplo anterior, mostrando os valores da espessura ótima p_{1final} e respectivos erros relativos à solução analítica, o número de iterações do método de Herskovits, o número de análises do modelo de elementos finitos e o número de soluções de problemas adjuntos. Em todos os casos foram obtidos resultados satisfatórios, sendo que o resultado mais preciso foi obtido com a malha de elementos quadráticos com refinamento (malha 4). A Figura 3.4 mostram a evolução da espessura e do volume do disco utilizando esta última discretização.

Tensão Equivalente de von Mises como Restrição

Utilizando a restrição funcional de se manter a tensão pontual máxima menor que 1500 kN/cm^2, otimizou-se a espessura do disco do exemplo anterior. A Tabela 3.4 mostra os resultados

Tabela 3.3: Espessuras ótimas do disco obtidas com o deslocamento radial máximo como restrição.

	$p_1 final$	Erro%	Iterações	Análises MEF	Problemas Adjuntos
Analítico	0.23238095	—	—	—	—
Malha 1	0.23151532	0.37250472	6	17	6
Malha 2	0.23196448	0.17921865	6	17	6
Malha 3	0.23205941	0.13836762	6	15	6
Malha 4	0.23241573	0.01496680	6	17	6



(a) Evolução da espessura do disco nas iterações.

(b) Evolução do volume do disco nas iterações.

Figura 3.4: Iterações de otimização da espessura do disco (Malha 4 — restrição no deslocamento resultante).

obtidos com as malhas 1 a 4 para o valor $\sigma_a = 1500$ kN/cm^2 como valor máximo da tensão média nos elementos. Observa-se que os resultados das duas útlimas malhas mostram a necessidade de se ajustar o limite nestes casos, como discutido na seção 3.5.1. Na Tabela 3.5, onde foram utilizados os valores $\sigma_a = 1500$ kN/cm^2 para as malhas 1 e 2, $\sigma_a = 1370$ kN/cm^2 para a malha 3 e $\sigma_a = 1380$ kN/cm^2 para a malha 4, os valores máximos de tensão nodal são mantidos abaixo de 1500 kN/cm^2. Assim como no funcional de deslocamento resultante, resultados mais precisos são obtidos com a malha mais adequada para a análise. Os gráficos da Figura 3.5 mostram a evolução da espessura e do volume do disco com a malha 4.

3.5.3 Avaliação Numérica da Análise de Sensibilidade

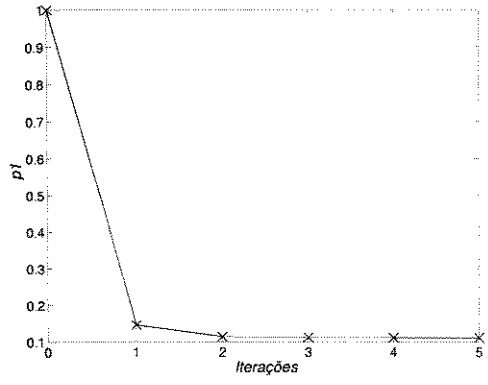
No caso geral, a avaliação da análise de sensibilidade somente pode ser feita de maneira numérica, comparando seus resultados com os obtidos por técnicas de diferenças finitas. Considere, por exemplo, o componente mostrado na Figura 3.6 sujeito às condições de contorno da

Tabela 3.4: Espessuras ótimas do disco obtidas com a tensão equivalente de von Mises como como restrição ($\sigma_a = 1500$ kN/cm^2 para todos os caso; σ_{MAX}: tensão nodal máxima da análise).

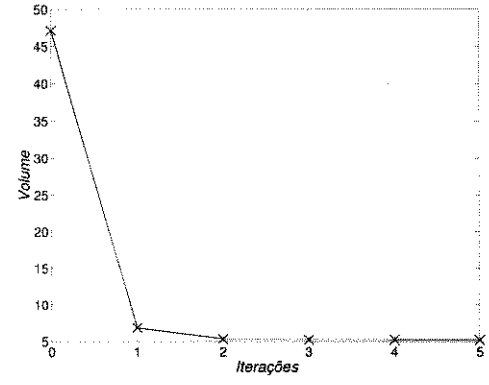
	$p_1 final$	Erro%	σ_{MAX}	Iterações	Análises MEF	Problemas Adjuntos
Analítico	0.11088332	—	1500.000	—	—	—
Malha 1	0.10511012	5.20655406	1499.682	5	14	5
Malha 2	0.10733739	3.19789306	1499.699	5	14	5
Malha 3	0.10014411	9.68514471	1639.755	5	14	5
Malha 4	0.10183929	8.15634893	1626.277	5	14	5

Tabela 3.5: Espessuras ótimas do disco obtidas com a tensão equivalente de von Mises como como restrição (σ_a = 1500 kN/cm^2 para as malhas 1 e 2, σ_a = 1370 kN/cm^2 para a malha 3 e σ_a = 1380 kN/cm^2 para a malha 4; σ_{MAX}: tensão nodal máxima na análise).

	$p_1 final$	Erro%	σ_{MAX}	Iterações	Análises MEF	Problemas Adjuntos
Analítico	0.11088332	—	1500.000	—	—	—
Malha 1	0.10511012	5.20655406	1499.682	5	14	5
Malha 2	0.10733739	3.19789306	1499.699	5	14	5
Malha 3	0.10963828	1.12283795	1497.759	5	14	5
Malha 4	0.11068734	0.17674435	1496.277	5	14	5



(a) Evolução da espessura do disco nas iterações.



(b) Evolução do volume do disco nas iterações.

Figura 3.5: Iterações de otimização da espessura do disco (Malha 4 — restrição na tensão de von Mises)

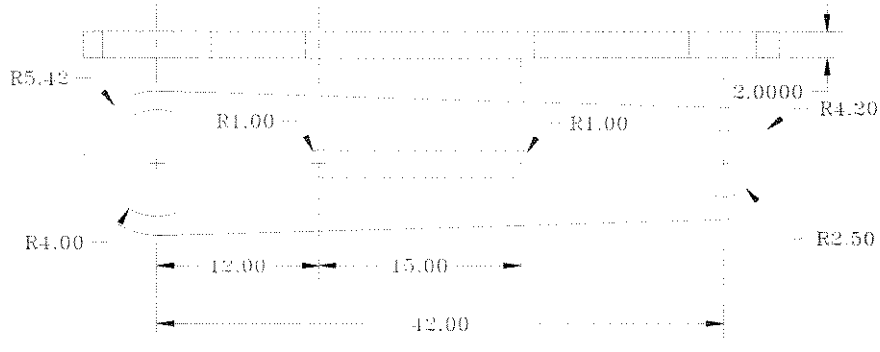


Figura 3.6: Dimensões do componente.

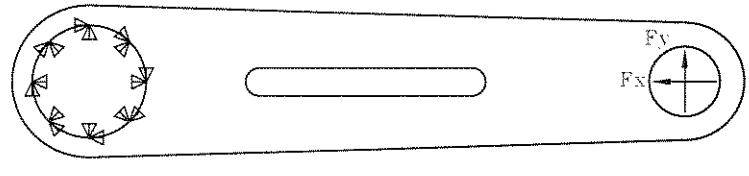


Figura 3.7: Condições de contorno do problema.

Tabela 3.6: Derivada da função $\psi_2/\sigma_a - 1$ em relação à espessura ($p_1 = 2$). Cálculo por diferenças finitas (MDF) com diferença à frente de 0.1% e por análise de sensibilidade (AS).

	MDF	AS	Erro%
$(d\psi_2/dp_1)/\sigma_a$	−0.46885000	−0.4693323	0.1028687

Figura 3.7.

Tomando-se uma espessura constante p_1 em todo o domínio, a avaliação se resume à comparação direta dos escalares obtidos por diferenças finitas (MDF) e por análise de sensibilidade (AS). Para os valores numéricos abaixo,

$$\tilde{E} = 2.1 \times 10^6 \text{ kN/cm}^2 \qquad \tilde{\nu} = 0.3 \qquad \sigma_a = 2500 \text{ kN/cm}^2 \qquad p_1 = 2 \text{ cm}$$

e aplicando a diferença à frente com 0.1% de variação, os resultados obtidos para o funcional ψ_2 de tensão de von Mises ponderada são mostrados na Tabela 3.6, onde se observa uma diferença de 0.10% em relação ao resultado obtido por diferenças finitas.

Suponha que o componente seja dividido em subregiões cujas espessuras possam variar independentemente. Nesse caso, a informação da variação é vetorial, sendo necessário estabelecer critérios para avaliação da precisão dos gradientes obtidos por análise de sensibilidade. TSENG e ARORA[37] apresentam os seguintes procedimentos de verificação:

- **Erro relativo da maior componente**. Pode-se comparar, componente a componente, o gradiente obtido por diferenças finitas e o resultado obtido pela análise de sensibilidade. Existem, contudo, dificuldades para aplicação dessa avaliação, especialmente quando existem grandes diferenças entre os valores absolutos das componentes do gradiente. O menor valor absoluto pode estar muito influenciado pelo ruído numérico e portanto, um grande erro no menor componente pode não significar um gradiente impreciso. O erro relativo da maior componente em valor absoluto pode ser usado para comparar os gradientes,

$$\text{Erro}_{MCA}\% = 100 \left| 1 - \frac{\left(\nabla\psi^{AS}\right)_i}{\left(\nabla\psi^{MDF}\right)_i} \right|$$

  onde $\left(\nabla\psi^{AS}\right)_i$ é a maior componente em valor absoluto calculado pela análise de sensibilidade e $\left(\nabla\psi^{MDF}\right)_i$ a componente correspondente calculada por diferenças finitas.

- **Produto interno normalizado**. Esse produto é definido por,

$$PIN = \frac{\nabla\psi^{AS} \cdot \nabla\psi^{MDF}}{\|\nabla\psi^{AS}\| \, \|\nabla\psi^{MDF}\|}$$

  Geometricamente é o cosseno do ângulo entre os dois vetores. Logo, a proximidade entre esse valor e 1 é uma medida da precisão alcançada. Assim,

$$\text{Erro}_{PIN} = 1 - PIN$$

Dividindo o componente da Figura 3.6 em duas subregiões de espessuras p_1 e p_2, mostradas na Figura 3.8, são obtidos os resultados da Tabela 3.7, mostrando os erros relativos de cada componente do vetor gradiente. Ainda que para a componente $\partial\psi_2/\partial p_1$, o erro relativo seja elevado, segundo os critérios acima obtém-se,

- $\text{Erro}_{MCA}\% = 0.1601043$,
- $\text{Erro}_{PIN} = 8.39 \times 10^{-10}$ (e erro de 0.1601040% no módulo do vetor gradiente).

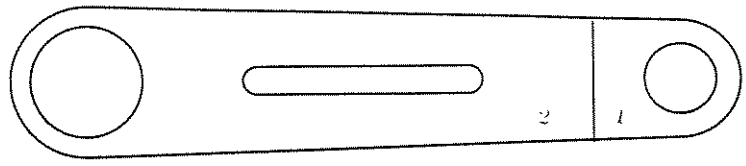


Figura 3.8: Divisão em 2 subdomínios.

Tabela 3.7: Derivadas parciais do funcional $\psi_2/\sigma_a - 1$ em relação às espessuras p_1 e p_2 em $(2,2)$. Cálculo por diferenças finitas (MDF) com diferença à frente de 0.1% e por análise de sensibilidade (AS).

	MDF	AS	Erro%
$(\partial\psi_2/\partial p_1)/\sigma_a$	0.0000500	0.0000304	**64.473684**
$(\partial\psi_2/\partial p_2)/\sigma_a$	−0.479250	−0.4800173	**0.1601043**

3.5.4 Otimização de Espessura de um Componente — Divisão em subdomínios

Com o objetivo de reduzir ao máximo o volume do componente do exemplo anterior, o domínio original foi subdividido em 2, 3 e 4 subdomínios (Figuras 3.9a) a 3.9d)) cujas espessuras podem variar de maneira independente. A Tabela 3.8 mostra os resultados obtidos com cada parametrização e o número de iterações e análises do modelo de elementos finitos.

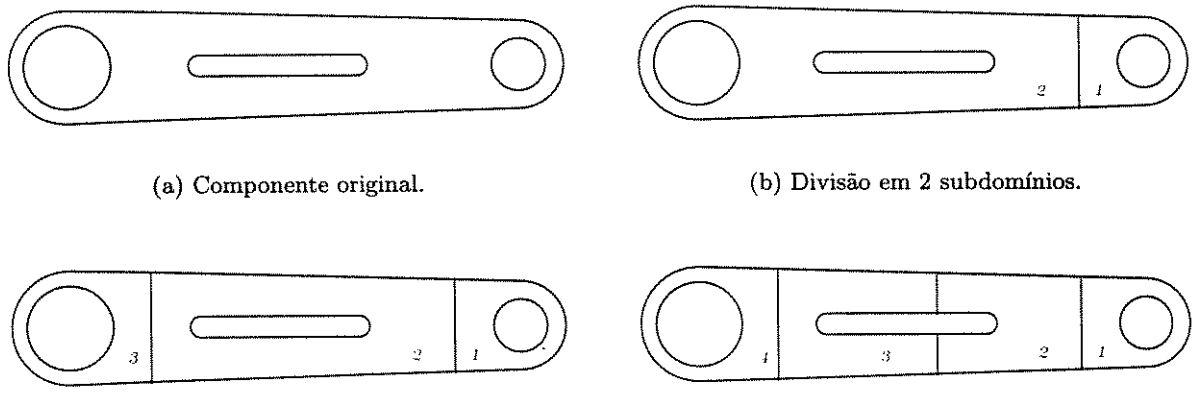
Um maior número de subdomínios permite que espessuras maiores sejam mantidas apenas em regiões de maior tensão, diminuindo a espessura nas demais. Entretanto, o aumento do número de variáveis de projeto torna o problema de otimização mais complexo, exigindo um número maior iterações para atingir a solução. Uma das causas dessa maior complexidade é a possibilidade de que ocorram discontinuidades no gradiente da função ψ_2, devido à oscilações do local de ocorrência da tensão máxima, exigindo passos pequenos, reduzindo a convergência em algumas iterações.

Os gráficos das Figuras 3.10 e 3.11 trazem a seqüência dos valores da função objetivo (volume) e das variáveis de projeto em parametrização. Pode-se observar, como ponto positivo dos algoritmos empregados, que não ocorrem oscilações dos valores da função objetivo e das variáveis de projeto do problema. A Tabela 3.9 mostra os projetos ótimos obtidos.

A Figura 3.12 traz a distribuição final da tensão de von Mises em cada caso. Com o maior número de subdivisões, observa-se um aproveitamento mais racional do material.

Tabela 3.8: Resultados da otimização com cada parametrização.

	Volume_{final}	$(\psi_2/\sigma_a - 1)_{final}$	Iterações	Análises MEF
Original	657.01902580	$-1.606058e-05$	2	2
2 Subdomínios	593.49351327	$-6.302077e-06$	7	17
3 Subdomínios	571.83410367	$-1.716869e-06$	16	52
4 Subdomínios	512.41437110	$-7.006431e-05$	20	61



(a) Componente original.

(b) Divisão em 2 subdomínios.

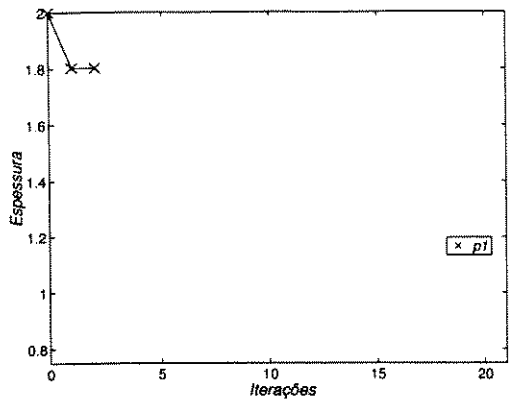
(c) Divisão em 3 subdomínios.

(d) Divisão em 4 subdomínios.

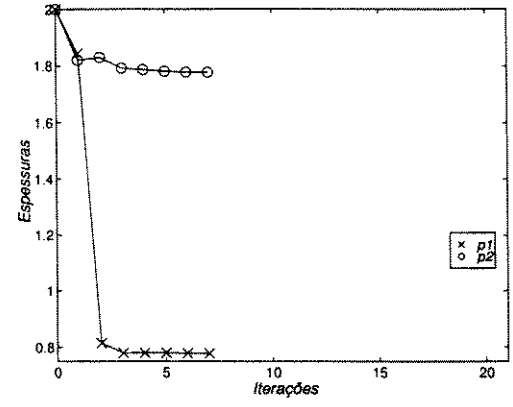
Figura 3.9: Compenente original de espessura única e divisões em subdomínios de espessuras independentes.

Tabela 3.9: Projetos ótimos obtidos com cada parametrização

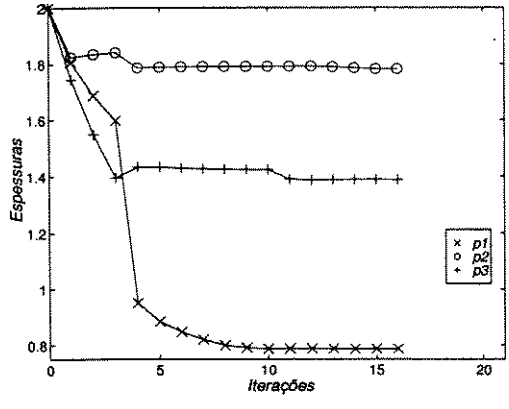
Parametrização	p_1	p_2	p_3	p_4	Volume
Original	1.80233091	—	—	—	657.01902580
2 Subdomínios	0.77780732	1.77758153	—	—	593.49351327
3 Subdomínios	0.78662093	1.78196663	1.38799162	—	571.83410367
4 Subdomínios	0.75572807	1.28249003	1.79727401	1.28239267	512.41437110



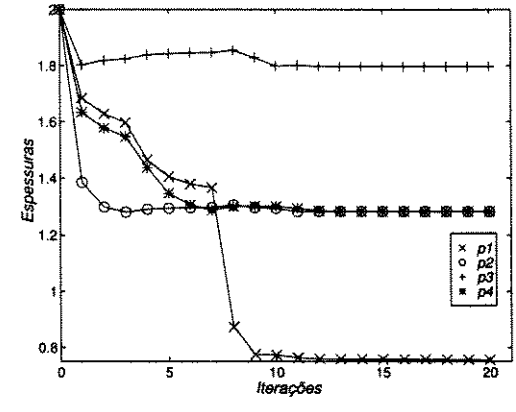
(a) Componente original.



(b) Divisão em 2 subdomínios.



(c) Divisão em 3 subdomínios.



(d) Divisão em 4 subdomínios.

Figura 3.10: Seqüência das variáveis de projeto (espessuras) em cada parametrização do problema.

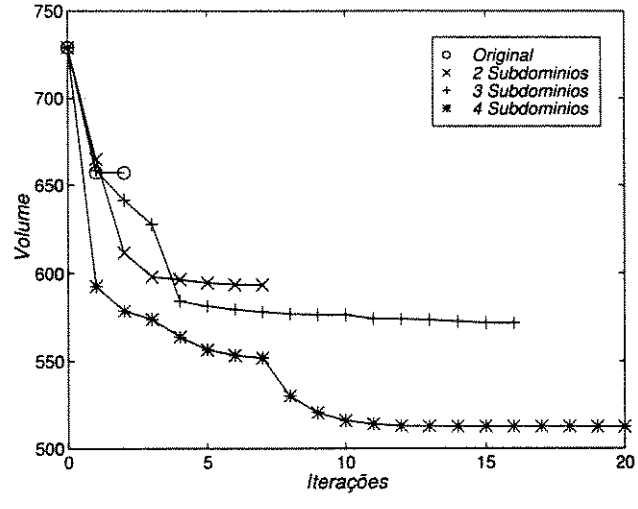
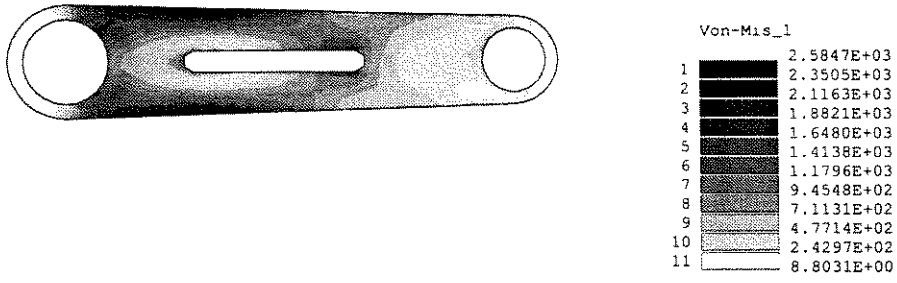
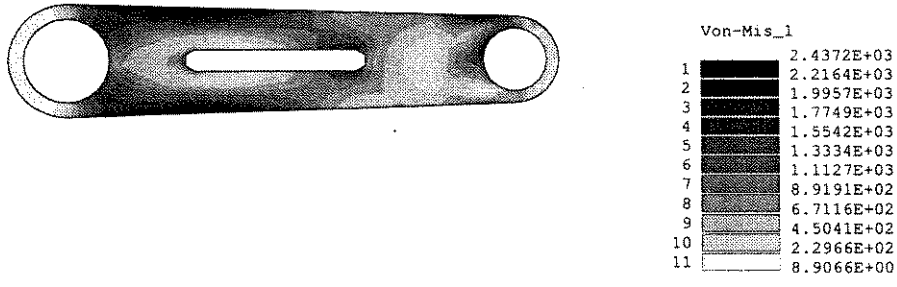


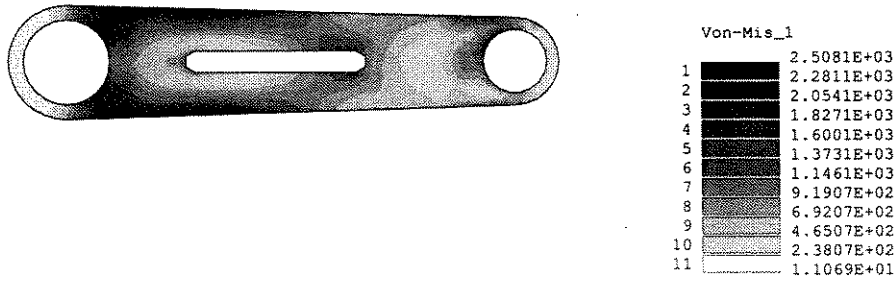
Figura 3.11: Evolução do volume do componente em cada parametrização do problema.



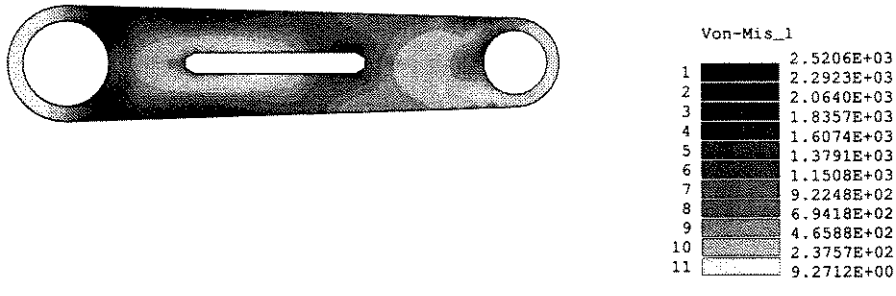
(a) Componente original.



(b) Divisão em 2 subdomínios.



(c) Divisão em 3 subdomínios.



(d) Divisão em 4 subdomínios.

Figura 3.12: Tensões de von Mises do projeto ótimo em cada parametrização.

Capítulo 4

Análise de Sensibilidade à Mudança de Forma

No capítulo anterior o vetor $\mathbf{p}$ parametrizava características do corpo elástico que não afetavam a geometria do domínio $\mathcal{B}$. Existem situações, entretanto, em que não apenas tais parâmetros, mas o próprio domínio $\mathcal{B}$, são variáveis do problema, exigindo a avaliação de funcionais e suas sensibilidades em relação à variação da forma de $\mathcal{B}$.

Será adotada a formulação contínua para o desenvolvimento das expressões de sensibilidade à mudança de forma, em conjunto com o método adjunto no caso de funcionais que dependam da resposta da equação de estado.

A formulação contínua da análise de sensibilidade à mudança de forma se baseia na definição de uma transformação descrevendo a variação do domínio em torno do domínio original. Esta transformação é descrita por um campo de velocidades, usado na determinação da derivada material do funcional, ou seja, sua derivada total. Os enunciados de sensibilidade dos funcionais são mantidos como integrais de domínio, pois o método de elementos finitos fornece uma melhor qualidade de resultados no interior do domínio.

O campo de velocidades é definido univocamente no contorno pela parametrização adotada para o mesmo. Nesse trabalho, adota-se uma parametrização em *NURBS* (*Non Uniform Rational B-Splines*), pois fornecem um enunciado simples e preciso para a descrição de formas geométricas usualmente empregadas em modelagem geométrica. O campo de velocidades no interior do domínio é definido de forma a tornar eficientes das integrais de domínio.

Por último, estes conceitos são empregados em exemplos de cálculo de sensibilidade e otimização.

4.1 Conceitos Gerais

Os corpos têm a propriedade física de ocuparem regiões no espaço euclidiano pontual $\mathcal{E}$. Apesar de nenhuma dessas regiões poder ser intrinsecamente associada ao corpo, convenciona-se adotar uma dessas regiões, denotada por $\mathcal{B}$, como *configuração de referência*, associando os pontos do corpo com suas posições em $\mathcal{B}$. Assim, 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* $\mathsf{X}^\tau(\mathbf{x},\tau)$ de um corpo $\mathcal{B}$ é uma função de classe C^3 tal que

$$\begin{aligned} \mathsf{X}^\tau : \ & \mathcal{B} \times \Re \rightarrow \mathcal{E} \\ & (\mathbf{x},\tau) \rightarrow \mathbf{x}^\tau \end{aligned} \tag{4.1}$$

onde, para um τ fixo, tem-se uma transformação injetora suave. Nessa transformação, $\mathbf{x}^\tau =$

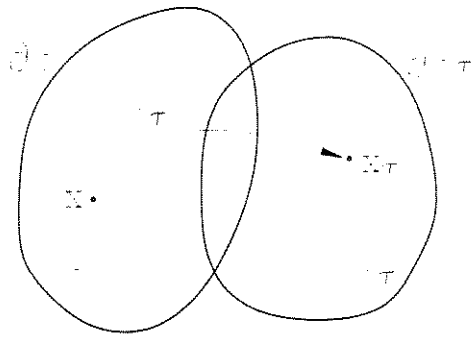


Figura 4.1: Movimento como uma seqüência de deformações.

$\mathsf{X}^\tau(\mathbf{x},\tau)$ é a posição do ponto $\mathbf{x}$ no tempo τ, enquanto $\mathcal{B}_\tau = \mathsf{X}^\tau(\mathcal{B},\tau)$ é a região ocupada pelo corpo em τ. Para cada τ fixo, $\mathsf{X}^\tau(\mathbf{x},\tau)$ representa uma deformação de $\mathcal{B}$. Logo, um movimento é uma família uniparamétrica de deformações em função do parâmetro τ, como ilustrado na Figura 4.1.

Define-se ainda a *trajetória* $\mathcal{T}$ de um ponto $\mathbf{x} \in \mathcal{B}$ como o conjunto de posições assumidas por $\mathbf{x}$ ao longo do tempo originadas pelo movimento, ou seja,

$$\mathcal{T} = \{(\mathbf{x}^\tau,\tau)\,|\mathbf{x}^\tau \in \mathcal{B}_\tau, \tau \in \Re\}$$

Como X^τ é um mapeamento injetor para todo τ, existe transformação inversa $\mathsf{X}(\cdot,\tau) \equiv [\mathsf{X}^\tau(\cdot,\tau)]^{-1}$, $\mathsf{X}(\cdot,\tau) : \mathcal{B}_\tau \to \mathcal{B}$ tal que

$$\mathsf{X}^\tau(\mathsf{X}(\mathbf{x}^\tau,\tau),\tau) = \mathbf{x}^\tau \qquad \mathsf{X}(\mathsf{X}^\tau(\mathbf{x},\tau),\tau) = \mathbf{x} \tag{4.2}$$

Logo, dado $(\mathbf{x}^\tau,\tau) \in \mathcal{T}$, então $\mathbf{x} = \mathsf{X}(\mathbf{x}^\tau,\tau)$ é o ponto material que ocupa o lugar $\mathbf{x}^\tau$ no tempo τ.

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

$$\Phi_s(\mathbf{x}^\tau,\tau) = \Phi(\mathsf{X}(\mathbf{x}^\tau,\tau),\tau)$$

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

$$\Psi_m(\mathbf{x},\tau) = \Psi(\mathsf{X}^\tau(\mathbf{x},\tau),\tau)$$

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

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

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

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

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

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

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

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

O divergente de um campo vetorial ou tensorial também pode ser definido em termos

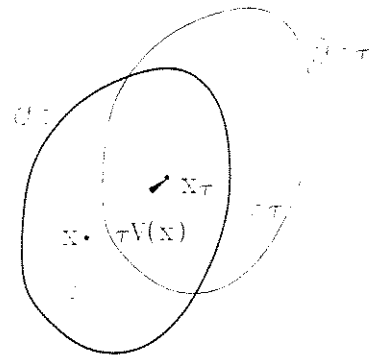


Figura 4.2: Variação do domínio $\mathcal{B}$ para $\mathcal{B}_\tau$.

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,

$$\dot{\Psi} = [(\Psi_m)^{\cdot}]_s = \frac{\partial}{\partial \tau}\Psi\left(\mathsf{X}^\tau(\mathbf{x},\tau),\tau\right)\Big|_{\mathbf{x}=\mathsf{X}(\mathbf{x}^\tau,\tau)} \tag{4.3}$$

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 a descrição espacial. Verifica-se ainda que a derivada material no tempo é comutativa com as transformações material e espacial, ou seja,

$$\left(\dot{\Phi}\right)_s = (\Phi_s)^{\cdot} = \dot{\Phi}_s \qquad \left(\dot{\Psi}\right)_m = (\Psi_m)^{\cdot} = \dot{\Psi}_m$$

Pela regra da cadeia obtém-se que, no caso de um campo escalar Ψ,

$$\dot{\Psi} = \Psi' + \mathbf{V}_s \cdot \text{ grad } \Psi \tag{4.4}$$

Se Ψ for um campo vetorial,

$$\dot{\Psi} = \Psi' + (\text{grad } \Psi)\,\mathbf{V}_s \tag{4.5}$$

O campo vetorial espacial $\mathbf{V}_s(\mathbf{x}^\tau,\tau)$ é definido como a descrição espacial da velocidade,

$$\mathbf{V}_s(\mathbf{x}^\tau,\tau) = \frac{\partial}{\partial \tau}\mathsf{X}_s^\tau = \frac{\partial}{\partial \tau}\mathsf{X}^\tau\left(\mathsf{X}(\mathbf{x}^\tau,\tau),\tau\right) = \dot{\mathsf{X}}^\tau\left(\mathsf{X}(\mathbf{x}^\tau,\tau),\tau\right) \tag{4.6}$$

4.2 Variação da Geometria do Domínio de um Problema Elástico

Como mencionado anteriormente, num problema de otimização de forma, as variáveis de projeto estão associadas a forma do domínio $\mathcal{B}$ do problema. Desta maneira, torna-se necessário definir um mapeamento entre os domínios original $\mathcal{B}$ e aquele obtido a partir de uma certa variação de $\mathcal{B}$ indicada como $\mathcal{B}_\tau$. Para isso, considera-se a definição de um movimento dado em (4.1), ilustrado na Figura 4.1.

Como a transformação $\mathsf{X}^\tau(\mathbf{x},\tau)$ é suave e contínua, toma-se a seguinte expansão na vizinhança de $\tau = 0$,

$$\mathsf{X}^\tau(\mathbf{x},\tau) = \mathsf{X}^\tau(\mathbf{x},0) + \tau\frac{\partial}{\partial \tau}\mathsf{X}^\tau(\mathbf{x},0) + o(\tau^n) \qquad n = 2,3,\ldots$$

Desprezando os termos de ordem superior $o(\tau^n)$ tem-se,

$$\mathbf{x}^\tau = \mathsf{X}^\tau(\mathbf{x},\tau) = \mathbf{x} + \tau\mathbf{V}(\mathbf{x}) \qquad \mathbf{x} \in \mathcal{B} \tag{4.7}$$

onde, a partir de (4.6), $\mathbf{V}(\mathbf{x}) = \mathbf{V}_s(\mathbf{x}^\tau,\tau)|_{\tau=0}$. O campo vetorial suave $\mathbf{V}(\mathbf{x})$ é denominado velocidade de mapeamento de projeto. A transformação (4.7), ilustrada na Figura 4.2, será

empregada para obter variações do domínio $\mathcal{B}$ pelo campo de velocidades $\mathbf{V}(\mathbf{x})$, sendo o novo domínio indicado por $\mathcal{B}_\tau = \mathsf{X}^\tau(\mathcal{B},\tau)$ com contorno $\partial\mathcal{B}_\tau = \mathsf{X}^\tau(\partial\mathcal{B},\tau)$. Logo,

$$\mathcal{B}_\tau = \left\{\mathbf{x}^\tau \in \Re^3 : \forall \mathbf{x} \in \mathcal{B}, \mathbf{x}^\tau = \mathbf{x} + \tau\mathbf{V}(\mathbf{x}), \mathbf{x}_0 = \mathbf{x}\right\} \tag{4.8}$$

$$\partial\mathcal{B}_\tau = \left\{\mathbf{x}^\tau \in \Re^3 : \forall \mathbf{x} \in \partial\mathcal{B}, \mathbf{x}^\tau = \mathbf{x} + \tau\mathbf{V}(\mathbf{x}), \mathbf{x}_0 = \mathbf{x}\right\} \tag{4.9}$$

Em cada τ, $\mathsf{X}^\tau(\cdot,\tau)$ é uma transformação injetora de $\mathcal{B}$ para $\mathcal{B}_\tau$, possuindo inversa $\mathsf{X}(\cdot,\tau) : \mathcal{B}_\tau \rightarrow \mathcal{B}$ tal que as relações em (4.2) são válidas.

A transformação X^τ é tal que em cada configuração $\mathcal{B}_\tau$, pode-se definir um problema elastostático linear análogo a (1.1),

$$\begin{cases} \mathbf{E}^\tau = \mathbf{E}(u^\tau) = \frac{1}{2}\left(\text{grad } u^\tau + \text{grad}^T u^\tau\right) \\ \mathbf{T}^\tau = \mathbf{T}(u^\tau) = \mathsf{C}[\mathbf{E}^\tau] \\ \text{div } \mathbf{T}^\tau + \mathbf{b}^\tau = \mathbf{0} \end{cases} \tag{4.10}$$

com condições de contorno,

$$\begin{array}{ll} u^\tau \equiv \mathbf{0} & \mathbf{x}_\tau \in \Gamma_\tau^1 \\ \mathbf{Tn} \equiv \mathbf{\Phi}^\tau & \mathbf{x}_\tau \in \Gamma_\tau^2 \end{array} \tag{4.11}$$

onde $\partial\mathcal{B}_\tau = \Gamma_\tau^0 \cup \Gamma_\tau^1 \cup \Gamma_\tau^2$ e $\overset{\circ}{\Gamma}{}_\tau^0 \cap \overset{\circ}{\Gamma}{}_\tau^1 \cap \overset{\circ}{\Gamma}{}_\tau^2 = \emptyset$, sendo Γ_τ^0 uma região do contorno sem condições especificadas. Aplicando resíduos ponderados ou o princípio dos trabalhos virtuais obtem-se,

$$\int_{\mathcal{B}_\tau} \mathbf{T}(u^\tau)\cdot\mathbf{E}(v^\tau)\ dV = \int_{\partial\mathcal{B}_\tau} \mathbf{T}(u^\tau)\,\mathbf{n}\cdot v^\tau\ dA + \int_{\mathcal{B}_\tau} \mathbf{b}^\tau\cdot v^\tau\ dV \tag{4.12}$$

Considerando as condições de contorno (4.11), chega-se ao seguinte problema variacional,

Determinar u^τ tal que

$$a_{\mathcal{B}_\tau}(u^\tau,v^\tau) = l_{\mathcal{B}_\tau}(v^\tau) \qquad \forall v^\tau \in \mathcal{V}_\tau \tag{4.13}$$

onde $\mathcal{V}_\tau$ é o espaço das soluções cinematicamente possíveis para o domínio $\mathcal{B}_\tau$.

Assim, a partir de (4.10) e (4.12), as formas bilinear e linear do problema elastostático são dadas por

$$a_{\mathcal{B}_\tau}(u^\tau,v^\tau) = \int_{\mathcal{B}_\tau} \mathbf{T}(u^\tau)\cdot\mathbf{E}(v^\tau)\ dV \tag{4.14}$$

$$l_{\mathcal{B}_\tau}(v^\tau) = \int_{\Gamma_\tau^2} \mathbf{\Phi}^\tau\cdot v^\tau\ dA + \int_{\mathcal{B}_\tau} \mathbf{b}^\tau\cdot v^\tau\ dV \tag{4.15}$$

Portanto, u^τ é a solução da equação de estado do problema elastostático linear no domínio $\mathcal{B}_\tau$.

Será adotada a seguinte notação simplificada para o problema variacional (4.13),

$$\tau \rightarrow \mathcal{B}_\tau : \qquad a_{\mathcal{B}_\tau}(u^\tau,v^\tau) = l_{\mathcal{B}_\tau}(v^\tau) \rightarrow a^\tau(u^\tau,v^\tau) = l^\tau(v^\tau) \tag{4.16}$$

onde, na configuração de referência,

$$\tau = 0 \rightarrow \mathcal{B}_0 = \mathcal{B} : \qquad a_{\mathcal{B}}(u,v) = l_{\mathcal{B}}(v) \rightarrow a(u,v) = l(v) \tag{4.17}$$

Como a cada τ tem-se um problema variacional diferente, sua solução no domínio $\mathcal{B}_\tau$ é o campo vetorial espacial $u^\tau = u^\tau(\mathbf{x}^\tau,\tau)$.

A derivada material no tempo de u_τ pode ser obtida como indicado em (4.3), estando a diferenciabilidade de u_τ demonstrada em [31]. Portanto,

$$\dot{u}^\tau = (\dot{u_m^\tau})_s = \frac{\partial}{\partial\tau} u^\tau(\mathsf{X}_\tau(\mathbf{x},\tau),\tau)\Big|_{\mathbf{x}=\mathsf{X}(\mathbf{x}^\tau,\tau)}$$

Empregando (4.5) obtém-se,

$$\dot{u}^\tau(\mathbf{x}^\tau,\tau) = u^{\tau\prime}(\mathbf{x}^\tau,\tau) + (\text{grad } u^\tau)\,\mathbf{V}_s(\mathbf{x}^\tau,\tau) \tag{4.18}$$

onde $u^{\tau\prime}(\mathbf{x}^\tau,\tau) = \dfrac{\partial}{\partial\tau} u^\tau(\mathbf{x}^\tau,\tau)$ é a derivada espacial de u^τ com respeito a τ mantendo $\mathbf{x}^\tau$ fixo.

Avaliando (4.18) em $\tau = 0$, tem-se que,

$$\dot{u}^{\tau}(\mathbf{x}^{\tau},\tau)|_{\tau=0} = u^{\tau\prime}(\mathbf{x}^{\tau},\tau)|_{\tau=0} + (\text{grad } u^{\tau})\,\mathbf{V}_s(\mathbf{x}^{\tau},\tau)|_{\tau=0}$$

$$\dot{u}(\mathbf{x}) = u'(\mathbf{x}) + (\nabla u)\,\mathbf{V}(\mathbf{x}) \tag{4.19}$$

sendo $\dot{u}(\mathbf{x}) = \dot{u}^{\tau}(\mathbf{x}^{\tau},\tau)|_{\tau=0}$ e $u'(\mathbf{x}) = u^{\tau\prime}(\mathbf{x}^{\tau},\tau)|_{\tau=0}$; o gradiente espacial grad u^{τ} se reduz a sua representação material ∇u na configuração de referência $\mathcal{B}$, ou seja, $\nabla u(\mathbf{x}) = \text{grad } u^{\tau}(\mathbf{x}^{\tau},\tau)|_{\tau=0}$.

O campo vetorial espacial u^{τ} tem a seguinte descrição material,

$$u^{\tau}(\mathbf{x}^{\tau},\tau) = u^{\tau}(\mathsf{X}_{\tau}(\mathbf{x},\tau),\tau) = u^{\tau}_{m}(\mathbf{x},\tau) \equiv u_{\tau}(\mathbf{x},\tau)$$

O gradiente material de u^{τ} é dado por,

$$\nabla u^{\tau}_{m} = (\text{grad } u^{\tau})_{m}\,\mathbf{F} \;\Rightarrow\; (\text{grad } u^{\tau})_{m} = \nabla u^{\tau}_{m}\mathbf{F}^{-1} \equiv \nabla u_{\tau}\mathbf{F}^{-1} \tag{4.20}$$

sendo $\mathbf{F}$ o gradiente material de X^{τ}, ou seja,

$$\mathbf{F} = \nabla \mathsf{X}^{\tau} = \mathbf{I} + \tau\nabla\mathbf{V}(\mathbf{x}) \tag{4.21}$$

A seguinte relação é válida para a derivada do determinante de $\mathbf{F}$ [32]:

$$(\det\mathbf{F})^{\cdot} = (\det\mathbf{F})\,(\mathbf{V}_s)_{m} \tag{4.22}$$

Observa-se que não é possível obter $\dot{u}(\mathbf{x})$ a partir de (4.19) pois a forma funcional de u^{τ} não é conhecida.

Considerando a transformação $\mathsf{X}_{\tau}(\mathbf{x},\tau)$ dada em (4.1) e um campo escalar contínuo φ em $\mathcal{B}_{\tau} = \mathsf{X}_{\tau}(\mathcal{B},\tau)$, então a seguinte relação é válida [32]:

$$\int_{\mathcal{B}_{\tau}} \varphi(\mathbf{x}^{\tau})\; dV = \int_{\mathcal{B}} \varphi(\mathsf{X}_{\tau}(\mathbf{x},\tau))\det\mathbf{F}(\mathbf{x})\; dV = \int_{\mathcal{B}} \varphi_{m}(\mathbf{x})\det\mathbf{F}(\mathbf{x})\; dV \tag{4.23}$$

Além disso, de forma geral, sendo φ e $\mathbf{w}$, respectivamente campos escalr e vetorial, vem que [32]:

$$\text{div}\;(\varphi\mathbf{w}) = \varphi\;\text{div}\;\mathbf{w} + \mathbf{w}\cdot\nabla\varphi \tag{4.24}$$

As expressões (4.22), (4.23) e (4.24) serão empregadas na próxima seção.

4.3 Sensibilidade de Funcionais Definidos num Domínio Variável

Considere, inicialmente, um funcional do tipo

$$\psi^{\tau} = \int_{\mathcal{B}_{\tau}} \mathcal{G}(\mathbf{x}^{\tau})\; dV \tag{4.25}$$

onde f^{τ} é um campo escalar espacial suave. Sua derivada material de tempo é a seguinte,

$$\dot{\psi}^{\tau} = \frac{d}{d\tau}\int_{\mathcal{B}_{\tau}} \mathcal{G}(\mathbf{x}^{\tau})\; dV = \frac{d}{d\tau}\int_{\mathcal{B}}^{\tau} \mathcal{G}_{m}(\mathbf{x})\det\mathbf{F}\; dV$$

Empregando (4.23) pode-se expressar esta derivada no domínio original $\mathcal{B}$, sendo possível passar a derivação para dentro da integral, pois o domínio de referência $\mathcal{B}$ é independente de τ. Logo,

$$\frac{d}{d\tau}\int_{\mathcal{B}}^{\tau} \mathcal{G}_{m}(\mathbf{x})\det\mathbf{F}\; dV = \int_{\mathcal{B}} (\mathcal{G}_{m}(\mathbf{x})\det\mathbf{F})^{\cdot}\; dV = \int_{\mathcal{B}} \dot{\mathcal{G}}_{m}(\mathbf{x})\det\mathbf{F} + \mathcal{G}_{m}(\mathbf{x})(\det\mathbf{F})^{\cdot}\; dV$$

Substituindo (4.22) na equação anterior, obtem-se,

$$\int_{\mathcal{B}} \left(\dot{\mathcal{G}} + \mathcal{G}\;\text{div}\;\mathbf{V}_s\right)_{m}\det\mathbf{F}\; dV$$

Avaliando esta derivada na configuração inicial $\tau = 0$ e utilizando (4.5), obtém-se,

$$\dot{\psi} = \int_{\mathcal{B}} \dot{\mathcal{G}} + \mathcal{G}\;\text{Div}\;\mathbf{V}\; dV = \int_{\mathcal{B}} \mathcal{G}' + \mathbf{V}\cdot\nabla\mathcal{G} + \mathcal{G}\;\text{Div}\;\mathbf{V}\; dV \tag{4.26}$$

onde $\mathcal{G}(\mathbf{x}) = \dfrac{\partial}{\partial\tau}\mathcal{G}^\tau(\mathbf{x}^\tau)\Big|_{\tau=0}$ e $\det\mathbf{F} = \det(\mathbf{I}+\tau\nabla\mathbf{V}(\mathbf{x}))_{\tau=0} = 1$. Como este funcional independe da solução da equação de estado, não são necessárias informações adicionais para o cálculo da sua sensibilidade.

Considere agora um funcional ψ^τ com a seguinte forma genérica,

$$\psi^\tau = \int_{\mathcal{B}_\tau} \mathcal{G}(u^\tau, \nabla u^\tau)\ dV \tag{4.27}$$

onde u^τ é a solução do problema variacional (4.13). Logo, empregando um procedimento análogo ao anterior, a derivada material no tempo é dada por,

$$\begin{aligned}\dot{\psi}^\tau &= \frac{d}{d\tau}\int_{\mathcal{B}_\tau}\mathcal{G}(u^\tau,\nabla u^\tau)\ dV = \frac{d}{d\tau}\int_{\mathcal{B}}\mathcal{G}(u_\tau,\nabla u_\tau)\ (\det\mathbf{F})\ dV\\ &= \int_{\mathcal{B}}[\mathcal{G}(u_\tau,\nabla u_\tau)\ (\det\mathbf{F})]^{\cdot}\ dV\\ &= \int_{\mathcal{B}}\left(\dot{\mathcal{G}}+\mathcal{G}\mathrm{div}\ \mathbf{V}_s\right)_m(\det\mathbf{F})\ dV = \int_{\mathcal{B}}\left(\mathcal{G}'+\mathbf{V}_s\cdot\mathrm{grad}\ \mathcal{G}+\mathcal{G}\ \mathrm{div}\mathbf{V}_s\right)_m(\det\mathbf{F})\ dV\end{aligned}$$

Para $\tau = 0$,

$$\dot{\psi} \equiv \frac{d\psi^\tau}{d\tau}\Big|_{\tau=0} = \int_{\mathcal{B}}\mathcal{G}' + \mathcal{G}\ \mathrm{Div}\mathbf{V} + \mathbf{V}\cdot\nabla\mathcal{G}\ \ dV$$

A partir de (4.26) e da regra da cadeia determina-se que $\mathcal{G}' = \mathcal{G}_{,u}\cdot u' + \mathcal{G}_{,\nabla u}\cdot\nabla u'$. Portanto,

$$\dot{\psi} = \int_{\mathcal{B}}[\mathcal{G}_{,u}\cdot u' + \mathcal{G}_{,\nabla u}\cdot\nabla u' + \mathcal{G}\ \mathrm{Div}\mathbf{V}]\ dV + \int_{\mathcal{B}}\mathbf{V}\cdot\nabla\mathcal{G}\ dV$$

Substituindo (4.19) na equação acima,

$$\begin{aligned}\dot{\psi} &= \int_{\mathcal{B}}[\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u} + -\mathcal{G}_{,u}\cdot\nabla u\mathbf{V} - \mathcal{G}_{,\nabla u}\cdot\nabla(\nabla u\mathbf{V})]\ dV\\ &\quad + \int_{\mathcal{B}}[\mathcal{G}\ \mathrm{Div}\mathbf{V} + \mathbf{V}\cdot\nabla\mathcal{G}]\ dV\end{aligned} \tag{4.28}$$

Entretanto, observa-se que,

$$\mathbf{V}\cdot\nabla\mathcal{G} = \mathbf{V}\cdot\left[\nabla u^T\mathcal{G}_{,u} + \nabla(\nabla u)^T\mathcal{G}_{,\nabla u}\right] = \nabla u\mathbf{V}\cdot\mathcal{G}_{,u} + \nabla(\nabla u)\mathbf{V}\cdot\mathcal{G}_{,\nabla u}$$

Substituindo as relações anteriores na equação (4.28), tem-se,

$$\dot{\psi} = \int_{\mathcal{B}}[\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u}]\ \ dV + \int_{\mathcal{B}}[\mathcal{G}\ \mathrm{Div}\mathbf{V} - \mathcal{G}_{,\nabla u}\cdot\nabla u\nabla\mathbf{V}]\ dV \tag{4.29}$$

Como através de (4.24) tem-se $\mathcal{G}\ \mathrm{Div}\mathbf{V} + \mathbf{V}\cdot\nabla\mathcal{G} = \mathrm{Div}(\mathcal{G}\mathbf{V})$ e pelo teorema da divergência $\int_{\mathcal{B}}\mathrm{Div}(\mathcal{G}\mathbf{V})\ dV = \int_{\partial\mathcal{B}}\mathcal{G}\mathbf{V}\cdot\mathbf{n}\ dA$, a partir da equação (4.28), obtem-se,

$$\begin{aligned}\dot{\psi} &= \int_{\mathcal{B}}[\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u}]\ dV - \int[\mathcal{G}_{,u}\cdot(\nabla u\mathbf{V}) + \mathcal{G}_{,\nabla u}\cdot\nabla(\nabla u\mathbf{V})]\ dV\\ &\quad + \int_{\partial\mathcal{B}}\mathcal{G}\mathbf{V}\cdot\mathbf{n}\ dA\end{aligned} \tag{4.30}$$

sendo $\mathbf{n}$ o vetor normal unitário a $\partial\mathcal{B}$.

As expressões (4.29) e (4.30) são equivalentes do ponto de vista analítico. Na prática, porém, deve-se analisar se o método numérico utilizado para resolver a equação de estado obtém melhores resultados no contorno ou no domínio. Através do método de elementos finitos, por exemplo, são obtidos resultados mais precisos no interior do domínio em comparação com os resultados do contorno. Nesse caso opta-se por utilizar a expressão (4.29) [34].

Em resumo, a sensibilidade do funcional ψ é descrita pelo seguinte enunciado,

$$\dot{\psi} = \psi_{,u}\cdot\dot{u} + \psi_{,\nabla u}\cdot\nabla\dot{u} + \psi'_V \tag{4.31}$$

onde ψ'_V envolve os termos das expressões (4.29) e (4.30) em função do campo de velocidades.

Novamente o enunciado da sensibilidade do funcional envolve os termos $\dot{u}$ e $\nabla\dot{u}$, os quais não são conhecidos analiticamente. O estudo da sensibilidade da equação de estado do problema fornece meios para empregar o enunciado (4.31) na determinação da sensibilidade deste tipo de funcional.

4.4 Sensibilidade da Equação de Estado

A derivada material em relação ao tempo da forma bilinear (4.14) é expressa como,

$$\begin{aligned}\dot{a}^{\tau}\left(u^{\tau},v^{\tau}\right) &= \frac{d}{d\tau}\int_{\mathcal{B}_{\tau}}\Phi\left(\text{grad } u^{\tau},\text{grad } v^{\tau}\right)\,dV = \frac{d}{d\tau}\int_{\mathcal{B}}\Phi\left(\nabla u_{\tau}\mathbf{F}^{-1},\nabla v_{\tau}\mathbf{F}^{-1}\right)\det\mathbf{F}\,dV\\ &= \int_{\mathcal{B}}\left[\Phi\left(\nabla u_{\tau}\mathbf{F}^{-1},\nabla v_{\tau}\mathbf{F}^{-1}\right)\det\mathbf{F}\right]^{\cdot}\,dV\end{aligned}$$

onde empregaram-se as expressões (4.20) e (4.23), passando a integração para o domínio original $\mathcal{B}$.

A partir daí, usando (4.5) vem que,

$$\dot{a}^{\tau}\left(u^{\tau},v^{\tau}\right) = \int_{\mathcal{B}}\left(\dot{\Phi}+\Phi\ \text{div}\mathbf{V}_s\right)_m\det\mathbf{F}\,dV = \int_{\mathcal{B}}\left(\Phi'+\mathbf{V}_s\cdot\text{grad }\Phi+\Phi\ \text{div}\mathbf{V}_s\right)_m(\det\mathbf{F})\,dV$$

Para $\tau=0$,

$$\left[a\left(u,v\right)\right]^{\cdot}\equiv\frac{d}{d\tau}a^{\tau}\left(u^{\tau},v^{\tau}\right)\Big|_{\tau=0} = \int_{\mathcal{B}}\Phi'+\mathbf{V}\cdot\nabla\Phi+\Phi\ \text{Div}\mathbf{V}\,dV \tag{4.32}$$

Na equação (4.32) tem-se,

$$\Phi\ \text{Div}\mathbf{V} = \mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla v^{S}\ \text{Div}\mathbf{V}$$

$$\begin{aligned}\mathbf{V}\cdot\nabla\Phi &= \left\{\mathsf{C}\left[\nabla u^{S}\right]\cdot\left(\nabla\nabla v\right)^{S}\right\}\cdot\mathbf{V}+\left\{\mathsf{C}\left[\left(\nabla\nabla u\right)^{S}\right]\cdot\nabla v^{S}\right\}\cdot\mathbf{V}\\ &= \mathsf{C}\left[\nabla u^{S}\right]\cdot\left(\nabla\nabla v\mathbf{V}\right)^{S}+\mathsf{C}\left[\left(\nabla\nabla u\mathbf{V}\right)^{S}\right]\cdot\nabla v^{S}\end{aligned}$$

$$\begin{aligned}\Phi' &= \mathsf{C}\left[\nabla u'^{S}\right]\cdot\nabla v^{S}+\mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla v'^{S}\\ &= \mathsf{C}\left[\nabla\dot{u}^{S}\right]\cdot\nabla v^{S}-\mathsf{C}\left[\nabla\left(\nabla u\mathbf{V}\right)^{S}\right]\cdot\nabla v^{S}-\mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla\left(\nabla v\mathbf{V}\right)^{S}\\ &= \mathsf{C}\left[\nabla\dot{u}^{S}\right]\cdot\nabla v^{S}-\mathsf{C}\left[\left(\nabla\nabla u\mathbf{V}+\nabla u\nabla\mathbf{V}\right)^{S}\right]\cdot\nabla v^{S}-\mathsf{C}\left[\nabla u^{S}\right]\cdot\left(\nabla\nabla v\mathbf{V}+\nabla v\nabla\mathbf{V}\right)^{S}\end{aligned}$$

observando que $\dot{v}=v'+\left(\nabla v\right)\mathbf{V}=0$, pois o espaço $\mathcal{V}$ não depende de τ.

Dessa forma, a partir de (4.32) tem-se,

$$\begin{aligned}\left[a\left(u,v\right)\right]^{\cdot} &= \int_{\mathcal{B}}\mathsf{C}\left[\nabla\dot{u}^{S}\right]\cdot\nabla v^{S}\,dV+\\ &\quad+\int_{\mathcal{B}}[\mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla v^{S}\ \text{Div}\mathbf{V}\\ &\quad-\mathsf{C}\left[\left(\nabla u\nabla\mathbf{V}\right)^{S}\right]\cdot\nabla v^{S}-\mathsf{C}\left[\nabla u^{S}\right]\cdot\left(\nabla v\nabla\mathbf{V}\right)^{S}]\,dV\end{aligned} \tag{4.33}$$

Assim como na seção anterior, a equação (4.32) também pode ser reescrita convertendo a integral do último termo para uma integral de contorno. Logo,

$$\begin{aligned}\left[a\left(u,v\right)\right]^{\cdot} &= \int_{\mathcal{B}}\mathsf{C}\left[\nabla\dot{u}^{S}\right]\cdot\nabla v^{S}\,dV\\ &\quad-\int_{\mathcal{B}}\left\{\mathsf{C}\left[\nabla\left(\nabla u\mathbf{V}\right)^{S}\right]\cdot\nabla v^{S}+\mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla\left(\nabla v\mathbf{V}\right)^{S}\right\}\,dV\\ &\quad+\int_{\partial\mathcal{B}}\left(\mathsf{C}\left[\nabla u^{S}\right]\cdot\nabla v^{S}\right)\mathbf{V}\cdot\mathbf{n}\,dV\end{aligned} \tag{4.34}$$

Em qualquer uma das formas, entretanto, tem-se que,

$$[a(u,v)]^{\cdot} = a(\dot{u},v) + a'_V(u,v) \tag{4.35}$$

Para a forma linear (4.15), obtem-se pelo mesmo procedimento,

$$\dot{l}(v) \equiv \frac{d}{d\tau} l^{\tau}\Big|_{\tau=0} = \int_{\mathcal{B}} \left(\dot{\mathbf{b}} + \mathbf{b}\,\mathrm{Div}\mathbf{V}\right)\cdot v\,dV + \int_{\Gamma^2} \left(\dot{\mathbf{\Phi}} + \mathbf{\Phi}\;(\mathbf{V}\cdot\mathbf{n})\,\mathrm{Div}\,\mathbf{n}\right)\cdot v\,dA \tag{4.36}$$

Deve-se observar que o parâmetro τ não tem significado físico, tendo sido introduzido somente para descrever matematicamente a variação do domínio geométrico do problema em torno da posição original. Assim, pode-se afirmar que parâmetros físicos do problema independem diretamente de τ, dependendo somente da variação espacial. Logo,

$$\mathbf{b}' = \mathbf{0} \qquad \dot{\mathbf{b}} = \mathbf{b}' + (\nabla\mathbf{b})\,\mathbf{V} = (\nabla\mathbf{b})\,\mathbf{V}$$

Além disso, observa-se que o termo $\dot{\mathbf{\Phi}}$ está relacionado apenas a $\mathbf{V}$: se $\mathbf{\Phi}$ é independente da superfície, $\dot{\mathbf{\Phi}} = \mathbf{0}$; se $\mathbf{\Phi}$ é dependente da orientação da superfície[1], $\dot{\mathbf{\Phi}} = \dot{\mathbf{\Phi}}(\mathbf{V})$ pois as coordenadas dos pontos da superfície são lineares em τ. Logo, $\dot{l}(v)$ envolve apenas termos realcionados a $\mathbf{V}$, ou seja,

$$\dot{l}(v) = l'_V(v)$$

Assumindo que o problema variacional (4.13) é satisfeito para $\forall\tau$, pode-se determinar $\dot{u}$ através da relação,

$$a(\dot{u},v) = l'_V(v) - a'_V(u,v) \tag{4.37}$$

4.5 Método Adjunto

De modo semelhante ao cálculo da sensibilidade à variação de parâmetros distribuídos, a formulação de um problema adjunto permite eliminar a dependência direta das equações (4.29) e (4.30) em relação à $\dot{u}$ e $\nabla\dot{u}$. Seguindo o procedimento apresentado no Capítulo 3, uma equação adjunta é introduzida substituindo-se $\dot{u}$ em (4.29) e (4.30) por um deslocamento virtual $v \in \mathcal{V}$. Igualando-se a soma dos termos envolvendo v à forma bilinear (4.14), dá-se origem ao seguinte problema variacional *adjunto* em $\tau = 0$

Determinar ς *tal que*

$$a(\varsigma,v) = \int_{\mathcal{B}} \mathcal{G}_{,u}\cdot v + \mathcal{G}_{,\nabla u}\cdot\nabla v\,dV \qquad \forall v \in \mathcal{V} \tag{4.38}$$

Este problema é idêntico ao problema adjunto (3.17), sendo ς o estado adjunto. Esse fato é bastante vantajoso tanto do ponto de vista da implementação computacional, quanto para o cálculo simultâneo da sensibilidade à parâmetros e à forma.

Considerando $\dot{u}$ como deslocamento virtual em (4.38) obtem-se,

$$a(\varsigma,\dot{u}) = \int_{\mathcal{B}} \left[\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u}\right]\,dV$$

Avaliando a identidade (4.37) para $v = \varsigma$,

$$a(\dot{u},\varsigma) = l'_V(\varsigma) - a'_V(u,\varsigma)$$

e, a partir da simetria da forma bilinear,

$$\int_{\mathcal{B}} \left[\mathcal{G}_{,u}\cdot\dot{u} + \mathcal{G}_{,\nabla u}\cdot\nabla\dot{u}\right]\,dV = l'_V(\varsigma) - a'_V(u,\varsigma)$$

Logo, a sensibilidade do funcional ψ, dada em (4.29), pode ser escrita como,

$$\dot{\psi} = l'_V(\varsigma) - a'_V(u,\varsigma) + \int_{\mathcal{B}} \mathcal{G}\,\mathrm{Div}\mathbf{V} - \mathcal{G}_{,\nabla u}\cdot\nabla u\nabla\mathbf{V}\,dV$$

[1]Este é o exemplo de carregamentos relacionados à pressão, que são sempre normais à superfície, ou carregamentos relacionados a fenômenos de atrito, que são sempre tangentes a esta.

Substituindo as expressões (4.14) e (4.15), obtém-se,

$$\begin{aligned}\dot{\psi} &= \int_{\mathcal{B}} \left[(\nabla \mathbf{b})\,\mathbf{V}\cdot\varsigma + \mathbf{b}\cdot\varsigma\ \mathrm{Div}\mathbf{V}\right]\,dV + \int_{\Gamma^2}\left[\Phi' + (\nabla\Phi)\,\mathbf{V}\cdot\varsigma + \Phi\cdot\varsigma\ \ (\mathbf{V}\cdot\mathbf{n})\,\mathrm{Div}\ \mathbf{n}\right]\,dA \\ &+ \int_{\mathcal{B}} -\left\{\mathsf{C}\left[\nabla u^S\right]\cdot\nabla\varsigma^S\ \mathrm{Div}\mathbf{V} - \mathsf{C}\left[(\nabla u\nabla\mathbf{V})^S\right]\cdot\nabla\varsigma^S - \mathsf{C}\left[\nabla u^S\right]\cdot(\nabla\varsigma\nabla\mathbf{V})^S\right\}dV \\ &+ \int_{\mathcal{B}}\left[\mathcal{G}\ \mathrm{Div}\mathbf{V} - \mathcal{G}_{,\nabla u}\cdot\nabla u\nabla\mathbf{V}\right]\,dV \end{aligned} \tag{4.39}$$

Uma vez resolvido o problema adjunto (4.38), a expressão (4.39) permite obter a sensibilidade do funcional ψ para uma variação do domínio descrita pelo campo de velocidades $\mathbf{V}$. Observa-se que o problema variacional (4.38) tem a mesma forma bilinear de (4.13) em torno da configuração de referência ($\tau = 0$).

Suponha que os pontos do contorno $\partial\mathcal{B}$ são especificados por um vetor posição $\mathbf{x}(\mathbf{p}) \in \Re^3$, onde $\mathbf{p} \in \Re^n$ é o vetor com n variáveis de parametrização. Uma vez que o vetor $\mathbf{p} \in \Re^n$ tenha sido definido, as expressões de análise de sensibilidade podem ser expressas em termos de uma variação $\delta\mathbf{p}$. Para isto considera-se,

$$\mathbf{p}^\tau = \mathbf{p} + \tau\delta\mathbf{p} \tag{4.40}$$

onde $\mathbf{p}^\tau$ define o contorno $\partial\mathcal{B}_\tau$. O campo de velocidades no contorno é então definido como,

$$\mathbf{V}(\mathbf{x}) = \frac{d}{d\tau}\left[\mathbf{x}^\tau(\mathbf{p}^\tau)\right]\Big|_{\tau=0} = \nabla_{\mathbf{p}}\mathbf{x}(\mathbf{p})\,\frac{d\mathbf{p}^\tau}{d\tau}\Big|_{\tau=0} = \nabla_{\mathbf{p}}\mathbf{x}(\mathbf{p})\,\delta\mathbf{p} \tag{4.41}$$

Cada coluna de $\nabla_{\mathbf{p}}\mathbf{x}(\mathbf{p})$ expressa a contribuição de uma variável de projeto no campo de velocidades. Dessa forma,

$$\mathbf{V}(\mathbf{x})_{p_k} = \frac{\partial\mathbf{x}(\mathbf{p})}{\partial p_k}\delta p_k = \delta p_k \begin{Bmatrix} \dfrac{\partial x_1}{\partial p_k} \\ \dfrac{\partial x_2}{\partial p_k} \\ \dfrac{\partial x_3}{\partial p_k} \end{Bmatrix} \qquad k = 1,\ldots,n$$

para o caso geral de problemas tridimensionais.

Considerando que o campo de velocidades tenha sido definido no interior do domínio $\mathcal{B}$ respeitando as hipóteses de continuidade exigidas, a sensibilidade dos funcionais dependentes da solução da equação de estado pode ser expressa em função das variáveis de projeto, ou seja,

$$\dot{\psi} = \nabla_{\mathbf{p}}\psi\cdot\delta\mathbf{p}$$

A componente k do gradiente do funcional ψ em relação às variáveis de projeto é obtida a partir da seguinte expressão,

$$\begin{aligned}(\nabla_{\mathbf{p}}\psi)_k &= \int_{\mathcal{B}}\left\{(\nabla\mathbf{b})\,\frac{\partial\mathbf{x}}{\partial p_k}\cdot\varsigma + \mathbf{b}\cdot\varsigma\ \mathrm{Div}\frac{\partial\mathbf{x}}{\partial p_k}\right\}\,dV \\ &+ \int_{\Gamma^2}\left\{\frac{\partial\Phi}{\partial p_k}\cdot\varsigma + \frac{\partial\mathbf{x}}{\partial p_k}(\nabla\Phi)\cdot\varsigma + \Phi\cdot\varsigma\ \left(\frac{\partial\mathbf{x}}{\partial p_k}\cdot\mathbf{n}\right)\mathrm{Div}\ \mathbf{n}\right\}\,dA \\ &- \int_{\mathcal{B}}\left\{\mathsf{C}\left[\nabla u^S\right]\cdot\nabla\varsigma^S\ \mathrm{Div}\frac{\partial\mathbf{x}}{\partial p_k}\right\}dV \\ &+ \int_{\mathcal{B}}\left\{\mathsf{C}\left[\left(\nabla u\nabla\frac{\partial\mathbf{x}}{\partial p_k}\right)^S\right]\cdot\nabla\varsigma^S + \mathsf{C}\left[\nabla u^S\right]\cdot\left(\nabla\varsigma\nabla\frac{\partial\mathbf{x}}{\partial p_k}\right)^S\right\}dV \\ &+ \int_{\mathcal{B}}\left\{\mathcal{G}\ \mathrm{Div}\frac{\partial\mathbf{x}}{\partial p_k} - \mathcal{G}_{,\nabla u}\cdot\nabla u\nabla\frac{\partial\mathbf{x}}{\partial p_k}\right\}\,dV \end{aligned} \tag{4.42}$$

O cálculo do gradiente do funcional em relação às variáveis de projeto segue então os seguintes passos:

1. Constrói-se o termo independente de (4.38).

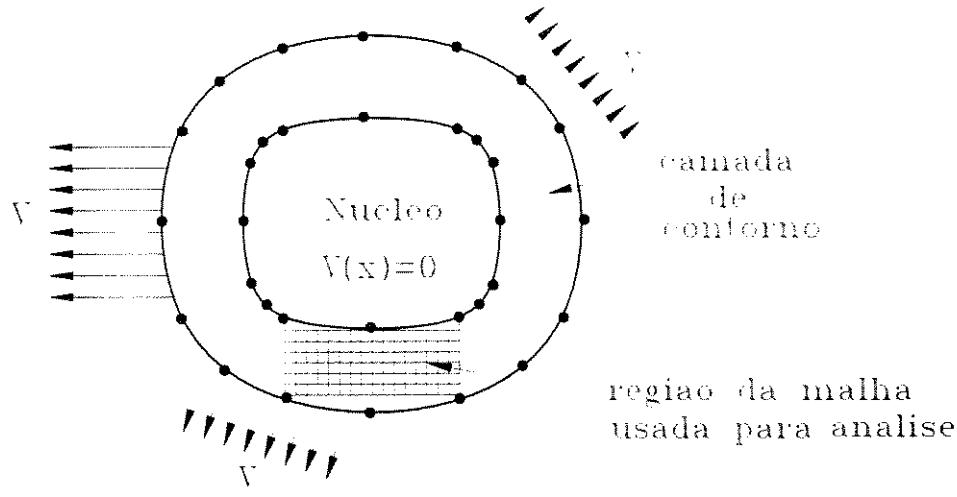


Figura 4.3: Camada de contorno definida por malha auxiliar isoparamétrica.

2. Soluciona-se o problema adjunto do funcional, obtendo-se a variável adjunta ς.

3. Aplica-se a expressão (4.42) para $i = 1, \ldots, n$.

4.6 Campo de Velocidades

Numericamente, o campo de velocidades $\mathbf{V}$ é definido de maneira unívoca no contorno pela parametrização adotada para $\partial\mathcal{B}$. Contudo, não existe uma regra geral para a definição do campo de velocidades no interior do domínio, podendo-se empregar diversas técnicas [33, 36, 35, 77, 78]. Qualquer que seja o método utilizado, entretanto, as seguintes exigências matemáticas devem ser respeitadas [78]:

- os campos de velocidades devem ter a mesma regularidade do campo de deslocamentos e,
- devem depender linearmente da variação dos parâmetros que controlam a geometria.

A primeira exigência surge da própria expressão de sensibilidade (4.39), sendo respeitada apenas utilizando-se as funções de forma de deslocamento para representar os campos de velocidades. A segunda se deve à formulação da análise de sensibilidade, na qual a derivada material do campo de deslocamento $\dot{u}$ depende linearmente do campo de velocidades. Como os coeficientes de sensibilidade fornecem derivadas de primeira ordem dos funcionais de performance em relação ao parâmetros de projeto e dependem linearmente do campo de velocidades, o campo de velocidades deve depender linearmente dos parâmetros de projeto.

Dentre as diversas abordagens de definição do campo de velocidades, pode ser citado o método de camada de contorno (*boundary layer approach*) apresentado por Seong e Choi [33]. Seu objetivo é apresentar um balanço entre precisão e eficiência no cálculo da sensibilidade de funcionais. Neste método, define-se um campo de velocidades não-nulo apenas em regiões do domínio adjacentes às superfícies parametrizadas do contorno. Isso evita que as integrais precisem ser realizadas em todo domínio $\mathcal{B}$.

Nesse método, dada uma região parametrizada do contorno, a porção do domínio com velocidades não nulas é constituída por uma camada de elementos de uma malha grossa isoparamétrica hexaédrica, como ilustrado na Figura 4.3. Através de esquema de interpolação entre as velocidades conhecidas no contorno e a velocidade nula assumida para a outra extremidade da

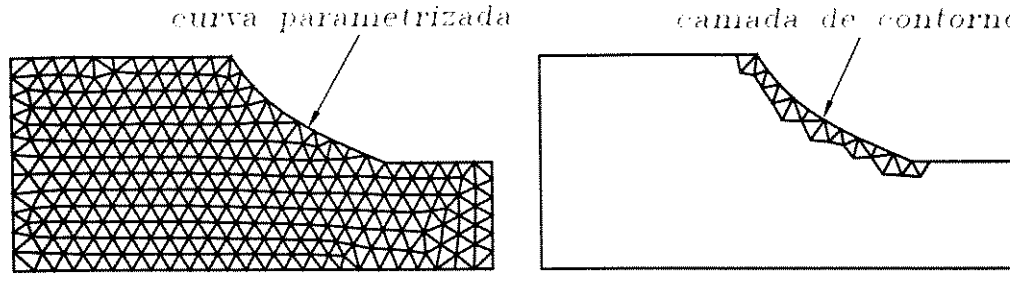


Figura 4.4: Camada de elementos no contorno utilizada na definição do campo de velocidades.

camada de elementos isoparamétricos, determina-se o campo de velocidades nos nós da malha fina utilizada para análise. A exigência de dependência linear é respeitada utilizando-se uma interpolação linear nesse caso.

Outros autores têm empregado uma variação desta técnica, na qual a mesma malha utilizada na análise é usada para definir o campo de velocidades, evitando-se a necessidade de definir a camada de contorno isoparamétrica hexaédrica. Adota-se uma camada de contorno com a espessura de um elemento, como mostrado na Figura 4.4, gerando a malha a cada alteração da geometria. Este procedimento tem a vantagem de ser ao mesmo tempo eficiente e simples de ser utilizado em um processo automatizado de otimização, pois utiliza a informação obtida da geração automática de malhas, tendo conduzido a resultados satisfatórios, ainda que não seja a abordagem mais precisa [21]. Entretanto, pode-se controlar a qualidade da malha a cada etapa de geração, estimando o erro da análise por elementos finitos ou da análise de sensibilidade [70].

Neste trabalho, emprega-se esta última abordagem na análise da sensibilidade e otimização de forma de domínios bidimensionais com contornos parametrizados por curvas *NURBS*.

4.6.1 Controle da Geometria por NURBS

Nos últimos anos, curvas e superfícies genéricas tiveram impacto significativo no projeto auxiliado por computador. Em particular, curvas e superfícies de Bezier se tornaram bastante populares devido a sua flexibilidade no controle de geometrias. Este tipo de formulação tem sido desenvolvida desde os anos sessenta, sendo as B-splines racionais não-uniformes (*NURBS - Non Uniform Rational B-Splines*) a inovação mais recente [79, 80].

As curvas *NURBS* fornecem uma forma matemática simples e precisa capaz de representar as formas analíticas comuns – linhas, planos, curvas cônicas incluindo círculos, curvas de formas livres e superfícies quadráticas ou não — usadas em gráficos de computadores e sistemas de CAD. Neste trabalho, as geometrias são descritas através de *NURBS* por apresentarem características bastante eficientes para otimização de formas estruturais como, por exemplo, a possibilidade de controlar as tangentes dos pontos extremos da curva e seu nível de oscilação. Isso permite a obtenção de formas suaves e simples de serem construídas. Além disso, como tais curvas ficam contidas em um polígono de controle claramente definido, pode-se evitar a obtenção de geometrias degeneradas durante o processo automatizado de otimização.

As curvas *NURBS* são definidas por um polígono cujos vértices correspondem a um conjunto de pontos de controle e os respectivos pesos de cada vértice, como ilustrado na Figura 4.5. As coordenadas de um ponto de uma curva desse tipo definida por q pontos de controle de

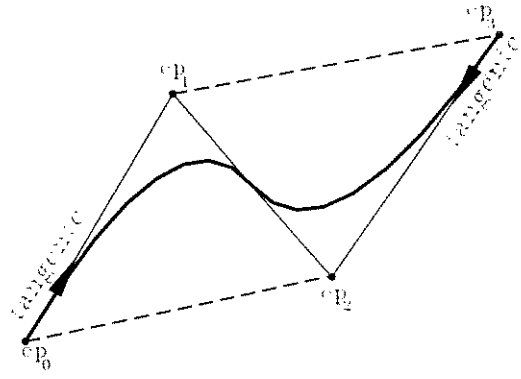


Figura 4.5: Curva NURBS.

coordenadas $\mathbf{X}_j = \{X_{j1}, X_{j2}, X_{j3}\}$ e pesos β_j são dadas por

$$\mathbf{x}(r) = \sum_{j=1}^{q} \mathbf{X}_j \hat{N}_{j\chi}(r) \tag{4.43}$$

com

$$\hat{N}_{j\chi}(r) = \frac{\beta_j N_{j\chi}(r)}{\sum_{i=1}^{q} \beta_i N_{i\chi}(r)} \tag{4.44}$$

O conjunto $\left\{\hat{N}_{j\chi}(r)\right\}$ é a base de *NURBS* associada a uma base $\{N_{j\chi}(r)\}$ de B-splines. Por sua vez, os valores β_j são pesos associados aos pontos de controle (vértices do polígono que define a curva) como uma quarta coordenada em adição às demais coordenadas físicas. A equação (4.43) define uma função racional, na qual o numerador e o denominador são polinômios de ordem χ, isto é, polinômios de grau $\chi - 1$. A quantidade adimensional r é o parâmetro interno da curva.

A base B-spline $\{N_{j\chi}(r)\}$ pode ser calculada pela fórmula recursiva de Cox-deBoor [80],

$$N_{j1}(r) = 1, \qquad \kappa_j \leq r \leq \kappa_{j+1}$$
$$N_{j1}(r) = 0, \qquad \text{nos demais casos}$$

e

$$N_{jk}(r) = \frac{(r - \kappa_j) N_{jk-1}(r)}{\kappa_{j+k-1} - \kappa_j} + \frac{(\kappa_{j+k} - r) N_{j+1k-1}(r)}{\kappa_{j+k} - \kappa_{j+1}}, \qquad k = 2, \ldots, \chi$$

onde os termos κ_j são os elementos do vetor de nós $\boldsymbol{\kappa}$ e tem dimensão $q + \chi$.

As características das curvas *NURB* que as tornam interessantes para aplicações de otimização estrutural são:

1. a curva fica contida dentro do maior polígono convexo formado pelos pontos de controle;
2. as tangentes dos pontos inicial e final da curva são definidas, respectivamente, pelo primeiro e pelo último lado do polígono;
3. a função (4.43) possui derivadas contínuas até a ordem $\chi - 2$ em toda extensão da curva.

Além disso, como as funções B-splines $\{N_{j\chi}(r)\}$ têm a seguinte propriedade,

$$\sum_{i=1}^{q} N_{i\chi}(r) = 1$$

as *NURBS* são idênticas a B-splines não-racionais no caso dos pesos serem todos unitários.

As *NURBS* podem ser controladas variando-se o número e a posição dos pontos de controle $\mathbf{X}_j$, o valor dos pesos β_j, alterando a ordem χ e alterando o vetor de nós $\boldsymbol{\kappa}$. São de interesse para a otimização de formas estruturais a variação das coordenadas dos pontos de controle da curva e dos pesos.

Em resumo, o contorno $\partial\mathcal{B}$ é parametrizado utilizando *NURBS*, sendo as coordenadas dos pontos de controle da curva e seus respectivos pesos, as variáveis de projeto $\mathbf{p}$ que controlam a geometria do domínio. São utilizadas funções B-splines $N_{i\chi}(r)$ de ordem 3 neste trabalho.

4.6.2 Definição do Campo de Velocidades

Considere o campo de velocidades no contorno (4.41) e a definição de uma curva *NURBS* dada em (4.43). Quando a variável p_k é uma coordenada l ($l = 1, 2, 3$) do ponto de controle j, ou seja, $p_k = X_{jl}$, observa-se que o gradiente de (4.43) em relação às coordenadas de um ponto de controle é definido pelas próprias funções de interpolação:

$$\frac{\partial \mathbf{x}}{\partial p_k} = \frac{\partial \mathbf{x}}{\partial X_{jl}} = \hat{N}^l_{j\chi} \tag{4.45}$$

onde

$$\hat{N}^1_{j\chi} = \begin{Bmatrix} \hat{N}_{j\chi}(r) \\ 0 \\ 0 \end{Bmatrix} \qquad \hat{N}^2_{j\chi} = \begin{Bmatrix} 0 \\ \hat{N}_{j\chi}(r) \\ 0 \end{Bmatrix} \qquad \hat{N}^3_{j\chi} = \begin{Bmatrix} 0 \\ 0 \\ \hat{N}_{j\chi}(r) \end{Bmatrix}$$

Quando a variável p_k é um o peso do ponto de controle j, ou seja, $p_k = w_j$, tem-se que

$$\frac{\partial \mathbf{x}}{\partial p_k} = \frac{\partial \mathbf{x}}{\partial \beta_j} = \frac{1}{\beta_j}\left[\mathbf{X}_j - \mathbf{x}(r)\right]\hat{N}_{j\chi}(r) \tag{4.46}$$

O campo de velocidades nos nós do contorno pode ser determinado a partir das expressões (4.45) e (4.46). Considera-se, então, a camada de contorno formada pelos elementos adjacentes à curva parametrizada, assumindo nula a velocidade dos nós que não estão no contorno, definindo assim o campo de velocidades.

4.7 Gradiente dos Funcionais de Performance Estrutural

As expressões anteriores podem ser aplicadas a funcionais usuais de performance estrutural, como funcionais de massa, energia de deformação, tensão e deslocamento.

Funcional de Massa : o seguinte funcional define a massa do corpo $\mathcal{B}_\tau$ durante a variação de sua geometria,

$$\psi_1 = \int_{\mathcal{B}_\tau} \tilde{\rho}^\tau(\mathbf{x}^\tau)\, dV \tag{4.47}$$

De (4.26) é obtida sua sensibilidade em relação à variação do domínio,

$$\dot{\psi}_1 = \int_{\mathcal{B}} \left[\mathbf{V}\cdot\nabla\tilde{\rho} + \tilde{\rho}\,\mathrm{Div}\mathbf{V}\right] dV$$

ou seja,

$$\dot{\psi}_1 = \int_{\mathcal{B}} \mathrm{Div}\left(\tilde{\rho}\mathbf{V}\right) dV = \int_{\partial\mathcal{B}} \tilde{\rho}\mathbf{V}\cdot\mathbf{n}\, dA \tag{4.48}$$

A componente k do vetor gradiente é dada por,

$$\left(\nabla_{\mathbf{p}}\psi_1\right)_k = \int_{\mathcal{B}} \mathrm{Div}\left(\tilde{\rho}\frac{\partial \mathbf{x}}{\partial p_k}\right) dV = \int_{\partial\mathcal{B}} \left(\tilde{\rho}\frac{\partial \mathbf{x}}{\partial p_k}\right)\cdot\mathbf{n}\, dA \tag{4.49}$$

Funcionais de Tensão e Deslocamento : estes funcionais dependem da solução da equação de estado do problema e sua sensibilidade pode ser obtida pelo método adjunto. Como observado anteriormente, 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 definições de funcionais de tensão e deslocamento do Capítulo 3 e seus respectivos carregamentos adjuntos.

Funcional de Energia de Deformação : a energia de deformação do corpo $\mathcal{B}_\tau$ sujeito ao problema elastostático (4.10) é definida pela expressão,

$$\psi_4 = \frac{1}{2}\int_{\mathcal{B}_\tau} \mathsf{C}\left[(\text{grad } u^\tau)^S\right]\cdot(\text{grad } u^\tau)^S\, dV$$

A sua sensibilidade é definida como

$$\dot{\psi}_4 = \int_{\mathcal{B}} \mathsf{C}\left[\nabla\dot{u}^S\right]\cdot\nabla u^S\, dV - \int_{\mathcal{B}} \mathsf{C}\left[(\nabla u\nabla\mathbf{V})^S\right]\cdot\nabla u^S\, dV + \frac{1}{2}\int_{\mathcal{B}} \mathsf{C}\left[\nabla u^S\right]\cdot\nabla u^S\, dV$$

Substituindo (4.34) e (4.36) na expressão anterior com $v = u$, obtém-se,

$$\dot{\psi}_4 = \int_{\mathcal{B}} \{(\nabla\mathbf{b})\,\mathbf{V}\cdot u + \mathbf{b}\cdot u\ \text{Div}\mathbf{V}\}\ dV +$$
$$\int_{\Gamma^2} \{\mathbf{\Phi}' + (\nabla\mathbf{\Phi})\,\mathbf{V}\cdot u + \mathbf{\Phi}\cdot u\ \ (\mathbf{V}\cdot\mathbf{n})\,\text{Div}\ \mathbf{n}\}\ dA +$$
$$\int_{\mathcal{B}} \left\{\mathsf{C}\left[(\nabla u\nabla\mathbf{V})^S\right]\cdot\nabla u^S - \frac{1}{2}\mathsf{C}\left[\nabla u^S\right]\cdot\nabla u^S\right\}\ dV$$

Devido a forma do funcional ψ_4, não é necessário resolver o problema adjunto para determinar sua sensibilidade.

4.8 Condições de Regularidade

Durante a realização das operações de derivação, assumiu-se que os operadores e variáveis são regulares o suficiente para torná-las válidas. As considerações sobre a transformação (4.7) que permitem transportar as expressões definidas no domínio transformado $\mathcal{B}_\tau$ para o domínio de referência $\mathcal{B}$ são apresentadas em [21].

O domínio $\mathcal{B}$ deve ser C_k-regular (contorno fechado, limitado e C^k-regular por partes), com $k \geq 1$ para problemas de segunda ordem e $k \geq 2$ para problemas de quarta ordem. O campo de velocidades $\mathbf{V}(\mathbf{x})$, que deve ter a mesma regularidade do campo de deslocamentos e ser definido numa vizinhança do fecho de $\mathcal{B}$. Assim, para τ suficientemente pequeno, o mapeamento X^τ é uma transformação injetora com inversa contínua. Nesse caso, prova-se que o espaço de Sobolev considerado na solução por elementos finitos é preservado na transformação. Eventuais singularidades das equações de estado e adjunta se localizam no contorno, tendo pouca influência em expressões discretizadas no domínio.

4.9 Estudo de Casos

A seguir são apresentados resultados da otimização de forma de 3 problemas elásticos bidimensionais com elementos lineares. Para os dois primeiros casos, ao se gerar uma nova malha, efetuou-se um refinamento aplicando o estimador de erros Zhu-Zienkievicz[71]. os resultados em termos de tensão nodal consideram uma técnica de ajuste por mínimos quadrados.

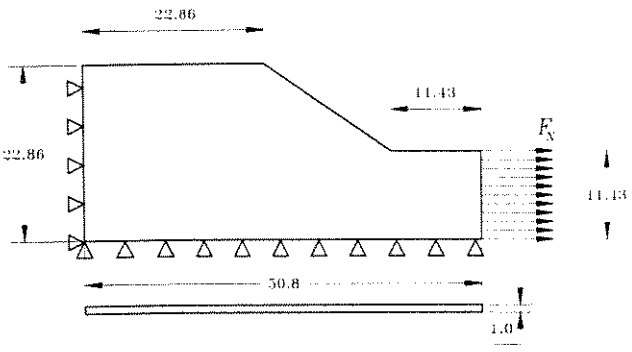


Figura 4.6: Dimensões [cm] e condições de contorno da chapa com mudança de seção transversal.

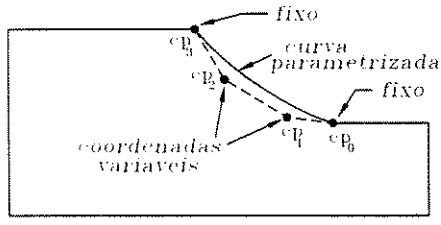


Figura 4.7: Modelo parametrizado da chapa.

4.9.1 Chapa com Mudança de Seção

Neste exemplo, tem-se uma chapa sob tração apresentando uma mudança de seção transversal, como mostrado esquematicamente na Figura 4.6. As constantes de material são $\tilde{E} = 2.1 \times 10^6$ kN/cm^2 e $\tilde{\nu} = 0.3$ sendo a tração $F_x = 175.127$ kN/cm . Observa-se que a forma dessa mudança de seção pode ocasionar altos níveis de tensão, sendo de interesse determinar uma *geometria mais eficiente* para o componente.

Um modelo geométrico para analisar este problema é mostrado na Figura 4.7 considerando a simetria na direção longitudinal, onde a região do contorno correspondente a mudança de seção transversal é definida por uma B-spline de terceira ordem com quatro pontos de controle. São consideradas como variáveis de interesse as coordenadas dos pontos de controle internos da curva, indicados por $\mathbf{cp}_1$ e $\mathbf{cp}_2$ na Figura 4.7, totalizando 4 variáveis ($\mathbf{cp}_1x$, $\mathbf{cp}_1y$, $\mathbf{cp}_2x$, $\mathbf{cp}_2y$). A tensão máxima de von Mises ocorre na região adjacente à curva parametrizada. Define-se como objetivo da análise minimizar o valor do funcional de tensão ψ_2 (1.14), modificando-se as coordenadas iniciais dos pontos de controle $\mathbf{cp}_1$ (32.63734800, 15.56171800) e $\mathbf{cp}_2$ (25.85452800, 20.49467800), tendo como restrições apenas os seguintes limites superior e inferior de cada variável,

$$\begin{aligned} 30.0000000 &\leq \mathbf{cp}_1x \leq 38.0000000 \\ 12.2000000 &\leq \mathbf{cp}_1y \leq 16.0000000 \\ 23.5000000 &\leq \mathbf{cp}_2x \leq 27.0000000 \\ 18.0000000 &\leq \mathbf{cp}_2y \leq 22.0000000 \end{aligned}$$

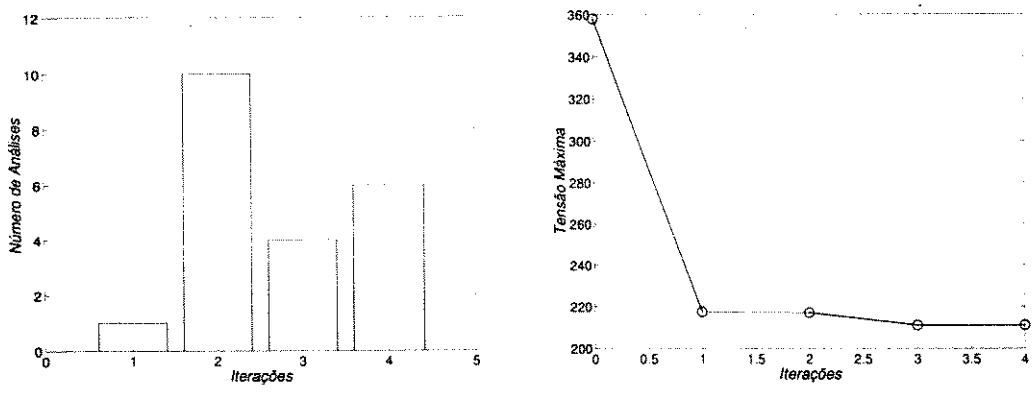
O algoritmo de Herskovits foi aplicado na solução numérica do problema anterior com 4 iterações, exigindo 22 análises de elementos finitos distribuídas nas iterações conforme mostrado na Figura 4.8(a). Obteve-se como solução,

$\mathbf{cp}_1$ (30.25878349, 13.04664418) $\mathbf{cp}_2$ (23.63328712, 19.98500496)

A evolução do valor da função objetivo na seqüência das iterações é mostrada da Figura 4.8(b). As Figuras 4.9 e 4.10 mostram, respectivamente, as malhas de elementos dos projetos inicial e final. Por sua vez, as Figuras 4.11 e 4.12 ilustram os resultados de tensões nodais de von Mises em cada caso.

4.9.2 Chapa com Furo

Considere uma chapa quadrada com um furo central sujeita às condições mostradas na Figura 4.13. As constantes de material são $\tilde{E} = 3000$ kN/cm^2 e $\tilde{\nu} = 0.3$ enquanto as trações laterais são $F_x = 15$ kN/cm e $F_y = 30$ kN/cm. A Figura 4.14 mostra a parametrização do



(a) Número de análises. (b) Tensão máxima.

Figura 4.8: Resultados da otimização da chapa.

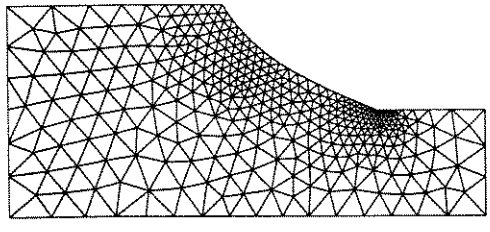


Figura 4.9: Projeto inicial.

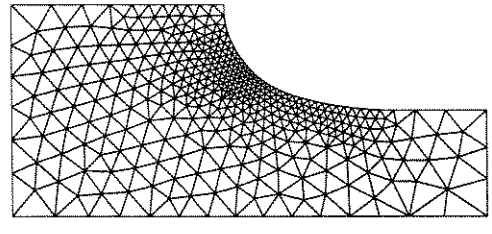


Figura 4.10: Projeto otimizado.

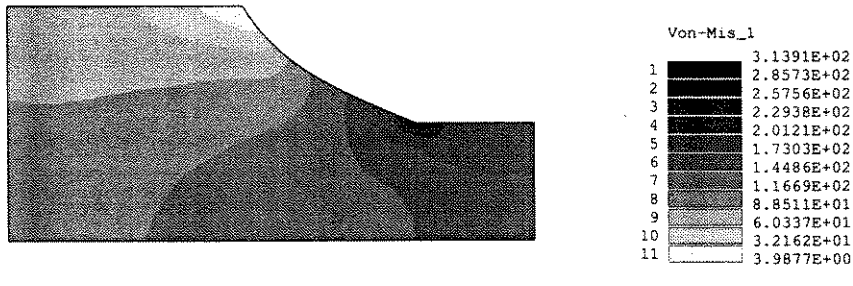


Figura 4.11: Tensões de von Mises no projeto inicial.

Figura 4.12: Tensões de von Mises no projeto otimizado.

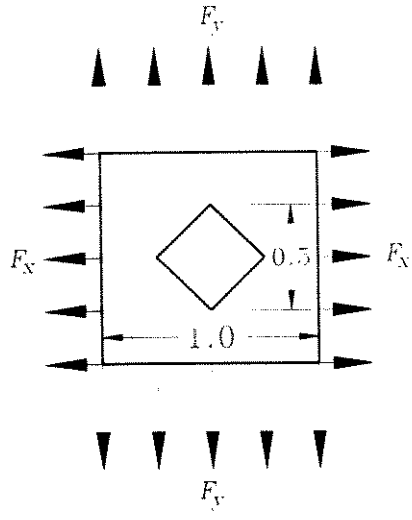


Figura 4.13: Dimensões [cm] e condições de contorno da chapa sob tração com furo quadrado.

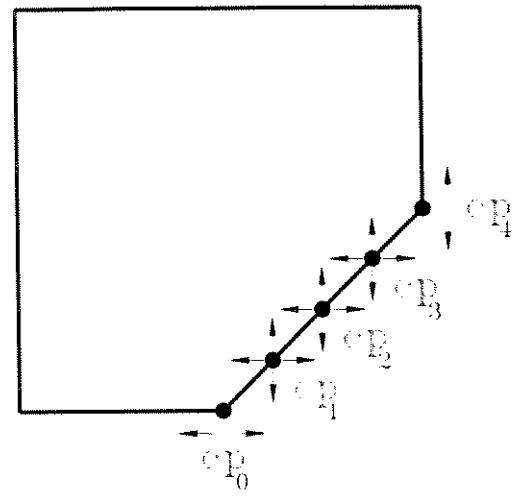


Figura 4.14: Parametrização do problema. As setas indicam as variáveis do problema.

problema considerando a simetria, que consiste na definição das lados do furo como uma B-spline de ordem 3 e cinco pontos de controle. As variáveis de projeto são a coordenada x de $\mathbf{cp}_0$, as coordenadas x e y de $\mathbf{cp}_1$, $\mathbf{cp}_2$ e $\mathbf{cp}_3$ e a coordenada y de $\mathbf{cp}_4$ totalizando 8 variáveis. Neste problema o objetivo é determinar a geometria que minimiza a energia de deformação ψ_4 (1.16), estando-se sujeito ao seguintes limites superior e inferior em cada variável,

$$0.24900 \leq \mathbf{cp}_0 x \leq 0.40000 \qquad 0.10000 \leq \mathbf{cp}_2 y \leq 0.30000$$
$$0.31200 \leq \mathbf{cp}_1 x \leq 0.40000 \qquad 0.32000 \leq \mathbf{cp}_3 x \leq 0.43760$$
$$0.06000 \leq \mathbf{cp}_1 y \leq 0.18000 \qquad 0.18000 \leq \mathbf{cp}_3 y \leq 0.40000$$
$$0.31200 \leq \mathbf{cp}_2 x \leq 0.41000 \qquad 0.18000 \leq \mathbf{cp}_4 y \leq 0.40000$$

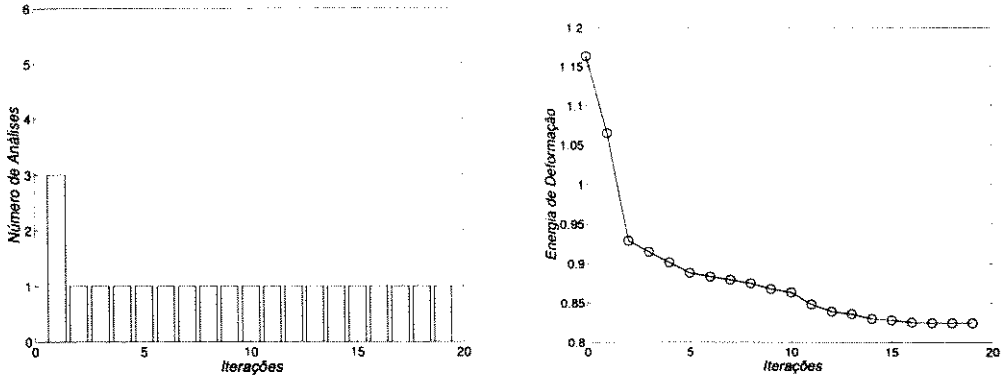
O algoritmo de Herskovits foi aplicado na solução numérica do problema em 19 iterações, exigindo 21 análises de elementos finitos, distribuídas conforme o gráfico da Figura 4.15(a). A evolução do valor da função objetivo na seqüência das iterações é mostrada da Figura 4.15(b). As Figuras 4.16(a), 4.16(b), 4.16(c), 4.16(d) ilustram, respectivamente, a geometria inicial, as formas alcançadas nas iterações 10 e 15 e a geometria final.

4.9.3 Projeto de um Componente

Dado o projeto inicial de um componente mostrado Figura 4.17, sujeito às condições de contorno da Figura 4.18, o objetivo é minimizar seu volume através da alteração da geometria, tendo como restrições o deslocamento máximo na direção y e o nível de tensão. A parametrização adotada para o problema é mostrada na Figura 4.19, onde 3 curvas do contorno foram definidas como B-splines de terceira ordem e as coordenadas de 6 pontos de controle determinam a geometria do componente. O vetor de variáveis de projeto $\mathbf{p}$ é então composto de 12 variáveis de acordo com a Tabela 4.1.

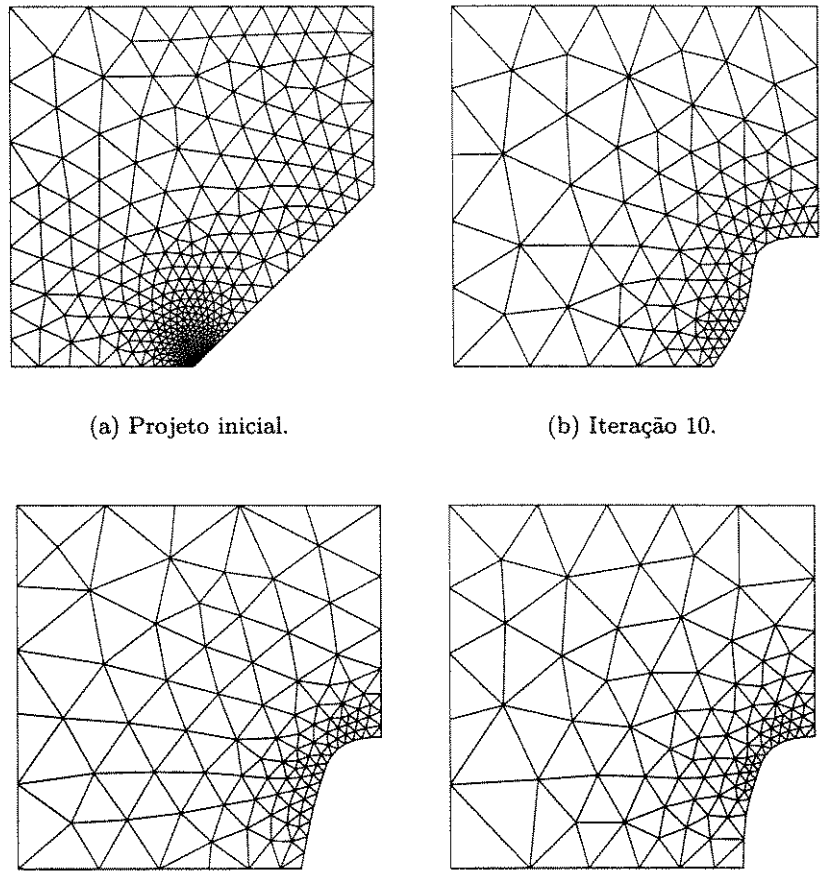
Inicialmente, verificam-se a precisão dos gradientes dos funcionais de volume, deslocamento máximo na direção y, energia de deformação e tensão de von Mises máxima para esta configuração inicial, aplicando os conceitos de avaliação numérica de gradientes da Seção 3.5.3 aos resultados das Tabelas 4.2 a 4.5.

O gradiente do funcional de volume (Tabela 4.2) apresenta 11.3311742% de erro no critério



(a) Número de análises. (b) Energia de deformação.

Figura 4.15: Resultados da otimização da chapa com furo.



(a) Projeto inicial. (b) Iteração 10.

(c) Iteração 15. (d) Projeto final.

Figura 4.16: Geometria obtidas no processo de otimização.

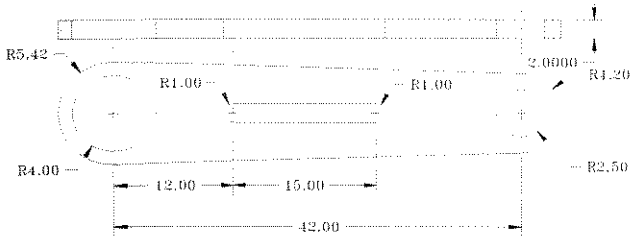


Figura 4.17: Dimensões [cm] do componente.

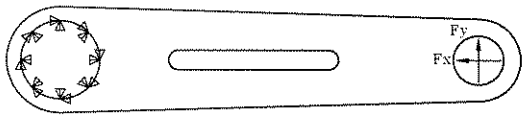


Figura 4.18: Condições de contorno ($F_x = 1000$ kN, $F_y = 2000$ kN).

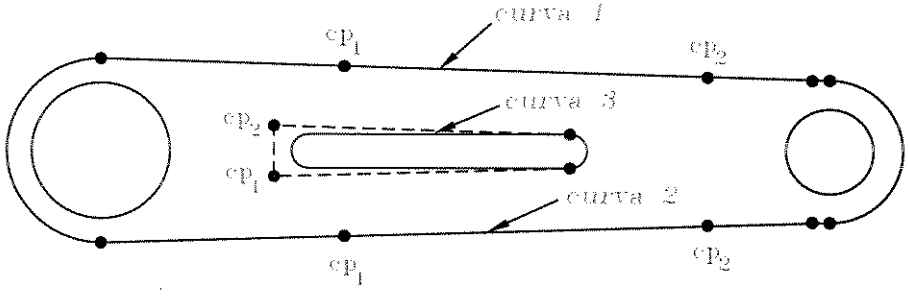


Figura 4.19: Parametrização da geometria do componente.

Tabela 4.1: Variáveis de projeto para o componente.

Variável	Significado	Variável	Significado
p_1	$\mathbf{cp}_1$ / curva 1 / coordenada x	p_7	$\mathbf{cp}_2$ / curva 2 / coordenada x
p_2	$\mathbf{cp}_1$ / curva 1 / coordenada y	p_8	$\mathbf{cp}_2$ / curva 2 / coordenada y
p_3	$\mathbf{cp}_2$ / curva 1 / coordenada x	p_9	$\mathbf{cp}_1$ / curva 3 / coordenada x
p_4	$\mathbf{cp}_2$ / curva 1 / coordenada y	p_{10}	$\mathbf{cp}_1$ / curva 3 / coordenada y
p_5	$\mathbf{cp}_1$ / curva 2 / coordenada x	p_{11}	$\mathbf{cp}_2$ / curva 3 / coordenada x
p_6	$\mathbf{cp}_1$ / curva 2 / coordenada y	p_{12}	$\mathbf{cp}_2$ / curva 3 / coordenada x

Tabela 4.2: Funcional de volume: comparação entre os gradientes determinados pelo método de análise de sensibilidade (AS) e por diferenças finitas com diferença central ajustada para manter a topologia da malha original (MDF_c).

	$\nabla_{\mathbf{p}}\psi$ (AS)	$\nabla_{\mathbf{p}}\psi$ (MDF_c)	Erro%
$\partial\psi/\partial p_1$	0.5636614	0.8966444	**37.1400809**
$\partial\psi/\partial p_2$	18.6407777	29.6532048	**37.1373926**
$\partial\psi/\partial p_3$	0.8466538	0.8239422	2.7564555
$\partial\psi/\partial p_4$	27.9995635	27.2499909	2.7507261
$\partial\psi/\partial p_5$	0.5636616	0.8966528	**37.1371394**
$\partial\psi/\partial p_6$	−18.6407777	−29.6531599	**37.1372975**
$\partial\psi/\partial p_7$	0.8466535	0.8240055	2.7485253
$\partial\psi/\partial p_8$	−27.9995635	−27.2507150	2.7479958
$\partial\psi/\partial p_9$	2.1290012	1.9751850	7.7874326
$\partial\psi/\partial p_{10}$	11.3343909	11.2724100	0.5498468
$\partial\psi/\partial p_{11}$	2.1290012	1.9751850	7.7874326
$\partial\psi/\partial p_{12}$	−11.3343909	11.2724100	0.5498468

Tabela 4.3: Funcional de deslocamento máximo na direção y: comparação entre os gradientes determinados pelo método de análise de sensibilidade (AS) e por diferenças finitas com diferença central ajustada para manter a topologia da malha original (MDF_c).

	$\nabla_{\mathbf{p}}\psi$ (AS)	$\nabla_{\mathbf{p}}\psi$ (MDF_c)	Erro%
$\partial\psi/\partial p_1$	−0.0006115	−0.0006653	8.0835073
$\partial\psi/\partial p_2$	−0.0199030	−0.0231571	14.0524368
$\partial\psi/\partial p_3$	−0.0005379	−0.0001743	**208.5847368**
$\partial\psi/\partial p_4$	−0.0171168	−0.0086182	**98.6126582**
$\partial\psi/\partial p_5$	−0.0005813	−0.0006347	8.4166302
$\partial\psi/\partial p_6$	0.0188955	0.0217293	13.0413603
$\partial\psi/\partial p_7$	−0.0005013	−0.0001450	**245.8335443**
$\partial\psi/\partial p_8$	0.0158969	0.0075450	**110.6944997**
$\partial\psi/\partial p_9$	−0.0012684	−0.0011450	10.7772926
$\partial\psi/\partial p_{10}$	−0.0017570	−0.0020800	15.5288462
$\partial\psi/\partial p_{11}$	−0.0012606	−0.0011350	11.0660793
$\partial\psi/\partial p_{12}$	0.0017059	0.0020500	16.7853659

Tabela 4.4: Funcional de energia de deformação: comparação entre os gradientes determinados pelo método de análise de sensibilidade (AS) e por diferenças finitas com diferença central ajustada para manter a topologia da malha original (MDF_c).

	$\nabla_{\mathbf{p}}\psi$ (AS)	$\nabla_{\mathbf{p}}\psi$ (MDF_c)	Erro%
$\partial\psi/\partial p_1$	−0.5615628	−0.6052861	7.2235775
$\partial\psi/\partial p_2$	−18.2738532	−20.9182333	12.6415080
$\partial\psi/\partial p_3$	−0.4788265	−0.1136560	**321.2946563**
$\partial\psi/\partial p_4$	−15.2032020	−7.0730182	**114.9464572**
$\partial\psi/\partial p_5$	−0.5075574	−0.5441597	6.7263931
$\partial\psi/\partial p_6$	16.4718109	19.0063574	13.3352562
$\partial\psi/\partial p_7$	−0.4197197	−0.0763385	**449.8136896**
$\partial\psi/\partial p_8$	13.2313084	5.6127400	**135.7370625**
$\partial\psi/\partial p_9$	−1.2945720	−1.1647400	11.1468654
$\partial\psi/\partial p_{10}$	−1.7983408	−2.1649350	16.9332659
$\partial\psi/\partial p_{11}$	−1.2780532	−1.1376200	12.3444736
$\partial\psi/\partial p_{12}$	1.6128734	1.9731350	18.2583351

Tabela 4.5: Funcional de tensão de von Mises máxima: comparação entre os gradientes determinados pelo método de análise de sensibilidade (AS) e por diferenças finitas com diferença central ajustada para manter a topologia da malha original (MDF_c).

	$\nabla_{\mathbf{p}}\psi$ (AS)	$\nabla_{\mathbf{p}}\psi$ (MDF_c)	Erro%
$\partial\psi/\partial p_1$	−78.7672884	−8.4982403	**826.8658666**
$\partial\psi/\partial p_2$	1063.4240662	−321.0086048	**431.2758756**
$\partial\psi/\partial p_3$	170.0732919	1.1349853	**14884.6247998**
$\partial\psi/\partial p_4$	826.3410465	−11.2909364	**7418.6228306**
$\partial\psi/\partial p_5$	−8.8754226	−24.9839417	64.4754910
$\partial\psi/\partial p_6$	283.7915625	305.1753106	7.0070374
$\partial\psi/\partial p_7$	−6.9233392	0.5796422	**1294.4159998**
$\partial\psi/\partial p_8$	203.7412981	47.7750250	**326.4598461**
$\partial\psi/\partial p_9$	−30.9734462	−20.7812300	49.0452981
$\partial\psi/\partial p_{10}$	−42.7572638	−78.0263950	45.2015388
$\partial\psi/\partial p_{11}$	−31.8074108	−18.5984350	71.0219747
$\partial\psi/\partial p_{12}$	86.1511334	95.6489450	9.9298655

do produto interno normalizado e 15.0678962% de erro no valor do módulo desse vetor. Isto está associado ao erro em torno de 37% das derivadas parciais em relação às variáveis p_1, p_2, p_5 e p_6 (ponto de controle $\mathbf{cp}_1$ das curvas 1 e 2). Como o erro nas demais componentes está bem abaixo deste valor e a sensibilidade do funcional de volume não envolve grandezas relacionadas à resposta estrutural do componente, pode-se concluir que esta imprecisão pode estar associada alguma inadequação do campo de velocidades para os cálculos de sensibilidade deste funcional em relação à estas variáveis no projeto inicial. Nesse sentido, considerando a malha de elementos da Figura 4.21(a), pode-se observar que a camada de contorno tem forma bastante regular (praticamente paralela ao contorno), exceto na região adjacente ao ponto de controle $\mathbf{cp}_1$ nas curvas 1 e 2.

O gradiente do funcional de deslocamento (Tabela 4.3) apresenta 6.3545994% de erro no produto interno normalizado, com 6.6261071% de imprecisão em seu módulo. O gradiente da energia de deformação (4.4) tem erros de 7.0040527% e 7.0019635% no produto interno e no módulo do vetor. Nos dois casos, entretanto, as derivadas parciais em relação às variáveis p_3, p_4, p_7 e p_8 (ponto de controle $\mathbf{cp}_2$ das curvas 1 e 2) apresentam erro elevado. Como o procedimento de cálculo dos dois gradientes é diferente (o funcional de deslocamento utiliza o método adjunto e o segundo não pois é calculado diretamente) e os erros em cada componente são muito semelhantes, esta imprecisão deve estar associada principalmente a espessura da camada de contorno na região adjacente ao ponto de controle $\mathbf{cp}_2$ das curvas 1 e 2.

Apesar disso, a precisão alcançada nos gradientes dos três funcionais anteriores é satisfatória para a aplicação em otimização estrutural, como atestam os resultados desse exemplo.

O funcional de tensão ψ_2 dado por (1.14), porém, somente fornece resultados precisos para as variáveis associadas a pontos de controle próximos ao elemento onde ocorre o valor máximo de tensão devido a forma do seu carregamento adjunto, que deforma praticamente apenas a região próxima a esse elemento, de forma que a solução do problema adjunto é dominado por erro numérico nas demais regiões do domínio. A grande imprecisão do gradiente desse funcional, para a maioria das variáveis, pode ser observada na quarta coluna da Tabela (4.5). Estes resultados mostram que se torna inviável utilizar este funcional neste problema de otimização.

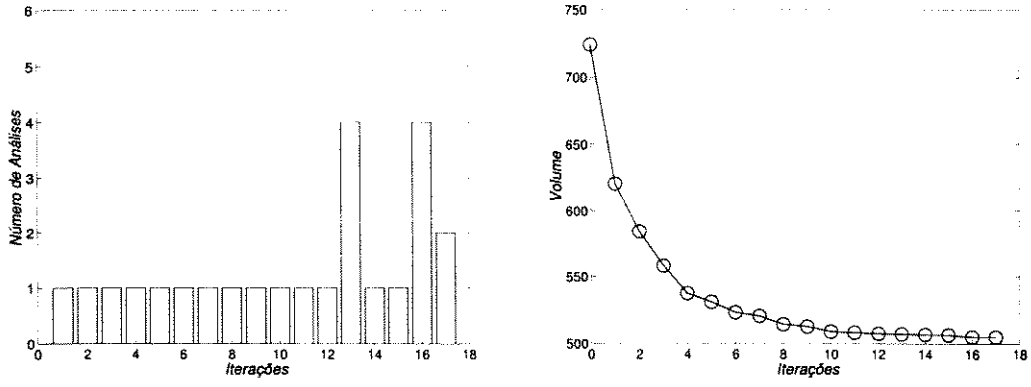
Baseando-se nos resultados de precisão dos gradientes de funcionais de performance, adotam-se os funcionais de deslocamento máximo na direção y (valor máximo admissível de 0.25 cm) e energia de deformação (valor máximo admissível de 200 kNcm) como restrições à otimização do volume do componente.

O volume foi minimizado em 17 iterações, exigindo 24 análises de elementos finitos distribuídos conforme a Figura 4.20(a), tendo a restrição de energia de deformação ativa na solução. A evolução do volume é mostrado na Figura 4.20(b). As Figuras 4.21(a), 4.21(b), 4.21(c) e 4.21(d) ilustram, respectivamente, a geometria inicial, as formas alcançadas ao final da primeira e da sétima iteração e a forma final. As tensões nodais de von Mises em cada caso são mostradas nas Figuras 4.22(a), 4.22(b), 4.22(c) e 4.22(d).

4.10 Discussão dos Resultados

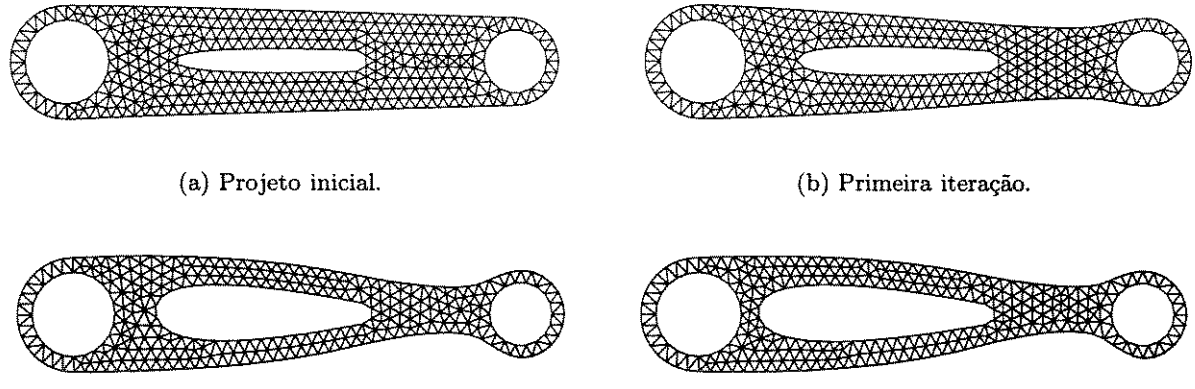
A partir dos resultados anteriores observa-se que grande parte do sucesso da análise se deve à escolha dos funcionais de performance mais adequados. O funcional de tensão ψ_2 dado por (1.14), por exemplo, somente fornece resultados satisfatórios se as variáveis utilizadas tiverem grande influência em seu comportamento.

No exemplo da Seção 4.9.1 a tensão máxima ocorre sempre na região tangente à curva parametrizada, tornando o funcional bastante sensível a qualquer variação das coordenadas dos pontos de controle. Já na Seção 4.9.3, poucas variáveis têm grande influência no valor deste



(a) Número de análises. (b) Volume.

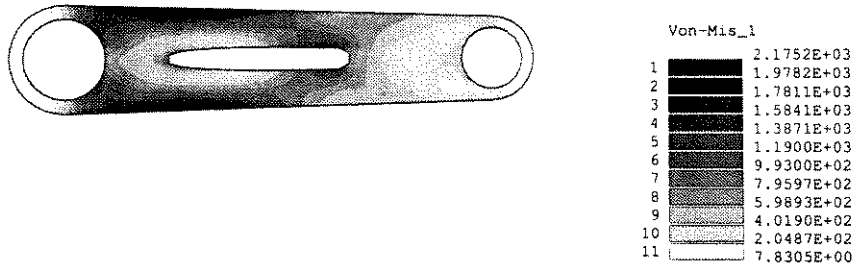
Figura 4.20: Resultados da otimização do componente.



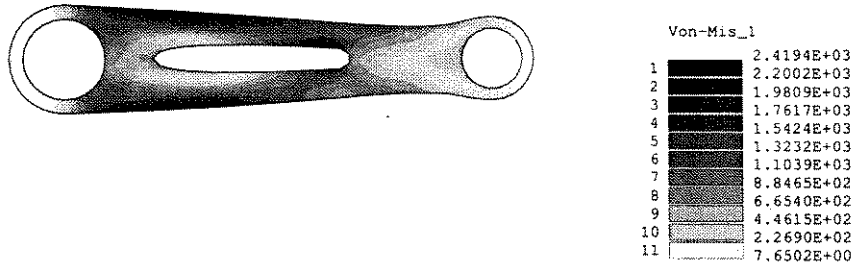
(a) Projeto inicial. (b) Primeira iteração.

(c) Sétima iteração. (d) Projeto final.

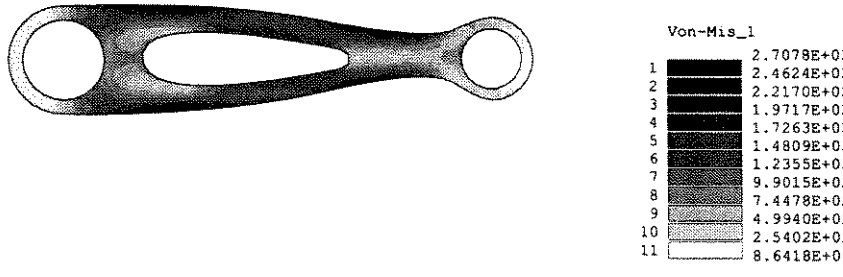
Figura 4.21: Geometria obtidas no processo de otimização.



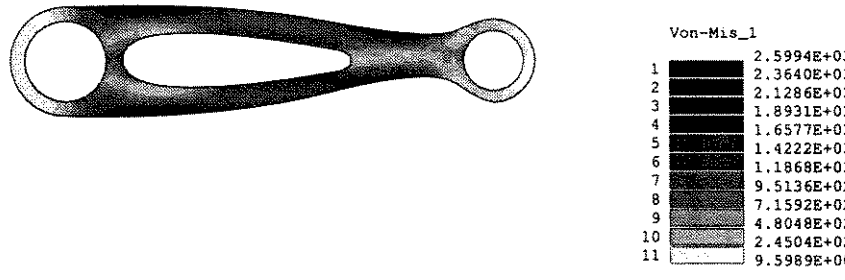
(a) Projeto inicial.



(b) Primeira iteração.



(c) Sétima iteração.



(d) Projeto final.

Figura 4.22: Tensões de von Mises em geometrias obtidas durante o processo de otimização.

funcional, tornam sua utilização inviável, como observado na Tabela (4.5).

Isto se deve tanto à possibilidade de mudança do elementos de ocorrência da tensão máxima, pois o funcional considera a sensibilidade no valor da tensão tomada no elemento de tensão máxima da tensão corrente, quanto à forma do carregamento adjunto deste funcional, que deforma apenas a região adjacente ao elemento de tensão máxima. Portanto, o estado adjunto na maior parte do domínio pode sofrer muita influência de imprecisões numéricas.

Em termos do procedimento global de otimização, observa-se que a maior parcela do decréscimo na função objetivo é atingida nas primeiras iterações, fazendo com que resultados práticos possam ser obtidos com menor custo. Isto pode ser comprovado no exemplo da Seção 4.9.3 onde, do total de 24 análises, 12 são realizadas somente nas 5 últimas iterações, que causam decréscimo bastante pequeno no volume do componente.

Capítulo 5

Implementação dos Algoritmos de Análise de Sensibilidade e Otimização

Uma das maiores preocupações no desenvolvimento de programas é adotar procedimentos que garantam produtividade e, ao mesmo tempo, confiabilidade e qualidade aos programas produzidos. Isso envolve a adoção de ferramentas aumentando a mantenabilidade do código (facilidade em testá-lo, corrigi-lo e atualizá-lo). Neste sentido, o aumento da modulação do programa e da reutilização de códigos (aproveitamento total ou parcial de códigos já existentes em novas aplicações) são algumas das soluções visando direcionar a prática da programação na direção das metas anteriores.

A programação orientada por objetos tem se tornado uma das mais importantes tecnologias para o desenvolvimento de programas. Este paradigma tem grande potencial para aumentar a produtividade e a mantenabilidade da programação, pois uma das mais importantes características das linguagens orientadas por objetos é fornecer ao programador recursos para estruturar, ordenar e modular o código.

Na programação orientada a procedimentos tradicional existe uma separação intrínsica entre estruturas de dados e procedimentos que manipulam tais dados. Em outras palavras, primeiro desenvolve-se a estrutura de dados necessária ao tipo de aplicação a ser desenvolvida. Em seguida, implementa-se um conjunto de procedimentos necessários para o tipo de problema a ser tratado. Como tais funções ficam dependentes dos tipos de dados manipulados, a possibilidade de reutilização de códigos diminui. Por sua vez, grande parte das estruturas de dados pode ser diretamente acessada e manipulada pelos procedimentos, aumentando o risco que alguma alteração indevida cause perda de informações relevantes. Além disso, qualquer necessidade de alteração da estrutura de dados requer a revisão de grande parte do programa em busca de possíveis dependências.

De fato, códigos procedurais de aplicação profissional (milhares de linhas de código) são freqüentemente instáveis sob pequenas alterações, ou seja, revisões de pequenos problemas podem originar um grande número de efeitos colaterais. Quando o código é maior que um certo limite (10^5 linhas de código), as técnicas procedurais parecem atingir seu limite. A complexidade e a implementação tornam os projetos extremamente difíceis e caros, e em muitos casos o custo de manutenção se torna maior que o custo de desenvolvimento. A programação orientada por objetos tem sido largamente aceita entre desenvolvedores de programas e grandes corporações, pois suas técnicas permitem a construção de programas extensos (até dezenas de milhões de linhas de código), estáveis e mais confiáveis.

A programação orientada por objetos é baseada no desenvolvimento de tipos abstratos de dados, permitindo agrupar, numa mesma entidade, um conjunto de dados e métodos que manipulam tais dados. Em C++, os tipos abstratos são implementados a partir do conceito de classe, constituindo-se na unidade fundamental a partir da qual uma variável deste tipo, denominada instância ou objeto, pode ser criada. É possível controlar a forma de acesso aos dados e métodos de um tipo abstrato definindo-se, por exemplo, se o acesso é restrito apenas a classe ou ao longo de todo o programa.

Na programação procedural, os procedimentos são a unidade básica de modulação, enquanto que na programação orientada por objetos as unidades básicas são os tipos abstratos. A dinâmica de funcionamento do programa passa então a ser vista como a interação entre os diversos objetos definidos. Como os dados são protegidos da manipulação externa, estas interações são efetuadas pela troca de mensagens, as quais são interpretadas pelos objetos que as recebem de acordo com as suas características, executando então uma determinada ação. Como conseqüência, esta técnica fornece uma estrutura que se assemelha muito à organização das abstrações matemáticas dos modelos físicos geralmente aplicados no meio científico, tais como operadores, matrizes, vetores, dentre outros.

No ambiente de engenharia, o desenvolvimento de programas exige a combinação de trabalhos de um número razoável de pessoas e a incorporação freqüente de novas técnicas. Além disso, as estruturas de dados exigidas para manipular e armazenar grandes os volumes de dados (necessários para a descrição dos modelos físicos) de maneira eficiente são geralmente complexas e de difícil utilização. Deve-se ressaltar também que essa classe de programas é aplicada na análise, simulação e projeto de componentes e sistemas, onde a obtenção de resultados precisos e de forma eficiente é um requisito básico. Em outras palavras, o ambiente de desenvolvimento de programas para engenharia se caracteriza pela necessidade de revisão e atualização freqüente de códigos complexos, sendo fundamental a garantia da confiabilidade.

A aplicação da programação orientada por objetos no contexto de engenharia tem sido relativamente pequena. Isto se deve parcialmente ao fato das primeiras linguagens orientadas por objetos comprometerem seriamente a eficiência. Entretanto, linguagens como C++ têm performance igual às linguagens procedurais tradicionais, permitindo a construção de códigos também altamente eficientes. Deve-se ressaltar ainda que um grande volume de código, escrito em linguagens procedurais, está disponível. A migração destes programas para uma linguagem orientada por objetos é na maioria dos casos inviável técnica e economicamente.

Por último, deve-se considerar a elevada taxa de crescimento da sofisticação dos sistemas computacionais nas suas capacidades de processamento, armazenamento, troca de informações e interação com os usuários. Os desenvolvedores de programas também devem estar atentos à incorporação das possibilidades oferecidas por processadores de grande capacidade numérica, computação paralela, arquiteturas cliente-servidor e alta capacidade gráfica, visando a produção de aplicações que permitam o tratamento de um espectro maior de problemas práticos de maneira mais fácil e eficiente. Devido às suas características, não é nenhuma surpresa que a programação orientada por objetos realmente facilite a simulação e a visualização de fenômenos físicos, explorando completamente as possibilidades de tratamento de informações resultantes da análise de sistemas complexos (visualização científica, multimídia, realidade virtual, etc.), possibilitando aindo o emprego de arquiteturas e características dos processadores e redes de computadores atuais.

5.1 Características Principais da Programação Orientada por Objetos

A metodologia de programação orientada por objetos consiste em definir e implementar hierarquias de tipos abstratos de dados. Em C++, a implementação de tais estruturas é feita através da construção de entidades auto-suficientes denominadas *classes.* Como dito anteriormente, esta é a estrutura através da qual podem ser agrupados conjuntos de *dados* e *métodos.* A classe é então a definição do *tipo.* Por sua vez, um objeto dessa classe é uma *variável* (instância) desse tipo. Assim, os objetos são declarados como variáveis de tipos abstratos definidos pelo usuário (`Matrix`, `Vector`, `Nodes`, `FEModel`, etc.) da mesma maneira como são definidas variáveis de tipos pré-definidos da linguagem (`int`, `double`, `char`, etc.). Por exemplo, o código a seguir define os objetos `b` e `u` da classe `Vector`, além da instância `A` da classe `Matrix`:

```
Vector b, u;
Matrix A;
```

Supondo que o método `GaussSolver`, para a solução de equações da forma `Au = b`, seja implementado na classe Matrix e que os parâmetros A, u e b estejam adequadamente inicializados, o seguinte comando resolve o sistema de equações:

```
A.GaussSolver(u,b);
```

A programação orientada por objetos é baseada nos princípios de abstração, encapsulamento, modulação e herança. Todas as linguagens desse tipo apresentam recursos baseados nesses conceitos, sendo suas definições independentes da linguagem adotada. Contudo, detalhes específicos de implementação serão abordados considerando a linguagem C++, utilizada nos programas desenvolvidos neste trabalho.

Abstração

Abstração é um termo usual para qualquer cientista ou engenheiro interessado em modelagem de sistemas físicos. Consiste da extração das características mais importantes do sistema, fornecendo uma generalização adequada e desprezando os detalhes irrelevantes. A abstração na programação orientada por objetos significa determinar as características que devem definir os objetos de uma classe, diferenciando-os de objetos de outros tipos.

Em outras palavras, a abstração significa definir a *interface pública* da classe, ou seja, como seus objetos vão ser acessados ou interagir com os demais objetos. Podem ser aplicados os mais diversos níveis de abstração, desde a reprodução fiel de características de objetos reais até a definição de classes puramente abstratas. Uma abstração eficiente fornece uma interface pública simples e concisa, independentemente da estrutura interna da classe, e, ao mesmo temo, genérica o suficiente para que os objetos da classe possam ser aplicados em uma ampla variedade de casos.

Encapsulamento

Dado um objeto, procedimentos externos e demais objetos do programa podem não conhecer ou acessar sua estrutura interna. A programação orientada por objetos tem a capacidade de ocultar a estrutura interna dos objetos, deixando visível apenas os componentes de sua interface pública. Este mecanismo é denominado *encapsulamento* ou *ocultamento de informação.*

Toda a interação com os objetos é efetuado através de *mensagens.* Quando um objeto recebe uma mensagem, executa a método associado, atuando sobre a sua estrutura interna.

Dessa forma, o encapsulamento não permite o acesso direto aos dados dos objetos, o qual pode ser feito somente através do conjunto de métodos definidos na interface pública da classe.

Assim, a organização dos dados, a estrutura, posição física de memória e detalhes de implementação são invisíveis ao código que utiliza o objeto. Além disso, se a interface pública for mantida inalterada, a implementação interna dos métodos pode ser modificada (atualizada, reimplementada de maneira mais eficiente, estendida, etc.) com total independência do restante do código. Dessa maneira, as modificações podem ser feitas em nível mais localizado, diminuindo o risco de efeitos colaterais indesejados. Em resumo, o encapsulamento dos dados de um objeto proporciona confiabilidade nos resultados e facilidades na manutenção.

Em C++, existem três níveis de acesso: `public`, `protected`, `private`. Os componentes da classe situados no nível `public` podem ser acessados externamente, ou seja, formam a interface pública. O nível `protected` engloba dados e procedimentos acessados somente pelos métodos da classe e de classes derivadas desta (ver a definição de herança). Os componentes classificados como `private` somente podem ser acessados pelos métodos da própria classe, sendo invisíveis inclusive para as classes derivadas.

Modulação

Modulação permite que um sistema possa ser decomposto em um conjunto de módulos fracamente dependentes. Com a utilização dos recursos anteriores de abstração e encapsulamento, um programa pode ser construído de forma que o funcionamento se dê através da interação de objetos pouco acoplados, facilitando a atualização e a detecção de erros. Entretanto, para cada sistema existe um nível adequado de modulação, acima do qual pode haver queda na eficiência e dificuldades no gerenciamento do código. De fato, um dos principais aspectos da programação orientada por objetos é a definição dos módulos descrevendo o problema a ser codificado.

Herança

Dada uma classe, a mesma pode ser utilizada para construir subclasses, *herdando* as características (dados e métodos) da classe antecessora através do recurso de derivação de classes. Dessa forma, as classes podem ser definidas formando uma *estrutura hierárquica*. Ao se descer nesta hierarquia, tem-se a especialização da informação. Indo para os níveis mais altos, verifica-se uma maior generalização ou abstração.

O conceito de herança diminui a quantidade de código a ser elaborado durante a implementação, pois módulos já existentes, testados e confiáveis podem ser aproveitados em situações novas através de classes derivadas, tornando a reutilização, aliada ao encapsulamento de informações, uma prática segura e confiável, reduzindo o tempo de desenvolvimento.

Polimorfismo

Através do *polimorfismo*, objetos de uma família de classes recebendo a mesma mensagem, cada um respondem de maneira particular, através de seu próprio método. Por exemplo, a mensagem + pode ser interpretada como uma concatenação de caracteres para a classe `String` ou como adição para a classe `Matrix`.

A combinação de herança e poliformismo permite alguns resultados interessantes. Uma estratégia bastante utilizada é a definição de classes base altamente genéricas, onde alguns métodos, denominados virtuais, não são necessariamente definidos neste nível básico. Objetos de tais classes também, denominadas *virtuais*, têm apenas a função de representar um conjunto de objetos mais especializados a serem definidos posteriormente. Por exemplo, considera-se uma

classe virtual genérica `FiniteElement` representando todos tipos de elementos finitos. Como dados, tem-se partâmetros comuns a todos os elementos, tais como números de nós e graus de liberdade. Métodos para o clículo de matriz de rigidez, de massa, tensões e deformações são declarados como virtuais nesta classe. A implementação específica é feita nas subclasses de `FiniteElement`.

Assim, durante o cálculo das tensões, o método a ser executado depende do tipo de elemento considerado durante a execução. Para que isto seja possível, declara-se um apontador para a classe base, no caso `FiniteElement`, e dependendo do nome do elemento usado no procedimento `NewElement`, diferentes métodos com o mesmo nome, no caso `GetStress`, serão executados pelo programa. Tem-se então o seguinte código:

```
FiniteElement *pFE = NewElement(FEName);
pFE->GetStress();
```

De forma geral, um ponteiro para um objeto da classe base pode também apontar para qualquer objeto de uma classe derivada. Assim, um objeto da classe base age como *representante* dos objetos das classes derivadas. Dessa forma, pode ser construído um código que decida o conjunto de métodos a serem executados somente em tempo de execução, dando origem a um programa genérico que pode facilmente incluir novas características pela inclusão de novos tipos derivados.

A aplicação dos conceitos anteriores para programas de engenharia têm conduzido a resultados bastante satisfatórios [21, 61, 62, 79, 81, 82, 83]. De forma geral, é implementada a estrutura mais adequada à manipulação e armazenamento dos dados do problema, considerando as exigências acerca da eficiência no acesso e economia de memória. Entretanto, a complexidade dessa estrutura fica, do ponto de vista da utilização dos objetos, oculta pela interface pública da classe. Em outras palavras, pode-se combinar a estrutura de dados mais apropriada e eficiente, independente de sua complexidade, com uma interface de utilização simples e prática.

Outro aspecto importante, devido aos conceitos de modulação, herança e encapsulamento, é a possibilidade de combinar com mais facilidade e rapidez os resultados dos trabalhos de determinado grupo de pessoas em um único código, dando origem a um sistema que possibilita, além do acesso eficiente à várias tecnologias, a sua utilização combinada. Claramente esse sistema pode dar origem a resultados de maior valor agregado que uma série de programas estanques e isolados.

Em geral, engenheiros e cientistas utilizando as práticas tradicionais de programação têm como principal objetivo a obtenção de resultados para trabalhos específicos, não havendo a preocupação de desenvolver códigos para o uso de terceiros. Neste sentido, não é necessário se concentrar suficientemente na implementação de estruturas de dados eficientes, devido não somente à dificuldade de implementação, mas também à forma direta e complexa de sua utilização em linguagens procedurais. Por exemplo, apesar de grande número de problemas de engenharia darem origem a sistemas de matrizes esparsas de grandes dimensões, procedimentos eficientes de alocação, armazenamento e solução de tais sistemas não são muito empregados, mesmo sendo geralmente códigos de domínio público. Soma-se a isto a dificuldade de combinar os programas procedurais tradicionais. Esses dados mostram que na prática os programas produzidos no meio acadêmico utilizando conceitos de orientação por objetos tendem a ser não somente de maior confiabilidade, mas também mais eficientes numericamente.

Deve-se deixar claro, contudo, que a utilização dos recursos de orientação por objetos em programas de cálculo numérico devem estar subordinados ao balanço entre a facilidade de administração do código (manutenção e atualização) e a eficiência. Um programa altamente modular e encapsulado não terá uma aceitação razoável pelos usuários se sua eficiência numérica for muito baixa. Dessa forma, é importante ser criterioso na aplicação da derivação de classes,

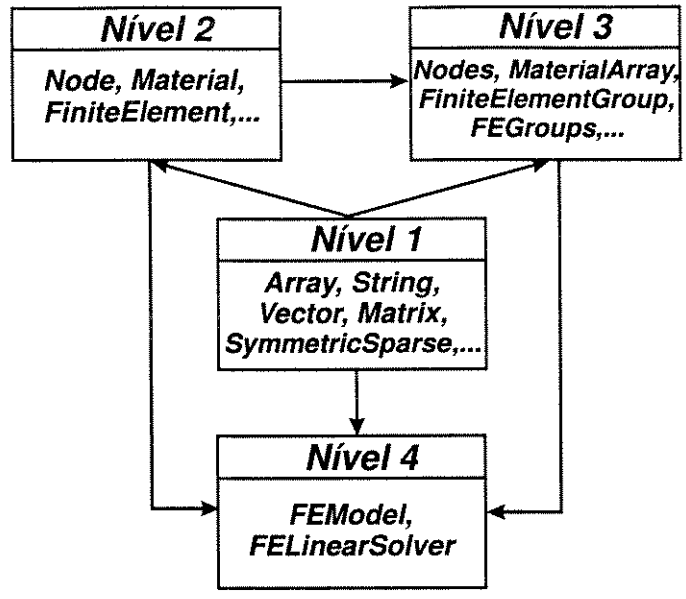


Figura 5.1: Níveis de organização das classes de elementos finitos.

procurando, por exemplo, alcançar códigos de fácil estensibilidade, mas evitando hierarquias de classes de vários níveis. A razão é que quando um objeto recebe uma mensagem, procura-se o método correspondente à mensagem. Caso não seja achado, a procura continua na classe antecessora, repetindo-se o processo até o método ser localizado, implicando num custo adicional e gasto de tempo.

Pode-se citar a implementação dos tipos de elementos, como classes derivadas de uma classe virtual genérica `FiniteElement`. Dessa forma, todas as manipulações de elementos finitos são implementadas independentemente do tipo de elemento utilizado, sendo o tipo específico de elemento a ser empregado na análise definido apenas durante o tempo de execução do programa. Além disso, novos tipos de elementos podem ser introduzidos sem que o código original precise ser alterado. Nesse caso, tem-se apenas um nível de hierarquia.

A mesma observação pode ser feita com relação aos tipos de materiais (isotrópicos, ortotrópicos, etc.) e no caso da análise de sensibilidade e otimização estrutural, na implementação dos funcionais.

Em resumo, utilizando os recursos de abstração e encapsulamento de informações, deve-se ter como objetivo combinar a estrutura de dados mais apropriada para o problema (implementação interna da classe) a uma interface pública simples e concisa, sem, entretanto, comprometer a eficiência numérica do programa em favor apenas da simplicidade.

5.2 Classes para Elementos Finitos

Os algoritmos de análise de sensibilidade e otimização foram implementados utilizando um conjunto de classes para análise linear estática por elementos finitos isoparamétricos desenvolvidas previamente [61] e revisadas no início do trabalho. Estas classes estão organizadas nos 4 níveis seguintes, ilustrados na Figura 5.1:

Nível 1 : Compreende o conjunto de classes definindo as estruturas de dados empregadas nos demais níveis. Tem como objetivo gerenciar a alocação dinâmica de memória. Compreende as classes **Matrix**, **SymmetricMatrix**, **SymmetricSkyline**, **SymmetricSparse**,

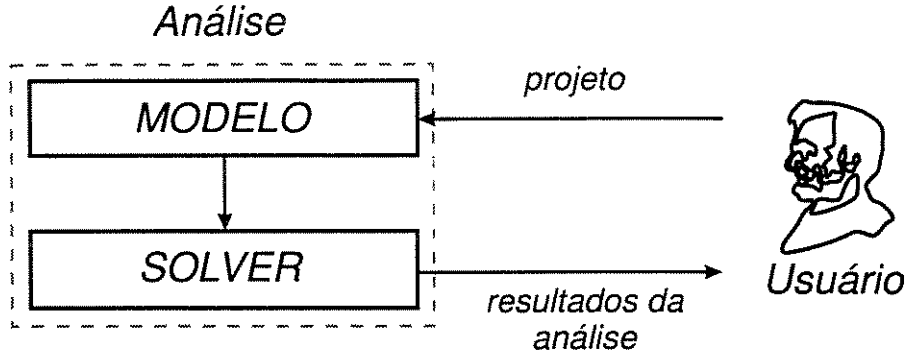


Figura 5.2: Utilização convencional de um programa de análise.

Vector, **String** e **Array**.

Nível 2 : Neste nível estão presentes as classes relativas a estruturas básicas do modelo de elementos finitos. Compreende as classes `Node`, **DefinitionDOF**, **EliminatedDOF**, **PrecribedBC**, **Equations**, **GeometricProperties**, **ElasticMaterial**, **ElasticIsotropicMaterial**, **FiniteElement**, **PlaneStressTriangular**, **TriangularShapeFunctions** e **TriangularGaussLegendre**.

Nível 3 : As classes desse nível consistem de conjuntos das classes dos níveis anteriores, organizando as informações das características do modelo de elementos finitos. Compreende as classes **Nodes**, **DOFBoundaryConditions**, **MaterialArray**, **FiniteElementGroup**, **FiniteElementGroups** e **LoadCase**.

Nível 4 : Neste nível estão as classes de armazenamento de todos os atributos do modelo de elementos finitos e solução do sistema de equações. Compreende as classes **FEModel** e **FELinearSolver**.

Observa-se que foram utilizados procedimentos do sistema ACDP [84] para o tratamento de erros, gravação e recuperação de dados em disco.

5.3 Implementação da Análise de Sensibilidade e da Otimização Estrutural

5.3.1 Características do Problema de Otimização

Na grande maioria dos casos, o desenvolvimento de um programa de otimização estrutural evolui a partir de um programa de análise estrutural já existente. A utilização de um programa de análise segue, em geral, as cinco etapas ilustradas na Figura 5.2, ou seja,

1. definição de um projeto especificando as dimensões relevantes;
2. geração do modelo de elementos finitos, definindo atributos como malha, materiais, condições de contorno e carregamento;
3. análise e obtenção dos resultados (eventualmente redefinição da discretização para reduzir erros);
4. julgamento dos resultados;

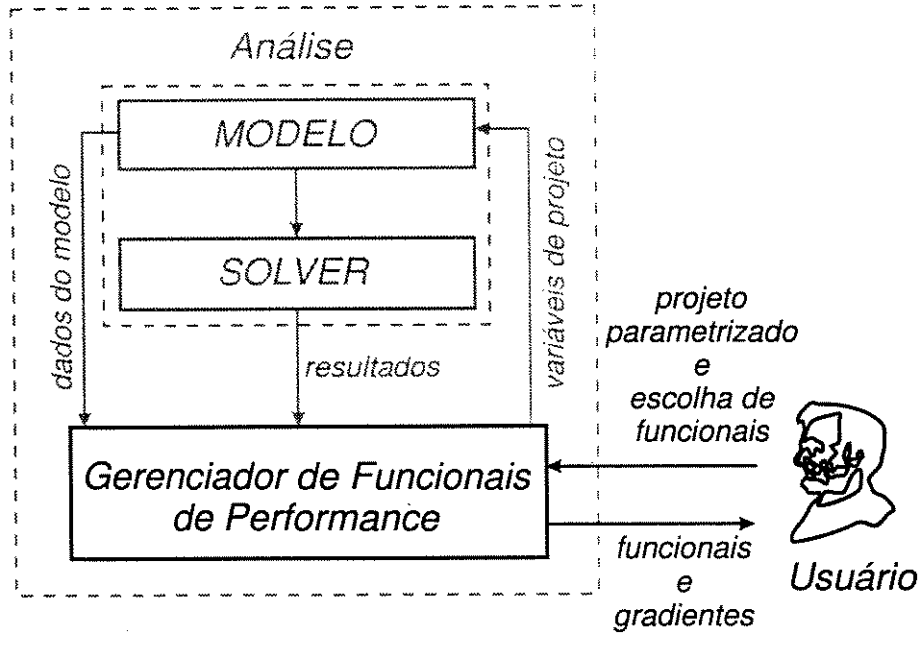


Figura 5.3: Utilização de funcionais de performance na análise.

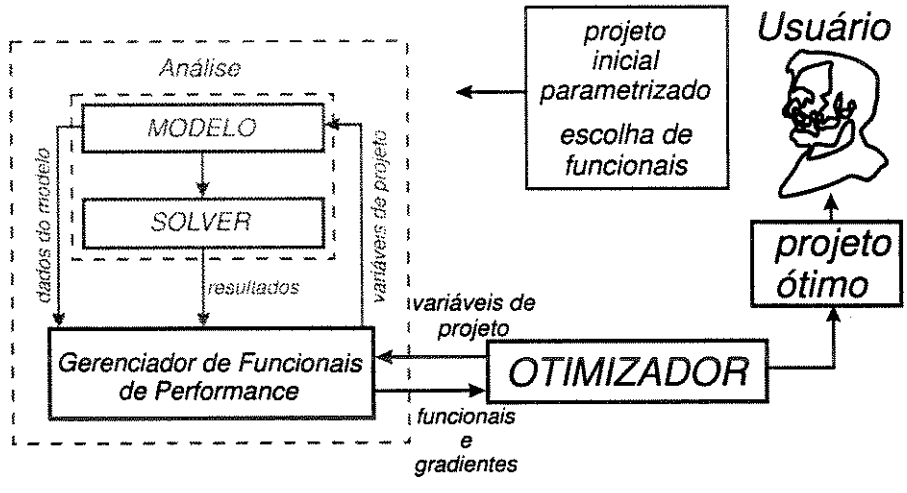


Figura 5.4: Utilização de um programa de otimização estrutural.

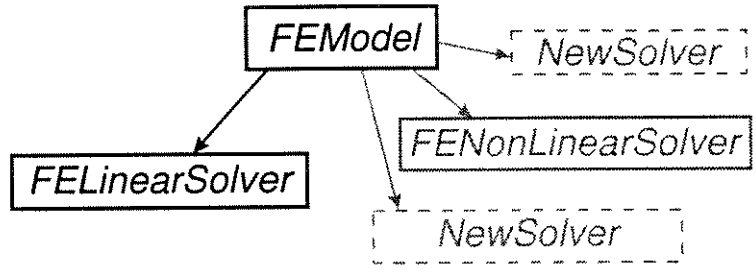


Figura 5.5: Hierarquia das classes `FEModel` e `FELinearSolver`.

5. redefinição do modelo.

Por mais recursos que se possa incorporar ao programa de análise estrutural, é impossível eliminar a necessidade de interação intensiva com usuário nas etapas 1, 2, 4 e 5 quando este módulo é utilizado de maneira isolada. A introdução do conceito de funcionais de peformance tem o objetivo de fornecer recursos para que a etapa 4 seja realizada de maneira mais objetiva. A introdução de um módulo de avaliação de funcionais e gradientes é mostrado na Figura 5.3.

A inclusão de um algoritmo de otimização visa automatizar o ciclo anterior, direcionando o trabalho do analista na definição do projeto (parametrização) e na definição dos funcionais de performance mais adequados. Este último estágio é ilustrado na Figura 5.4, podendo-se observar:

- um programa de otimização estrutural pode ser obtido a partir da incorporação de recursos de análise de sensibilidade e otimização a um programa de análise já existente;
- o processo de otimização é caracterizado pela intensa troca de informações entre as diversas etapas e conseqüentemente entre os diversos módulos do programa (solver, otimizador, gerenciador de funcionais). Assim, a performance global do processo depende da eficiência numérica do programa de análise, convergência do algoritmo de otimização, número de análise requeridas e eficiência na troca de informações entre os módulos;
- o algoritmo de otimização somente manipula variáveis de projeto, valores de funcionais e gradientes, sendo independente (do ponto de vista numérico) de como são organizadas as estruturas anteriores;
- a estrutura que realiza a avaliação dos funcionais e seus gradientes constitui uma interface entre o programa de análise e o algoritmo de otimização, devendo ser capaz de acessar as informações relevantes do modelo e os recursos de solução de sistemas do módulo de análise (para a solução dos problemas adjuntos) e, ao mesmo tempo, mantê-las protegidas de acessos indevidos.

A partir dessas conclusões, são definidas as características necessárias às classes e a maneira como seus objetos devem interagir.

5.3.2 Modelo de Elementos Finitos

O módulo de análise usado na implementação dos procedimentos de otimização estrutural está definido na classe `FEModel` e em uma classe derivada chamada `FELinearSolver`. A classe `FEModel` reúne a estrutura de dados e métodos que descrevem um modelo de elementos finitos:

```
class FEModel
{
protected:
   DefinitionDOF DOFs;          //graus de liberdade
   Nodes Coords;                //coordenadas nodais
   LoadCases LCases;            //casos de carregamentos
   FEGroups Groups;             //grupos de elementos
   MaterialArray Materials;     //tipos de materiais
   Equations DOFEq;             //equacoes
   DOFBoundaryConditions BC;    //condicoes de contorno

public:
   void Read(FILE *File);       //le arquivo de entrada
   void Print(FILE *File);      //imprime dados

   //funcoes de acesso aos dados
              .
              .
              .
};
```

Os dados são objetos de um conjunto de classes definindo nós, graus de liberdade, casos de carregamento, grupos de elementos, materiais, propriedades geométricas, equações e condições de contorno, equações e casos de carregamento. Os métodos públicos são constituídos por funções de alocação, leitura, impressão de dados e funções de acesso aos métodos de cada um desse objetos. Como a mesma estrutura do modelo de elementos finitos pode ser utilizada para uma variedade de tipos de análise, o *solver* proprieamente dito é definido como uma classe derivada da anterior, permitindo que outros tipos de análises possam ser incorporados, compartilhando a estrutura de `FEModel` através do recurso de derivação, conforme descrito na Figura 5.5.

A classe derivada `FELinearSolver` destina-se à análise de problemas estruturais lineares estáticos:

```
class FELinearSolver: public FEModel
{
protected:
   SymmetricSparse GlobalMatrix;   //matriz global do problema
   Vector Load, U;                 //vetores de carregamento e solucao

   void SetMatrix();               //constroi matriz global
   void SaveSolution2D();          //salva resultados

public:
   void Solver();                  //executa analise
};
```

Os dados são matriz de rigidez armazenada empregando uma classe para matriz [61], além dos vetores de carregamento e solução como instâncias da classe `Vector`. A interface pública se resume ao método `Solver()` que realiza a análise de elementos finitos:

```
char *DataFile = "modelo.fem";       //declara nome do arquivo de dados
FELinearSolver Modelo(DataFile);     //declara objeto a ser analisado
```

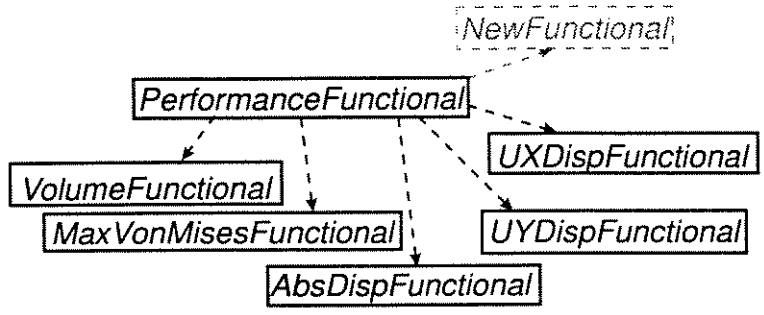


Figura 5.6: Hierarquia das classes de funcionais.

```
Modelo.Solver();                    //analise
```

Recebendo esta mensagem, o objeto `Modelo` executa os métodos internos de leitura dos dados do arquivo `modelo.fem`, construção da matriz de rigidez e do vetor de carregamentos, solução do sistema e preparação dos dados para visualização.

5.3.3 Funcionais

As exigências para a implementação dos funcionais de performance são as seguintes:

1. a estrutura deve permitir que novos tipos de funcionais possam ser incluídos exigindo apenas alterações localizadas, sem influir na estrutura principal do programa;
2. a escolha do conjunto de funcionais a serem utilizados em cada análise, dentre os tipos disponíveis, deve ser feita apenas em tempo de execução do programa;
3. cada tipo de funcional já dever conter seus métodos de análise de sensibilidade em sua estrutura interna, simplificando a sua utilização.

A solução que permite satisfazer esse conjunto de exigências é a definição de uma classe virtual genérica `PerformanceFunctional`, a partir da qual todos os tipos específicos de funcionais de performance são derivados, como mostrado na Figura 5.6. A estrutura básica dessa classe é a seguinte,

```
class PerformanceFunctional
{
protected:
    String FuncName;        //nome do funcional
    double TestVal;         //valor de teste

    virtual void WriteAdjointLoad(FELinearSolver &Model) = 0;

public:
    virtual double Eval(FELinearSolver &Model) = 0;
    virtual void Gradient(FELinearSolver &Model, Vector &Grad) = 0;
};
```

Como se trata de uma classe destinada principalmente a definição de métodos, o conjunto de dados é simples, consistindo apenas de um nome, usado na impressão de informações, além de um valor de teste, pois funcionais usados como restrições são usualmente comparados com

valores máximos ou mínimos. A interface pública da classe é formada por funções virtuais, as quais são implementadas somente nas classes derivadas de acordo com o funcional específico, declarando os métodos de avaliação do funcional, determinação de seu gradiente e construção do carregamento adjunto para a análise de sensibilidade. Na declaração desses métodos também se observa como a programação orientada por objetos permite definir uma estrutura de código muito próxima da estrutura de solução do problema. Como os funcionais de performance são entidades definidas sobre o modelo matemático do projeto, torna-se uma decisão natural definir métodos que tenham como argumento um objeto da classe `FELinearSolver`.

A partir desta classe, são derivadas as especializações dos diversos tipos de funcionais da Figura 5.5, necessários ao tratamento do problema de otimização estrutural: funcional de volume na classe `VolumeFunctional`; funcional de máxima tensão ponderada de von Mises dado em (1.14) na classe `MaxVonMisesFunctional`; funcional de deslocamento resultante dado em (1.15) na classe `AbsDispFunctional`; funcionais de deslocamento absoluto nas direções x e y nas classes `UXDispFunctional` e `UYDispFunctional`; funcional de energia de deformação dado em (1.16) na classe `StrainEnergyFunctional`. Por exemplo, para o cálculo do funcional de deslocamento resultante segundo a equação (1.15) com 0.05 como valor máximo admissível, escreve-se,

```
AbsDispFunctional ResMax(0.05);
double res = ResMax.Eval(Modelo);
```

O cálculo de seu gradiente é feito da seguinte forma,

```
Vector Grad;
ResMax.Gradient(Modelo, Grad);
```

para variáveis não-relacionadas à forma do domínio e

```
ResMax.Gradient(Modelo, Grad, Velocity);
```

para variáveis de forma, sendo `Velocity` o vetor de campo de velocidades nos nós.

Os métodos `Gradient` acessam internamente os procedimentos de construção do carregamento adjunto e solução do problema adjunto.

Como todas as classes de funcionais descendem de uma mesma classe base, o acesso aos objetos que os implementam pode ser feito utilizando somente ponteiros para objetos da classe `PerformanceFunctional`, sendo, portanto, independente do tipo específico de funcional acessado. O tipo de funcional pode ser definido somente em tempo de execução via, por exemplo, arquivo de dados.

5.3.4 Gerenciador de Funcionais

A classe `FunctionalManager` visa automatizar a utilização dos funcionais de performance nos problemas de análise estrutural, proporcionando, ao mesmo tempo, uma troca eficiente de informações entre o *solver* e a estrutura de funcionais. Nesse sentido, essa classe agrupa como dados um ponteiro para `PerformanceFunctional` (função objetivo), uma estrutura `FunctionalArray` (array de ponteiros `PerformanceFunctional`) correspondendo aos funcionais de restrição, um array com o campo de velocidades, um objeto `FELinearSolver`, correspondendo ao modelo de elementos finitos, as dimensões do problema de otimização, variáveis armazenando os valores dos funcionais e seus gradientes, um objeto `DesignVariablesDefs` com a definição das variáveis e um objeto `BoundaryLayerVelocity` para determinação do campo de velocidades. Como métodos tem-se aqueles para avaliação dos funcionais e seus gradientes. Essa classe tem a seguinte estrutura básica:

```
class FunctionalManager
```

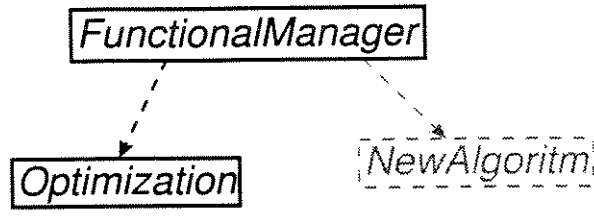


Figura 5.7: Hierarquia das classes `FunctionalManager` e `Optmization`.

```
{
private:
   PerformanceFunctional   *ObjFunc;     //funcao objetivo
   FunctionalArray ConstrSet;            //array de funcionais de restricao
   Array<double> Vi;                     //velocidades de projeto
protected:
   FELinearSolver Model;                 //modelo FE a ser otimizado
   long n, m, p;                         //dimensoes
   double f;                             //valor da funcao objetivo
   Vector X,                             //valores das variaveis de projeto
          G,                             //valor do vetor de restricoes
          Lambda,                        //multiplicadores de Lagrange
          C;                             //gradiente da funcao objetivo
   Matrix XLimits,                       //limites das variaveis
          A;                             //gradiente dos funcionais
   DesignVariablesDefs Variables;        //definicao das variaveis de projeto
   BoundaryLayerVelocity VelocityField; //determinacao da velocidade
public:
   void Eval();                          //avalia funcionais
   void Derivative();                    //determina gradientes
};
```

Como um dos objetivos dessa classe é constituir uma interface entre o *solver* e o algoritmo de minimização em um código de otimização estrutural, os funcionais são mantidos inacessíveis para classes derivadas, pois os algoritmos de minimização devem acessar somente os valores dos funcionais e de seus gradientes e não as estruturas de dados e métodos que permitem a sua obtenção.

Supondo que o modelo e o conjunto de funcionais relevantes para a análise tenham sido definidos pelo usuário, respectivamente nos arquivos de dados `DataFile` e `FuncFile`, as seguintes linhas de código determinam a avaliação dos funcionais e gradientes:

```
FunctionalManager Modelo(DataFile, FuncFile);
Modelo.Eval();
Modelo.Derivative();
```

A mensagem `Eval` internamente executa a análise de elementos finitos para o objeto `Model` que é então acessado pelos funcionais. Da mesma forma, `Derivative` determina a execução dos procedimentos necessários a determinação dos gradientes, inclusive os procedimentos de análise de sensibilidade se necessário.

5.3.5 Algoritmo de Minimização

A classe `Optimization` implementa o algoritmo de minimização de Herkovits discutido no Capítulo 2. Para ter acesso direto aos valores dos funcionais e gradientes foi definida como uma classe derivada de `FunctionalManager`, como pode ser visto na Figura 5.7. Além disso, essa opção permite que a mesma classe de otimização seja utilizada para diferentes tipos de análise, dependendo da parte privada da classe `FunctionalManager`, não necessariamente uma análise por elementos finitos (os exemplos dos Capítulos 2, 3 e 4 foram obtidos com a mesma implementação da classe `Optimization`). Foi definida a seguinte estrutura básica,

```
class Optimization: public FunctionalManager
{
protected:
  //variaveis auxiliares
          .
          .
          .

  void SearchDirection();   //determina direcao de busca
  void LinearSearch();      //busca linear
  void Update();            //atualiza dados
public:
  int Iterate();            //iteracao do algoritmo
};
```

Os dados dessa classe consistem apenas de variáveis auxiliares utilizadas nos cálculos intermediários do algoritmo de otimização. Os métodos internos principais consistem das funções `SearchDirection` e `LinearSearch`, implementando respectivamente os procedimentos de determinação da direção de busca e de busca linear, além da função `Update` que atualiza os valores do algoritmo para a próxima iteração. A interface pública dessa classe consiste apenas da função `Iterate`, realizando uma iteração do algoritmo, retornando 1 em caso de convergência e 0 caso contrário. Para aplicar o algortimo basta executar esta função em um *loop* condicional, até que haja convergência ou um limite de iterações seja atingido.

Deve-se ressaltar que durante o processo de otimização, a troca de informações entre os diversos módulos que compõem o código é feita com as variáveis mantidas em memória, sem recorrer a troca de arquivos de dados em disco. Da mesma forma, a preparação de dados para visualização é feita somente na última análise.

5.4 Otimização Estrutural e Análise de Sensibilidade Orientadas por Objetos

A práticas tradicionais de programação são, em geral, barreiras ao aumento de eficiência dos programas de otimização. Isto ocorre pois, ao se incorporar módulos de análise de sensibilidade e programação matemática, a um programa de análise estrutural já existente, toda a comunicação é feita através de troca de arquivos de dados. De outra maneira, uma maior integração entre os novos módulos e o programa de análise exigiria não só a reimplementação de parte do código de análise, mas também o acesso direto às suas estruturas de dados. Essa opção, além de aumentar o tempo de desenvolvimento, exige que se conheça detalhes da implementação do código de análise.

A utilização dos recursos de programação orientada por objetos, discutidos neste capítulo, permitem a integração eficiente entre os diversos módulos de um programa de otimização sem comprometer a independência dos módulos e a segurança dos dados.

Em primerio lugar, a incorporação dos recursos de avaliação de funcionais e seus gradientes à análise de elementos finitos foi feita declarando-se uma instância da classe para análise de elementos finitos, previamente desenvolvida e testada [61], como um dado da classe. Esta instância, através de sua interface pública, disponibiliza todos os métodos de análise estrutural por elementos finitos, como construção e solução de sistemas, e métodos de acesso aos dados do modelo e resultados da análise, mantidos em memória, durante a avaliação dos funcionais e análise de sensibilidade. Ao implementar o algoritmo de programação matemática como uma classe derivada da anterior, o programa de otimização herda as mesmas características.

Dessa forma, elimina-se o *overhead* associado à troca de arquivos, não sendo necessário para isso reimplementar partes do código de análise estrutural ou acessar diretamente sua estrutura de dados. Os conceitos de orientação por objetos permitem então atingir eficiência e confiabilidade em programas de otimização estrutural e análise de sensibilidade.

Capítulo 6

Conclusões e Perspectivas Futuras

A partir das discussões e resultados dos capítulos anteriores podem ser apresentadas as seguintes conclusões acerca da análise de sensibilidade e otimização em problemas lineares:

- é indispensável considerar as caracteríticas dos problemas estruturais na escolha e implementação de algoritmos eficientes de programação matemática para problemas estruturais. Nesse sentido, o algoritmo de Herskovits se mostra bastante adequado para esta classe de problemas devido à alta taxa de convergência. Além disso, por ser um método de ponto interior, fornece uma solução viável mesmo que não haja convergência no processo iterativo. Permite ainda incorporar uma estratégia de restrições ativas para aumentar a eficiência global do processo iterativo. Sua desvantagem está na necessidade de um ponto inicial viável, o que pode se tornar difícil na presença de grande número de restrições ou mesmo restriçõs complicadas, exigindo a utilização de procedimentos numéricos para a sua determinação [67];

- como se pode observar nos exemplos de otimização estrutural dos capítulos anteriores, a maior parcela do decréscimo na função objetivo é atingido já nas primeiras iterações. Isso permite que soluções razoáveis para casos práticos possam ser obtidos a um custo relativamente baixo, limitando-se o número de iterações ao mínimo necessário para se obter um decréscimo satisfatório na função objetivo;

- funcionais de performance estrutural podem ser definidos sem associação a topologia da malha de elementos. Isso fornece a flexibilidade de manipular apenas modelos geométricos, permitindo gerar novas malhas durante a otimização;

- a utilização de curvas de Bezier, B-splines e NURBS na descrição de geometrias para otimização de forma não exige nenhuma restrição adicional para se evitarem geometrias degeneradas, além dos limites máximos e mínimos de cada variável, os quais são restrições lineares;

- com o refinamento da malha de elementos finitos, em geral, o erro da solução do problema estrutural cai mais acentuadamente que o erro associado à determinação numérica de gradientes. Isto sugere a necessidade de determinar estimativas de erro da malha para o cálculo da sensibilidade, obtendo-se a discretização mais conveniente [70];

- deve-se ter atenção especial na escolha das variáveis de projeto e funcionais de performance mais convenientes para análise. Os resultados de sensibilidade relativos à variáveis com pequena influência no funcional tendem a ser dominados por erros numéricos;

- a abordagem adotada para a determinação dos campos de velocidades de projeto se mostrou bastante eficiente. Primeiro porque são considerados campos de velocidades não-nulos apenas nos elementos adjacentes ao contorno parametrizado, não exigindo, portanto, procedimentos numéricos adicionais para a determinação de velocidades no interior do domínio. Como conseqüência, as integrais de domínio somente são avaliadas nesse conjunto de elementos. Essa abordagem, apesar de apresentar resultados satisfatórios, pode apresentar imprecisões em alguns casos, pois um campo de velocidades não-nulo em uma região com espessura de apenas um elemento pode não ser sempre a opção mais conveniente em termos da precião. Em tais situações torna-se necessário aplicar outros métodos de cálculo de campos de velocidades [78];

- a utilização do paradigma de orientação por objetos permite obter um programa de otimização estrutural com ênfase na eficiência na troca de informações entre os módulos de análise estrutural, análise de sensibilidade e programação matemática, sem a necessidade de reimplementar o programa de análise por elementos finitos original. Dessa forma, pode-se efetivamente aumentar a produtividade e a qualidade dos programas produzidos com a utilização de tais conceitos. A linguagem C++ permite que os recursos de orientação por objetos sejam empregados mantendo-se a eficiência numérica, tornando possível obter programas melhores em termos de utilização e manutenção se comparada às linguagens procedurais tradicionais.

Como atividades a serem desenvolvidades a partir dos resultados desse trabalho podem ser citadas:

- revisão das classes de análise de sensibilidade e otimização e integração em um ambiente interativo gráfico;

- incorporação de uma estratégia de restrições ativas à implementação do algoritmo de Herskovits;

- incorporar novos funcionais de performance estrutural ao programa, principalmente outras opções de funcionais de tensão;

- estudo de um estimador de erro para a análise de sensibilidade;

- consideração de métodos alternativos para determinação do campo de velocidades.

Referências Bibliográficas

[1] TIMOSHENKO, Stephen P. *History of Strength of Materials.* McGraw-Hill, New York, 1953.

[2] MICHELL, A. G. M. The limits of economy of material in frame structures. *Philosophical Magazine*, v. 8, n. 47, p. 589–595, November 1904.

[3] KLEIN, B. Direct use of extremal principles in solving certain optimization problems involving inequalities. *Journal of the Operations Research Society of America*, v. 3, p. 168–175, 1955.

[4] SCHIMT, Lucien A. Structural design by systematic synthesis. In: *Conference on Eletronic Computation*, 2, 1960, New York. American Society of Civil Engineering, 1960. p. 105–122.

[5] VANDERPLAATS, Garret N. Structural optimization - past, present and future. *AIAA Journal*, v. 20, n.7, p. 992–1000, July 1982.

[6] SCHMIT, L. A., FARSHI, B. Some approximation concepts for structural concepts for structural synthesis. *AIAA Journal*, v. 12, p. 692–699, May 1974.

[7] SCHMIT, Lucien A. Structural synthesis - its genesis and development. *AIAA Journal*, v. 19, n. 10, 1249–1263, October 1981.

[8] TEMPLEMAN, Andrew B. Optmization methods in structural design practice. *Journal of Structural Engineering*, v. 109, n. 10, p. 2420–2433, October 1983.

[9] LAVI, Abrahim, VOGL, Thomas P., editor. *Recent Advances in Optimization Techniques*, New York, 1966. IEEE and Optical Society of America, John Wiley & Sons.

[10] LUENBERGER, David G. *Optimization by Vector Space Methods.* John Wiley & Sons, New York, 1969.

[11] POLAK, E. *Computational Methods in Optimization: A Unified Approach*, volume 77 of *Mathematics in Science and Engineering.* Academic Press, New York, 1971.

[12] ARORA, Jasbir S., HAUG, Edward J. *Applied Optimal Design - Mechanical and Structural Systems.* John Wiley & Sons, New York, 1979.

[13] LEVY, Robert, LEV, Ovadia E. Recent developments in structural optimization. *Journal of Structural Engineering*, v. 113, n. 9, p. 1939–1962, September 1987.

[14] BELEGUNDU, Ashok D., ARORA, Jasbir S. A study of mathematical programming methods for structural optimization. part i: Theory. *International Journal for Numerical Methods in Engineering*, v. 21, p. 1583–1599, 1985.

[15] BELEGUNDU, Ashok D., ARORA, Jasbir S. A study of mathematical programming methods for structural optimization. part ii: Numerical results. *International Journal for Numerical Methods in Engineering*, v. 21, 1601–1623, 1985.

[16] LIM, O. K., ARORA, J. S. An active set rqp algorithm for engineering design optimization. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 51–65, September 1986.

[17] HERSKOVITS, José. A two-stage feasible directions algorithm for nonlinear constrained optimization. *Mathematical Programming*, v. 36, p. 19–38, 1986.

[18] PANIER, Eliane R., TITS, André L., HERSKOVITS, José. A qp-free, globally convergent, locally superlinear convergent algorithm for inequality constrained optimization. *SIAM Journal of Control and Optimization*, v., 26, n. 4, p. 788–810, July 1988.

[19] HERSKOVITS, José, COELHO, C. A. B. An interior points algorithm for structural optimization problems. In BREBBIA, C. A., HERNANDEZ, S. (Eds.), *Computer aided optimum design of structures: recent advances - Proceedings of the First International Conference, Southampton, June, 1989*, Southampton. Computational Mechanics Publications - Springer Verlag, 1989, p. 231–241.

[20] EVSUKOFF, Alexandre G. Sobre a introdução de um algoritmo de ponto interior no ambiente de projeto de engenharia. Rio de Janeiro: COPPE, UFRJ, 1992. Dissertação (Mestrado) - COPPE, Universidade Federal do Rio de Janeiro, 1992.

[21] FANCELLO, Eduardo A. *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.* Rio de Janeiro: COPPE, UFRJ, 1993. Tese (Dotourado) - COPPE, Universidade Federal do Rio de Janeiro, Julho 1993.

[22] FANCELLO, E., A., HASLINGER, J., FEIJOÓ, R. A. Numerical comparison between two cost functions in contact shape optimization. *Structural Optimization*, v. 9, n. 1, p. 57–68, February 1995.

[23] WANG, Shu-Yu, SUN, Yanbing, GALLAGHER, R. H. Sensitivity analysis in shape optimization of continuum structures. *Computers & Structures*, v. 20, n.5, p. 855–867, 1985.

[24] BENNETT, J. A., BOTKIN, M. E. Structural shape optimization with geometric description and adaptive mesh refinement. *AIAA Journal*, v. 23, n. 3, p. 458–464, March 1985.

[25] DING, Yunliang. Shape optimization of structures: A literature survey. *Computers & Structures*, v. 24, n. 6, p. 985–1004, 1986.

[26] KIKUCHI, Noboru, CHUNG, Kyoon Y., TORIGAKI, Toshikazu, TAYLOR, John E. Adaptive finite element methods for shape optimization of linearly elastic structures. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 67–89, September 1986.

[27] HAFTKA, Raphael T., GRANDHI, Ramana V. Structural shape optimization — a survey. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 91–106, September 1986.

[28] CHOI, K. K., HAUG, E. J. Shape design sensitivity analysis of elastic structures. *Journal of Structural Mechanics*, v. 11, p. 231–269, 1983.

[29] YANG, Ren-Jye, CHOI, Kyung K. Accuracy of finite element based shape design sensitivity analysis. *Journal of Structural Mechanics*, v. 13, n. 2, p. 223–239, 1985.

[30] CHOI, Kyung K., SEONG, Hwal G. A domain method for shape design sensitivity analysis of built-up structures. *Computer Methods in Applied Mechanics and Engineering*, v. 57, n. 1, p. 1–15, September 1986.

[31] HAUG, Edward J., CHOI, Kyung K., KOMKOV, Vadim. *Design sensitivity analysis of structural systems*, volume 177 of *Mathematics in Science and Engineering*. Academic Press, Orlando, 1986.

[32] GURTIN, Morton E. *An Introduction to Continuum Mechanics*, volume 158 of *Mathematics in Science and Engineering*. Academic Press, 1981.

[33] SEONG, Hwal G., CHOI, Kyung K. Boundary-layer approach to shape design sensitivity analysis. *Mechanics of Structures and Machines*, v. 15, n. 2, p. 241–263, June 1987.

[34] YANG, R. J., BOTKIN, M. E. Accuracy of the domain material derivative approach to shape design sensitivities. *AIAA Journal*, v. 25, n. 12, p. 1606–1610, December 1987.

[35] YAO, Tse-Min, CHOI, Kyung K. 3-d shape optimal design and automatic finite element regridding. *International Journal for Numerical Methods in Engineering*, v. 28, n. 2, p. 369–384, Feb 1989.

[36] RAJAN, S. D., BELEGUNDU, A. D. Shape optimal design using fictitious loads. *AIAA Journal*, v. 27, n. 1, p. 102–107, Jan 1989.

[37] TSENG, C. H., ARORA, J. S. Numerical verification of design sensitivity analysis. *AIAA Journal*, v. 27, n. 1, p. 117–119, Jan 1989.

[38] CHOI, K. K., SANTOS, J. L. T. Design sensitivity analysis of nonlinear structural systems part i: theory. *International Journal for Numerical Methods in Engineering*, v. 24, p. 2039–2055, 1987.

[39] HAFTKA, R. T., ADELMAN, H. M. Recent developments in structural sensitivity analysis. *Structural Optimization*, v. 1, n. 3, p. 137–151, September 1989.

[40] BREBBIA, C. A., TELLES, J. C. F., WROBEL, L. C. *Boundary element technique*. Springer-Verlag, Berlin, 1984.

[41] HARTMANN, F. *Introduction to Boundary Elements. Theory and Applications*. Springer-Verlag, 1989.

[42] ZHAO, Zhiye. *Shape Design Sensitivity Analysis and Optimization Using the Boundary Element Method*, volume 62 of *Lecture Notes in Engineering*. Springer-Verlag, Berlin, 1991.

[43] VANDERPLAATS, Garret N., SUGIMOTO, Hiroyuki. A general-purpose optimization program for engineering design. *Computers & Structures*, v. 24, n. 1, p. 13–21, 1986.

[44] FLEURY, C. Conlin: An efficient dual optimizer based on convex approximation concepts. *Structural Optimization*, v. 1, p. 81–89, 1989.

[45] KIRSCH, Uri. *Optimum Structural Design: Concepts, Methods and Applications.* McGraw-Hill, New York, 1981.

[46] GILL, Philip E., MURRAY, Walter, WRIGHT, Margareth H. *Practical Optimization.* Academic Press, San Diego, 1981.

[47] MANGASARIAN, Olvi L., MEYER, Robert R., ROBINSON, Stephen M., editor. *Nonlinear Programming 4*, New York, 1981. Computer Sciences Department, University of Wisconsin-Madison, Academic Press.

[48] ARORA, Jasbir S. *Introduction to Optimum Design.* Mechanical Engineering Series. McGraw-Hill, New York, 1989.

[49] LUENBERGER, David G. *Linear and Nonlinear Programming.* Addison-Wesley, Massachusetts, 2nd edition, 1989.

[50] VANDERPLAATS, Garret N. *Numerical Optimization Techniques for Engineering Design: With Applications.* McGraw-Hill Series in Mechanical Engineering. McGraw-Hill, New York, 1984.

[51] KODIYALAM, S., VANDERPLAATS, G. N., MIURA, H. Structural shape optimization with msc/nastran. *Computers & Structures*, v. 40, n. 4, p. 821–829, 1991.

[52] CHARGIN, Mladen K., RAASCH, Ingo, BRUNS, Rudolf, DEUERMEYER, Dawson. General shape optimization capabiblity. *Finite Elements in Analysis and Design*, v. 7, n. 4, p. 343–354, February 1991.

[53] Ansys Inc. *Ansys Release 5.1 – Procedures Manual*, 1995.

[54] Ansys Inc. *Ansys Release 5.1 – Theory Manual*, 1995.

[55] BOTKIN, M. E. Three-dimensional shape optimization using fully automatic mesh generation. *AIAA Journal*, v. 30, n. 7, p. 1932–1934, July 1992.

[56] YANG, R. J., CHAHANDE, A. I. Automotive applications of topology optimization. *Structural Optimization*, v. 9, n. 3/4, p. 245–249, July 1995.

[57] TWU, Sung L., CHOI, Kyung K. Configuration design sensitivity analysis of built-up structures part i: Theory. *International Journal for Numerical Methods in Engineering*, v. 35, n. 5, p. 1127–1150, September 1992.

[58] VANDERPLAATS, Garret N. Thirty years of modern structural optimization. *Advances in Engineering Software*, v. 16, n. 2, p. 81–88, 1993.

[59] COX, B. *Object-Oriented Programming – An Evolutionary Approach.* Addison-Wesley, 1986.

[60] BITTENCOURT, Marco L. Análise estática e dinânica por subestruturação e programação orientada por objetos. Campinas: FEM, Unicamp, 1990. Dissertação (Mestrado) - Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1990.

[61] BITTENCOURT, Marco L. *Métodos Iterativos e Multigrid Adaptáveis em Malhas Não-Estruturadas.* Campinas: FEM, Unicamp, 1996. Tese (Mestrado) - Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1996.

[62] FANCELLO, E. A., GUIMARÃES, A. C., FEIJOÓ, R. A. Geração automática de malhas em programação orientada a objetos. In *XI Congresso Brasileiro de Engenharia Mecânica*, São Paulo, Brasil, Dezembro 1991.

[63] CARDOSO, João M. B., SANTOS, José L. T. Object oriented design sensitivity analysis and optimization. In HERSKOVITS, José (Ed.), *Proceedings of Structural Optimization 93 - The World Congress on Optimal Design of Structural Systems*, Rio de Janeiro, August 1993. Rio de Janeiro: ABCM and COPPE/UFRJ, 1993, p. 423–430.

[64] NEMIROVSKY, Adolfo M. Is shrödinger's cat object-oriented - why natural scientists should care about object-oriented technology. Taligent White Papers URL: http://www.taligent.com/Technology/WhitePapers/quantun-oo.html, April 1994.

[65] LEE, H. H., ARORA, Jasbir S. Object-oriented programming for engineering applications. *Engineering with Computers*, v. 7, n. 4, p. 225–235, Fall 1991.

[66] ARORA, J. S., CHAHANDE, A. I., PAENG, J. K. Multiplier methods for engineering optimization. *International Journal for Numerical Methods in Engineering*, v. 32, n. 7, p. 1485–1525, November 1991.

[67] ELWAKEIL, Ossama A., ARORA, Jasbir S. Methods for finding feasible points in constrained optimization. *AIAA Journal*, v. 33, n. 9, p. 1715–1719, Sep 1995.

[68] ARORA, J. S., ELWAKEIL, O. A., CHAHANDE, A. I. Global optimization methods for engineering applications - a review. *Structural Optimization*, v. 9, n. 3/4, p. 137–159, July 1995.

[69] HASLINGER, J., JEDELSKÝ, D. Genetic algorithms and fictitious domain based approaches in shape optimization. *Structural Optimization*, v. 12, n. 4, p. 257–264, December 1996.

[70] FUENMAYOR, F. J., OLIVER, J. L., RÓDENAS, J. J. Extension of the zienkiewicz-zhu error estimator to shape sensitivity analysis. *International Journal for Numerical Methods in Engineering*, v. 40, p. 1413–1433, 1997.

[71] ZIENKIEWICZ, O. C., ZHU, J. Z A simple error estimator and adaptive procedure for practical engineering analysis. *International Journal for Numerical Methods in Engineering*, v. 24, p. 337–359, 1987.

[72] BAZARAA, Mokhtar S., SHERALI, Hanif D., SHETTY, C. M. *Nonlinear Programming - Theory and Applications*. John Wiley & Sons, New York, 2nd edition, 1993.

[73] PERESSINI, Anthony L., SULLIVAN, Francis E., UHL Jr., J. J. *The Mathematics of Nonlinear Programming*. Undergraduate Textes in Mathematics. Springer-Verlag, New York, 1993.

[74] KODIYALAM, S., SAXENA, M., editor. *Geometry and Optimization Techniques for Structural Design*. International Series on Computational Engineering. Computational Mechanics Publications - Elsevier Applied Science, London, 1994.

[75] RAO, Singiresu S. *Engineering Optimization - Theory and Practice*. John Wiley & Sons, New York, 3rd edition, 1996.

[76] POPOV, Egor P. *Engineering Mechanics of Solids*. Prentice-Hall, 1st edition, 1990.

[77] YATHEENDHAR, M., BELEGUNDU, A. D. Analytical shape sensitivity by implicit differentiation for general velocity fields. *Computers & Structures*, v. 46, n. 4, p. 617–623, February 1993.

[78] CHOI, Kyung K., CHANG, Kuang-Hua. A sutdy of design velocity field computation for shape optimal design. *Finite Elements in Analysis and Design*, v. 15, n. 4, p. 317–341, February 1994.

[79] GALVÃO, M. C. Ambiente baseado em nurbs para a definição de contornos em malhas de elementos finitos. Relatório de iniciação científica (proc. 95/1621-9), FAPESP, 1995.

[80] ROGERS, David F., ADAMS, J. Alan. *Mathematical Elements for Computer Graphics.* McGraw-Hill, 2nd edition, 1990.

[81] GUIMARÃES, A. C. S., FANCELLO, E. A., FEIJOÓ, G. R., FEIJOÓ, R. A. *SAFE - integrated system for strutctural finite element analysis - Version 1.0.* Laboratório Nacional de Computação Científica (LNCC), Rio de Janeiro, 1994.

[82] RAMOS, S. H. P. Visualização de resultados de engenharia estrutural em ambiente windows. Relatório de iniciação científica (proc. 95/4948-9), FAPESP, 1996.

[83] THEES, A. M. Ferramentas computacionais para um programa de análise estrutural por elementos finitos. Relatório de iniciação científica (proc. 95/3148-9), FAPESP, 1996.

[84] GUIMARÃES, A. C. S., FEIJOÓ, R. A. O sistema acdp. Relatório Técnico 027/89, Laboratório Nacional de Computação Científica (LNCC), Rio de Janeiro, Agosto 1989.

Apêndice A

Métodos de Newton e Quasi-Newton

A.1 Solução de Sistemas de Equações pelo Método de Newton

Seja $\boldsymbol{\chi}(\mathbf{v}) = \mathbf{0}$ um sistema homogêneo de equações n_{eq} não-lineares, $\boldsymbol{\chi} : \Re^{n_{eq}} \rightarrow \Re^{n_{eq}}$, onde todas χ_i $(i = 1, \ldots, n_{eq})$ têm derivadas contínuas. O método de Newton para a solução desta classe de sistemas de equações não-lineares consiste do seguinte procedimento iterativo de ponto fixo,

$$\nabla\boldsymbol{\chi}\left(\mathbf{v}^{(k)}\right)\Delta\mathbf{v}^{(k)} = -\boldsymbol{\chi}\left(\mathbf{v}^{(k)}\right) \qquad \mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + \Delta\mathbf{v}^{(k)}$$

$$\text{até que} \qquad \left\|\Delta\mathbf{v}^{(k)}\right\| \leq \xi > 0 \text{ arbitrário}$$

Demonstra-se a convergência do método para os casos onde $\det\left[\nabla\boldsymbol{\chi}(\mathbf{v})\right] \neq 0$, $\forall\mathbf{v} \neq \mathbf{0}$ [49].

A.2 Método de Otimização de Newton

O método de otimização de Newton consiste na solução do sistema de equações (2.2) a (2.6), obtido das condições de otimalidade de Karush-Kuhn-Tucker (Teorema 2.1), pelo método de Newton para equações não-lineares. No desenvolvimento a seguir, considera-se que todas as funções têm derivadas contínuas até segunda ordem e que os gradientes de todas as restrições são linearmente independentes nos pontos de solução.

Considere o problema de otimização na forma geral (1.11)

$$\begin{aligned}&\min\ f(\mathbf{p})\\ &\text{sujeita à}\\ &\qquad g_i(\mathbf{p}) \leq 0 \qquad i = 1, 2, \ldots, m\\ &\qquad h_j(\mathbf{p}) = 0 \qquad j = 1, 2, \ldots, p\end{aligned}$$

O Lagrangeano do problema é definido por (2.1),

$$L(\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{p}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{p}) + \sum_{i=1}^{p} \mu_i h_i(\mathbf{p}) = f(\mathbf{p}) + \boldsymbol{\lambda}^T\mathbf{g}(\mathbf{p}) + \boldsymbol{\mu}^T\mathbf{h}(\mathbf{p})$$

onde $\mathbf{p} \in \Re^n$ são as variáveis de projeto, sendo $\boldsymbol{\lambda} \in \Re^m$ e $\boldsymbol{\mu} \in \Re^p$ são multiplicadores de Lagrange do problema. As condições de Karush-Kuhn-Tucker determinam que L seja estacionário ($\nabla L = \mathbf{0}$) com respeito a solução $(\mathbf{p}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$, ou seja,

$$\frac{\partial L}{\partial p_j} \equiv \frac{\partial f(\mathbf{p}^*)}{\partial p_j} + \sum_{i=1}^{m} \lambda_i^* \frac{\partial g_i(\mathbf{p}^*)}{\partial p_j} + \sum_{i=1}^{p} \mu_i^* \frac{\partial h_i(\mathbf{p}^*)}{\partial p_j} = 0 \qquad j = 1, 2, \ldots, n$$

$$g_i(\mathbf{p}^*) \leq 0 \qquad i = 1, 2, \ldots, m$$

$$\lambda_i^* g_i(\mathbf{p}^*) = 0 \qquad i = 1, 2, \ldots, m$$
$$h_i(\mathbf{p}^*) = 0 \qquad i = 1, 2, \ldots, p$$
$$\lambda_i^* \geq 0 \qquad i = 1, 2, \ldots, m$$

Considerando a notação

$f(\mathbf{p}) : \Re^n \longrightarrow \Re$	Função objetivo
$\mathbf{g}(\mathbf{p}) : \Re^n \longrightarrow \Re^m$	Vetor de restrições de desigualdade
$\mathbf{G}(\mathbf{p}) : \Re^n \longrightarrow \Re^{m \times m}$	Matriz diagonal onde $G_{ii}(\mathbf{p}) = g_i(\mathbf{p})$
$\mathbf{C}(\mathbf{p}) : \Re^n \longrightarrow \Re^n$	Gradiente de $f(\mathbf{p})$, ou seja, $C_i(\mathbf{p}) = \frac{\partial f}{\partial x_i}(\mathbf{p})$
$\mathbf{A}(\mathbf{p}) : \Re^n \longrightarrow \Re^{m \times n}$	Gradiente de $\mathbf{g}(\mathbf{p})$, ou seja, $A_{ij}(\mathbf{p}) = \frac{\partial g_i}{\partial x_j}(\mathbf{p})$
$\mathbf{h}(\mathbf{p}) : \Re^n \longrightarrow \Re^p$	Vetor de restrições de igualdade
$\mathbf{L}(\mathbf{p}) : \Re^n \longrightarrow \Re^{p \times n}$	Gradiente de $\mathbf{h}(\mathbf{p})$, ou seja, $L_{ij}(\mathbf{p}) = \frac{\partial h_i}{\partial x_j}(\mathbf{p})$
$\mathbf{H}(\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \nabla^2 L$	Matriz Hessiana

tem-se o sistema de equações $\boldsymbol{\chi}(\mathbf{v}^*) = \mathbf{0}$ onde,

$$\boldsymbol{\chi}(\mathbf{v}^*) = \begin{Bmatrix} \mathbf{C}(\mathbf{p}^*) + \mathbf{A}(\mathbf{p}^*)^T \boldsymbol{\lambda}^* + \mathbf{L}^T(\mathbf{p}^*) \boldsymbol{\mu}^* \\ \mathbf{G}(\mathbf{p}^*) \boldsymbol{\lambda}^* \\ \mathbf{h}(\mathbf{p}^*) \end{Bmatrix} \qquad \mathbf{v} = \begin{Bmatrix} \mathbf{p} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu} \end{Bmatrix}$$

sendo

$$\nabla \boldsymbol{\chi}(\mathbf{v}) = \begin{bmatrix} \mathbf{H}(\mathbf{p}, \boldsymbol{\lambda}, \boldsymbol{\mu}) & \mathbf{A}^T(\mathbf{p}) & \mathbf{L}^T(\mathbf{p}) \\ \boldsymbol{\Lambda}\mathbf{A}(\mathbf{p}) & \mathbf{G}(\mathbf{p}) & \mathbf{0} \\ \mathbf{L}(\mathbf{p}) & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Aplicando o método de Newton para determinar a solução $\mathbf{v}^*$ de $\boldsymbol{\chi}(\mathbf{v}) = \mathbf{0}$ tem-se,

$$\nabla \boldsymbol{\chi}\left(\mathbf{v}^{(k)}\right) \Delta \mathbf{v}^{(k)} = -\boldsymbol{\chi}\left(\mathbf{v}^{(k)}\right)$$

$$\begin{bmatrix} \mathbf{H}\left(\mathbf{p}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\mu}^{(k)}\right) & \mathbf{A}^T\left(\mathbf{p}^{(k)}\right) & \mathbf{L}^T\left(\mathbf{p}^{(k)}\right) \\ \boldsymbol{\Lambda}^{(k)}\mathbf{A}\left(\mathbf{p}^{(k)}\right) & \mathbf{G}\left(\mathbf{p}^{(k)}\right) & \mathbf{0} \\ \mathbf{L}\left(\mathbf{p}^{(k)}\right) & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \\ \boldsymbol{\lambda}^{(k+1)} - \boldsymbol{\lambda}^{(k)} \\ \boldsymbol{\mu}^{(k+1)} - \boldsymbol{\mu}^{(k)} \end{Bmatrix} =$$

$$= -\begin{Bmatrix} \mathbf{C}\left(\mathbf{p}^{(k)}\right) + \mathbf{A}\left(\mathbf{p}^{(k)}\right)^T \boldsymbol{\lambda}^{(k)} + \mathbf{L}^T\left(\mathbf{p}^{(k)}\right) \boldsymbol{\mu}^{(k)} \\ \mathbf{G}\left(\mathbf{p}^{(k)}\right) \boldsymbol{\lambda}^{(k)} \\ \mathbf{h}\left(\mathbf{p}^{(k)}\right) \end{Bmatrix}$$

$$\Rightarrow \begin{bmatrix} \mathbf{H}\left(\mathbf{p}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\mu}^{(k)}\right) & \mathbf{A}^T\left(\mathbf{p}^{(k)}\right) & \mathbf{L}^T\left(\mathbf{p}^{(k)}\right) \\ \boldsymbol{\Lambda}^{(k)}\mathbf{A}\left(\mathbf{p}^{(k)}\right) & \mathbf{G}\left(\mathbf{p}^{(k)}\right) & \mathbf{0} \\ \mathbf{L}\left(\mathbf{p}^{(k)}\right) & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{p}^{(k+1)} - \mathbf{p}^{(k)} \\ \boldsymbol{\lambda}^{(k+1)} \\ \boldsymbol{\mu}^{(k+1)} \end{Bmatrix} =$$

$$= -\begin{Bmatrix} \mathbf{C}\left(\mathbf{p}^{(k)}\right) \\ \mathbf{0} \\ \mathbf{h}\left(\mathbf{p}^{(k)}\right) \end{Bmatrix} \tag{A.1}$$

A solução do sistema linear (A.1) fornece os multiplicadores de Lagrange para a próxima iteração $\Delta \mathbf{p}^{(k)}$, sendo que, na versão original do método, $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + \Delta \mathbf{p}^{(k)}$. Contudo, melhores resultados são obtidos a solução do sistema (A.1) for considerada a etapa de determinação da direção de busca linear da iteração, sendo seguida por um procedimento de busca linear conveniente de acordo com a função de decréscimo adotada. Nesse caso, $\mathbf{p}^{(k+1)} = \mathbf{p}^{(k)} + t_k \Delta \mathbf{p}^{(k)}$, onde t_k é o passo determinado na busca linear.

A.3 Métodos Quasi-Newton

Foram desenvolvidos vários procedimentos de aproximação numérica da Hessiana a partir de informações de primeira ordem. Tais procedimentos usam o gradiente do Lagrangeano em dois pontos e a mudança no projeto nos algoritmos de atualização da aproximação da matriz hessiana $\mathbf{H}$. Os métodos que utilizam estes procedimentos são chamados métodos quasi-Newton.

A.3.1 Regra BFGS de Aproximação da Hessiana

A regra BFGS com modificação de Powell (para evitar problemas de singularidade e indeterminação na regra BFGS básica) é desenvolvida a seguir.

Determinam-se os seguintes vetores e escalares,

$\mathbf{s}^{(k)} = t_k \Delta \mathbf{p}^{(k)}$	vetor
$\mathbf{z}^{(k)} = \mathbf{H}^{(k)} \mathbf{s}^{(k)}$	vetor
$\mathbf{y}^{(k)} = \nabla L\left(\mathbf{p}^{(k+1)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\mu}^{(k)}\right) - \nabla L\left(\mathbf{p}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\mu}^{(k)}\right)$	vetor
$\vartheta_1 = \mathbf{s}^{(k)} \cdot \mathbf{y}^{(k)}$	escalar
$\vartheta_2 = \mathbf{s}^{(k)} \cdot \mathbf{z}^{(k)}$	escalar
$\theta = 1$ se $\vartheta_1 \geq 0.2\vartheta_2$, senão $\theta = 0.8\vartheta_2 / (\vartheta_2 - \vartheta_1)$	escalar
$\mathbf{w}^{(k)} = \theta \mathbf{y}^{(k)} + (1 - \theta)\, \mathbf{z}^{(k)}$	vetor
$\vartheta_3 = \mathbf{s}^{(k)} \cdot \mathbf{w}^{(k)}$	escalar

que são, em seguida, aplicados na atualização da aproximação $\mathbf{Q}$ da matriz hessiana $\mathbf{H}$ na iteração $k+1$ da seguinte forma:

$$\mathbf{H}^{(k+1)} \simeq \mathbf{Q}^{(k+1)} = \mathbf{Q}^{(k)} + \frac{1}{\vartheta_3} \mathbf{w}^{(k)} \left(\mathbf{w}^{(k)}\right)^T - \frac{1}{\vartheta_2} \mathbf{z}^{(k)} \left(\mathbf{z}^{(k)}\right)^T \tag{A.2}$$

Apêndice B

Interpolações

Métodos de busca linear de ordem zero[1] podem exigir muitas avaliação dos funcionais para determinar o passo na direção de busca. Portanto, podem ser ineficientes em aplicações práticas. Ao invés de fazer uma série de estimativas das funções pode-se fazer uma interpolação polinomial de um número limitado de pontos. Esta idéia pode ser aplicada tanto na determinação do mínimo da função objetivo na direção de busca corrente quanto na determinação de um passo viável nessa direção. Apesar de ser possível utilizar polinômios de maior ordem, em muitos casos são obtidos resultados satisfatórios em termos de precisão e eficiência com polinônios de segunda ordem apenas.

B.1 Interpolação Quadrática

Pode-se interpolar uma função $\varphi(t)$ de uma variável por uma curva quadrática conhecendo seu valor em dois pontos e sua derivada num deles ou seu valor em três pontos. Considere a seguinte curva de segundo grau na variável t,

$$\tilde{\varphi}(t) = a_0 + a_1 t + a_2 t^2 \tag{B.1}$$

Caso 1 No primeiro caso, são conhecidos $\varphi(0)$, $\varphi'(0)$ e $\varphi(t_1)$. Logo,

$$a_0 = \varphi(0) \qquad a_1 = \varphi'(0) \qquad a_2 = \varphi(t_1)/t_1^2 - \varphi'(0)/t2 - \varphi(0)/t_1^2 \tag{B.2}$$

Caso 2 No segundo, são conhecidos $\varphi(0)$, $\varphi(t_1)$ e $\varphi(t_2)$. Dessa forma,

$$a_0 = \varphi(0) \qquad a_1 = \frac{\varphi(t_1)-\varphi(0)}{t_1} - a_2 t_1 \qquad a_2 = \frac{1}{(t_2 - t_1)}\left[\frac{\varphi(t_2)-\varphi(0)}{t_2} - \frac{\varphi(t_1)-\varphi(0)}{t_1}\right] \tag{B.3}$$

B.2 Busca Linear com Interpolação Quadrática

Considerando o **Caso 1** na interpolação dos funcionais, a interpolação quadrática pode ser usada para minimizar a funcão objetivo ou para determinar um passo viável na direção de busca. Assume-se nas seções seguintes que a direção de busca é uma direção viável de decréscimo.

[1] Métodos de busca linear de ordem zero determinam uma nova estimativa como uma porcentagem da anterior.

B.2.1 Minimização da Função Objetivo

Deve-se determinar o passo t_k que respeite a condição de decréscimo de Armijo $f\left(\mathbf{p}^{(k)}+t_k\mathbf{d}\right) \leq f\left(\mathbf{p}^{(k)}\right) + \eta\mathbf{d}^T\nabla f^{(k)}t_k$, para uma direção de busca $\mathbf{d}$. Suponha que se conheça uma estimativa inicial de passo t_1. Faz-se $\varphi(t) = f\left(\mathbf{p}^{(k)}+t\mathbf{d}\right)$, seguindo o algoritmo:

1. Determina-se $\varphi(0)$, $\varphi'(0)$ e $\varphi(t_1)$.
2. Calculam-se os coeficientes de (B.1) de acordo com a (B.2).
3. Se $f\left(\mathbf{p}^{(k)}+\bar{t}\mathbf{d}\right) \leq f\left(\mathbf{p}^{(k)}\right) + \eta\mathbf{d}^T\nabla f^{(k)}\bar{t}$, sendo $\bar{t} = -\dfrac{1}{2a_2}a_1$ então $t_k = \bar{t}$ e termina o processo.
4. Senão $t_1 = \bar{t}$ e $\varphi(t_1) = f\left(\mathbf{p}^{(k)}+\bar{t}\mathbf{d}\right)$. Retorna ao passo 1.

B.2.2 Determinação de um Passo Viável

Deseja-se determinar o passo t_k que respeite a restrição $g_i \leq 0$. Suponha que seja conhecida uma estimativa inicial de passo t_1 inviável, ou seja, $g_i\left(\mathbf{p}^{(k)}+t_1\mathbf{d}\right) > 0$. Faz-se $\varphi(t) = g_i\left(\mathbf{p}^{(k)}+t\mathbf{d}\right)$, seguindo o algoritmo:

1. Determina-se $\varphi(0)$, $\varphi'(0)$ e $\varphi(t_1)$.
2. Calculam-se os coeficientes de (B.1) de acordo com a (B.2).
3. Se $g_i\left(\mathbf{p}^{(k)}+\bar{t}\mathbf{d}\right) \leq 0$, sendo $\bar{t}$ a menor raiz positiva de φ, então $t_k = \bar{t}$ e termina o processo.
4. Senão $t_1 = \bar{t}$ e $\varphi(t_1) = g_i\left(\mathbf{p}^{(k)}+\bar{t}\mathbf{d}\right)$. Retorna ao passo 1.