UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA MECÂNICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA

GERAÇÃO E OTIMIZAÇÃO DE MALHAS ESTRUTURADAS SOBRE DOMÍNIOS BIDIMENSIONAIS ARBITRÁRIOS

Dissertação apresentada

à Universidade Federal de Uberlândia por:

MÁRCIO RICARDO PIVELLO

como parte dos requisitos para obtenção do titulo de Mestre em

Engenharia Mecânica

Aprovada por:

Prof. Dr. Carlos Roberto Ribeiro (UFU) - Orientador
Prof. Dr. João Batista Campos Silva (UNESP – Ilha Solteira)
Prof. Dr. Aristeu da Silveira Neto (UFU)

Uberlândia, 21 de setembro de 2001.

FICHA CATALOGRÁFICA

Elaborado pelo Sistema de Bibliotecas da UFU / Setor de Catalogação e Classificação

P693g Pivello, Márcio Ricardo, 1975-
Geração e otimização de malhas estruturadas sobre domínios bidimensionais arbitrários / Márcio Ricardo Pivello. - Uberlândia, 2004.
79f. : il.
Orientador: Carlos Roberto Ribeiro.
Dissertação (mestrado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Mecânica.
Inclui bibliografia.
1. Equações diferenciais parciais – Soluções numéricas - Teses. 2. Geração numérica de malhas (Análise numérica) -Teses. 3. Método dos elementos finitos - Teses. 4. Engenharia mecânica - Teses. I. Ribeiro, Carlos Roberto. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Mecânica. III. Título.

517.95 (043.3)

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA MECÂNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
Av . João Naves de Ávila, 2160 - Campus Santa Mônica - Uberlândia – MG - 38400-902
Fone: 0XX3432394149 - FAX: 0XX3432394282

ALUNO: MÁRCIO RICARDO PIVELLO

NÚMERO DE MATRÍCULA: 5002609-4

ÁREA DE CONCENTRAÇÃO: Mecânica dos Sólidos e Vibrações

PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA: NÍVEL MESTRADO

TÍTULO DA DISSERTAÇÃO:

"Geração e Otimização de Malhas Estruturadas sobre Domínios Bidimensionais Arbitrários".

ORIENTADOR: Prof. Dr. Carlos Roberto Ribeiro

A Dissertação foi **APROVADA** em reunião pública, realizada no Anfiteatro do Bloco X do Campus Santa Mônica, em 21 de setembro de 2001, às 09:00 horas, com a seguinte Banca Examinadora:

NOME		ASSINATURA
Carlos Roberto Ribeiro, Prof. Dr.	UFU	
João Batista Campos Silva, Prof. Dr.	UNESP/FEIS	
Aristeu da Silveira Neto, Prof. Dr.	UFU	

Uberlândia, 21 de setembro de 2001.

a meus pais, Olívio e Neuza,

a minhas irmãs, Sabrina e Luciana,

à minha namorada,Tatiana.

AGRADECIMENTOS

Ao professor Dr. Carlos Roberto Ribeiro, pelo apoio e incentivo,

A todos os colegas estudantes e professores da FEMEC, que diretamente ou não, me apoiaram e incentivaram durante o desenvolvimento da pesquisa, em especial aos amigos Danuza e Patrick,

Ao apoio financeiro oferecido pela CAPES – Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior.

SUMÁRIO

LISTA DE SÍMBOLOS

Letras Gregas

α, β, γ	-	Coeficientes das equações de Laplace, tendo como variáveis independentes ξ e η
Δt_k	-	Variação na coordenada paramétrica t do nó k
Δg_k	-	Variação na perturbação imposta à coordenada paramétrica t do nó k
ε	-	Erro cometido na aproximação da solução de um sistema de equações lineares; distorção de um elemento de discretização de um domínio qualquer
Γ	-	Fronteira do domínio Ω
$\Gamma_S, \Gamma_E, \Gamma_N, \Gamma_W$	-	Faces componentes da fronteira Γ (em referência a sul, leste, norte e oeste)
η	-	Ordenada de um ponto num sistema genérico de coordenadas curvilíneas
θ	-	Ordenada de um ponto num sistema de coordenadas polares
τ	-	Coordenada paramétrica de um ponto, normalizada para o intervalo [0..1]
ξ	-	Abcissa de um ponto num sistema genérico de coordenadas curvilíneas
$\nabla^2, \tilde{\nabla}^2$	-	Operador laplaciano nos sistemas de coordenadas cartesiano e curvilíneo genérico, respectivamente
Ω	-	Domínio a ser discretizado

Letras Latinas

$A_N, A_E, A_N, A_S, A_{NE}, A_{NW}, A_{SE}, A_{SW}, A_P$	-	Coeficientes de ponderação usados na resolução das equações de geração da malha por diferenças finitas centrais
a,b,c	-	Coeficientes das equações de Laplace, tendo como variáveis independentes x e y
a, b, c, d	-	Parâmetros que quantificam a deformação de um elemento de malha

B_i - Coeficientes do polinômio interpolador

d_f, d_t - Diferenciais no sistema cartesiano e curvilíneo genérico, respectivamente

E - Módulo de elasticidade

I - Índice

f - Função que relaciona os sistemas de coordenadas cartesiano e curvilíneo genérico

F - Vetor de componentes da coordenada paramétrica de um ponto, numa curva de interpolação

F'(t) - Derivada do vetor de componentes da coordenada paramétrica de um ponto em relação a t

f_{xx}, f_{yy}, f_x, f_y - Derivadas parciais de f em relação a x e a y

f_1, f_2, ... - Faces de subdomínios internas ao domínio

G - Vetor de coordenadas e inclinações dos pontos localizados nas extremidades de um intervalo, sobre uma curva de interpolação

g_k - Parte decimal da coordenada paramétrica do nó k; perturbação introduzida nesta coordenada

I_{2h}^{h} - Prolongação (transferência de resultados da malha grosseira 2h para a malha refinada h)

J - Jacobiano da transformação do sistema cartesiano para o sistema curvilíneo genérico

k - Número de identificação de um nó; parte inteira de sua coordenada paramétrica

m - Número de elementos localizados sobre uma face de um domínio / subdomínio quadrilateral

M - Matriz das coordenadas paramétricas dos pontos que definem uma curva de interpolação

M - Momento fletor

P(t) - Coordenadas cartesianas do ponto de coordenada paramétrica t

P'(t) - Inclinação da curva de interpolação no ponto t

P''(t) - Raio de curvatura da curva de interpolação no ponto t

P' - Vetor contendo a inclinação da curva em cada ponto

r - Abcissa de um ponto num sistema de coordenadas polares

t	-	Intervalo de interpolação; coordenada paramétrica de um ponto
Sa, Sb, Sc, Sd	-	Parcelas do erro quadrático de uma malha
u	-	Variável dependente num sistema curvilíneo genérico
$\tilde{u}$	-	Aproximação para a solução de um sistema de equações lineares
u_N, u_E, u_N, u_S, u_{NE}, u_{NW}, u_{SE}, u_{SW}, u_P	-	Pontos de discretização na resolução das equações de geração da malha por diferenças finitas centrais
$u_{\xi\xi}$, $u_{\eta\eta}$, $u_{\xi\eta}$, u_ξ, u_η	-	Derivadas parciais de u em relação a ξ e η
v_1, v_2, ...	-	Vértices de subdomínios internos ao domínio
w	-	Fator de ponderação do método SOR
x,y	-	Coordenadas de um ponto no sistema cartesiano

LISTA DE FIGURAS

Pivello, M. R., 2001, "Geração e Otimização de Malhas Estruturadas Sobre Domínios Bidimensionais Arbitrários", Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia, MG.

RESUMO

Este trabalho apresenta uma metodologia para geração e otimização de malhas estruturadas sobre domínios bidimensionais arbitrários. O método consiste em dividir o domínio em áreas ou subdomínios quadrilaterais e discretizá-los individualmente para depois conectar a malha entre áreas vizinhas. Uma formulação paramétrica da interpolação por spline cúbica foi empregada tanto na caracterização do domínio como na aplicação das condições de contorno, e a geração da malha foi feita através de um sistema acoplado de equações elípticas de Laplace. O processo de otimização consiste em realocar os nós de fronteira de modo a se obter elementos quadrados, tendo como variáveis de projeto as coordenadas paramétricas destes nós. Estas servem como condições de contorno de Dirichlet para o sistema gerador de malhas, que realocará os nós internos ao domínio. Para garantir a continuidade da malha, impõe-se o mesmo conjunto de condições de contorno a faces comuns a áreas adjacentes. Quando a forma dos subdomínios não deve ser modificada, aplicam-se condições de Dirichlet. Quando permite-se esta alteração, condições de Neumann são usadas para garantir uma transição mais suave da malha entre os subdomínios adjacentes e melhor adaptada ao domínio. Esta opção mostrou-se particularmente interessante quando aplicada a geometrias típicas de CFD, possibilitando a concentração da malha em certas regiões, e mantendo ainda as propriedades de malha estruturada. Foram testados dois métodos de busca, um por Algoritmos Genéticos e outro por gradiente, sendo que somente este último mostrou-se viável do ponto de vista do custo computacional. Os resultados confirmam a rapidez do método, permitindo que seja empregado em problemas que exigem remalhagem durante o processo iterativo.

Palavras chave: Malhas estruturadas, otimização, domínios bidimensionais arbitrários.

Pivello, M. R., 2001, "Generation and Optimization of Structured Meshes on Generic Two Dimensional Domains", M. Sc. Dissertation, Universidade Federal de Uberlândia, Uberlândia, MG.

ABSTRACT

The aim of this work is to develop an algorithm for optimizing structured meshes generated over generic two dimensional domains, by splitting complex shapes into a collection of quadrilateral sub domains, which are then meshed by a Laplacian elliptic mesh generator, and the mesh smoothness between adjacent areas is imposed later. A parametric formulation for cubic spline interpolation is used both to define the domain and apply the boundary conditions. In order to get square elements, a nodal repositioning algorithm is used, and two search methods are tested: a genetic algorithm and a gradient based method. The algorithm reallocates the boundary nodes and uses their parametric coordinates as Dirichlet boundary conditions to solve the Laplacian generator system and update the interior nodes coordinates. To get a continuous mesh between adjacent areas, the user gives the same boundary conditions to the nodes located on the common boundaries. If the original geometry of the areas is important to the problem, and must remain unchanged, Dirichlet conditions are applied. Otherwise, the shape of the subdomains is changed so that a smoother mesh is obtained. To ensure this smoothness, Neumann conditions are aplied to the modified boundaries. This method has shown very interesting results when applied to domains that are often found in CFD problems. Because of the large number of design variables, the genetic algorithm had a poor time performance, and could be applied only to simple quadrilateral domains. On the other hand, the gradient based method proved to be fast enough to be applied to problem where remeshing during the iterative process is necessary. The results also show that this method may be adapted to work as a shape optimization algorithm.

Keywords: structured meshes, optimization, general shape two dimensional domains

Capítulo 1

Introdução

O método dos elementos finitos é um dos métodos de discretização mais usados na resolução de problemas de engenharia, valendo-se geralmente de malhas não estruturadas. A preferência por este tipo de malha vem de sua melhor adaptação a geometrias complexas, fazendo com que sejam empregadas até mesmo em campos onde os problemas são predominantemente analisados com malhas estruturadas, como CFD, por exemplo. A principal diferença entre malhas estruturadas e não estruturadas é que nas primeiras qualquer nó interior ao domínio deve ser compartilhado sempre pelo mesmo número de elementos, o que não ocorre com malhas não estruturadas, permitindo-lhes melhor adaptação a geometrias complexas. Apesar disso, são mais onerosas na resolução do problema físico subjacente.

Uma condição necessária, embora não suficiente para que a condição de malha estruturada seja satisfeita é que a geometria seja quadrilateral e apresente o mesmo número de elementos em faces opostas, o que é difícil de se conseguir para domínios complexos. Para resolver este problema, este trabalho propõe uma metodologia para geração e otimização de malhas estruturadas em domínios bidimensionais arbitrários através da decomposição do domínio em subdomínios quadrilaterais. Sobre cada subdomínio gera-se uma malha estruturada usando-se um sistema de equações elípticas de Laplace e depois impõe-se a continuidade da malha entre áreas adjacentes.

Os métodos de otimização de malhas podem ser divididos em dois grandes grupos, um que mantém constante o número de elementos do modelo e outro que altera este número. São chamados de métodos h quando alteram a quantidade de elementos no modelo e métodos r quando realocam os nós da malha.

Os métodos h são normalmente aplicados a malhas não estruturadas. Steve Owen (Owen, 1997) faz uma rápida revisão sobre alguns destes métodos, relacionando as técnicas de refinamento à forma dos elementos empregados na malha. Para elementos triangulares, o procedimento básico consiste em gerar novos elementos pela bisseção de suas faces ou simplesmente pela introdução de novos nós no interior dos elementos. Para elementos quadrilaterais, os métodos podem ser indiretos, se estes elementos forem obtidos a partir de uma malha triangular, ou diretos, se forem gerados diretamente na forma quadrilateral. Shimada, Liao e Itoh (Shimada et al., 1998) partem de uma triangularização de Delaunay construída a partir de células quadradas alocadas sobre o domínio, segundo um vetor de direcionalidade definido pelo usuário. A malha triangular é convertida para uma malha

predominante ou totalmente quadrangular que satisfaça a direcionalidade original. Bern e Epstein (Bern e Epstein, 1997) usam empacotamento por círculos para gerar malhas quadrilaterais para domínios poligonais. A idéia do método é, antes da geração da malha, preencher o domínio com círculos alocados próximos uns dos outros, de modo que as regiões vazias estejam cercadas por 3 ou 4 círculos. A malha é então construída fixando-se os vértices dos elementos nos centros dos círculos, nos pontos de tangência e no interior das regiões vazias.

Ao contrário dos métodos h, os métodos r têm como característica principal não alterar a quantidade de elementos do modelo. Seu objetivo é realocar os nós da malha de modo a minimizar a distorção dos elementos. Um exemplo simples consiste em realocar um nó fazendo com que sua posição seja a média das posições dos nós vizinhos. Outros métodos baseiam-se em simular propriedades físicas tais como forças de atração e repulsão entre nós para buscar a posição ótima. Cheng, por exemplo, associa a distorção do elemento ao seu peso e simula forças de atração gravitacional de modo que elementos distorcidos tendem a atrair os nós vizinhos para o seu centróide (Cheng, 1993). Em (Lohner et al.,1986), os autores simulam a força entre nós vizinhos como sistemas de molas interagindo entre si. Shimada (Shimada,1997) trata os nós como centros de esferas que se reposicionam para manter o equilíbrio. Samareh apresenta um resumo dos principais avanços e problemas para geração de malhas em problemas de otimização multidisciplinar, onde alguns métodos de realocação nodal são analisados. São destacados os trabalhos de Batina (Batina, 1989) e Crumpton e Giles (Crumpton e Giles, 1997). O primeiro propõe que as arestas dos elementos sejam modeladas como molas, cuja rigidez é inversamente proporcional ao seu comprimento. O segundo trabalho se concentra em problemas com grandes perturbações na malha, e propõe uma formulação baseada na equação da condução de calor, onde a condutividade térmica é inversamente proporcional à área (ou volume, em 3D) do elemento.

Estes métodos são normalmente aplicados a malhas não estruturadas, embora a princípio também possam ser aplicados a malhas estruturadas. Porém, a conectividade regular deste tipo de malha permite que outros algoritmos sejam aplicados, diminuindo o esforço computacional. Alguns destes métodos utilizam métodos de geração elípticos para otimizar malhas geradas algebricamente, usando as coordenadas dos nós como ponto de partida (Shanmugasundaram et al., 1997). Para diminuir o esforço computacional, pode-se utilizar um esquema de multigrelha, onde o sistema gerador é resolvido na malha grosseira e a solução é então interpolada para a malha fina (Mastin, 1995).

Um ponto tão importante quanto o método de otimização utilizado, é a definição do critério de malha ótima. Estes critérios geralmente não dependem do método de geração ou otimização empregados, mas sim do problema físico a ser estudado. Em problemas estruturais,

por exemplo, é comum estimar-se o erro pela energia de distorção ou energia total da malha (Cheng, 1993), enquanto em problemas transientes pode-se levar em conta também o movimento dos nós durante o processo para se analisar a deformação da malha.

Percebe-se então a dificuldade de se desenvolver um critério genérico de malha ótima, que forneça resultados satisfatórios para problemas de diferentes áreas. Porém, segundo os trabalhos de Ribeiro (Ribeiro, 2000) e Berle (Berle, 1998), um bom ponto de partida seria a regularidade geométrica dos elementos, uma vez que a solução numérica do problema subjacente tende a ser mais precisa se empregada uma malha uniforme. Zhou e Shimada (Zhou et al., 2000), embora trabalhando com malhas não estruturadas, também buscam a regularidade da malha por realocação nodal segundo critérios puramente geométricos.

O critério empregado neste trabalho também é puramente geométrico. Porém, por utilizar malhas estruturadas, a otimização da malha é restrita ao reposicionamento dos nós. Para que o gerador elíptico possa ser empregado, domínios complexos, ou seja, de forma não quadrilateral, devem ser decompostos pelo usuário em subdomínios quadrilaterais, que serão discretizados independentemente.

Para garantir a continuidade da malha no caso de domínios complexos, o usuário tem as opções de alterar ou não a forma das faces comuns dos subdomínios fornecidos ao programa como dados de entrada durante o processo de otimização, quando estas não participarem da definição da fronteira do domínio. Para o caso de manter a geometria inicial inalterada, impõem-se condições de Dirichlet. Este método via de regra é mais rápido que o segundo, que trata as faces internas ao domínio como isotermas. Neste caso, a continuidade nas faces comuns é obtida impondo-se condições de Neumann às faces comuns, garantindo uma transição suave entre subdomínios vizinhos. Este método, se mal empregado, pode recair sobre a principal limitação quanto ao uso de equações de Laplace na geração de malhas, que é a concentração de linhas equipotenciais próximas às quinas, afastando-as dos cantos. Porém, configurando os subdomínios de modo adequado, pode-se contornar esta limitação e conseguir efeitos de concentração de malha tais como os conseguidos pelo método de refinamento normalmente aplicado a malhas não estruturadas, sem que seja necessário aumentar o número de nós. Do mesmo modo que se concentram elementos triangulares nas malhas não estruturadas, pode-se concentrar subdomínios em regiões que precisam de refinamento.

O baixo tempo de cálculo necessário para gerar malhas ótimas sobre domínios complexos sugere a aplicação do método a problemas onde a remalhagem do domínio durante o processo iterativo seja necessária. Uma vez gerada a malha ótima, espera-se que durante algumas iterações seja possível aproveitar sua configuração para realocar somente os nós referentes à parte modificada do domínio, já que, para um domínio com partes móveis, a

mudança de posição entre duas iterações sucessivas deve ser menor que o comprimento de um elemento. Assim, definindo-se algum critério de tolerância é possível reduzir o número de otimizações de malha durante a resolução do problema, o que reduziria consideravelmente o tempo de cálculo.

A apresentação do trabalho segue a ordem de construção de uma malha para um domínio genérico. O capítulo 2 mostra a formulação paramétrica do algoritmo de interpolação por splines cúbicas, usado tanto na definição do domínio como na aplicação das condições de contorno. O capítulo 3 trata da geração de malhas estruturadas sobre domínios de forma irregular usando-se sistemas de equações diferenciais elípticas de Laplace através da técnica de mudança de domínio. O capítulo 4 apresenta a descrição do algoritmo, desde a caracterização paramétrica do domínio até a obtenção da malha ótima. Os resultados apresentados no capítulo 5 mostram alguns domínios discretizados, a configuração original e final dos subdomínios, bem como o tempo de cálculo necessário para cada caso. Os resultados obtidos pelos dois métodos de continuidade da malha são comparados, analisando-se principalmente suavidade na transição entre subdomínios e o tempo de cálculo. As conclusões a que se chegaram ao final do trabalho são apresentadas no capítulo 6, e as referências bibliográficas utilizadas são listadas no capítulo 7.

Capítulo 2

Interpolação por Spline Cúbica

2.1 Parametrização de Curvas através de Interpolação por Spline Cúbica

O uso de curvas definidas parametricamente na representação de domínios arbitrários é preferível em relação à formulação não paramétrica devido à independência dos eixos coordenados, uma vez que a variável independente está localizada sobre a curva a ser representada. Esta formulação também facilita a utilização de sistemas de coordenadas coincidentes com a fronteira, o que será necessário neste trabalho. Além disso, a formulação paramétrica reduz o número de variáveis independentes do problema, diminuindo assim seu custo computacional. O domínio da variável é dado pelo comprimento da curva, mas pode ser normalizado para intervalos de comprimentos constantes, definidos pelo usuário, de modo a diminuir o esforço computacional necessário para a caracterização da geometria. Além disso, a independência em relação ao sistema referencial facilita a aplicação de transformações algébricas como rotação, escalonamento, reflexão entre outros, geralmente presentes em algoritmos de computação gráfica.

O modo mais simples de se representar uma curva definida por um conjunto de pontos é por segmentos de reta, embora na maioria das vezes não seja o modo mais indicado, uma vez que curvas com pequenos raios de curvatura necessitariam de um grande número de segmentos para representá-las. Por isso, o uso de curvas não lineares, geralmente polinômios, é a prática mais usual na representação gráfica de curvas. Estas, porém, não podem ter grau muito elevado para evitar o *overfitting*, que faz com que a curva represente perfeitamente a curva nas regiões próximas aos pontos definidos pelo usuário, mas seja altamente oscilatória nos intervalos entre estes pontos. Para minimizar estes problemas, geralmente são utilizados polinômios interpoladores de terceiro grau, pois é o polinômio de menor grau a permitir pontos de inflexão no domínio, além de ser uma curva de continuidade C^2, o que é suficiente para a maioria das aplicações em problemas de engenharia, e neste caso em particular. Duas das formulações mais conhecidas são o polinômio interpolador de Lagrange e as splines. A grande diferença entre os dois métodos é que o primeiro utiliza um único polinômio para representar a curva, enquanto a spline usa um polinômio para cada intervalo definido entre dois pontos consecutivos, o que faz com que represente melhor o domínio.

A figura 2.1a mostra uma coleção de n pontos que servem de suporte para a construção da curva da figura 2.1b. Para gerar esta curva através de uma spline cúbica, geram-se (n-1) polinômios de 3° grau, cada um definido entre 2 pontos consecutivos P_i e P_{i+1}. Sua formulação

matemática, baseada na equação da deflexão da viga de Euler – Bernoulli, é desenvolvida de tal forma que lhe garante algumas características importantes como curva interpoladora (Rogers, 1990):

1. Tem continuidade C^2 entre intervalos adjacentes;
2. Reduz instabilidade numérica decorrente do uso de curvas de interpolação de grau elevado;
3. O polinômio de 3° grau é o mínimo necessário para que haja pontos de inflexão no domínio.

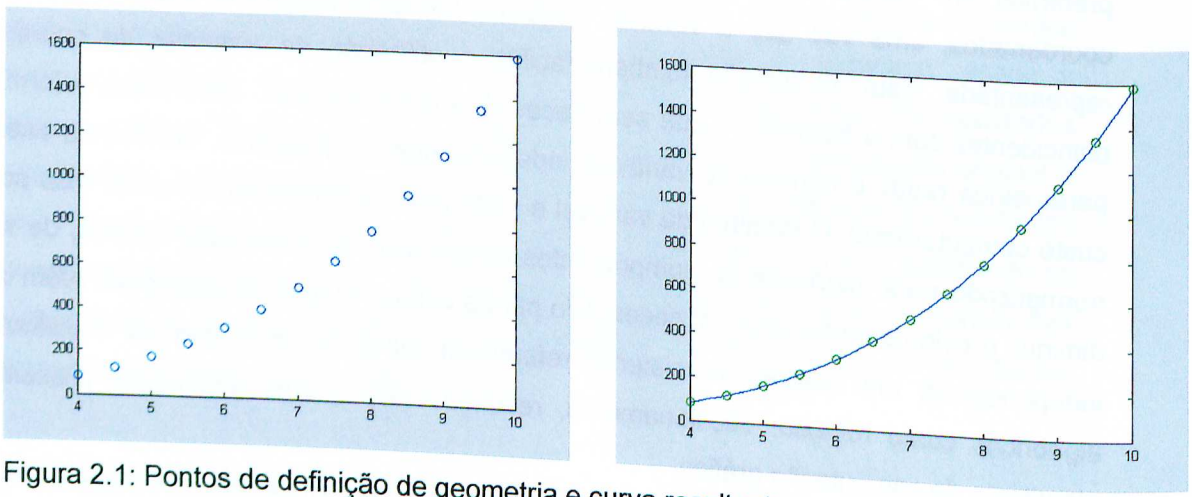


Figura 2.1: Pontos de definição de geometria e curva resultante.

Estas características advêm da analogia feita durante sua formulação. Admitindo que a spline represente uma viga no regime elástico simplesmente apoiada sobre os n pontos que definem a curva, o momento aplicado sobre um ponto qualquer terá uma relação linear com x. Neste caso a equação da deflexão, que é dada por

$$y'' = \frac{\boldsymbol{M}(x)}{EI} \tag{2.1}$$

pode ser escrita como

$$y'' = \frac{A \cdot x + B}{EI} \tag{2.2}$$

e integrando-se duas vezes obtém-se um polinômio de 3° grau que descreverá a curva em cada intervalo entre dois pontos consecutivos. Isso já garante as características 2 e 3. Além disso, mesmo que a curva seja definida por mais de 2 pontos, como geralmente ocorre, como a

viga está no regime elástico por hipótese, haverá continuidade no raio de curvatura nos pontos de apoio, o que garante a 1ª característica.

Um ponto genérico *P(t)* pertencente ao intervalo $t_i \le t \le t_{i+1}$ pode ser descrito então pela equação 2.3:

$$P(t) = B_1 + B_2 \cdot t + B_3 \cdot t^2 + B_4 \cdot t^3, \qquad t_i \le t \le t_{i+1} \tag{2.3}$$

Logo, para determinar as coordenadas do ponto *P* é necessário determinar os coeficientes B_i da equação acima. Note que, para o caso bidimensional, *P(t)* tem componentes *x(t)* e *y(t)* e, consequentemente, *B* tem dimensões (n x 2), uma vez que a coordenada paramétrica *t* é escalar.

Para determinar os coeficientes B_i são necessárias 4 equações. As duas primeiras são obviamente obtidas pela aplicação da eq. (2.3) nas extremidades do intervalo, onde se conhece o valor de *P(t)*. As outras são obtidas pelas derivadas destas aplicadas aos mesmos pontos.

$$P(t_i) = P_i = B_1 + B_2 \cdot t_i + B_3 \cdot t_i^2 + B_4 \cdot t_i^3 \tag{2.4a}$$

$$P(t_{i+1}) = P_{i+1} = B_1 + B_2 \cdot t_{i+1} + B_3 \cdot t_{i+1}^2 + B_4 \cdot t_{i+1}^3 \tag{2.4b}$$

$$P'(t_i) = P_i' = B_2 + 2 \cdot B_3 \cdot t_i + 3 \cdot B_4 \cdot t_i^2 \tag{2.4c}$$

$$P'(t_{i+1}) = P_{i+1}' = B_2 + 2 \cdot B_3 \cdot t_{i+1} + 3 \cdot B_4 \cdot t_{i+1}^2 \tag{2.4d}$$

Como os polinômios são definidos em cada intervalo, pode-se fazer $t_i=0$ sem que haja perda na precisão dos resultados. Desta forma, os valores de *{B}* são dados por:

$$B_1 = P_i \tag{2.5a}$$

$$B_2 = P_i' \tag{2.5b}$$

$$B_3 = 3 \cdot \frac{P_{i+1} - P_i}{t_{i+1}^2} - 2 \cdot \frac{P_i'}{t_{i+1}} - \frac{P_{i+1}'}{t_{i+1}} \tag{2.5c}$$

$$B_4 = 2 \cdot \frac{P_i - P_{i+1}}{t_{i+1}^3} + \frac{P_i'}{t_{i+1}^2} + \frac{P_{i+1}'}{t_{i+1}^2} \tag{2.5d}$$

Esta equação é usada para descrever um único intervalo. Porém, a curva é descrita pela união de todos os intervalos, apresentando, portanto, pontos internos, como no caso da figura 2.2.

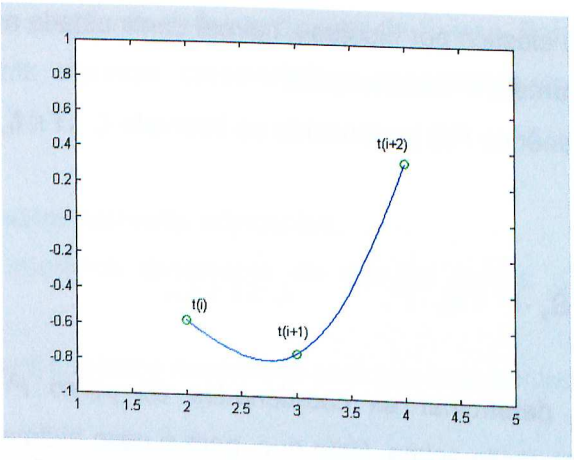


Figura 2.2 Spline formada por 3 pontos não co-lineares.

Tomando-se os 3 pontos t_i, t_{i+1} e t_{i+2} e lembrando-se da condição de continuidade de curvatura entre intervalos, obtém-se o raio de curvatura para o ponto t_{i+1} a partir da eq. (2.6).

$$P''(t) = 2 \cdot B_3 + 6 \cdot B_4 \cdot t \tag{2.6}$$

Avaliando-se esta equação nos intervalos $[t_i..t_{i+1}]$ e $[t_{i+1}..t_{i+2}]$ em torno do ponto t_{i+1}, tem-se:

$$P''(t_{i+1}) = 2 \cdot B_3 + 6 \cdot B_4 \cdot t_{i+1}, \qquad 0 \le t \le t_{i+1} \tag{2.7a}$$

$$P''(0) = 2 \cdot B_3, \qquad 0 \le t \le t_{i+2} \tag{2.7b}$$

Substituindo-se (2.5c) e (2.5d) nas equações (2.7) e lembrando-se da condição de continuidade de 2ª derivada entre intervalos adjacentes obtém-se a eq. (2.8) após algumas manipulações algébricas.

$$t_{i+2} \cdot P'_i + 2 \cdot (t_{i+2} + t_{i+1}) \cdot P'_{i+1} + t_{i+1} \cdot P'_{i+2} = \frac{3}{t_{i+1} t_{i+2}} \cdot \left(t_{i+1}^2 \cdot (P_{i+2} - P_{i+1}) + t_{i+2}^2 \cdot (P_{i+1} - P_i) \right) \tag{2.8}$$

A aplicação da equação (2.8) ao longo de todos os segmentos que definem a curva fornece um sistema de equações que permite calcular sua inclinação em todos os pontos internos, o que resulta em (n-2) equações para n pontos. Fixando-se as inclinações nas extremidades da curva, obtém-se o sistema linear da eq. (2.9), que apresentará a matriz M tridiagonal e diagonal dominante, o que, além de reduzir o esforço computacional, garante a unicidade da solução.

$$[M]\cdot\{P'\}=\{R\} \tag{2.9}$$

onde

$$M=\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
t_3 & 2\cdot(t_2+t_3) & t_2 & & & & & 0\\
0 & t_4 & 2\cdot(t_3+t_4) & t_3 & & & & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
0 & & t_{i+1} & 2\cdot(t_i+t_{i+1}) & t_i & & & 0\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
0 & & & & t_{n-1} & 2\cdot(t_{n-1}+t_n) & t_{n-1} & 0\\
0 & & & & & & 1 &
\end{bmatrix}$$

$$P'=\begin{Bmatrix}P'_1\\P'_2\\P'_3\\\vdots\\P'_i\\\vdots\\P'_{n-1}\\P'_n\end{Bmatrix} \quad e \quad R=\begin{Bmatrix}P'_1\\\frac{3}{t_2t_3}\left[t_2^2\cdot(P_3-P_2)+t_3^2\cdot(P_2-P_1)\right]\\\frac{3}{t_3t_4}\left[t_3^2\cdot(P_4-P_3)+t_4^2\cdot(P_3-P_2)\right]\\\vdots\\\frac{3}{t_it_{i+1}}\left[t_i^2\cdot(P_{i+1}-P_i)+t_{i+1}^2\cdot(P_i-P_{i-1})\right]\\\vdots\\\frac{3}{t_{n-1}t_n}\left[t_{n-1}^2\cdot(P_n-P_{n-1})+t_n^2\cdot(P_{n-1}-P_{n-2})\right]\\P'_n\end{Bmatrix}$$

O vetor P', que contém as inclinações de todos os pontos que definem a curva é então obtido através da eq (2.10).

$$\{P'\}=[M]^{-1}\cdot\{R\} \tag{2.10}$$

O resultado obtido em (2.10) permite agora resolver a eq. (2.5), que pode ser escrita matricialmente de acordo com a eq. (2.11)

$$\begin{Bmatrix}B_{1k}\\B_{2k}\\B_{3k}\\B_{4k}\end{Bmatrix}=\begin{bmatrix}1 & 0 & 0 & 0\\0 & 1 & 0 & 0\\\frac{-3}{t_{k+1}^2} & \frac{-2}{t_{k+1}} & \frac{3}{t_{k+1}^3} & \frac{-1}{t_{k+1}}\\\frac{2}{t_{k+1}^3} & \frac{1}{t_{k+1}^2} & \frac{-2}{t_{k+1}^3} & \frac{1}{t_{k+1}^2}\end{bmatrix}\cdot\begin{Bmatrix}P_k\\P_{k+1}\\P'_k\\P'_{k+1}\end{Bmatrix} \tag{2.11}$$

Uma vez determinado o vetor *B* pode-se generalizar a equação (2.3) para determinar as coordenadas dos pontos $P_i(t)$ sobre a curva:

$$P_i(t) = B_{1i} + B_{2i} \cdot t + B_{3i} \cdot t^2 + B_{4i} \cdot t^3 \tag{2.12}$$

ou, em notação matricial,

$$P_i(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \cdot \begin{bmatrix} B_{1i} \\ B_{2i} \\ B_{3i} \\ B_{4i} \end{bmatrix} \tag{2.13}$$

então, substituindo (2.11) em (2.13) e rearranjando, obtém-se

$$P_i(\tau) = \begin{bmatrix} F_1(\tau) & F_2(\tau) & F_3(\tau) & F_4(\tau) \end{bmatrix} \cdot \begin{bmatrix} P_i \\ P_i' \\ P_{i+1} \\ P_{i+1}' \end{bmatrix} \tag{2.14}$$

onde

$$\tau = \frac{t}{t_{i+1}}, \qquad 0 \le \tau \le 1, \qquad 1 \le i \le n-1$$

e

$$\begin{aligned} F_1(\tau) &= 2 \cdot \tau^3 - 3 \cdot \tau^2 + 1 \\ F_2(\tau) &= -2 \cdot \tau^3 + 3 \cdot \tau^2 \\ F_3(\tau) &= \tau \cdot (\tau^2 - 2 \cdot \tau + 1) \cdot t_{i+1} \\ F_4(\tau) &= \tau \cdot (\tau^2 - \tau) \cdot t_{i+1} \end{aligned} \tag{2.15}$$

Se

$$F = \{F_1(\tau) \quad F_2(\tau) \quad F_3(\tau) \quad F_4(\tau)\} \qquad \text{e} \qquad G = \{P_i \quad P_i' \quad P_{i+1} \quad P_{i+1}'\}^T$$

Pode-se escrever

$$P_i(\tau) = \{F\} \cdot \{G\} \tag{2.16}$$

De acordo com a equação (2.16), a coordenada de cada ponto *P(t)* no interior de um intervalo é dada pela soma ponderada das coordenadas dos pontos nas extremidades do intervalo e da inclinação da curva nestes dois pontos, sendo que o vetor {F} da eq. (2.15) age como função de ponderação. Além disso, esta equação também mostra que *P(t)* depende dos pontos de suporte. Esta parece uma observação óbvia, mas enfatiza que a escolha dos pontos de suporte para a coordenada paramétrica determina a suavidade da curva, que não é garantida simplesmente pela continuidade C^2 entre intervalos. Para isso, é necessário determinar o espaçamento apropriado entre os pontos de suporte de modo a minimizar B_3 e B_4, uma vez que estes coeficientes são os responsáveis pela não linearidade da curva interpoladora. Uma maneira eficaz de se conseguir uma curva suave é usar pontos equiespaçados na definição das curvas. Outro cuidado a se tomar é com o esforço computacional, já que a determinação dos coeficientes B_i envolve a inversão da matriz [M]. A normalização de todos os intervalos para um domínio de mesmo comprimento, por exemplo, faz com que [M] fique constante, sendo necessária então somente uma inversão. Porém, se os pontos de suporte não estiverem equiespaçados isso pode gerar concentração de pontos interpolados em torno de intervalos menores.

Se *t* for normalizado para o intervalo [0..1], a eq. (2.15) fica

$$\begin{aligned} F_1(t) &= 2 \cdot t^3 - 3 \cdot t^2 + 1 \\ F_2(t) &= -2 \cdot t^3 + 3 \cdot t^2 \\ F_3(t) &= t^3 - 2 \cdot t^2 + t \\ F_4(t) &= t^3 - t^2 \end{aligned} \tag{2.17}$$

ou, matricialmente,

$$\{F\} = \{T\} \cdot [N] = \{t^3 \quad t^2 \quad t \quad 1\} \cdot \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{2.18}$$

e a equação (2.16) agora se tornou

$$P(t) = \{T\} \cdot [N] \cdot \{G\} \tag{2.19}$$

A equação (2.9) agora é escrita da seguinte forma:

$$\begin{bmatrix} 1 & 0 & & & & & & & \\ 1 & 4 & 1 & 0 & & & & & \\ 0 & 1 & 4 & 1 & 0 & & & & \\ & & \ddots & & & & & & \\ & & & \ddots & & & & & \\ & & & & \ddots & & & & \\ & & & & & 1 & 4 & 1 \\ & & & & & & 0 & 1 \end{bmatrix} \cdot \begin{Bmatrix} P'_1 \\ P'_2 \\ P'_3 \\ \vdots \\ P'_i \\ \vdots \\ \vdots \\ P'_n \end{Bmatrix} = \begin{Bmatrix} P'_1 \\ 3 \cdot \left((P_3 - P_2) - (P_2 - P_1) \right) \\ 3 \cdot \left((P_4 - P_3) - (P_3 - P_2) \right) \\ \vdots \\ 3 \cdot \left((P_{i+1} - P_i) - (P_i - P_{i-1}) \right) \\ \vdots \\ 3 \cdot \left((P_{n-1} - P_{n-2}) - (P_{n-2} - P_{n-3}) \right) \\ P'_n \end{Bmatrix} \tag{2.20}$$

Embora reduza consideravelmente o tempo de cálculo em problemas de grande porte, a normalização das coordenadas paramétricas pode causar alguns inconvenientes se os pontos de suporte não estiverem uniformemente distribuídos. Neste caso, se estes pontos forem simplesmente numerados seqüencialmente, a curva resultante pode não representar adequadamente a curva. Uma alternativa que apresenta bons resultados, e será empregada neste trabalho, é a numeração dos pontos de acordo com sua posição sobre a curva. Assim, para a curva genérica apresentada na fig. 2.3, supondo que seu comprimento seja 1.0, os 3 pontos que a definem seriam numerados da seguinte forma:

- Numeração seqüencial: $t_1 = 1$, $t_2 = 2$, $t_3 = 3$.
- Numeração pelo comprimento da curva: $t_1 = 0.0$, $t_2 = 0.35$, $t_3 = 1.0$.

Isto faz com que a curva resultante seja mais suave, representando melhor a geometria do problema (Rogers, 1990). Ou seja, o modo como as coordenadas paramétricas dos pontos de suporte são numeradas definirá a suavidade da curva resultante.

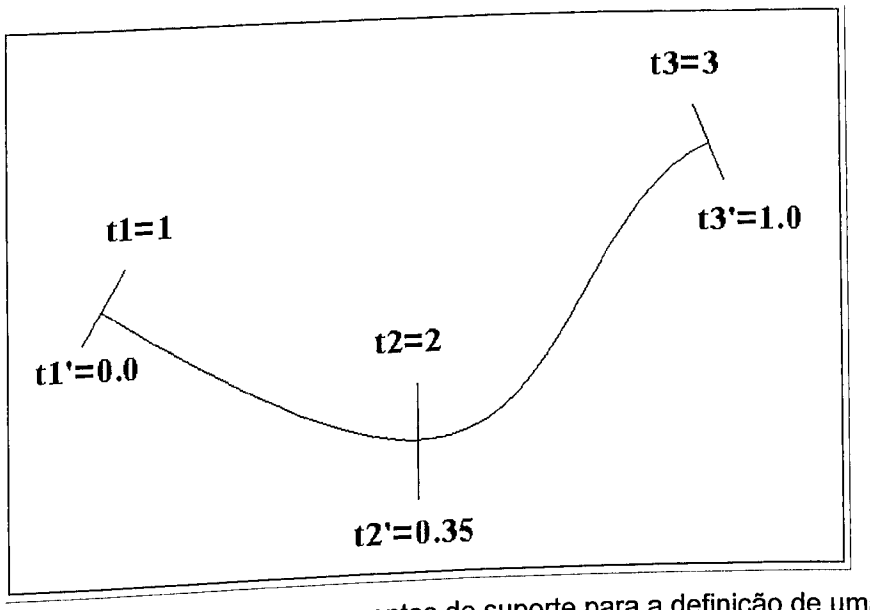


Figura 2.3: Dois modos de se numerar os pontos de suporte para a definição de uma spline

Além da suavidade da curva, outra grande diferença relacionada à numeração dos pontos de suporte é a concentração de pontos em intervalos menores no caso de distribuição não uniforme e numeração seqüencial. Este fato é de grande importância neste trabalho, já que as splines serão empregadas não somente na definição da geometria, mas também na aplicação das condições de contorno para o gerador elíptico de malhas. Uma vez que serão usadas condições de Dirichlet, estas serão as próprias coordenadas dos nós na fronteira do domínio. Assim, a numeração sequencial pode ser usada para concentrar elementos em regiões de interesse, aumentando-se a quantidade de pontos de suporte nestas regiões. Entretanto, esta abordagem não é a mais recomendada, uma vez que *o usuário fornecerá as coordenadas cartesianas dos pontos de suporte e não as coordenadas paramétricas, e seu controle sobre a densidade da malha não será preciso.* Recomenda-se, nestes casos, a concentração de subdomínios nas regiões de interesse, e este assunto será discutido em detalhes no capítulo 4.

Capítulo 3

Gerador Elíptico de Malhas Estruturadas Bidimensionais

3.1 Introdução

Tendo em vista que a otimização da malha será feita puramente por reposicionamento dos nós do modelo buscando a regularidade geométrica dos elementos, a escolha de um gerador elíptico em vez de um algébrico foi feita buscando diminuir o número de variáveis de projeto no problema. Considerando uma malha quadrada de m X m elementos, um método algébrico teria m(m-1) variáveis de projeto, que seriam os nós do modelo, exceto os nós que definem os vértices do domínio. Já o método elíptico empregado analisa apenas os elementos que compõem as fronteiras do domínio e usa somente as coordenadas dos nós de fronteira como condições de contorno de Dirichlet, totalizando 4(m-1) variáveis de projeto. Além disso, variações bruscas de derivada nas fronteiras não se propagam para o interior do domínio, o que também garante que as linhas coordenadas de uma mesma família nunca se interceptarão (Thompson, 1985), (Milioli, 1985). Já para o método algébrico, esta garantia poderia elevar o custo computacional, uma vez que as funções de interpolação não possuem esta propriedade, e testes adicionais deveriam ser implementados.

Entretanto, o uso de sistemas de equações diferenciais parciais para geração de malhas estruturadas, apesar de apresentar resultados bastante interessantes, pode ter um elevado custo computacional se certos cuidados não forem tomados. Isto porque o método exige, obrigatoriamente, a resolução de sistemas de equações sobre domínios arbitrários, o que é inviável tanto pela resolução do sistema de equações em si como pela aplicação das condições de contorno.

Para evitar estes problemas, o método de geração empregado neste trabalho faz uma mudança de domínios – ou mudanças de variáveis – de modo que o sistema de equações seja resolvido sobre um domínio retangular e a resolução já forneça as coordenadas dos nós de discretização diretamente para o domínio original ou domínio físico.

A mudança de domínios é feita considerando que o domínio físico seja estabelecido sobre um sistema de coordenadas curvilíneas. Estabelecem-se então relações de transformação de coordenadas que mapeam um domínio bidimensional arbitrário para um domínio retangular. A resolução do sistema de equações neste domínio passa a ter menor

custo computacional e o resultado é uma malha estruturada coincidente com a fronteira do domínio original.

O domínio circular da figura 3.1, por exemplo, pode ser representado tanto no sistema cartesiano como no sistema polar, que é obviamente o mais adequado. Enquanto para este sistema basta especificar pares (r,θ) com $0 \le r \le R$ e $0 \le \theta \le 2\pi$, no sistema cartesiano o domínio seria representado pela equação implícita

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

$$\begin{aligned} &0 \le r \le R \\ &x_o - r \le x \le x_o + r \\ &y_o - r \le x \le y_o + r \end{aligned} \qquad (3.1)$$

Um modo mais simples de se caracterizar este domínio seria estabelecer relações entre os sistemas de coordenadas, obtendo as equações (3.2) e (3.3):

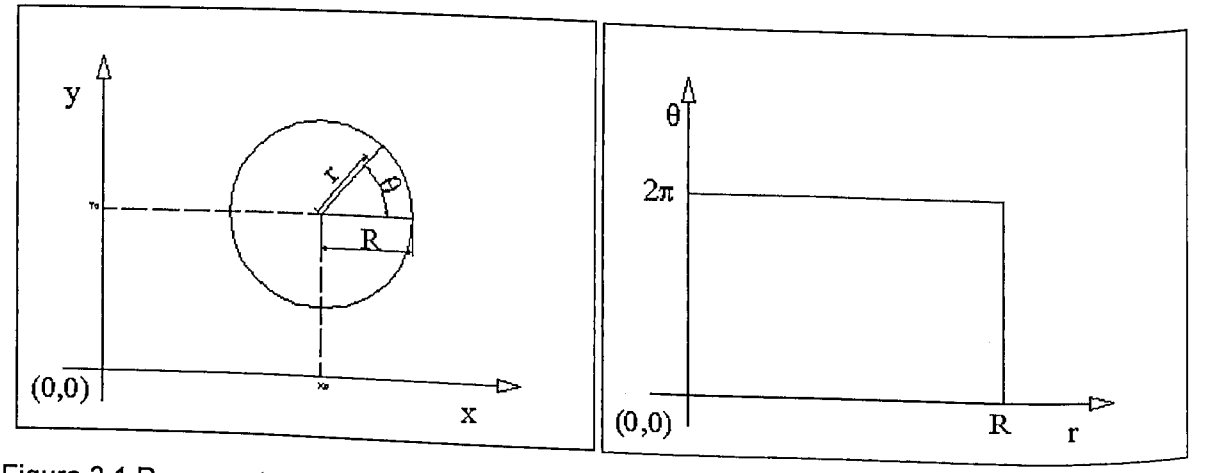


Figura 3.1 Representação de um domínio em dois sistemas de coordenadas.

$$x = x_0 + r \cdot cos(\theta) \qquad (3.2)$$

$$y = y_0 + r \cdot sin(\theta) \qquad (3.3)$$

ou, inversamente,

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2} \qquad (3.4)$$

$$\theta = atan\left(\frac{x - x_0}{y - y_0}\right) \qquad (3.5)$$

Ou seja, é possível determinar uma relação entre os sistemas de coordenadas de modo que o domínio circular, mais facilmente representado no sistema polar, seja representado no sistema cartesiano determinando-se relações entre as bases dos dois sistemas. Agora, para gerar uma malha estruturada sobre o domínio basta especificar pares (r,θ), gerando uma malha retangular sobre este domínio, e aplicar as equações 3.2 e 3.3 para se obter as coordenadas (x,y) dos nós.

Esta idéia será agora expandida para domínios bidimensionais arbitrários, com a condição de que sejam quadrilaterais. Ao contrário do exemplo descrito acima, porém, a mudança de variáveis não será feita para as equações que definem o domínio, mas para um sistema de equações elípticas que modelam um problema genérico de campo de potenciais.

3.2 Desenvolvimento das Equações Geradoras

O sistema gerador consiste de um sistema de equações elípticas de Laplace formulado para um sistema de coordenadas curvilíneas coincidentes com a fronteira do domínio. Se, no domínio Γ da figura 3.2a (x,y) forem as coordenadas independentes e (ξ,η) as variáveis dependentes, as equações de geração serão dadas por:

$$\nabla^2 \xi(x,y) = 0 \tag{3.6}$$

$$\nabla^2 \eta(x,y) = 0 \tag{3.7}$$

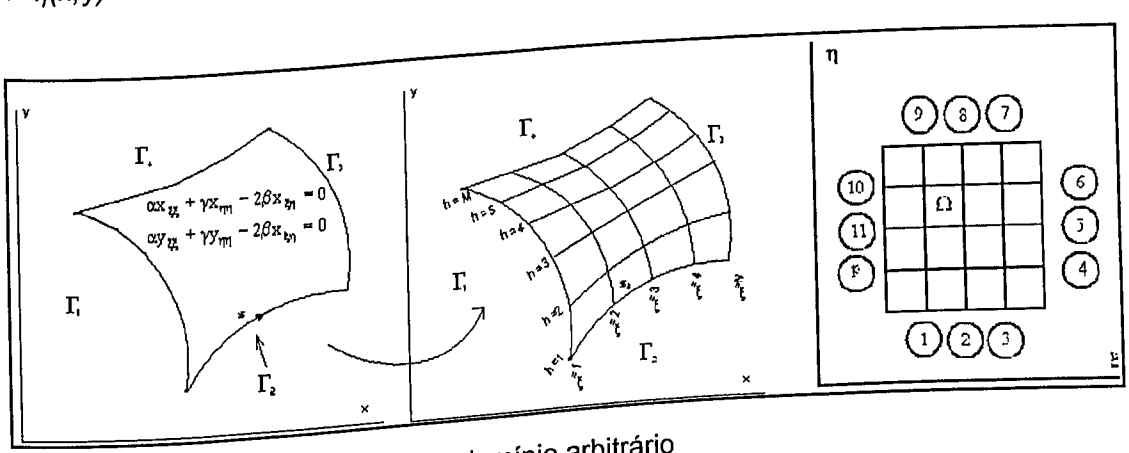


Figura 3.2: Malha estruturada sobre domínio arbitrário

O objetivo aqui é determinar exatamente o oposto, ou seja: expressar as coordenadas (x,y) dos pontos interiores ao domínio Γ em função de (ξ,η). As equações 3.6 e 3.7 então serão escritas como

$$\tilde{\nabla}^2 x(\xi,\eta) = 0 \tag{3.8}$$

$$\tilde{\nabla}^2 y(\xi,\eta) = 0 \tag{3.9}$$

onde ∇^2 e $\tilde{\nabla}^2$ são os operadores laplacianos nos referenciais cartesiano (x,y) e curvilíneo (ξ,η), respectivamente.

A aplicação de (3.6) ou (3.7) a uma função genérica f resulta em

$$\nabla^2 f(\xi(x,y),\eta(x,y)) = 0 \tag{3.10}$$

As derivadas parciais de segunda ordem f_{xx} e f_{yy} são dadas por

$$f_{xx} = f_\xi \xi_{xx} + f_\eta \eta_{xx} + \xi_x^2 f_{xx} + \eta_x^2 f_{\eta\eta} + 2\xi_x \eta_x f_{\xi\eta} \tag{3.11}$$

$$f_{yy} = f_\xi \xi_{yy} + f_\eta \eta_{yy} + \xi_y^2 f_{yy} + \eta_y^2 f_{\eta\eta} + 2\xi_y \eta_y f_{\xi\eta} \tag{3.12}$$

Sendo $f(\xi,\eta)$ a função que relaciona as coordenadas no sistema curvilíneo às coordenadas no sistema cartesiano, deseja-se então:

$$f(\xi,\eta) = x$$
$$f(\xi,\eta) = y \tag{3.13}$$

Então, fazendo f = x e f = y no domínio na equação 3.11, obtém-se:

$$x_\xi \xi_{xx} + x_\eta \eta_{xx} + E_1 = 0 \tag{3.14}$$

$$y_\xi \xi_{xx} + y_\eta \eta_{xx} + F_1 = 0 \tag{3.15}$$

onde

$$E_1 = \xi_x^2 x_{\xi\xi} + \eta_x^2 x_{\eta\eta} + 2\xi_x \eta_x x_{\xi\eta} \tag{3.16}$$

$$F_1 = \xi_x^2 y_{\xi\xi} + \eta_x^2 y_{\eta\eta} + 2\xi_x \eta_x y_{\xi\eta} \tag{3.17}$$

Procedimento análogo para a equação 3.12 resulta em

$$x_\xi \xi_{yy} + x_\eta \eta_{yy} + E_2 = 0 \tag{3.18}$$

$$y_\xi \xi_{yy} + y_\eta \eta_{yy} + F_2 = 0 \tag{3.19}$$

onde

$$E_2 = \xi_y^2 x_{\xi\xi} + \eta_y^2 y_{\eta\eta} + 2\xi_y \eta_y y_{\xi\eta} \tag{3.20}$$

$$F_2 = \xi_y^2 y_{\xi\xi} + \eta_y^2 y_{\eta\eta} + 2\xi_y \eta_y y_{\xi\eta} \tag{3.21}$$

Reescrevendo-se os sistemas das equações 3.14 – 3.15 e 3.18 – 3.19 na forma matricial, obtêm-se

$$\begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \cdot \begin{Bmatrix} \xi_{xx} \\ \eta_{xx} \end{Bmatrix} = -\begin{Bmatrix} E_1 \\ F_1 \end{Bmatrix} \tag{3.22}$$

$$\begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \cdot \begin{Bmatrix} \xi_{yy} \\ \eta_{yy} \end{Bmatrix} = -\begin{Bmatrix} E_2 \\ F_2 \end{Bmatrix} \tag{3.23}$$

A resolução dos sistemas dados pelas equações 3.22 e 3.23 fornece

$$\xi_{xx} = -\left(E_1 \xi_x + F_1 \xi_y\right) \tag{3.24}$$

$$\eta_{xx} = -\left(E_1 \eta_x + F_1 \eta_y\right) \tag{3.25}$$

$$\xi_{yy} = -\left(E_2 \xi_x + F_2 \xi_y\right) \tag{3.26}$$

$$\eta_{yy} = -\left(E_2 \eta_x + F_2 \eta_y\right) \tag{3.27}$$

Substituindo-se (3.24) a (3.27) em (3.6) e (3.7) obtém-se

$$E_1 \xi_x + F_1 \xi_y - \left(E_2 \xi_x + F_2 \xi_y\right) = 0 \tag{3.28}$$

$$E_1 \eta_x + F_1 \eta_y - \left(E_2 \eta_x + F_2 \eta_y\right) = 0 \tag{3.29}$$

$$\left(E_1 + E_2\right) \cdot \xi_x - \left(F_1 + F_2\right) \cdot \xi_y = 0 \tag{3.30}$$

$$\left(E_1 + E_2\right) \cdot \eta_x - \left(F_1 + F_2\right) \cdot \eta_y = 0 \tag{3.31}$$

Fazendo $E = E_1 + E_2$, $F = F_1 + F_2$ pode-se reagrupar estas 4 equações na forma matricial

$$\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \cdot \begin{Bmatrix} E \\ F \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \tag{3.32}$$

que só admite a solução trivial

$$E = 0 \tag{3.33}$$

$$F = 0 \tag{3.34}$$

Substituindo-se então os valores de E_1 e E_2, F_1 e F_2 nas equações 3.33 e 3.34, obtém-se:

$$a \cdot x_{\xi\xi} + b \cdot x_{\eta\eta} + 2c \cdot x_{\xi\eta} = 0 \tag{3.35}$$

$$a \cdot y_{\xi\xi} + b \cdot y_{\eta\eta} + 2c \cdot y_{\xi\eta} = 0 \tag{3.36}$$

onde

$$a = \xi_x^2 + \xi_y^2 \tag{3.37}$$

$$b = \eta_x^2 + \eta_y^2 \tag{3.38}$$

$$c = \xi_x \eta_x + \xi_y \eta_y \tag{3.39}$$

Como as equações (3.35) e (3.36) possuem variáveis representadas nos dois domínios, o último passo para obtenção das equações de geração para o sistema curvilíneo é então efetuar uma mudança de variáveis nas expressões 3.37 a 3.39, de modo que apresentem (ξ,η) como variáveis independentes. Dados

$$\xi = \xi(x,y) \tag{3.40}$$

$$\eta = \eta(x,y) \tag{3.41}$$

Os diferenciais $d\xi$ e $d\eta$ são obtidos pela aplicação da regra da cadeia:

$$d\xi = \xi_x dx + \xi_y dy \tag{3.42}$$

$$d\eta = \eta_x dx + \eta_y dy \tag{3.43}$$

Escrevendo na forma matricial,

$$\begin{Bmatrix} d\xi \\ d\eta \end{Bmatrix} = \begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \cdot \begin{Bmatrix} dx \\ dy \end{Bmatrix} \tag{3.44}$$

o que é equivalente a escrever

$$(d_T) = [A] \cdot (d_F) \tag{3.45}$$

onde d_T e d_F são, respectivamente, os diferenciais no domínio transformado e domínio físico. Analogamente, os diferenciais dx e dy, para (x,y) como variáveis dependentes, são dados na forma matricial por

$$\begin{Bmatrix} dx \\ dy \end{Bmatrix} = \begin{bmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{bmatrix} \cdot \begin{Bmatrix} d\xi \\ d\eta \end{Bmatrix} \tag{3.46}$$

ou

$$\{d_F\} = [B] \cdot \{d_T\} \tag{3.47}$$

Comparando-se (3.47) e (3.45) chega-se à relação

$$[A] = [B]^{-1} = J \cdot \begin{bmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{bmatrix} \tag{3.48}$$

onde

$$J = \frac{1}{\det(B)} = \frac{1}{x_\xi y_\eta - x_\eta y_\xi} \tag{3.49}$$

A lei de transformação do domínio físico para o domínio transformado é dada então por

$$\begin{aligned} \xi_x &= J y_\eta & \xi_y &= J x_\eta \\ \eta_x &= -J y_\xi & \eta_y &= -J x_\xi \end{aligned} \tag{3.50}$$

Aplicando-se a equação 3.50 às equações 3.37 a 3.39, obtém-se o sistema gerador formulado para o sistema curvilíneo:

$$\alpha \cdot x_{\xi\xi} + \beta \cdot x_{\eta\eta} - 2 \cdot \gamma \cdot x_{\xi\eta} = 0 \tag{3.51}$$

$$\alpha \cdot y_{\xi\xi} + \beta \cdot y_{\eta\eta} - 2 \cdot \gamma \cdot y_{\xi\eta} = 0 \tag{3.52}$$

onde

$$\alpha = \left(x_{\eta}^2 + y_{\eta}^2\right) \cdot J^2 \tag{3.53}$$

$$\beta = \left(x_{\xi}^2 + y_{\xi}^2\right) \cdot J^2 \tag{3.54}$$

$$\gamma = \left(x_{\xi} x_{\eta} + y_{\xi} y_{\eta}\right) \cdot J^2 \tag{3.55}$$

ou, na forma matricial,

$$\begin{bmatrix} x_{\xi\xi} & x_{\eta\eta} \\ y_{\xi\xi} & y_{\eta\eta} \end{bmatrix} \cdot \begin{Bmatrix} \alpha \\ \beta \end{Bmatrix} = \begin{Bmatrix} x_{\xi\eta} \\ y_{\xi\eta} \end{Bmatrix} \tag{3.56}$$

O sistema gerador descrito pela equação 3.56 pode ser analisado de duas formas:

1. Foi feita uma mudança de variáveis de modo a expressar o Laplaciano num sistema de coordenadas curvilíneas genérico, coincidente com a fronteira do domínio, o que também facilita a aplicação das condições de contorno. Isto porque, uma vez que as coordenadas são definidas por uma variável paramétrica, pode-se associá-la a valores de (ξ,η) que definem as coordenadas dos nós de fronteira, que são as variáveis de projeto.
2. Foi feita uma mudança de domínio através de uma lei de mapeamento de modo que qualquer domínio bidimensional quadrilateral seja representado por um domínio retangular no domínio transformado. Assim como as equações 3.2 a 3.5 fazem as mudanças no exemplo apresentado na introdução deste capítulo, a equação 3.56 faz para o mapeamento neste caso.

3.3 Discretização do Sistema Gerador

A equação 3.57 é uma generalização do sistema gerador descrito pelas equações 3.51 e 3.52, utilizando-se a variável dependente genérica u, sendo os coeficientes α, β e γ dados pelas equações 3.53 a 3.55.

$$\alpha \cdot u_{\xi\xi} + \beta \cdot u_{\eta\eta} - 2\gamma \cdot u_{\xi\eta} = 0 \tag{3.57}$$

Para resolver este sistema por diferenças finitas centrais para o ponto genérico p da fig. 3.3, as equações discretizadas são:

$$u_{\xi\xi} = \frac{u_E + u_W - 2u_P}{\Delta\xi^2} \tag{3.58}$$

$$u_{\eta\eta} = \frac{u_S + u_N - 2u_P}{\Delta\eta^2} \tag{3.59}$$

$$u_{\xi\eta} = \frac{u_{SW} + u_{NE} - u_{NW} - u_{SE}}{4\Delta\eta\Delta\xi} \tag{3.60}$$

$$u_{\xi} = \frac{u_E - u_W}{2\Delta\xi} \tag{3.61}$$

$$u_{\eta} = \frac{u_N - u_S}{2\Delta\eta} \tag{3.62}$$

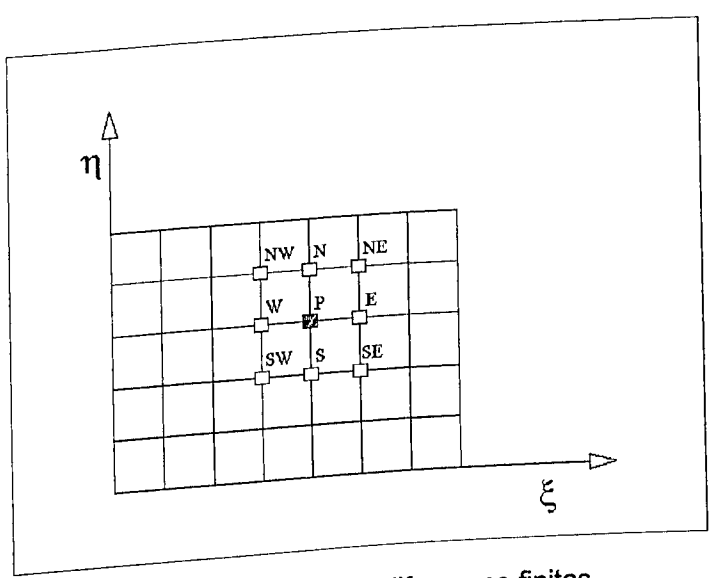


Figura 3.3. Esquema gráfico de discretização em diferenças finitas.

Substituindo 3.58 a 3.62 em 3.57 chegam-se às equações 3.63 e 3.64 após algumas manipulações algébricas.

$$A_p u_p = A_E u_E + A_W u_W + A_N u_N + A_S u_S + A_{NE} u_{NE} + A_{SE} u_{SE} + A_{NW} u_{NW} + A_{SW} u_{SW} \tag{3.63}$$

$$\begin{aligned} A_E &= A_W = \alpha \\ A_N &= A_S = \gamma \\ A_P &= 2 \cdot (\alpha + \gamma) \\ A_{NE} &= A_{SW} = \frac{-\beta}{2} \\ A_{SE} &= A_{NW} = \frac{\beta}{2} \end{aligned} \tag{3.64}$$

Neste trabalho, as equações 3.58 a 3.64 são resolvidas por um algoritmo SOR associado a um algoritmo multi grelha, para acelerar a convergência do método.

3.3.1 Fundamentos do método SOR

Para resolver o sistema linear

$$A \cdot u = f \tag{3.65}$$

o método SOR modifica o método iterativo clássico de Gauss-Seidel, ponderando a atualização do vetor ***u*** por um fator de relaxação w para acelerar a convergência do método. A fórmula de Gauss-Seidel na forma matricial para a iteração k é dada por (Press et al., 1992)

$$(L + D) \cdot u_k = -U \cdot u_{k-1} + b \tag{3.66}$$

onde D é a diagonal de A, L e U são os triângulos inferior e superior de A, respectivamente. Resolvendo-se (3.66) para u_k, adicionando-se e subtraindo-se u_{k-1} no lado direito da equação, obtém-se

$$u_k = u_{k-1} - (L + D)^{-1} \cdot [(L + D + U) \cdot u_{k-1} - b] \tag{3.67}$$

O termo entre colchetes em (3.67) representa o resíduo r da iteração k, e será chamado aqui de r_k. Então, pode-se rescrever esta equação como

$$u_k = u_{k-1} - (L + D)^{-1} \cdot r_{k-1} \tag{3.68}$$

Aplicando o fator de relaxação w na equação acima, obtém-se a iteração SOR na forma matricial

$$u_k = u_{k-1} - w(L+D)^{-1} \cdot r_{k-1} \tag{3.69}$$

Para se escrever a equação (3.69) a partir de um sistema originado de uma discretização em diferenças finitas parte-se da equação para um ponto genérico P

$$A_E^{\,k} \cdot u_E^k + A_W^{\,k} \cdot u_W^k + A_N^{\,k} \cdot u_N^k + A_S^{\,k} \cdot u_S^k + A_{NE}^{\,k} \cdot u_{NE}^k + \\ + A_{NW}^{\,k} \cdot u_{NW}^k + A_{SE}^{\,k} \cdot u_{SE}^k + A_{SW}^{\,k} \cdot u_{SW}^k - A_P^{\,k} \cdot u_P^k = f_P^k \tag{3.70}$$

e o resíduo em cada iteração será a diferença entre a aproximação u e a solução f:

$$r_p^{\,k} = A_E^{\,k} \cdot u_E^k + A_W^{\,k} \cdot u_W^k + A_N^{\,k} \cdot u_N^k + A_S^{\,k} \cdot u_S^k + A_{NE}^{\,k} \cdot u_{NE}^k + \\ + A_{NW}^{\,k} \cdot u_{NW}^k + A_{SE}^{\,k} \cdot u_{SE}^k + A_{SW}^{\,k} \cdot u_{SW}^k - A_P^{\,k} \cdot u_P^k - f_P^k \tag{3.71}$$

Para obter-se a iteração SOR, calcula-se primeiro u_p^k a partir de 3.72:

$$u_p^{\,k} = \frac{1}{A_P} * \begin{pmatrix} A_E^{\,k} \cdot u_E^k + A_W^{\,k} \cdot u_W^k + A_N^{\,k} \cdot u_N^k + A_S^{\,k} \cdot u_S^k + \\ + A_{NE}^{\,k} \cdot u_{NE}^k + A_{NW}^{\,k} \cdot u_{NW}^k + A_{SE}^{\,k} \cdot u_{SE}^k + A_{SW}^{\,k} \cdot u_{SW}^k - f_P^k \end{pmatrix} \tag{3.72}$$

e a iteração SOR é então definida pela equação 3.73

$$u_P^{k+1} = u_P^k + w \cdot r_P^{\,k} \tag{3.73}$$

Até a década de 1970, este era o método padrão para resolução de sistemas lineares. Porém, todos os métodos iterativos, inclusive o SOR, apresentavam a indesejada característica de convergência lenta nas etapas finais do processo. Para contornar este problema, surgiram nesta época os métodos de multi grelha, que procuram recriar as condições que implicam na alta taxa de convergência nas etapas iniciais do processo, quando o resíduo do processo se estagnar em um determinado nível acima do permitido.

3.3.2 Fundamentos da técnica Multi Grelha

O desenvolvimento dos métodos de multi grelha teve início quando se constatou que os métodos iterativos de solução de problemas lineares reduziam apenas os componentes oscilatórios do erro, no início do processo de solução (Wagner, 1998). Então, após algumas iterações, quando já havia ocorrido uma suavização do resíduo, era preciso introduzir novamente oscilações no processo de convergência, de modo que o algoritmo de solução recuperasse seu bom desempenho inicial. Os métodos de multi grelha fazem isso transferindo o processo estagnado para uma malha mais grosseira, de modo que a curva do erro, suave para a malha fina, passe a apresentar oscilações na nova malha, recriando o ambiente no qual os métodos tradicionais apresentam bons desempenhos. Efetuam-se então mais algumas iterações na malha grosseira até o processo se estagnar novamente. Pode-se então retornar à malha fina ou transferir os resultados para malhas ainda mais grosseiras, recursivamente, até se chegar à última malha, sempre relaxando o sistema a cada nova malha empregada. A transferência das malhas grosseiras para as mais finas pode ser feita diretamente por interpolação linear, sem perda de precisão no processo (Lebron, 2000).

Considere, por exemplo, a resolução do sistema Au=f dado pela equação 3.68 na seção anterior. Como a solução exata u não é conhecida, parte-se de uma aproximação $\tilde{u}$ tal que $u \approx \tilde{u}$. Então o erro cometido na aproximação é simplesmente

$$\varepsilon = u - \tilde{u} \tag{3.74}$$

Porém, se a solução exata do processo não é conhecida, o erro também não pode ser calculado. Utiliza-se então o resíduo ***r***, que determina a diferença entre f e $A \cdot \tilde{u}$ a cada iteração:

$$r = f - A \cdot \tilde{u} \tag{3.75}$$

ou

$$A \cdot \tilde{u} = f - r \tag{3.76}$$

Subtraindo-se a Eq. (3.76) da Eq. (3.65) obtém-se

$$A \cdot (u - \tilde{u}) = r \tag{3.77}$$

ou

$$A \cdot \varepsilon = r \tag{3.78}$$

que é chamada de equação residual. Assim, resolver o sistema da Eq. (3.65), como originalmente proposto, é o mesmo que resolver o sistema da Eq. (3.78). Esta equação residual é utilizada nos esquemas Multi Grelha porque a variável dependente ε oscila em torno de zero, facilitando a transferência de valores entre malhas distintas.

O algoritmo Multi Grelha pode ser formulado para duas malhas genéricas Ω^h e Ω^{2h}, com dimensões características h e $2h$, respectivamente, do seguinte modo (Briggs, 1987), (Wagner, 1998):

1. Relaxar o sistema $A^h u^h = f^h$ na malha Ω^h durante algumas iterações, obtendo uma aproximação inicial $\tilde{u}^h$;
2. Calcular o resíduo $r^h = (f^h - A^h \tilde{u}^h)$;
3. Transferir os valores do resíduo r^h para a malha mais grossa Ω^{2h} , para obter r^{2h} por interpolação;
4. Relaxar algumas vezes a equação residual $A^{2h}\varepsilon^{2h} = r^{2h}$ na malha Ω^{2h}, obtendo-se ε^{2h};
5. Transferir os valores do erro ε^{2h} para a malha mais fina Ω^h, para obter ε^h através de funções interpolantes;
6. Corrigir a aproximação da malha fina com $u^h \leftarrow \tilde{u}^h + \varepsilon^h$.

A transferência dos resultados para malhas mais finas, chamada de prolongação ou interpolação, é denotada por I_{2h}^h e pode ser feita da seguinte forma:

$$\begin{aligned}
&I_{2h}^h = \varepsilon^h \\
&\varepsilon_{2i,2j}^h = \varepsilon_{i,j}^{2h} \\
&\varepsilon_{2i+1,2j}^h = \tfrac{1}{2}(\varepsilon_{i,j}^{2j} + \varepsilon_{i+1,j}^{2j}) \\
&\varepsilon_{2i,2j+1}^h = \tfrac{1}{2}(\varepsilon_{i,j}^{2h} + \varepsilon_{i,j+1}^{2h}) \\
&\varepsilon_{2i+1,2j+1}^h = \tfrac{1}{4}(\varepsilon_{i,j}^{2h} + \varepsilon_{i+1,j}^{2h} + \varepsilon_{i,j+1}^{2h} + \varepsilon_{i+1,j+1}^{2h})
\end{aligned} \tag{3.79}$$

onde $0 < i < \frac{M}{2} - 1$, $0 < j < \frac{M}{2} - 1$ e M é o número de pontos em x e y.

Ou seja, o termo central é simplesmente copiado para a malha fina, enquanto os adjacentes são determinados por médias dos vizinhos.

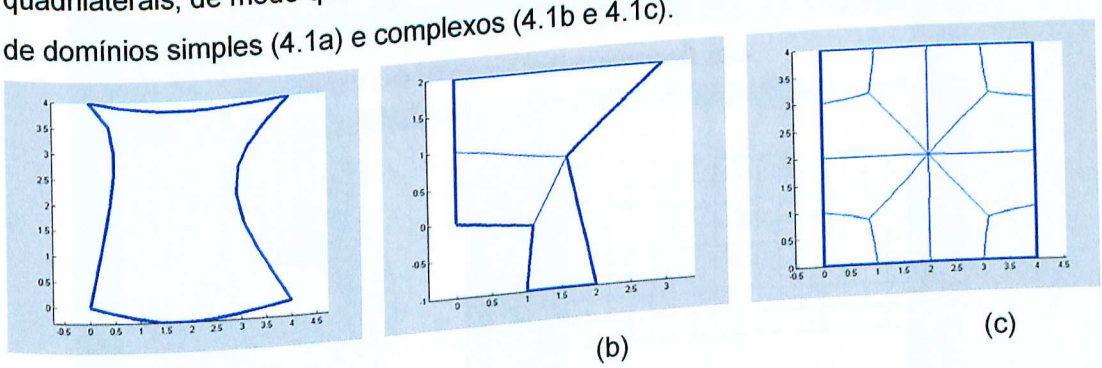
Este procedimento, ilustrado para duas malhas, pode ser estendido recursivamente até se atingir a malha mais grosseira possível, a de 3 pontos, e daí retornando-se à mais fina, onde reside a solução do problema original. O percurso de descida e subida pode ser realizado de formas diferentes, surgindo daí os diversos tipos de ciclos Multi Grelha. Uma descida simples até o fundo com retorno imediato à superfície é chamada de ciclo V. O ciclo W, como o próprio nome o sugere, desce da malha mais fina para a mais grosseira, como no ciclo V, porém durante o retorno interrompe a subida em uma malha intermediária, voltando novamente ao fundo, para só então retornar definitivamente à superfície. A convergência do problema em questão indica o tipo de ciclo adequado e o número de sub-malhas necessárias.

Capítulo 4

O algoritmo de Otimização

4.1 Introdução. O problema de otimização de malhas.

Os capítulos 2 e 3 forneceram a base para a construção de um domínio arbitrário através de curvas de interpolação por spline cúbica definidas parametricamente e sua discretização por malhas estruturadas obtidas através de um gerador elíptico com equações de Laplace. Neste capítulo, estes procedimentos serão empregados em conjunto na construção de um algoritmo para geração e otimização de malhas estruturadas em domínios bidimensionais arbitrários. O processo de otimização consiste em estabelecer um conjunto de condições de contorno para o gerador elíptico, visando obter-se a regularidade geométrica dos elementos. Como o emprego do gerador elíptico pressupõe que o domínio seja quadrilateral, domínios quadrilaterais simples são discretizados diretamente, ao passo que domínios complexos, isto é, que não apresentem forma quadrilateral, devem ser decompostos em subdomínios quadrilaterais, de modo que o gerador elíptico possa ser utilizado. A figura 4.1 mostra exemplos de domínios simples (4.1a) e complexos (4.1b e 4.1c).

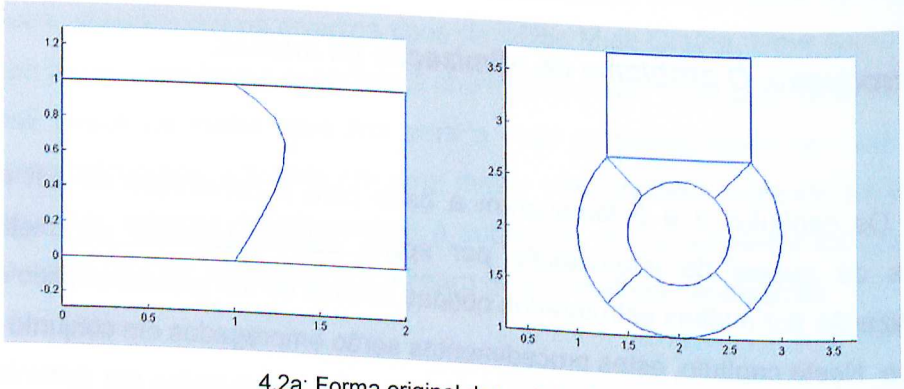


(a) (b) (c)

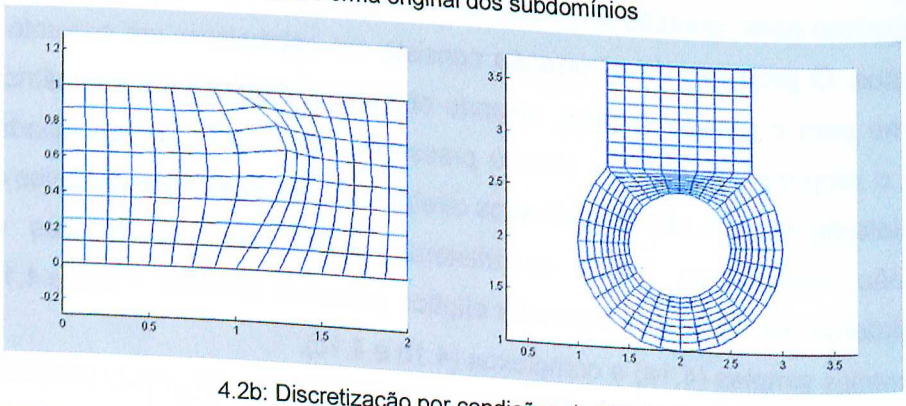
Figura 4.1: Exemplos de domínios simples e complexos

A diferença entre a otimização de uma malha gerada sobre um domínio simples e um domínio complexo é que, no segundo caso, deve-se garantir a continuidade da malha sobre as faces compartilhadas pelos subdomínios adjacentes. Esta continuidade pode ser construída de duas maneiras:De ordem C^0, onde se garante apenas a coincidência da malha na fronteira. Nesta abordagem impõem-se as mesmas condições de Dirichlet para os subdomínios adjacentes (fig.4.2b);De ordem C^1, onde, além da continuidade da malha exige-se também a continuidade da primeira derivada da malha ao cruzar a fronteira. Nesta abordagem impõem-se

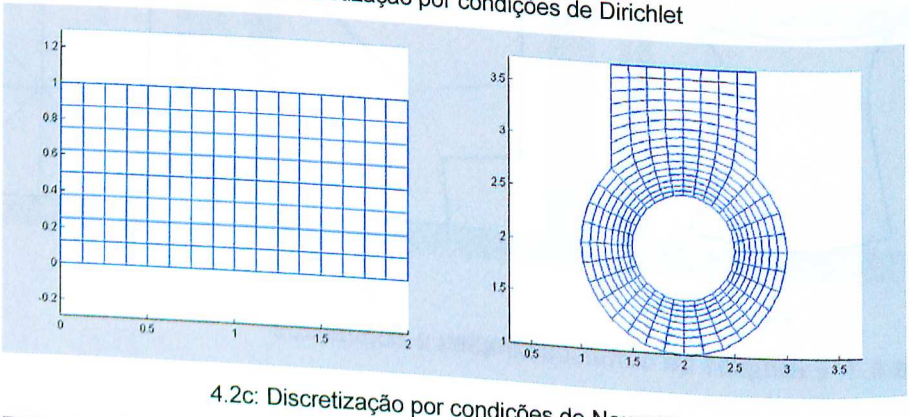
condições de Neumann para os subdomínios adjacentes. A consequência desta imposição é o reposicionamento da face que divide os subdomínios (fig. 4.2c). Portanto, não se deve utilizar esta abordagem quando a manutenção da forma dos subdomínios for importante para a discretização e análise subsequente.



4.2a: Forma original dos subdomínios



4.2b: Discretização por condições de Dirichlet



4.2c: Discretização por condições de Neumann

Figura 4.2: Dois domínios diferentes discretizados com condições de Dirichlet e de Neumann nas faces comuns aos subdomínios adjacentes.

A discretização de um domínio genérico é realizada em 3 fases:

- Caracterização do Domínio
- Geração da malha inicial
- Otimização da malha

A descrição de cada uma destas fases será feita primeiramente para um domínio quadrilateral simples, a partir do qual o método será generalizado.

4.2 O caso fundamental: geração da malha ótima para um domínio quadrilateral simples

Neste caso, o domínio Ω é modelado como um objeto computacional (fig. 4.3), composto pelas seguintes propriedades:

- Fronteira Γ, composta por quatro curvas splines cúbicas (Γ_s, Γ_E, Γ_N, Γ_W, em referência a sul, leste, norte e oeste), que serão chamadas de faces;
- Conjunto de nós internos (x,y) no interior do domínio;
- Quantidade de elementos presentes em cada face do domínio (m);
- Conjunto de condições de contorno definidas parametricamente sobre Γ pelo vetor $\{t\} = \{t_1, t_2, t_3, \ldots t_{4m}\}$;
- Um valor de mérito ε, que quantifica a distorção dos elementos localizados na fronteira Γ.

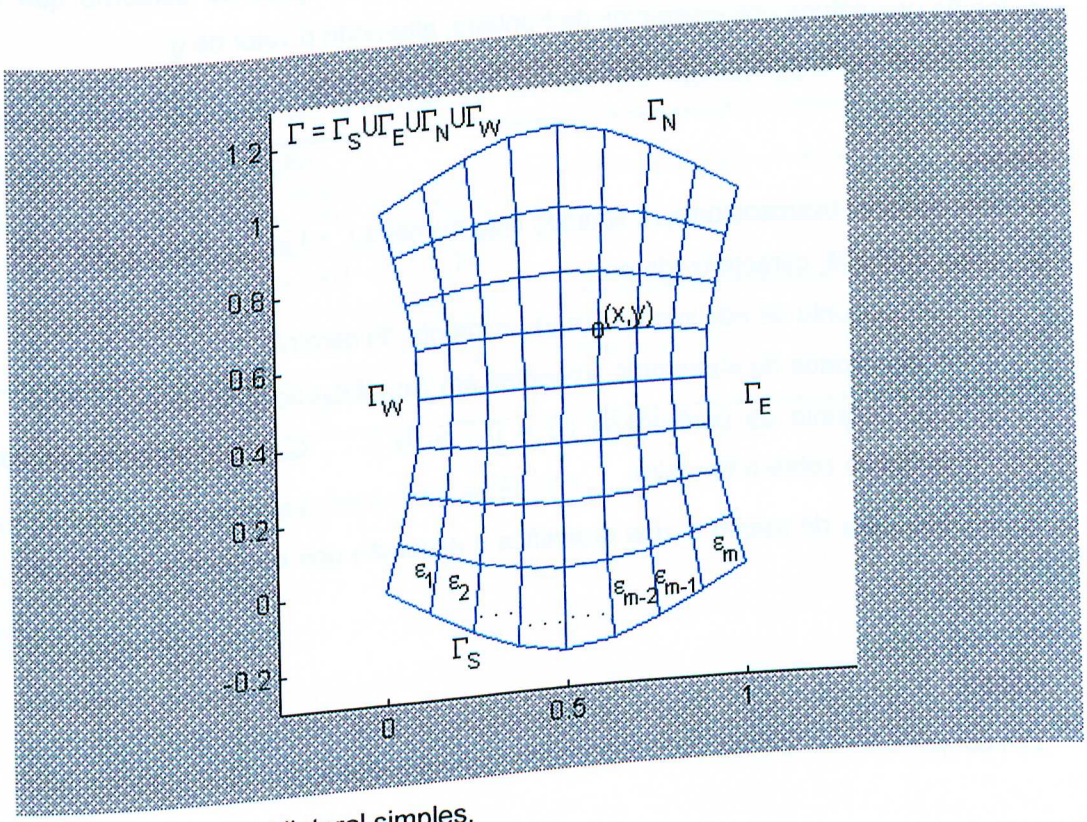


Figura 4.3: Domínio quadrilateral simples.

As faces ou fronteiras do domínio também são objetos computacionais, chamados de objetos face, definidos pelos coeficientes B_i do polinômio interpolador, calculados pela equação (2.11) a partir dos pontos fornecidos pelo usuário. A determinação das coordenadas cartesianas dos pontos da fronteira a partir de suas coordenadas paramétricas é feita pela equação (2.20).

Os objetos face também são utilizados para aplicar as condições de contorno ao problema. Tendo em vista a redução do esforço computacional, serão empregadas condições do tipo Dirichlet a não ser quando, na otimização da malha, as faces internas ao domínio forem modificadas. Para um nó genérico de fronteira k ($1 \leq k \leq 4m$), sua coordenada paramétrica t_k é dada por

$$t_k = k + g_k \tag{4.1}$$

onde k representa a parte inteira da coordenada e g_k sua parte decimal. Por exemplo, para t = 3.8, k = 3 e g = 0.8. Ou seja, após a normalização das distâncias, o ponto em questão está a uma distância de 0.8 unidades de comprimento do nó 3. Como a malha inicial é gerada com os nós igualmente espaçados na fronteira, na primeira iteração adota-se $g_k = 0$. A partir daí, inicia-se um processo de busca pelo conjunto ótimo de condições de contorno que minimize a distorção geométrica dos elementos de fronteira, alterando o valor de g.

Formula-se então o seguinte problema de otimização:

Dados:

- Um domínio bidimensional Ω, limitado pela fronteira $\Gamma = \Gamma_S \cup \Gamma_E \cup \Gamma_N \cup \Gamma_W$ (fig. 4.3);
- Uma Malha **M**, caracterizada por:
  - Um conjunto de nós internos **(x,y)** no interior do domínio,
  - A quantidade de elementos presentes em cada face do domínio (m),
  - Um conjunto de parâmetros $\{t\} = \{t_1, t_2, t_3, \ldots t_{4m}\}$ que definem as condições de contorno sobre a fronteira,
  - Um valor de mérito ε, que quantifica a distorção dos elementos localizados na fronteira Γ,

Obter:

Um conjunto de condições de contorno $\{t^*\}$ que minimize ε.

O processo de otimização consiste em medir a deformação dos elementos localizados na fronteira do domínio, e a partir deste valor, realocar os nós da fronteira, ou seja: redefinir as condições de contorno para o gerador elíptico, que então reposicionará os nós do interior do domínio. Este processo se repete até que se atinja um número máximo de iterações ou a média do erro se estabilize por duas iterações consecutivas. Para determinar o conjunto ótimo de condições de contorno foram utilizados dois métodos de busca, sendo um de busca genética e o outro de busca gradiente. Na seqüência de descrição do algoritmo, apresenta-se primeiramente a formulação do critério de otimização, e depois os métodos de busca.

4.2.1 O critério de otimização: cálculo da regularidade geométrica dos elementos.

Uma vez que todos os elementos são quadrilaterais, o critério adotado consiste em medir a distorção de cada elemento da fronteira em relação a um quadrado, e o desempenho da malha será medido pela soma destas distorções. Para o elemento da figura 4.4, a tabela 4.1 mostra as condições que determinam o desvio em relação à forma ideal, bem como as equações que quantificam este desvio.

Tabela 4.1 Condições a serem satisfeitas por um elemento quadrangular genérico

Condição	Equação	
1. A = F	$a = (x_0 - x_1) - (x_2 - x_3) = 0$	(4.2)
2. D = I	$b = (y_0 - y_1) - (y_2 - y_3) = 0$	(4.3)
3. D = F	$c = (y_1 - y_3) - (x_2 - x_3) = 0$	(4.4)
4. C = G	$d = (x_1 - x_3) - (y_3 - y_2) = 0$	(4.5)

O erro associado ao elemento será dado então por:

$$\varepsilon = a^2 + b^2 + c^2 + d^2 \tag{4.6}$$

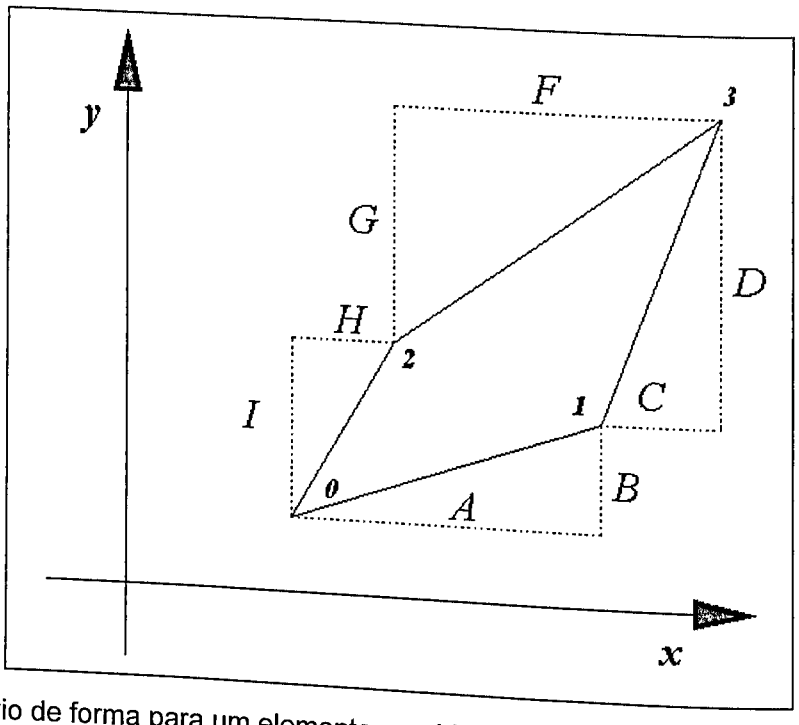


Figura 4.4 Desvio de forma para um elemento quadrilátero genérico.

Embora as condições adotadas sejam necessárias para que o elemento seja quadrado, elas não são suficientes. Entretanto, o desempenho deste critério foi testado contra outros que restringiam a forma do elemento a um quadrado estrito, mas estes sempre envolviam testes de ortogonalidade feitos via produtos internos, o que os tornava muito onerosos. Como a qualidade da malha obtida foi praticamente a mesma para todos os casos, optou-se por um critério menos rigoroso, a favor do desempenho computacional.

4.2.2 Métodos de Busca

4.2.2.1 Algoritmos Genéticos

Este método consiste em explorar o espaço de busca partindo de vários pontos simultaneamente, cada um codificando uma configuração das variáveis de projeto, gerada aleatoriamente. Cada ponto é chamado de indivíduo e o número total de pontos analisados é chamado de tamanho da população. Os valores das variáveis são chamados de genes.

A busca pelo ótimo começa avaliando cada indivíduo pela função objetivo, e este valor será sua aptidão, que determinará sua adaptação ao problema. Selecionam-se então os indivíduos mais aptos para formar uma população intermediária. Nesta população selecionam-

se alguns indivíduos aleatoriamente para que sejam modificados por operadores genéticos, normalmente chamados de mutação e cruzamento. Mutações consistem em sortear pontos no espaço de busca aleatoriamente e substituí-los por novos pontos também gerados aleatoriamente. Cruzamentos consistem em substituir dois indivíduos sorteados por combinações lineares dos dois, ponderadas por pesos também gerados aleatoriamente.

Como estes algoritmos não seguem uma direção de busca definida, seus principais parâmetros de controle estão relacionados à exploração do espaço de busca, que são o tamanho da população, responsável pelo número de pontos a serem analisados a cada iteração, e o número de iterações a serem executadas, aqui normalmente chamadas de gerações. Como este geralmente é o critério de parada, é importante estabelecer uma relação entre o tamanho da população e o número de gerações, a fim de se evitar que o processo se torne muito lento. É claro que, para um determinado número de gerações, quanto maior a população, maior o espaço de busca e a chance de se aproximar do ótimo global, porém maior será o custo computacional envolvido. Os outros parâmetros de controle, que são as taxas de ocorrência dos operadores genéticos, determinarão o número de pontos que serão efetivamente analisados a cada iteração. Por serem definidos em termos percentuais, é costume estabelecer uma razão inversa entre estas taxas e o tamanho da população.

Para que um AG seja implementado com sucesso, a implementação das seguintes etapas deve ser observada.

- Codificação das soluções potenciais.
- Formulação da função objetivo.
- Mecanismo de seleção para as gerações futuras.
- Implementação dos operadores genéticos do programa.
- Parâmetros de Controle:
  - Tamanho da população;
  - Critério de Parada;
  - Taxas de ocorrência de cruzamentos e mutações.

Neste caso, cada indivíduo representa uma malha, e as variáveis de projeto, que são as coordenadas paramétricas das condições de contorno, representam os genes. A função utilizada é a medida da deformação da malha, dada pelas equações 4.1 a 4.5.

O critério de seleção adotado foi o de seleção por torneio. Este critério consiste em selecionar o melhor indivíduo dentro de um conjunto de n indivíduos para a próxima geração. Este processo é repetido até que se forme uma nova população do tamanho da anterior. Um

valor típico indicado na literatura (Michalewickz, 1994) e utilizado aqui é n = 2. Ou seja, comparam-se os indivíduos dois a dois, selecionando o que apresentar maior aptidão.

O único operador genético utilizado foi o de mutação; como os genes estão representados em ponto flutuante, este operador consiste simplesmente em sortear o indivíduo, a posição da variável a ser alterada e depois alterá-la também aleatoriamente.

Leitores interessados em mais detalhes sobre este método têm à sua disposição vasta bibliografia sobre o assunto. Os trabalhos de Teixeira (TEIXEIRA, 2001) e Whitley (WITLEY, 1993) apresentam uma discussão breve e objetiva sobre o método, e são recomendados como leitura introdutória. Os livros de Michalewicz (Michalewicz, 1994), Goldberg (Goldberg, 1989) e Koza ((Koza, 1992) e (Koza, 1994)) tratam de diferentes enfoques sobre o assunto, mostrando a aplicabilidade do método não só como ferramenta de otimização, mas em diversos ramos da ciência e engenharia.

4.2.2.2 Busca por gradiente

O método de busca por gradiente usado neste trabalho é baseado no método da máxima descida, que atualiza as variáveis de projeto através de um passo ponderado na direção do gradiente da função objetivo, isto é:

$$X = X + \alpha \cdot \nabla F(X) \tag{4.7}$$

Neste trabalho, as variáveis de projeto são as coordenadas paramétricas dos nós de fronteira do domínio, e as variáveis dependentes são as distorções dos elementos que definem a fronteira. Portanto, interessa agora obter uma relação entre a variação do posicionamento destes nós e a distorção destes elementos, quantificada pelas eqs. 4.2 a 4.5.

Por analogia com a equação 4.7, a atualização de t_k na iteração i será dada por

$$t_k^{(i)} = \left(t_k - \Delta t_k \cdot \left(\frac{\partial \varepsilon}{\partial t_k} \right) \right)^{(i-1)} \tag{4.8}$$

Considerando que (ver equação 4.1) $t_k = k + g_k$, onde k é constante, tem-se

$$\Delta t_k = \Delta g_k$$

e a eq. 4.8 fica

$$g_k^{(i)} = \left(g_k - \Delta t_k \cdot \left(\frac{\partial \varepsilon}{\partial t_k} \right) \right)^{(i-1)} \tag{4.9}$$

Portanto,

$$g_k^{(i)} = \left(g_k - \Delta g_k \cdot \left(\frac{\partial \varepsilon}{\partial t_k} \right) \right)^{(i-1)} \tag{4.10}$$

onde g_k é a perturbação imposta à coordenada paramétrica e ε é a distorção ou erro de todos os elementos da fronteira analisada, ou seja, é o valor da função objetivo calculada para a iteração i. O sinal é negativo pois procura-se minimizar a função objetivo, isto é, caminha-se sempre na direção contrária ao gradiente.

No método clássico da máxima descida, o passo Δg_k deve ser atualizado a cada iteração, e mesmo assim o processo tende a se estagnar nas iterações finais, tornando-o lento. Por isso, foi utilizado um valor constante determinado experimentalmente de $\Delta g_k = 0.05$.

O desvio em cada elemento é dado pela equação 4.6, reescrita aqui:

$$\varepsilon = a^2 + b^2 + c^2 + d^2 \tag{4.6}$$

O erro associado à linha de elementos genérica representada na fig. 4.5 é então dado pela eq. 4.11. Nesta figura, os nós pertencentes à fronteira são indicados por (x_1, y_1), (x_2, y_2), ... e os nós localizados na linha de nós adjacentes à fronteira levam o super-escrito (*).

$$\varepsilon = \sum_{e=1}^{m} \left(a_e^{\,2} + b_e^{\,2} + c_e^{\,2} + d_e^{\,2} \right) = Sa + Sb + Sc + Sd \tag{4.11}$$

onde

$$Sa = \sum_{e=1}^{m} \left(a_e^{\,2} \right), \quad Sb = \sum_{e=1}^{m} \left(b_e^{\,2} \right), \quad Sc = \sum_{e=1}^{m} \left(c_e^{\,2} \right) \quad e \quad Sd = \sum_{e=1}^{m} \left(d_e^{\,2} \right)$$

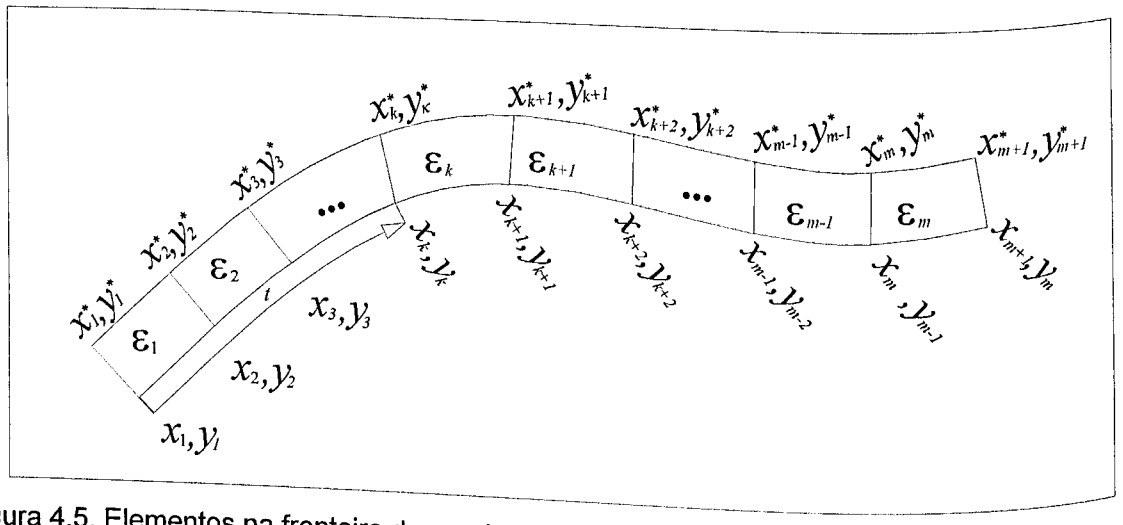


Figura 4.5. Elementos na fronteira de um domínio genérico.

Expandindo-se Sa, Sb, Sc e Sd a partir das eqs. (4.2) a (4.5), que quantificam as grandezas a, b, c, d, chega-se à eq. (4.12).

$$\begin{aligned} Sa = &\left[(x_1 - x_2) - (x_1^* - x_2^*)\right]^2 + \left[(x_2 - x_{3,}) - (x_2^* - x_3^*)\right]^2 + \cdots + \\ &\left[(x_{i-1} - x_i) - (x_{i-1}^* - x_i^*)\right]^2 + \left[(x_i - x_{i+1}) - (x_i^* - x_{i+1}^*)\right]^2 + \\ &\left[(x_{i+1} - x_{i+2}) - (x_{i+1}^* - x_{i+2}^*)\right]^2 + \cdots + \left[(x_{m-1} - x_m) - (x_{m-1}^* - x_m^*)\right]^2 \end{aligned} \tag{4.12a}$$

$$\begin{aligned} Sb = &\left[(y_1 - y_2) - (y_1^* - y_2^*)\right]^2 + \left[(y_2 - x_{3,}) - (y_2^* - y_3^*)\right]^2 + \cdots + \\ &\left[(y_{i-1} - y_i) - (y_{i-1}^* - y_i^*)\right]^2 + \left[(y_i - y_{i+1}) - (y_i^* - y_{i+1}^*)\right]^2 + \\ &\left[(y_{i+1} - y_{i+2}) - (y_{i+1}^* - y_{i+2}^*)\right]^2 + \cdots + \left[(y_{m-1} - y_m) - (y_{m-1}^* - y_m^*)\right]^2 \end{aligned} \tag{4.12b}$$

$$\begin{aligned} Sc = &\left[(y_1 - y_2) - (x_1^* - x_2^*)\right]^2 + \left[(y_2 - x_{3,}) - (x_2^* - x_3^*)\right]^2 + \cdots + \\ &\left[(y_{i-1} - y_i) - (x_{i-1}^* - x_i^*)\right]^2 + \left[(y_i - y_{i+1}) - (x_i^* - x_{i+1}^*)\right]^2 + \\ &\left[(y_{i+1} - y_{i+2}) - (x_{i+1}^* - x_{i+2}^*)\right]^2 + \cdots + \left[(y_{m-1} - y_m) - (x_{m-1}^* - x_m^*)\right]^2 \end{aligned} \tag{4.12c}$$

$$\begin{aligned} Sd = &\left[(x_2 - x_2^*) - (y_3^* - y_2^*)\right]^2 + \left[(x_2 - x_{3,}) - (y_2^* - y_3^*)\right]^2 + \cdots + \\ &\left[(x_{i-1} - x_i) - (y_{i-1}^* - y_i^*)\right]^2 + \left[(x_i - x_{i+1}) - (y_i^* - y_{i+1}^*)\right]^2 + \\ &\left[(x_{i+1} - x_{i+2}) - (y_{i+1}^* - y_{i+2}^*)\right]^2 + \cdots + \left[(x_{m-1} - x_m) - (y_{m-1}^* - y_m^*)\right]^2 \end{aligned} \tag{4.12d}$$

A variação do erro é calculada considerando-se que apenas os nós da fronteira estejam livres, uma vez que os nós interiores serão modificados pelo gerador elíptico e numa iteração serão considerados fixos. Então, as variáveis de projeto serão as coordenadas dos nós, que

podem ser representadas parametricamente pelo parâmetro t, definido no capítulo 2. As componentes do gradiente do erro serão calculadas pela eq.(4.13).

$$\frac{\partial \varepsilon}{\partial t_k} = \frac{\partial Sa}{\partial t_k} + \frac{\partial Sb}{\partial t_k} + \frac{\partial Sc}{\partial t_k} + \frac{\partial Sd}{\partial t_k} \tag{4.13}$$

onde

$$\frac{\partial Sa}{\partial t_k} = \frac{\partial Sa}{\partial x_k} \cdot \frac{\partial x_k}{\partial t_k} = \left\{\left[(x_{k,j} - x_{k+1,j}) - (x^*_{k-1} - x^*_{k+1})\right] - \left[(x_{k-1,j} - x_{k,j}) - (x^*_{k-1} - x^*_k)\right]\right\} \cdot \left(2 \cdot \frac{\partial x_k}{\partial t_k}\right)$$

$$\frac{\partial Sa}{\partial t_k} = 2 \cdot \frac{\partial x_k}{\partial t_k}\{a_k - a_{k-1}\} \tag{4.14a}$$

$$\frac{\partial Sb}{\partial t_k} = \frac{\partial Sb}{\partial y_k} \cdot \frac{\partial y_k}{\partial t_k} = \left\{\left[(y_k - y_{k+1}) - (y^*_k - y^*_{k+1})\right] - \left[(y_{k-1} - y_k) - (y^*_{k-1} - y^*_k)\right]\right\} \cdot \left(2 \cdot \frac{\partial y_k}{\partial t}\right)$$

$$\frac{\partial Sb}{\partial t_k} = 2 \cdot \frac{\partial y_k}{\partial t_k}\{b_k - b_{k-1}\} \tag{4.14b}$$

$$\frac{\partial Sc}{\partial t_k} = \frac{\partial Sc}{\partial y_k} \cdot \frac{\partial y_k}{\partial t_k} = \left[(y_k - y_{k+1}) - (x^*_k - x^*_{k+1})\right] \cdot \left(2 \cdot \frac{\partial y_k}{\partial t_k}\right)$$

$$\frac{\partial Sc}{\partial t_k} = 2 \cdot \frac{\partial y_k}{\partial t_k} \cdot c_k \tag{4.14c}$$

$$\frac{\partial Sd}{\partial t_k} = \left[(x_k - x_{k+1}) - (y^*_k - y^*_{k+1})\right] \cdot \left(2 \cdot \frac{\partial x_k}{\partial t_k}\right)$$

$$\frac{\partial Sd}{\partial t_k} = 2 \cdot \frac{\partial x_k}{\partial t_k} \cdot d_k \tag{4.14d}$$

onde a_k, b_k, c_k e d_k são dados pelas equações 4.2 a 4.5.

O gradiente do erro para cada elemento será dado então por

$$\nabla\varepsilon_k = 2 \cdot \left\{ \frac{\partial x_k}{\partial t_k} \cdot [a_k - a_{k-1} + d_k] + \frac{\partial y_k}{\partial t_k} \cdot [b_k - b_{k-1} + c_k] \right\} \tag{4.15}$$

O cálculo de $\frac{\partial x_k}{\partial t_k}$ e $\frac{\partial y_k}{\partial t_k}$ é feito derivando-se a eq. 2.17 em relação a t. O resultado é a equação 4.16 abaixo.

$$\begin{aligned} F_1'(t) &= 6 \cdot t^2 - 6 \cdot t \\ F_2'(t) &= -6 \cdot t^2 + 6 \cdot t \\ F_3'(t) &= 3 \cdot t^2 - 4 \cdot t + 1 \\ F_4'(t) &= 3 \cdot t^2 - 2 \cdot t \end{aligned} \tag{4.16}$$

A realocação dos nós que definem as condições de contorno será feita com base no sinal da equação 4.7 para cada elemento que define a fronteira do domínio, ou seja, a evolução do erro para o elemento determinará o reposicionamento dos nós de fronteira do modelo.

A figura 4.6 mostra um domínio quadrilateral simples, com os nós equiespaçados antes da aplicação das condições de contorno. Executando-se uma iteração obtém-se a malha da figura 4.7. Ao final do processo chega-se à malha da figura 4.8. A figura 4.9 mostra o fluxograma deste algoritmo.

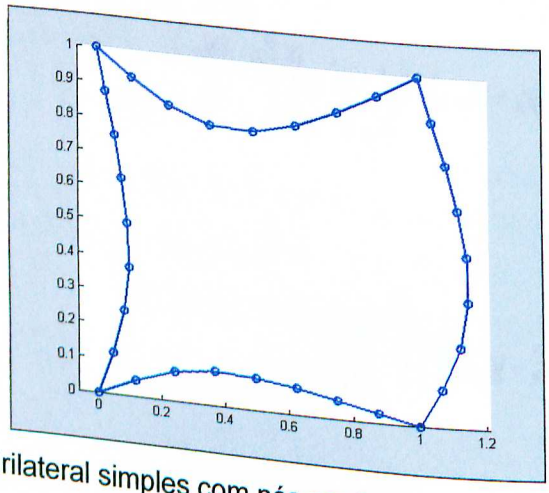


Figura 4.6: Domínio quadrilateral simples com nós equi-espaçados.

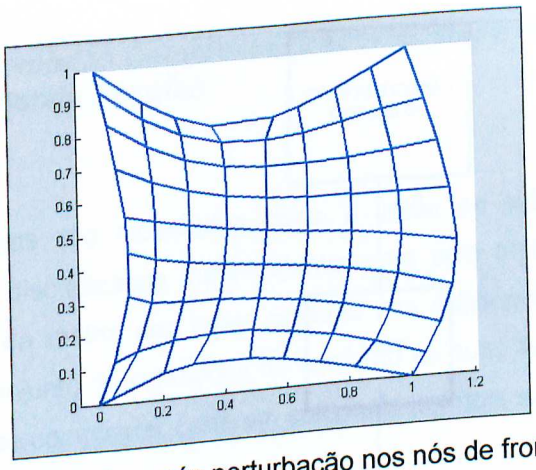


Figura 4.7: Posição dos nós e malha após perturbação nos nós de fronteira.

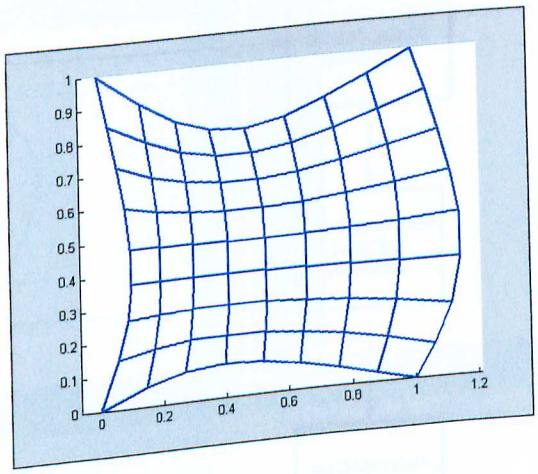


Figura 4.8: Malha final.

O caso acima foi apresentado apenas como exemplo do método. Numa geometria simples como esta, provavelmente a malha inicial seria aproximadamente ótima, e as condições de contorno não variariam tanto como mostrado nas figuras 4.7 e 4.8.

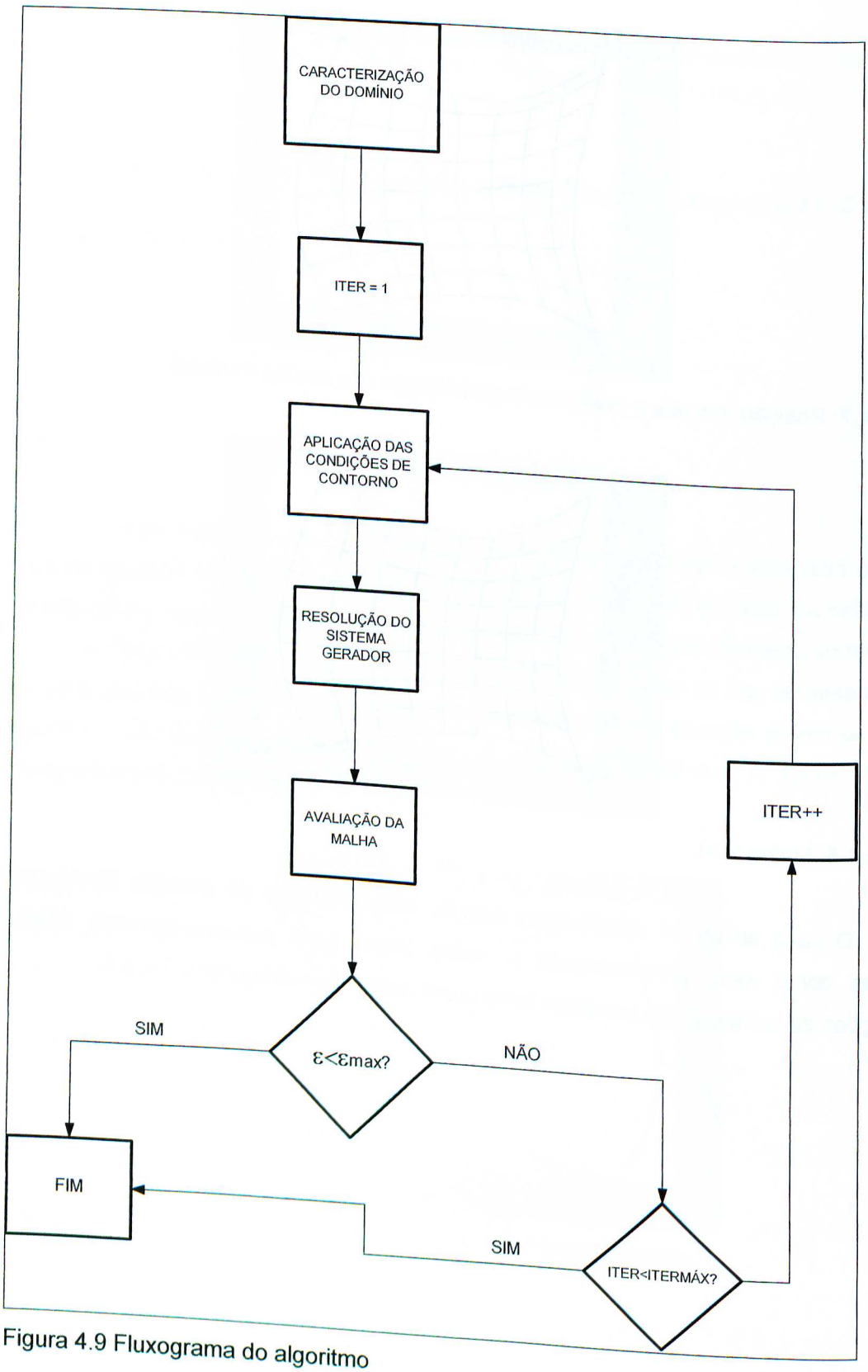
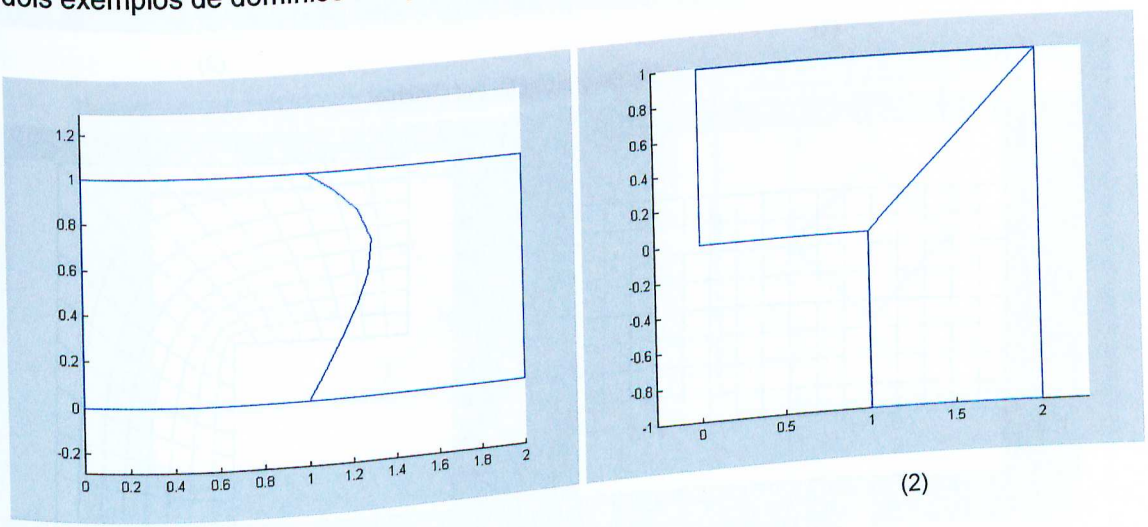


Figura 4.9 Fluxograma do algoritmo

4.3 Discretização de Domínios Complexos: Decomposição da Geometria em Subdomínios Quadrilaterais Simples

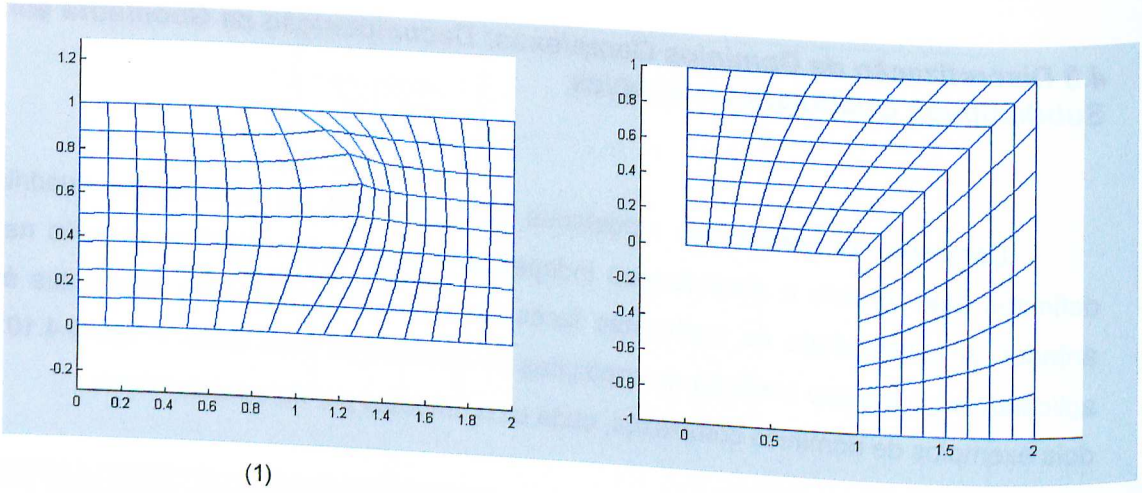
Domínios complexos são modelados como a união de subdomínios quadrilaterais, definidos pelo usuário e discretizados independentemente pelo algoritmo descrito na seção anterior. A continuidade da malha nas faces comuns aos subdomínios adjacentes é obtida aplicando-se o mesmo conjunto de condições de contorno às duas faces. A figura 4.10 mostra dois exemplos de domínios complexos, cada um composto por dois subdomínios.



(1) (2)

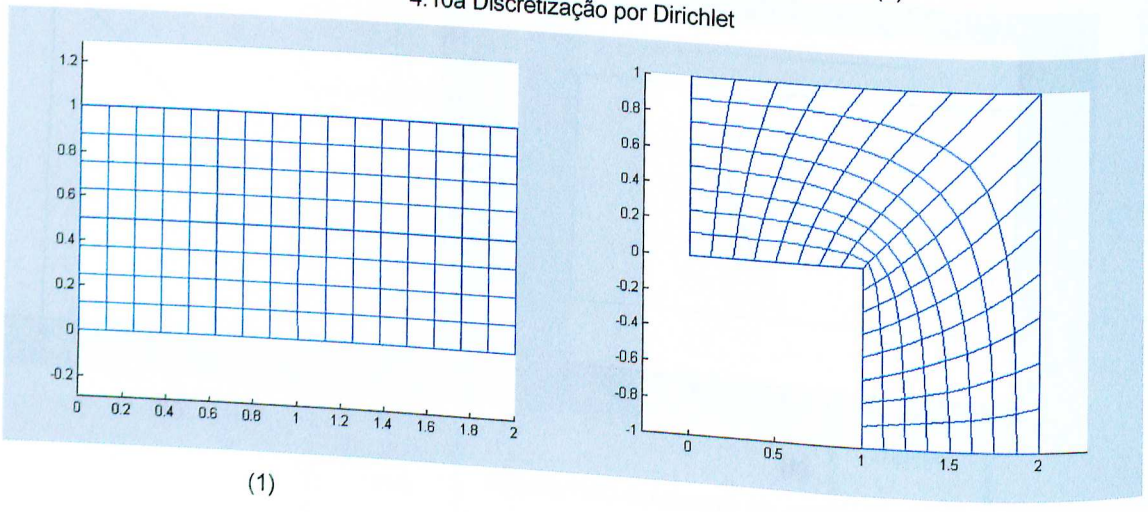
Figura 4.10: Dois casos de domínios complexos

A figura 4.11 apresenta a discretização destes domínios aplicando-se condições de Dirichlet (fig. 4.11a) e de Neumann (fig. 4.11b) às faces comuns. Note que, enquanto para o modelo 1 a continuidade por Dirichlet apresentou resultados piores, para o modelo 2 ocorreu o oposto. Assim, observa-se que nem sempre a suavidade na transição da malha oferece os melhores resultados. Entretanto, este método é bastante indicado quando há fronteiras curvas no problema, como será visto no capítulo 5.



(1) (2)

4.10a Discretização por Dirichlet



(1) (2)

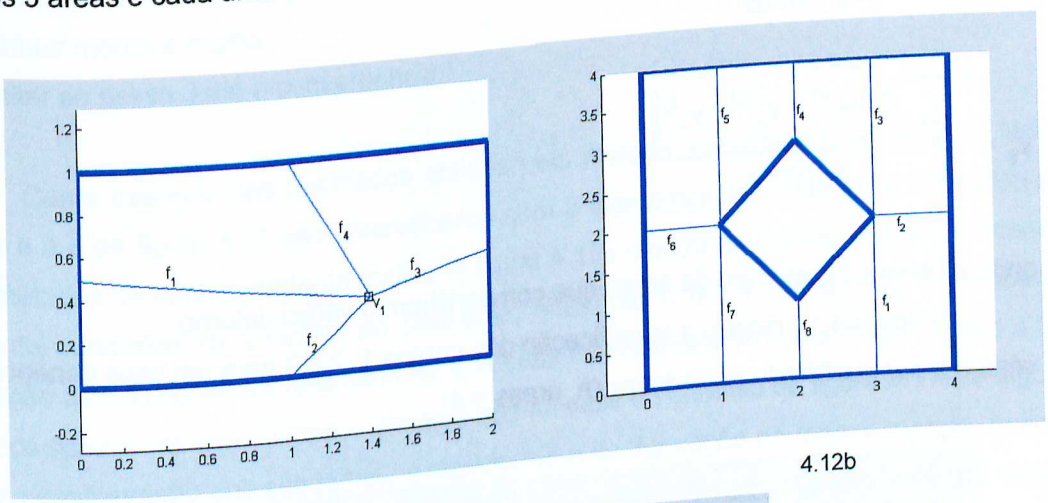
4.10b: Discretização por Neumann

Figura 4.10: Malhas geradas sobre os domínios apresentados na figura 4.9, com diferentes condições de continuidade.

Quando a forma das faces não é importante para a análise do problema, são tratadas como isolinhas, permitindo que o algoritmo as modifique, ou seja, reposicione livremente os nós localizados sobre elas, minimizando assim a distorção dos elementos vizinhos. O resultado é mostrado na figura 4.10b, para os dois casos. Para isto é necessário associar os nós localizados sobre a fronteira e seus vizinhos aos pontos N, S, E, W, NE, NW, SE, SW ilustrados na fig. (3.3), para que se possa resolver as equações 3.64 e 3.65 para estes nós durante a geração da malha. Neste caso, o nó P sempre se localizará sobre a fronteira comum e os demais pertencem a uma das duas áreas vizinhas. É claro que, além de P, os pontos N e S ou E e W também estarão sobre a fronteira, dependendo da configuração dos subdomínios.

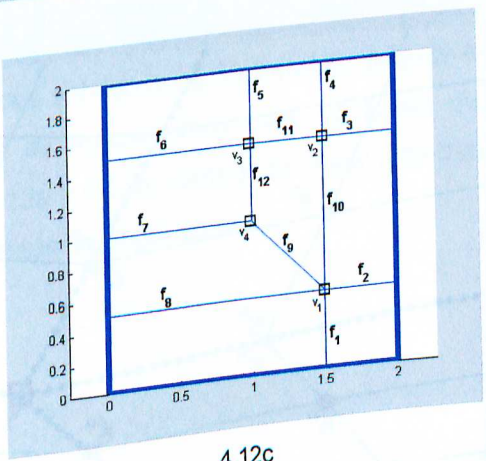
Em alguns casos, o modelo pode ter não somente faces internas (identificadas por $f_1, f_2, \ldots$ na fig. 4.11) mas também vértices internos ao domínio ($v_1, v_2, \ldots$), que também devem ser

realocados durante a alteração das faces na otimização da malha. A condição para que um vértice seja interno ao domínio é expressa observando-se os 3 casos da figura 4.11, onde as linhas finas indicam os subdomínios, e as linhas grossas as fronteiras do domínio. Embora o caso mais comum seja o de um vértice compartilhado por 4 áreas, como ilustrado na figura 4.11a, esta condição não é suficiente como se vê na fig. 4.11b. Comparando-se os dois casos nota-se que o vértice só é interno se, dado o conjunto de áreas que o compartilham, cada uma delas tiver duas vizinhas neste conjunto. A figura 4.11c mostra a generalização da condição: um ponto do domínio é um vértice interno se ele for compartilhado por um conjunto de pelo menos 3 áreas e cada uma tiver duas vizinhas neste conjunto.



4.12a

4.12b



4.12c

Figura 4.12: Critério para identificação de um vértice interno ao domínio.

Cada vértice interno é modelado como um objeto computacional, caracterizado pelas seguintes propriedades:

- Identificação dos subdomínios que o compartilham;
- Identificação das faces que o compartilham;

- Coordenadas locais do vértice e de seus vizinhos imediatos.

O reposicionamento do vértice interno é feito usando-se a equação 4.17. Ela pondera a posição deste ponto pela posição dos vizinhos, atribuindo peso 3 para os nós localizados *sobre* as faces que o compartilham ($x_1^{(i)}$ e $x_2^{(i)}$) e peso 1 para os nós vizinhos imediatos interiores ao domínio ($x_3^{(i)}$).

$$x_p = \frac{\sum_{i=1}^{n\_areas} \left[3 \cdot \left(x_1^{(i)} + x_2^{(i)}\right) + x_3^{(i)}\right]}{7 \cdot n\_areas}$$

$$y_p = \frac{\sum_{i=1}^{n\_areas} \left[3 \cdot \left(y_1^{(i)} + y_2^{(i)}\right) + y_3^{(i)}\right]}{7 \cdot n\_areas} \qquad (4.17)$$

onde ***n_areas*** é o número de áreas que compartilham o vértice interno.

A figura 4.13 mostra a identificação dos pontos 1, 2 e 3 para um caso genérico no qual o vértice é compartilhado por 4 áreas (n_areas = 4).

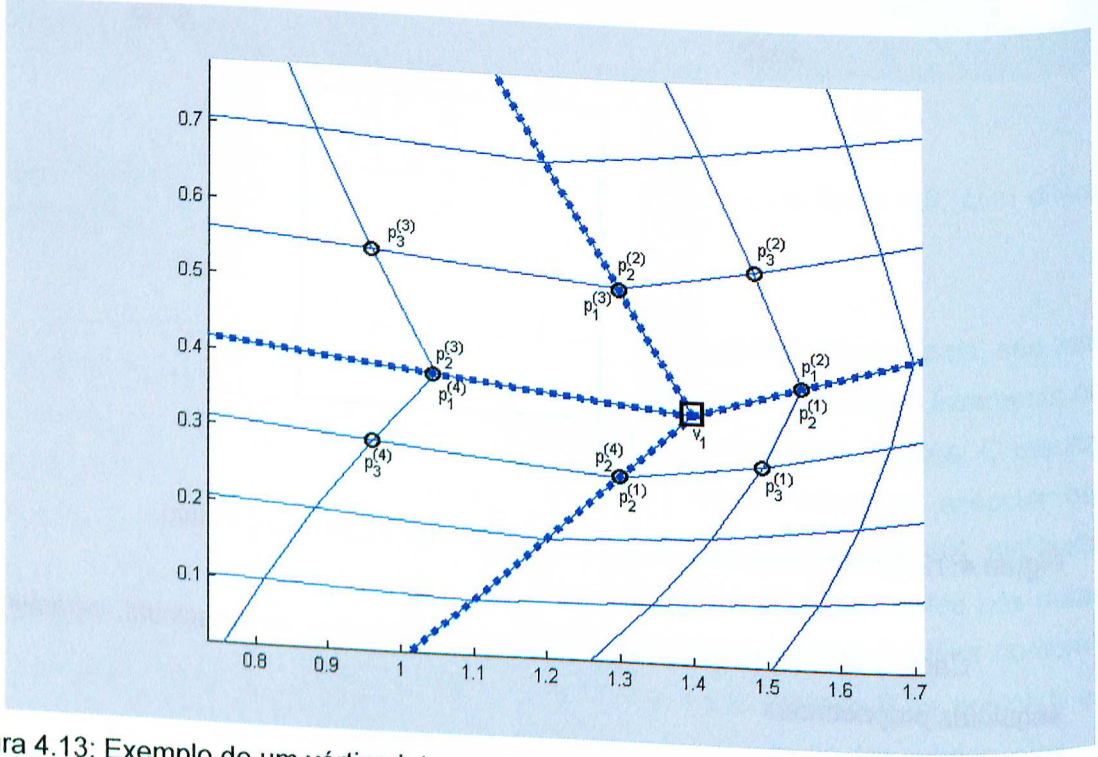


Figura 4.13: Exemplo de um vértice interno compartilhado por 4 áreas.

O algoritmo para geração e otimização de uma malha sobre um domínio bidimensional arbitrário pode ser assim resumido:

1. Definir a fronteira parametricamente a partir dos pontos de suporte fornecidos pelo usuário.
2. Analisar a conectividade dos subdomínios, identificando as áreas vizinhas, faces internas e nós internos ao domínio.
3. Aplicar um método de continuidade de malha.
4. Resolver o sistema de equações geradoras.
5. Realocar os vértices internos, se houverem.
6. Atribuir mérito à malha.
7. Voltar ao passo 3 até a convergência.

Como exemplo dos resultados obtidos pelo método, considere a figura 4.13. Nela, os casos a e c da figura 4.11 são discretizados com 8 elementos em cada face, utilizando-se os dois métodos de continuidade da malha. A figura 4.13a mostra as malhas obtidas aplicando-se somente condições de Dirichlet ao problema, enquanto a fig. 4.13b mostra o resultado da aplicação de condições de Neumann às fronteiras coincidentes. A figura 4.13c mostra a forma final dos subdomínios neste caso. Observe que no caso do domínio da figura 4.11a, o algoritmo levou a configuração dos subdomínios claramente à configuração de menor energia, buscando a forma retangular para os mesmos. No caso da figura 4.11c isto não foi possível, devido à configuração não simétrica dos subdomínios. Entretanto, nota-se que após a modificação dos subdomínios, os elementos se aproximaram mais da forma quadrada, como se observa comparando-se as figuras 4.13a e 4.13b, além de se obter uma transição mais suave da malha entre os subdomínios adjacentes no segundo caso.

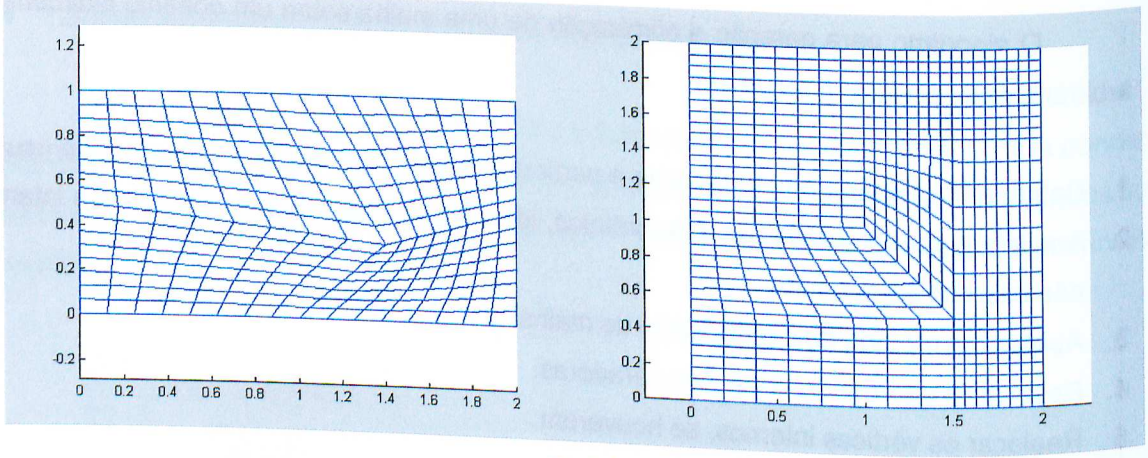


fig. 4.14a

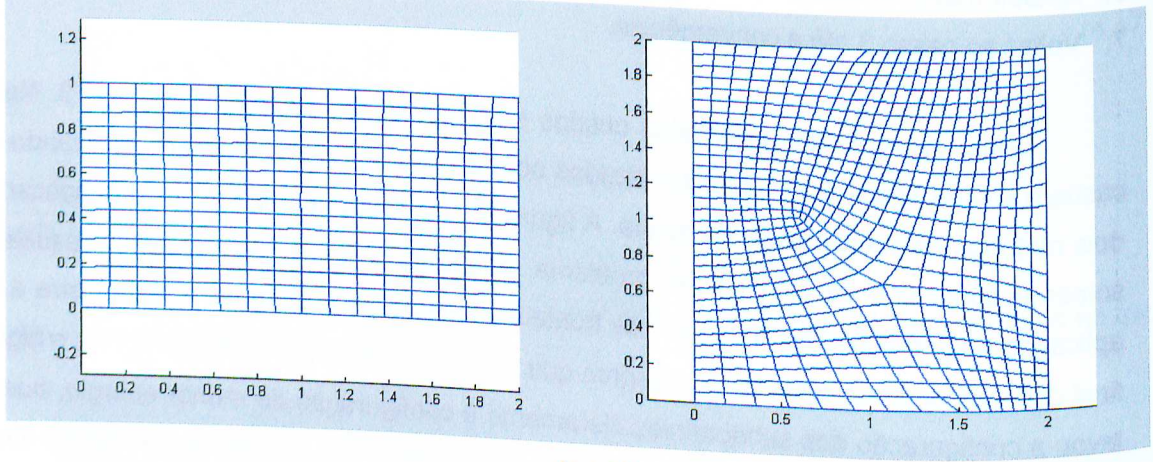


fig.4.14b

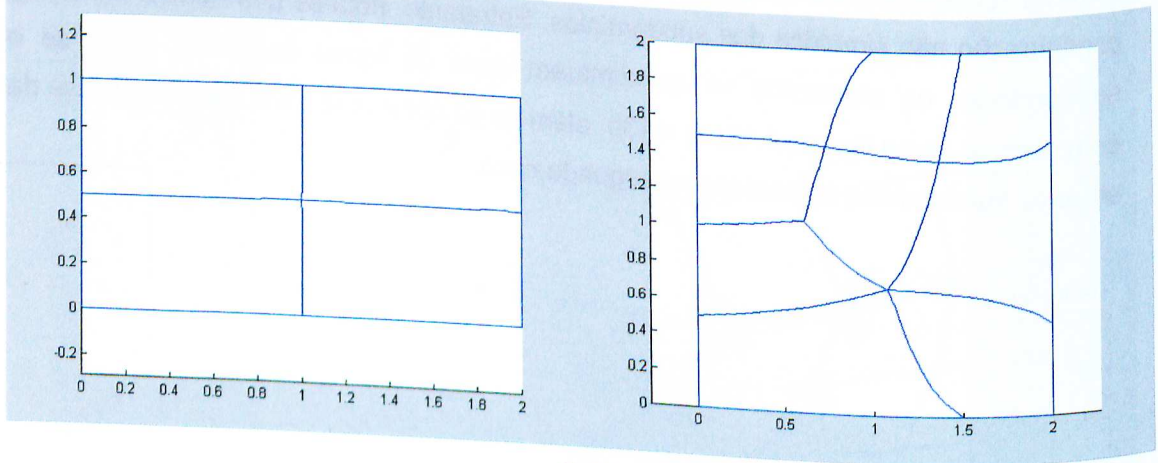


fig.4.14c

Figura 4.14: Efeito dos métodos de continuidade sobre a malha e a forma final dos subdomínios.

A figura 4.15 apresenta o algoritmo geral para o caso de um domínio composto.

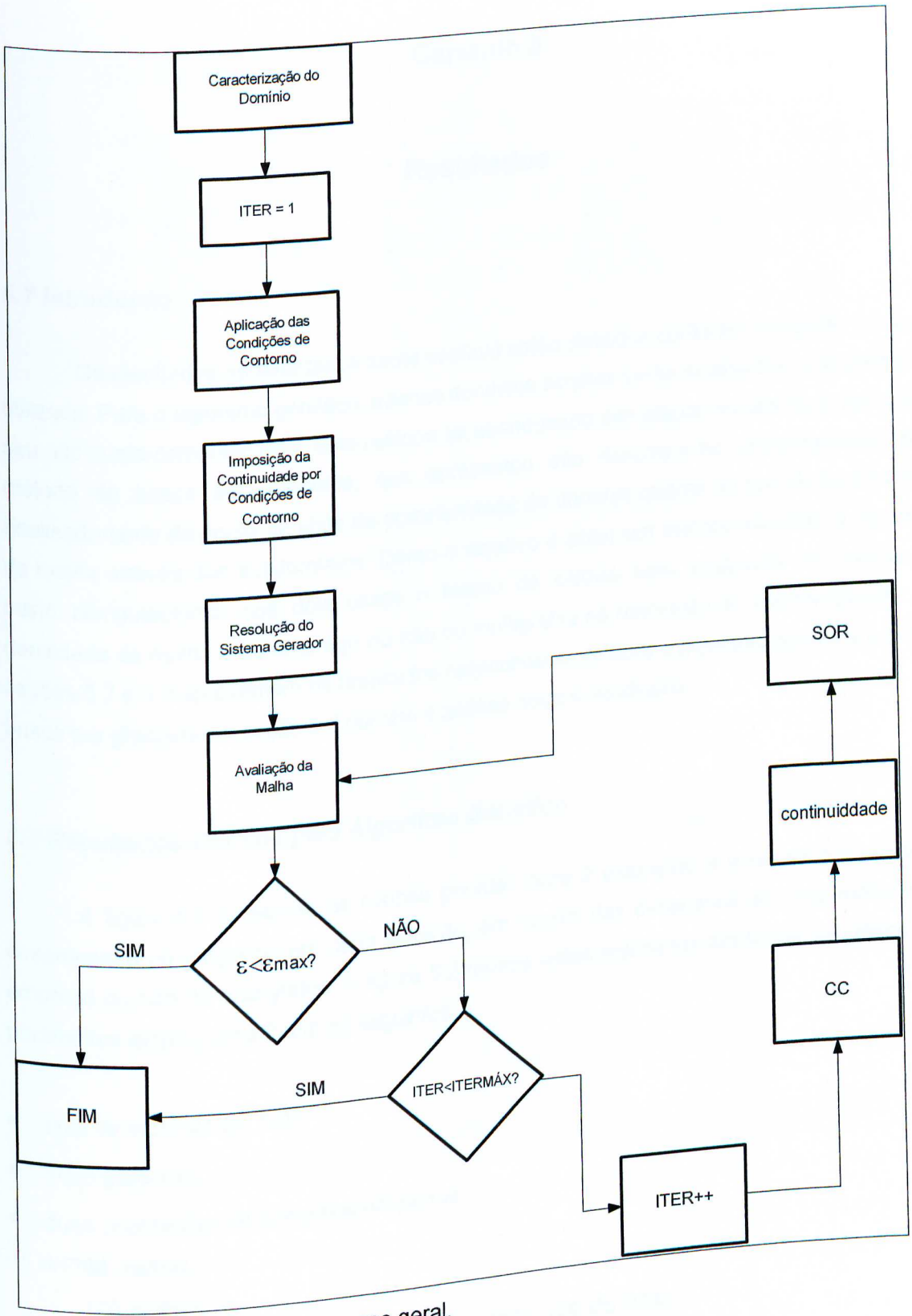


Figura 4.15: Fluxograma para o caso geral.

Capítulo 5

Resultados

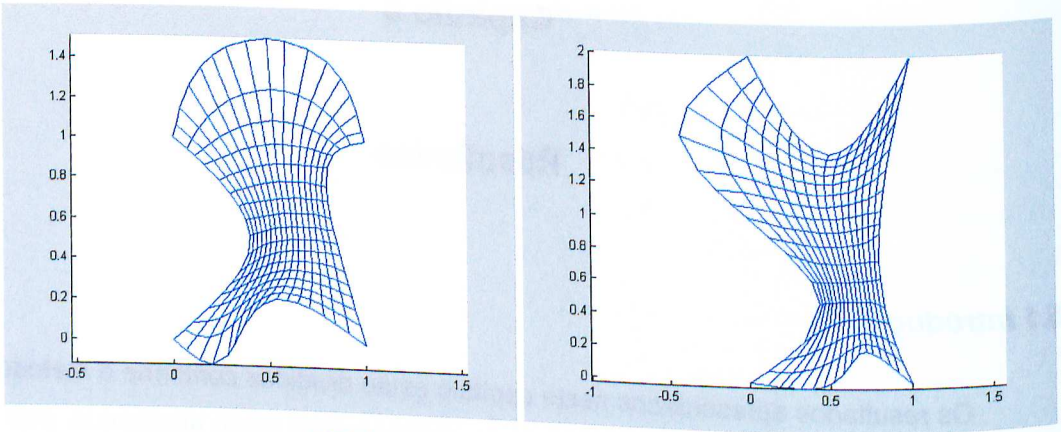
5.1 Introdução

Os resultados apresentados neste capítulo estão divididos conforme o método de busca utilizado. Para o algoritmo genético, apenas domínios simples serão analisados, pois devido ao seu alto custo computacional, este método foi abandonado nas etapas iniciais do projeto. Já o método de busca por gradiente, que apresentou alto desempenho computacional, será analisado tanto do ponto de vista da complexidade do domínio quanto do tipo de continuidade da malha através dos subdomínios. Como o objetivo é obter um método eficiente a um baixo custo computacional, nos dois casos o tempo de cálculo será analisado em função da densidade da malha e do emprego ou não de multigrelha na resolução do sistema gerador. As seções 5.2 e 5.3 apresentam os resultados respectivamente para o algoritmo genético e para a busca por gradiente. A seção 5.4 contém a análise destes resultados.

5.2 Resultados Obtidos pelo Algoritmo Genético

A figura 5.1 apresenta as malhas geradas para 2 exemplos e a tabela 5.1 mostra o desempenho do programa em cada situação, em função das dimensões de cada malha e do emprego ou não de multigrelha. A figura 5.2 mostra estes resultados em forma de gráfico. Os parâmetros empregados foram os seguintes:

- taxa de mutação de 0.02;
- 1000 gerações;
- duas populações de tamanhos diferentes:
  - 50 malhas;
  - 100 malhas.
- Recursos Computacionais: Pentium III 450 MHz, 128 Mb RAM.



caso 1 caso 2

Figura 5.1: Malhas resultantes para os dois casos resolvidos pelo algoritmo genético

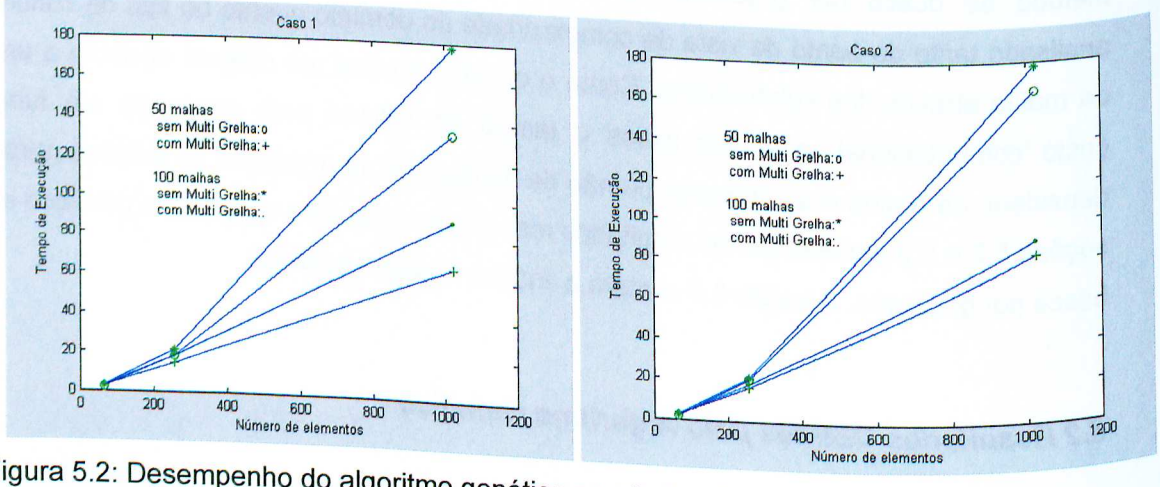


Figura 5.2: Desempenho do algoritmo genético na otimização das malhas.

Tabela 5.1: Desempenho do algoritmo genético na otimização das malhas.

	Número de elementos	Multi Grelha	Tempo em segundos	
			50 indivíduos	100 indivíduos
Caso 1	64	Não	2.3	2.5
		Sim	2.6	2.9
	256	Não	18.2	21.2
		Sim	14.0	18.0
	1024	Não	133.7	177.9
		Sim	66.2	89.1
Caso 2	64	Não	2.3	2.6
		Sim	2.7	2.9
	256	Não	20.0	21.3
		Sim	16.6	18.1
	1024	Não	165.2	177.8
		Sim	83.1	89.5

5.3 Resultados Obtidos pelo Método do Gradiente

Os casos estudados dividem-se em 3 categorias: domínios multiplamente conexos, domínios decompostos em subdomínios distorcidos e influência da configuração dos subdomínios no resultado da malha. Na primeira, procura-se analisar a suavidade da malha na presença de descontinuidades geométricas. Na segunda, são analisados domínios que foram divididos em subdomínios distorcidos, para testar a capacidade do algoritmo de minimizar a distorção dos elementos pela realocação dos subdomínios. Na terceira, apresentam-se dois casos para analisar a influência da configuração dos subdomínios no resultado da malha.

Para cada caso estão ilustrados o domínio, sua divisão em subdomínios e as malhas geradas para cada condição de continuidade. O desempenho do algoritmo é apresentado sob a forma de tabelas e gráficos.

5.3.1 Domínios com descontinuidades geométricas

Serão apresentados quatro casos típicos (figs. 5.3 a 5.6) de domínios multiplamente conexos. As figuras da esquerda mostram o domínio e as figuras da direita mostram a configuração dos subdomínios.

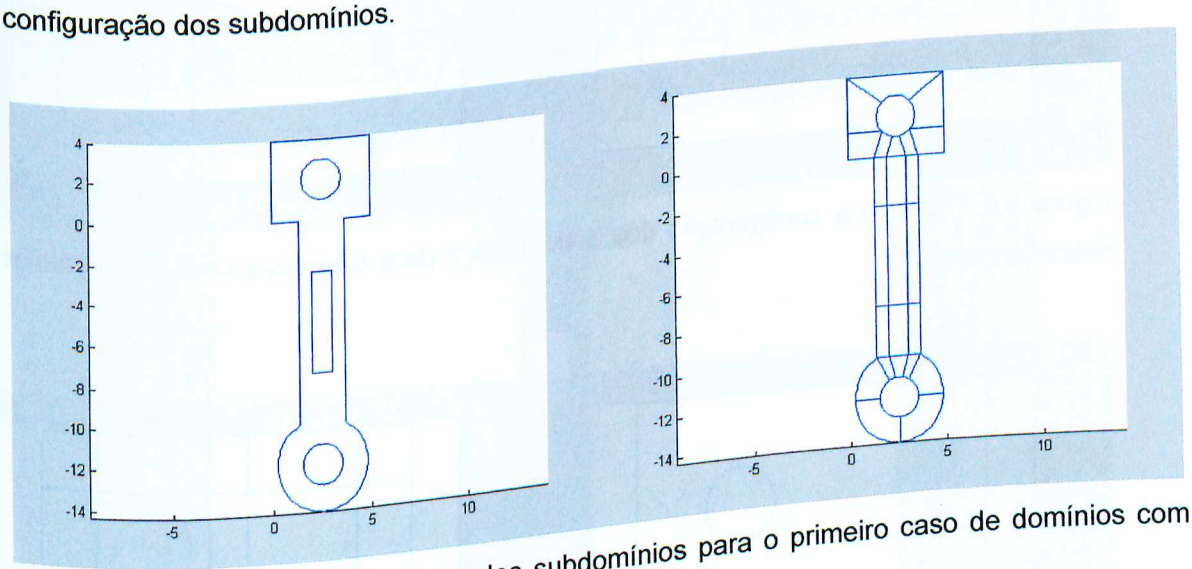


Figura 5.3: Domínio e configuração dos subdomínios para o primeiro caso de domínios com descontinuidades.

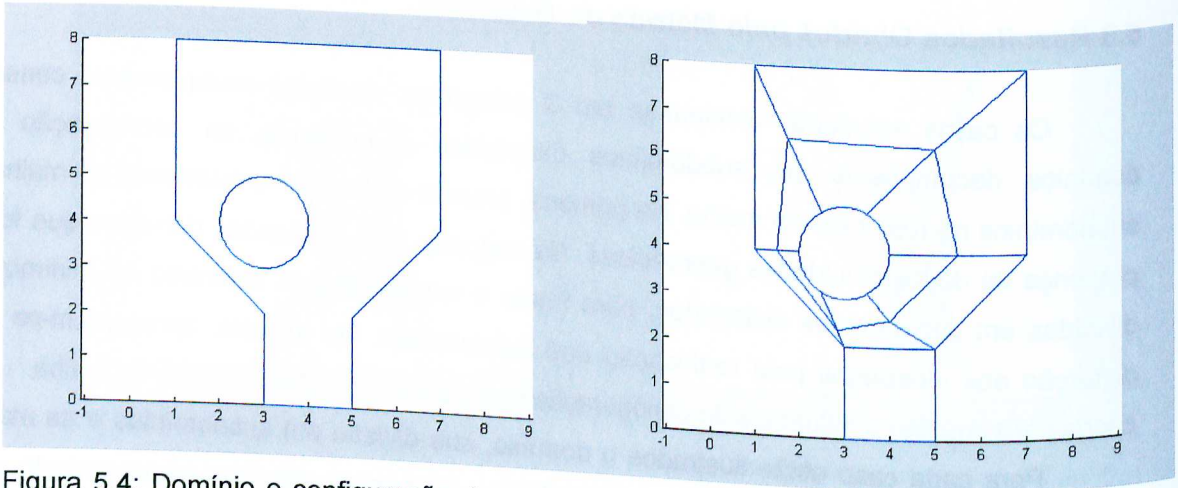


Figura 5.4: Domínio e configuração dos subdomínios para o segundo caso de domínios com descontinuidades.

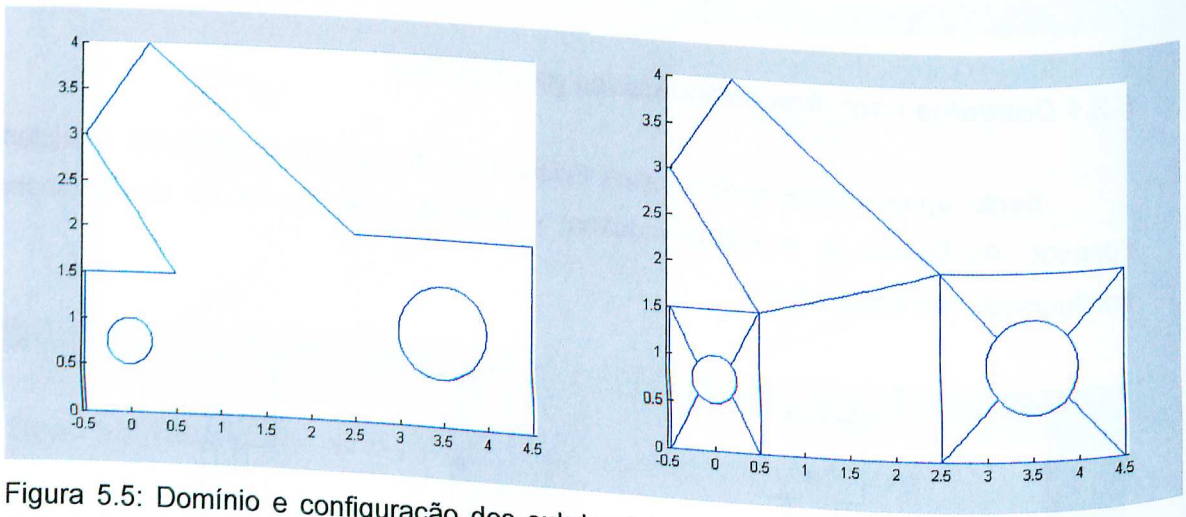


Figura 5.5: Domínio e configuração dos subdomínios para o terceiro caso de domínios com descontinuidades.

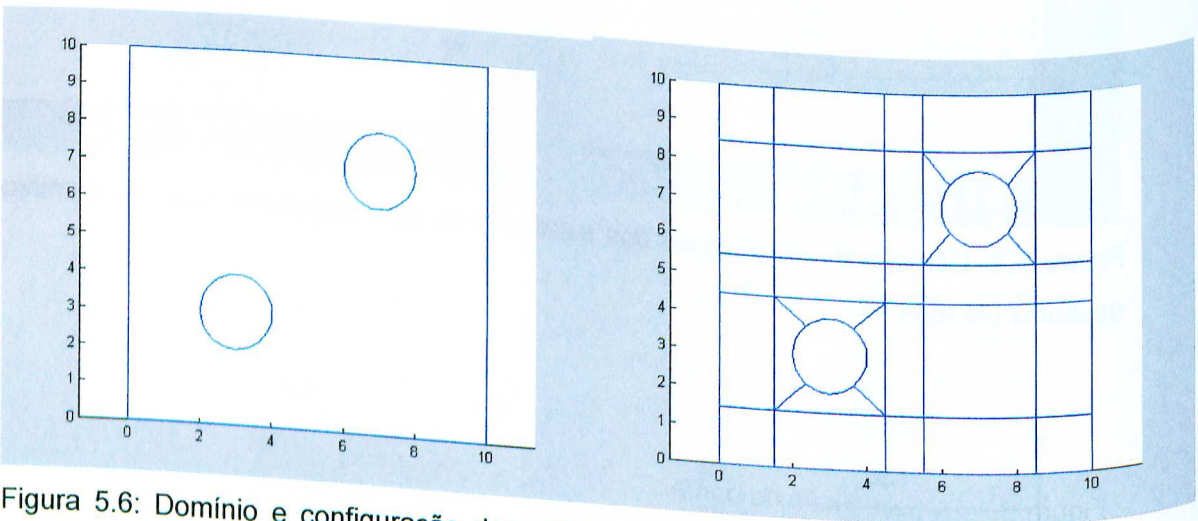


Figura 5.6: Domínio e configuração dos subdomínios para o quarto caso de domínios com descontinuidades.

As figuras 5.7 a 5.10 apresentam dois exemplos de malhas geradas sobre cada domínio analisado. As figuras superiores mostram a malha obtida impondo-se condições de Dirichlet nas fronteiras comuns, e as figuras inferiores mostram o resultado quando se aplicam condições de Neumann e permite-se ao algoritmo alterar os subdomínios.

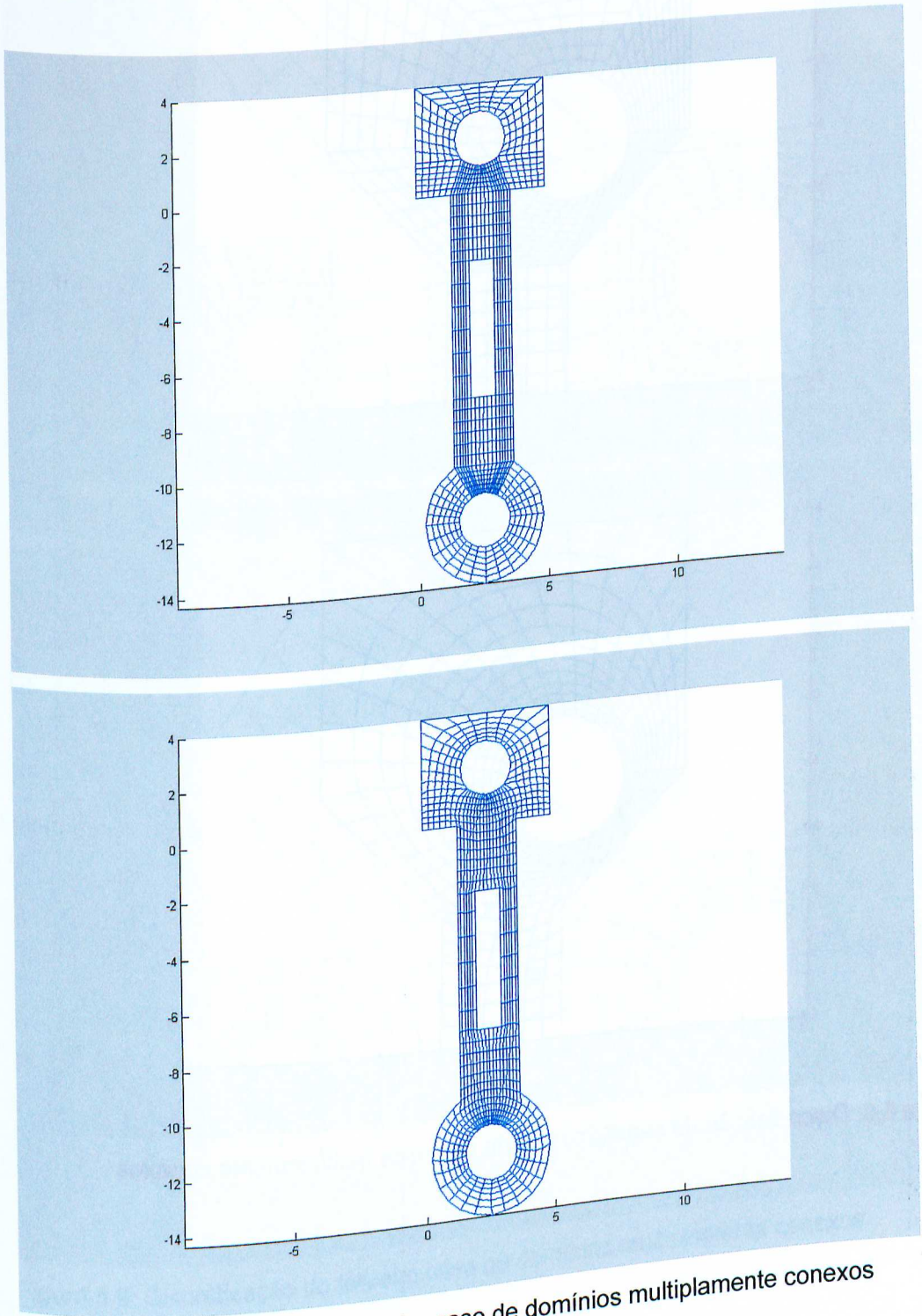
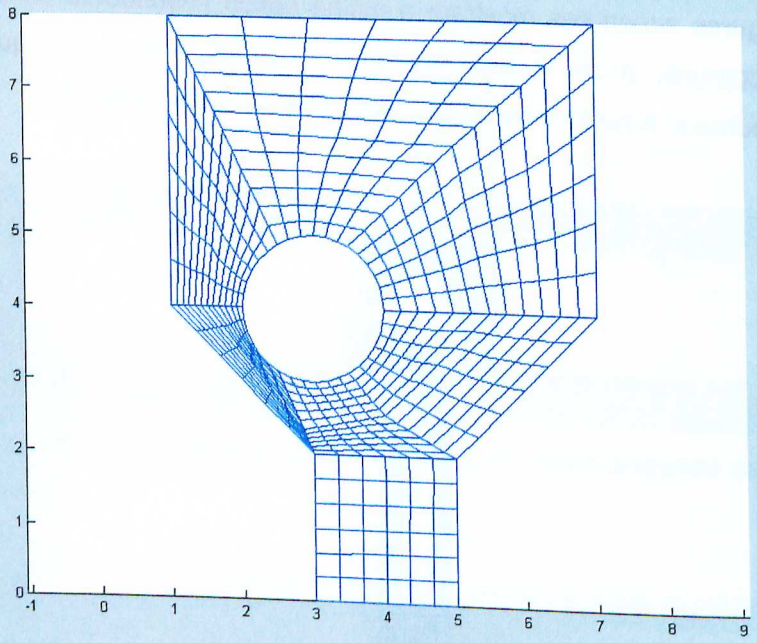


Figura 5.7: Discretização do primeiro caso de domínios multiplamente conexos



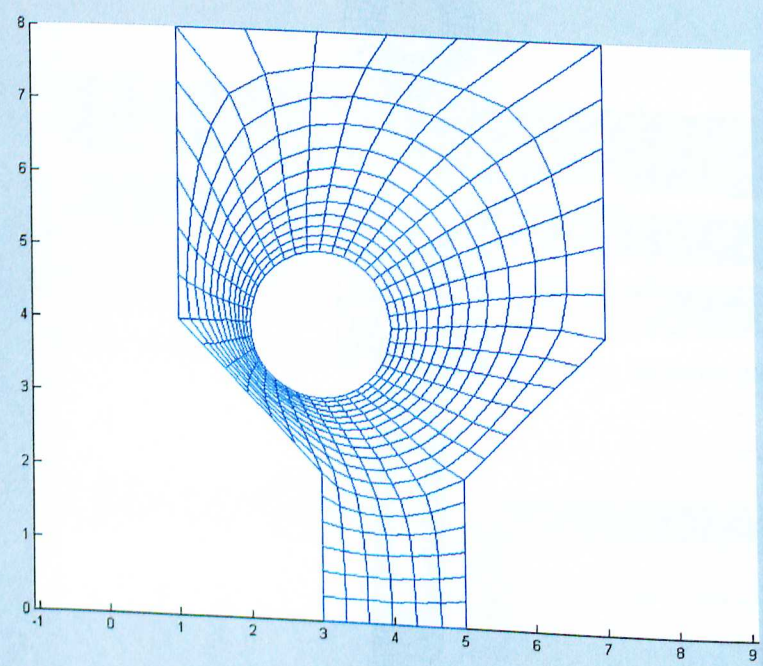


Figura 5.8: Discretização do segundo caso de domínios multiplamente conexos

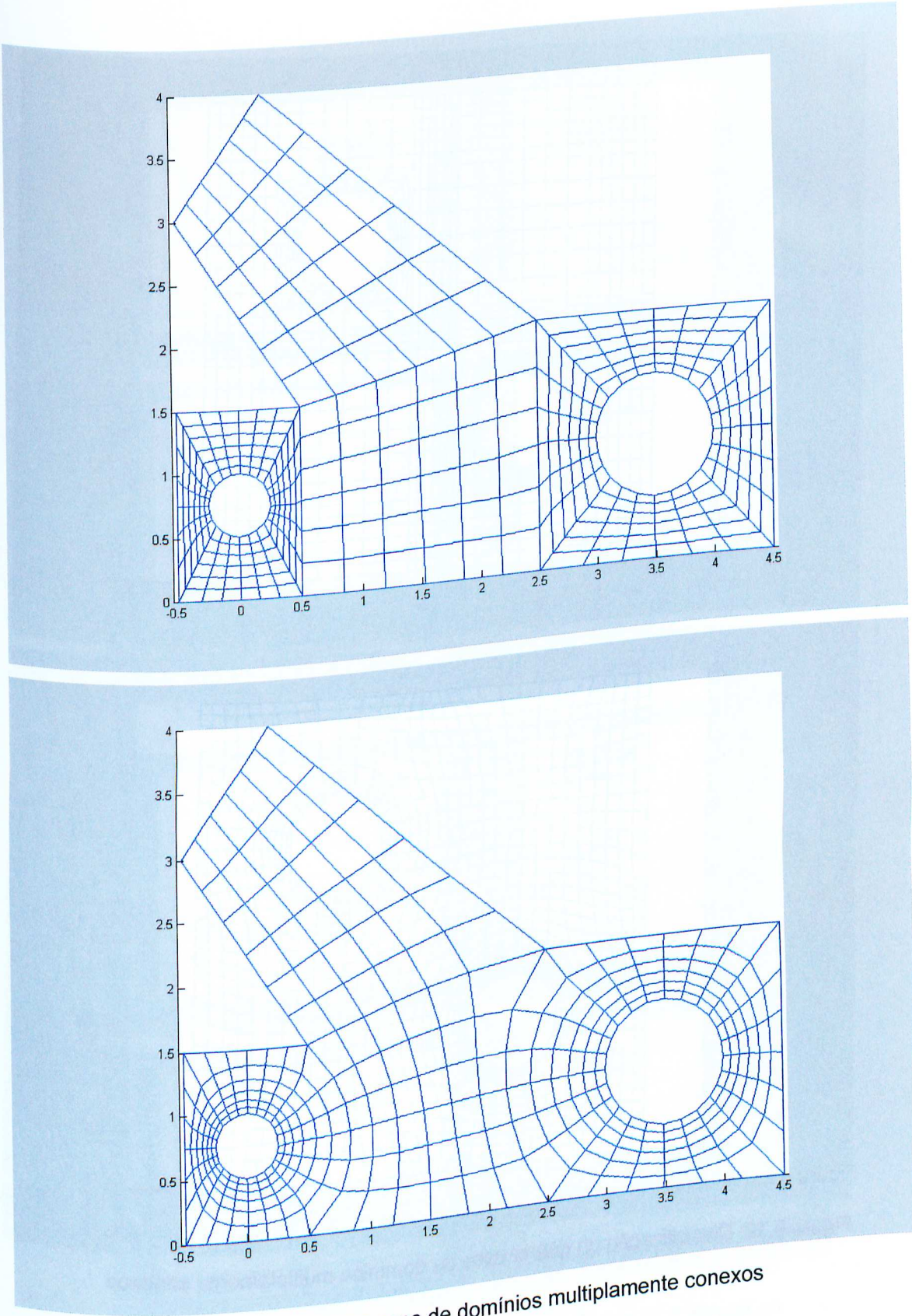


Figura 5.9: Discretização do terceiro caso de domínios multiplamente conexos

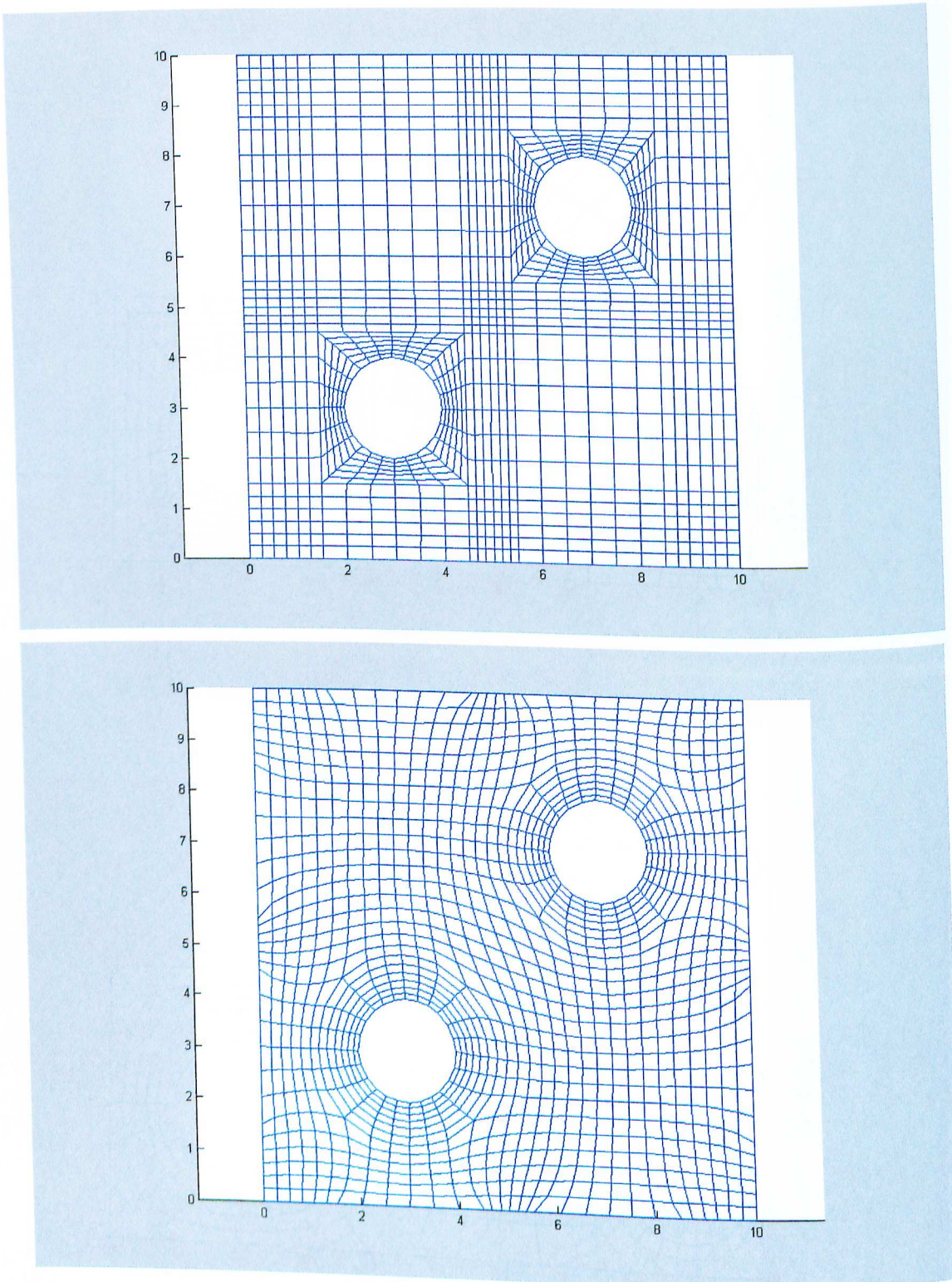


Figura 5.10: Discretização do quarto caso de domínios multiplamente conexos

As figuras 5.11 e 5.12 mostram o desempenho do algoritmo para cada caso em função do número de elementos. Para cada caso são analisados o tempo de execução (em segundos), o número de iterações e o erro máximo obtido.

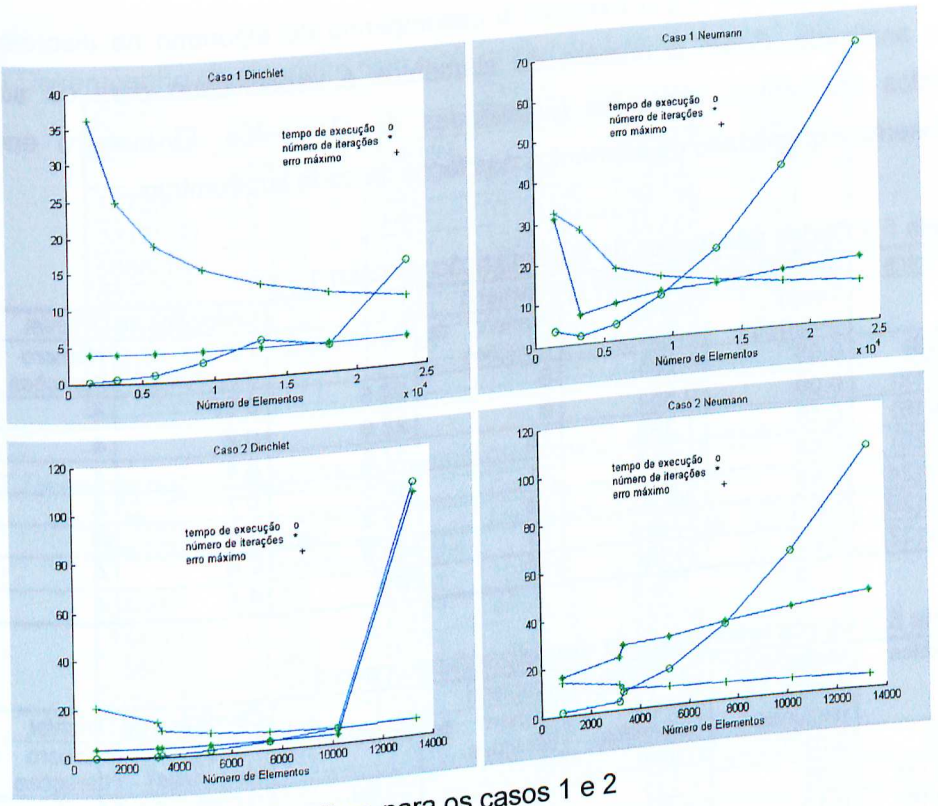


Figura 5.11: Desempenho do algoritmo para os casos 1 e 2

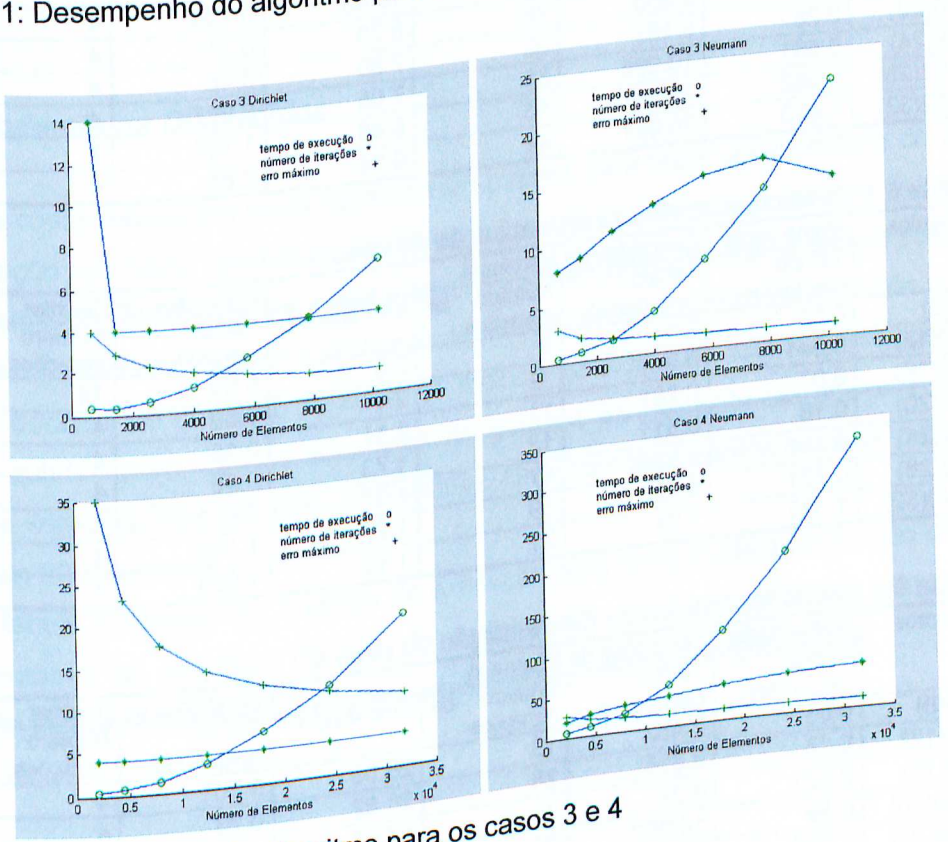


Figura 5.12: Desempenho do algoritmo para os casos 3 e 4

As tabelas 5.2 a 5.5 mostram o desempenho do algoritmo na discretização de cada caso analisado. Como o número de elementos é especificado para os subdomínios, os modelos apresentam diferentes quantidades de elementos. O número entre parênteses representa a quantidade de elementos nas faces de cada subdomínio.

Tabela 5.2 Dados resultantes da discretização do caso 1

Elementos	Malha Inicial	Condições de Neumann			Condições de Dirichlet		
	Tempo (segundos)	Tempo (segundos)	Número de Iterações	Erro Máximo	Tempo (segundos)	Número de Iterações	Erro Máximo (valor abs.)
1472(8)	0.05	0.281	6	33.5	0.09	4	36.8
3312(12)	0.09	0.851	8	22.9	0.2	4	24.8
5888(16)	0.16	1.382	10	17.6	0.4	4	18.7
9200(20)	0.44	3.165	11	14.5	0.7	4	15.0
13248(24)	0.99	6.9	13	12.4	1.5	4	12.6
18032(28)	1.9	13.0	15	10.9	2.6	4	10.8
23552(32)	3.2	21.35	16	9.8	4.4	4	9.5

Tabela 5.3 Dados resultantes da discretização do caso 2

Elementos	Malha inicial	Condições de Neumann			Condições de Dirichlet		
	Tempo (segundos)	Tempo (segundos)	Número de Iterações	Erro Máximo	Tempo (segundos)	Número de Iterações	Erro Máximo
832(8)	0.03	0.511	17	13.89	0.070	4	20.89
3172(12)	0.05	1.222	23	10.45	0.130	4	14.34
3328(16)	0.10	2.363	28	8.70	0.211	4	10.95
5200(20)	0.26	5.197	33	7.80	0.48	4	8.86
7488(24)	0.561	10.996	38	7.05	0.941	4	7.44
10192(28)	1.062	20.599	43	6.48	1.703	4	6.42
13312(32)	1.833	35.081	48	6.04	2.85	4	5.63

Tabela 5.4 Dados resultantes da discretização do caso 3

Elementos	Malha Inicial	Condições de Neumann			Condições de Dirichlet		
	Tempo (segundos)	Tempo (segundos)	Número de Iterações	Erro Máximo	Tempo (segundos)	Número de Iterações	Erro Máximo
640(8)	0.02	0.11	8	2.9	0.05	14	4.0
1440(12)	0.041	0.271	11	2.11	0.09	4	2.83
2560(16)	0.07	0.281	11	2.11	0.15	4	2.18
4000(20)	0.18	0.511	13	1.73	0.45	4	1.78
5760(24)	0.411	2.534	17	1.37	0.671	4	1.5
7840(28)	0.781	4.496	19	1.27	1.182	4	1.30
10240(32)	1.322	7.25	21	1.18	1.953	4	1.14

Tabela 5.5 Dados resultantes da discretização do caso 4

Elementos	Malha Inicial	Condições de Neumann			Condições de Dirichlet		
	Tempo (segundos)	Tempo (segundos)	Número de Iterações	Erro Máximo	Tempo (segundos)	Número de Iterações	Erro Máximo
1984(8)	0.05	2.452	18	26.56	0.101	4	30.412
4464(12)	0.13	3.335	25	20.82	0.23	4	20.44
7936(16)	0.221	6.38	31	17.63	0.541	4	15.42
12400(20)	0.59	14.982	36	15.54	0.861	4	12.38
17856(24)	1.502	31.465	42	14.07	1.783	4	10.36
24304(28)	2.574	57.233	47	12.95	3.255	4	8.89
31744(32)	4.437	101.115	52	12.15	5.448	4	7.8

Tabela 5.6 Desempenho do algoritmo com multi grelha de 1 nível, casos 1 e 2

Caso 1			Caso 2		
Elementos	Tempo		Elementos	Tempo	
	Neumann	Dirichlet		Neumann	Dirichlet
1472(8)	0.371	0.26	832(8)	1.943	0.08
3312(12)	2.073	0.45	3172(12)	10.755	0.17
5888(16)	2.714	0.571	3328(16)	26.698	0.41
9200(20)	5.859	1.302	5200(20)	42.151	0.771
13248(24)	10.245	1.993	7488(24)	47.178	1.342
18032(28)	17.606	3.144	10192(28)	68.698	2.273
23552(32)	28.271	5.167	13312(32)	144.177	4.506

Tabela 5.7 Desempenho do algoritmo com multi grelha de 1 nível, casos 3 e 4

Caso 3			Caso 4		
Elementos	Tempo		Elementos	Tempo	
	Neumann	Dirichlet		Neumann	Dirichlet
640(8)	0.16	0.091	1984(8)	1.913	0.120
1440(12)	0.53	0.261	4464(12)	5.407	0.300
2560(16)	0.991	0.271	7936(16)	12.538	0.701
4000(20)	2.073	0.501	12400(20)	27.369	1.502
5760(24)	3.635	0.881	17856(24)	51.794	2.393
7840(28)	6.319	1.432	24304(28)	98.011	3.815
10240(32)	10.375	2.364	31744(32)	168.703	6.48

5.3.2 Subdomínios Distorcidos

Os casos estudados nesta seção foram usados para testar a capacidade do algoritmo de realocação das faces internas ao domínio. Os resultados, apresentados nas figuras 5.13 a 5.17, constam das configurações inicial e final dos subdomínios, da malha gerada aplicando-se condições de Neumann durante o processo de otimização e do gráfico da evolução do erro da malha no decorrer do processo iterativo. As malhas apresentadas têm 8 elementos em cada face dos subdomínios. São apresentados os gráficos da evolução do erro em função do número de iterações para três malhas diferentes. Uma com 8 elementos por face, outra com 20 e outra com 32 elementos.

A tabela 5.8 apresenta o tempo de execução para cada modelo em função do número de elementos e do emprego ou não de multi grelha. Mais uma vez, os números entre parênteses indicam a quantidade de elementos em cada face dos subdomínios.

Tabela 5.8. Desempenho do algoritmo na otimização das faces internas dos domínios.

Caso	Número de elementos	Tempo de execução	
		Com multi grelha	Sem multi grelha
1	12288(32)	47.2	36.2
	4800(20)	2.8	1.6
	1280(8)	0.6	0.4
2	9216(9)	17.0	13.0
	3600(20)	3.9	2.4
	576(9)	0.3	0.2
3	11264(32)	15.1	11.4
	4400(20)	3.0	1.8
	704(8)	0.2	0.1
4	6144(32)	17.4	7.5
	2400(20)	2.0	1.1
	384(8)	0.1	0.1
5	5120(32)	29.202	20.339
	2000(20)	15.000	3.205
	320(8)	1.783	0.441

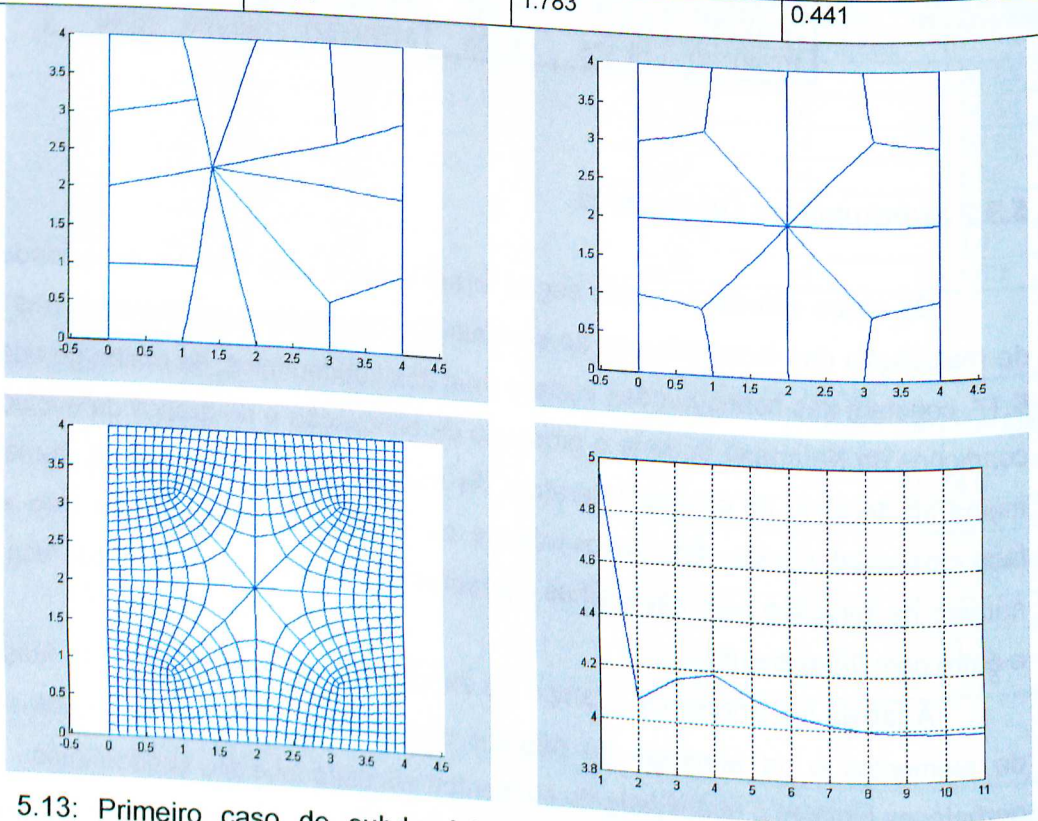
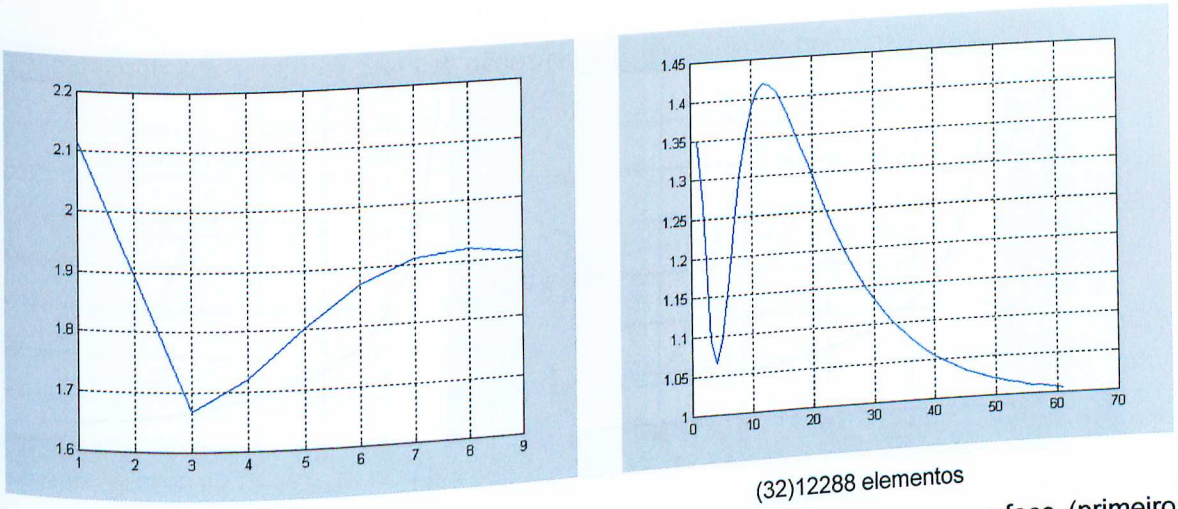


Figura 5.13: Primeiro caso de subdomínios distorcidos. Configuração inicial e final dos subdomínios, malha com 8 elementos por face e evolução do erro no decorrer do processo iterativo.



4800(20) elementos

(32)12288 elementos

Figura 5.14: Evolução do erro para subdomínios com 20 e 32 elementos por face (primeiro caso).

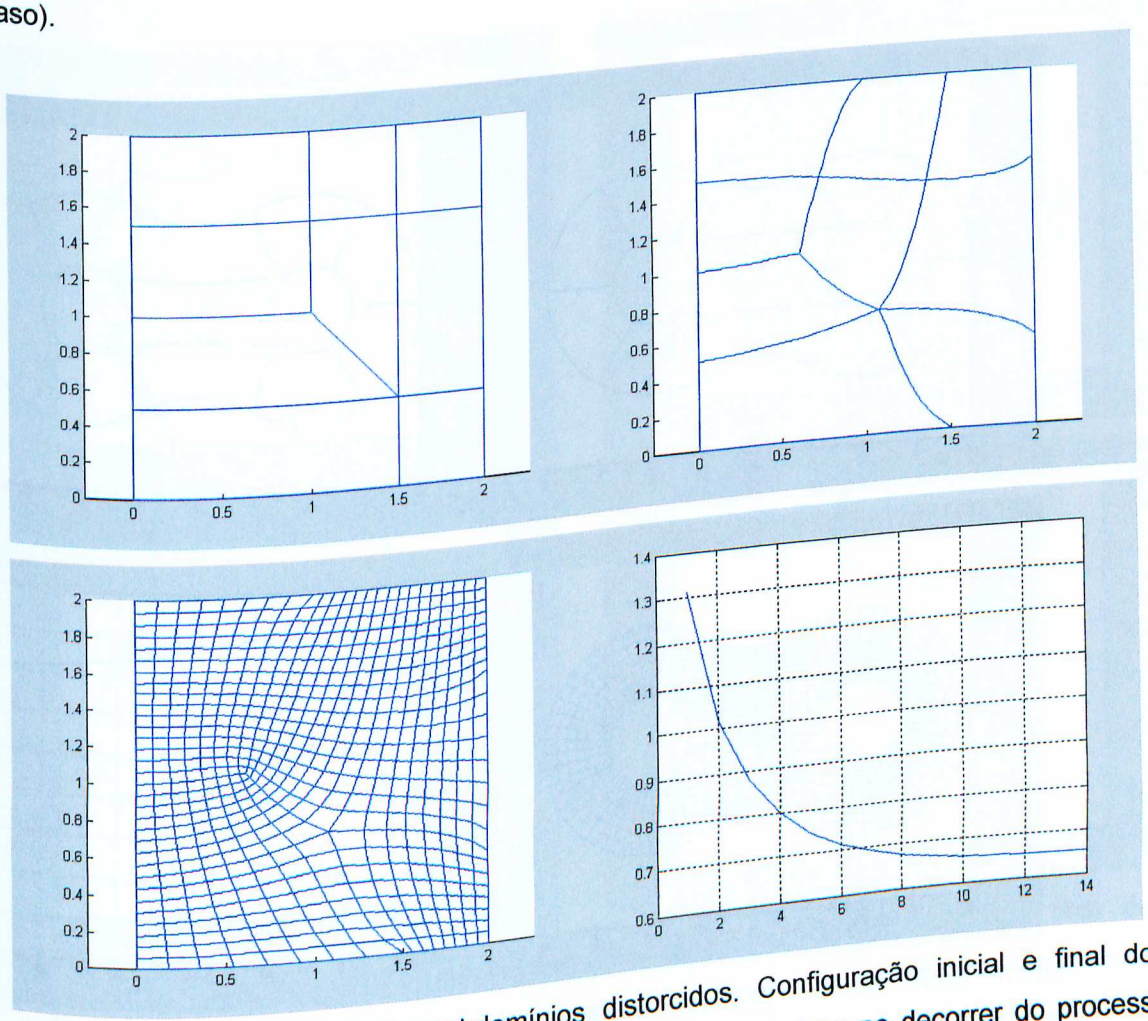
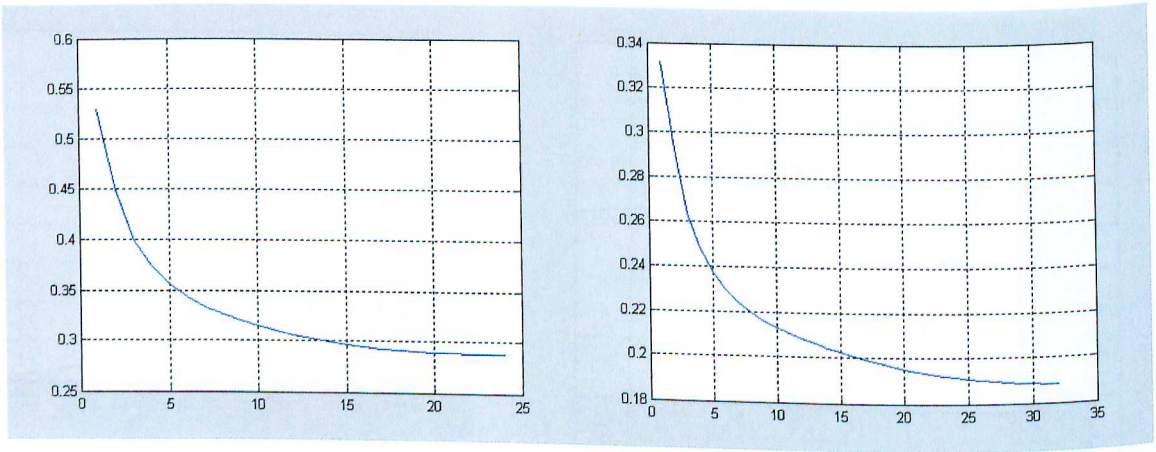


Figura 5.15: Segundo caso de subdomínios distorcidos. Configuração inicial e final dos subdomínios, malha com 8 elementos por face e evolução do erro no decorrer do processo iterativo.



3600(20) 9216(32)

Figura 5.16: Evolução do erro para subdomínios com 20 e 32 elementos por face (segundo caso).

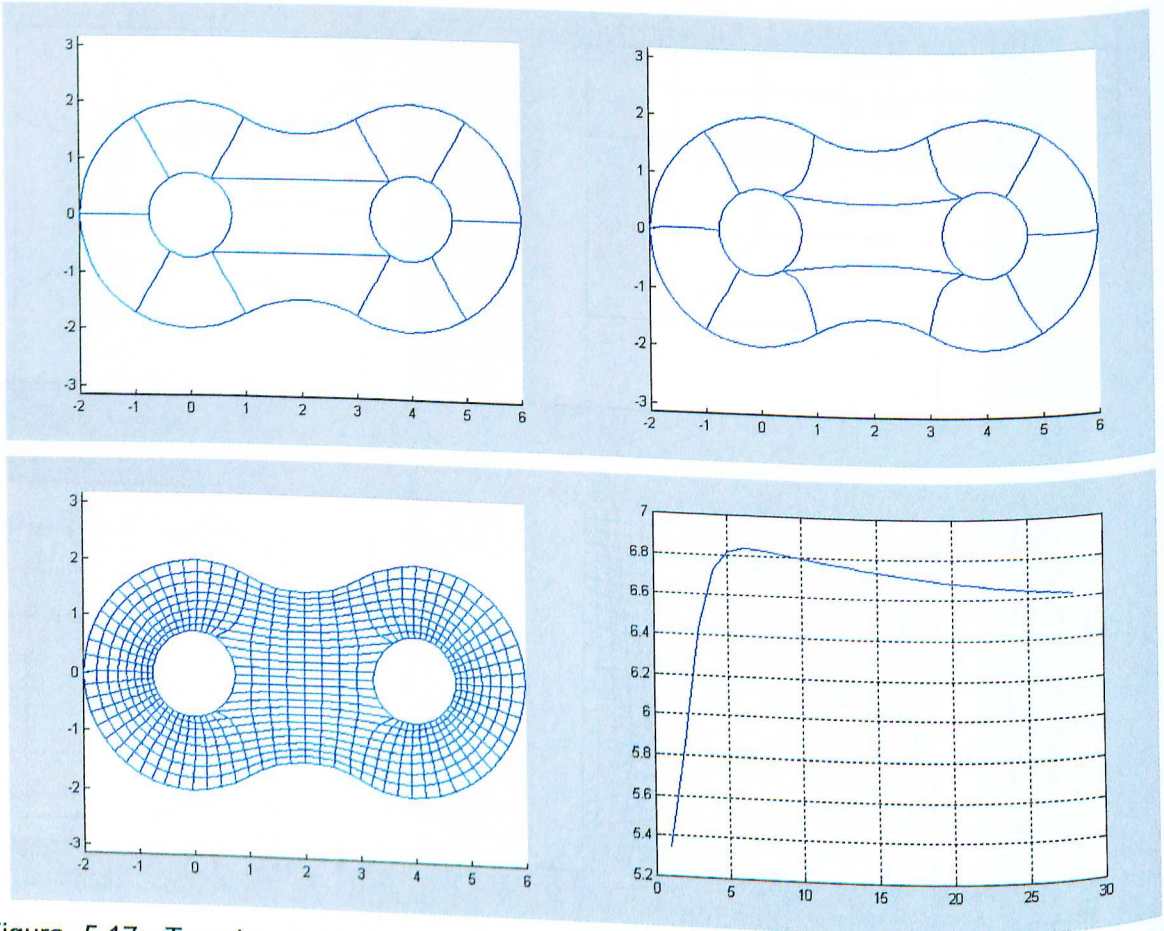
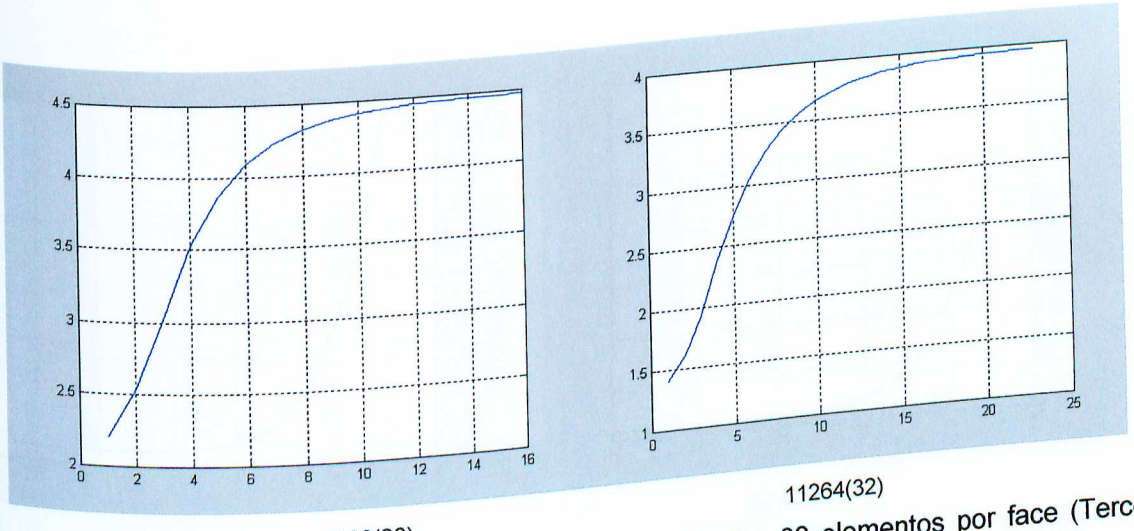


Figura 5.17: Terceiro caso de subdomínios distorcidos. Configuração inicial e final dos subdomínios, malha com 8 elementos por face e evolução do erro no decorrer do processo iterativo.



4400(20) 11264(32)

Figura 5.18: Evolução do erro para subdomínios com 20 e 32 elementos por face (Terceiro caso).

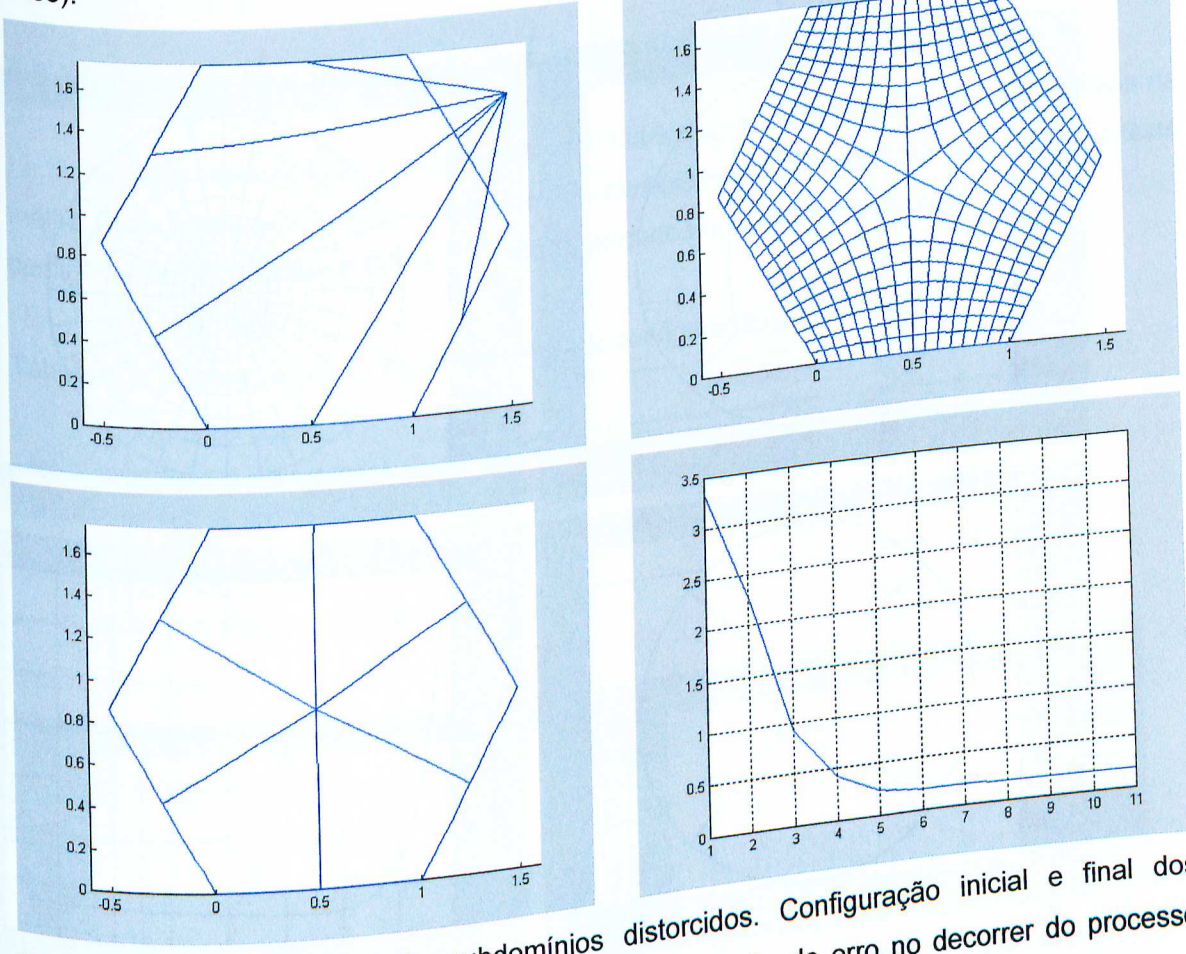
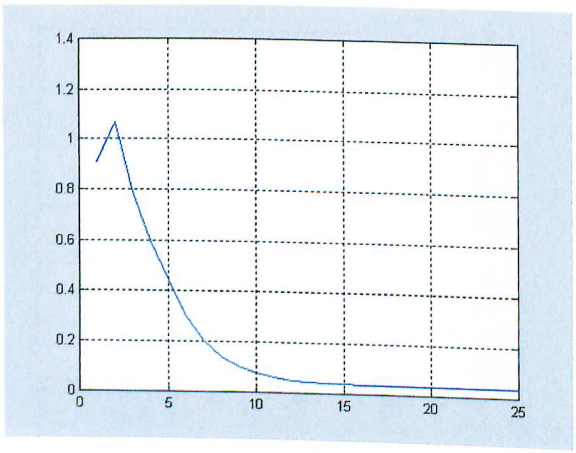
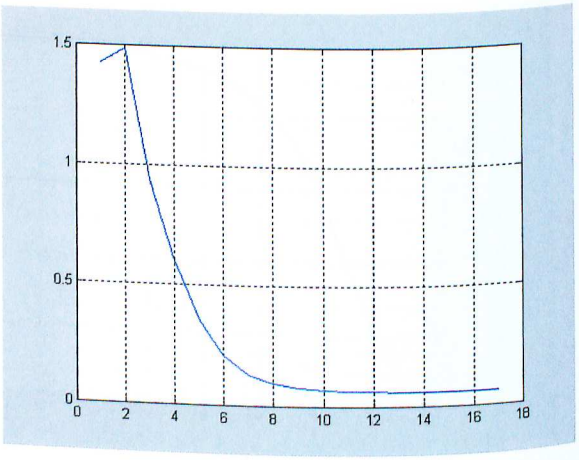


Figura 5.19: Quarto caso de subdomínios distorcidos. Configuração inicial e final dos subdomínios, malha com 8 elementos por face e evolução do erro no decorrer do processo iterativo.

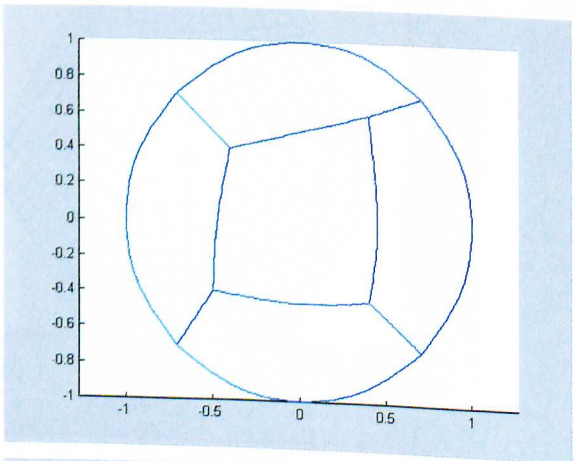


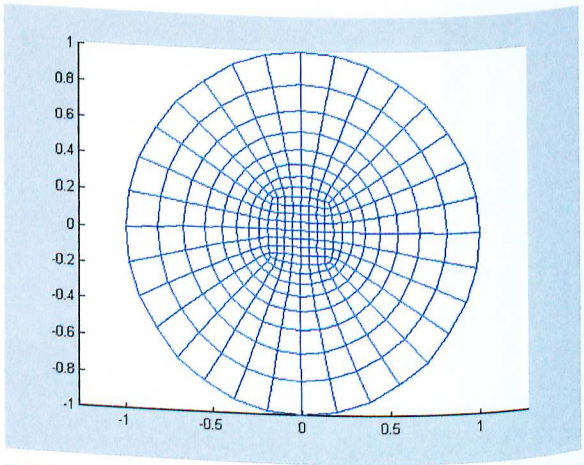
4800(20)

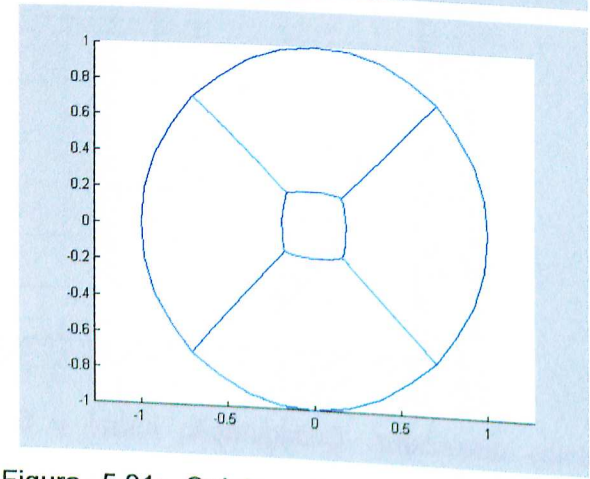


6144(32)

Figura 5.20: Evolução do erro para subdomínios com 20 e 32 elementos por face (Quarto caso).







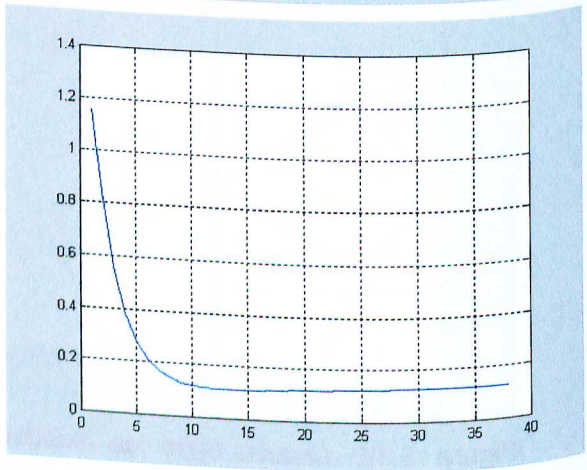
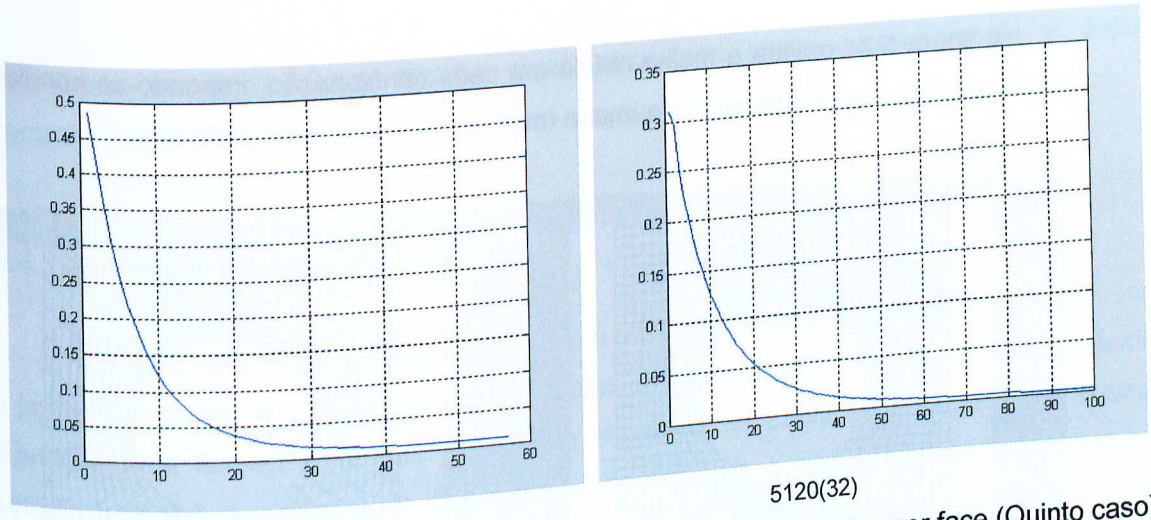


Figura 5.21: Quinto caso de subdomínios distorcidos. Configuração inicial e final dos subdomínios, malha com 8 elementos por face e evolução do erro no decorrer do processo iterativo.



2000(20) 5120(32)

Figura 5.22: Evolução do erro para subdomínios com 20 e 32 elementos por face (Quinto caso).

5.3.3 Influência da configuração dos subdomínios na malha final

O caso apresentado a seguir mostra a influência da configuração dos subdomínios na malha final. A figura 5.23 mostra um domínio multiplamente conexo modelado segundo duas configurações diferentes. A tabela 5.9 mostra os dados referentes a cada caso.

Tabela 5.9: Desempenho do algoritmo em cada configuração de subdomínios

Configuração (no. De elementos)	Tempo (s)	
	Condições de Neumann	Condições de Dirichlet
1 (576)	0.09	0.06
2 (1152)	0.25	0.22

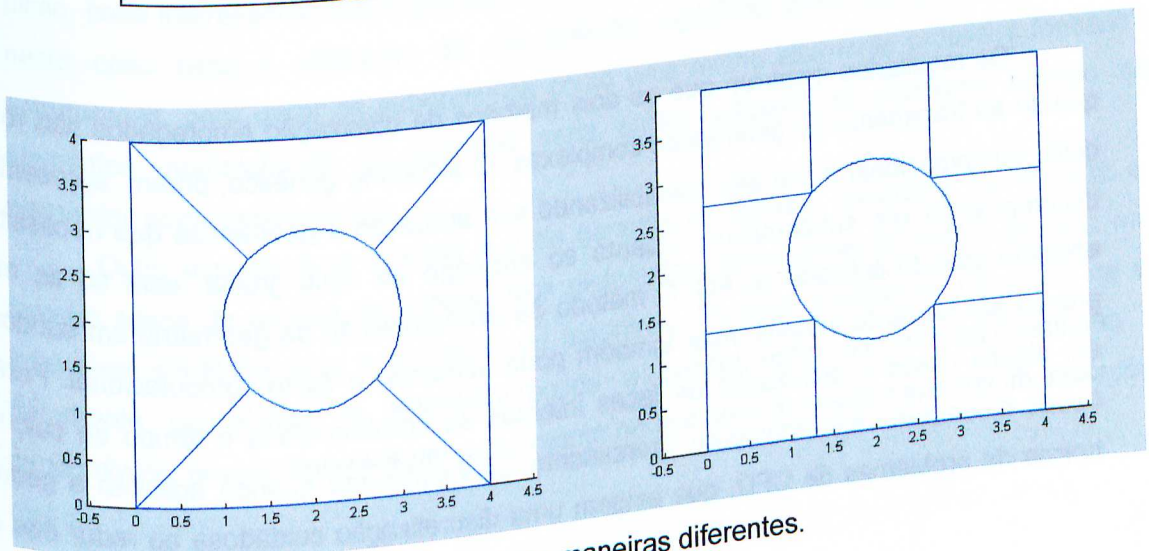


Figura 5.23: Domínio decomposto de duas maneiras diferentes.

A figura 5.24 mostra a malha obtida em cada configuração, impondo-se condições de Dirichlet (malhas superiores) e de Neumann (malhas inferiores) para cada caso.

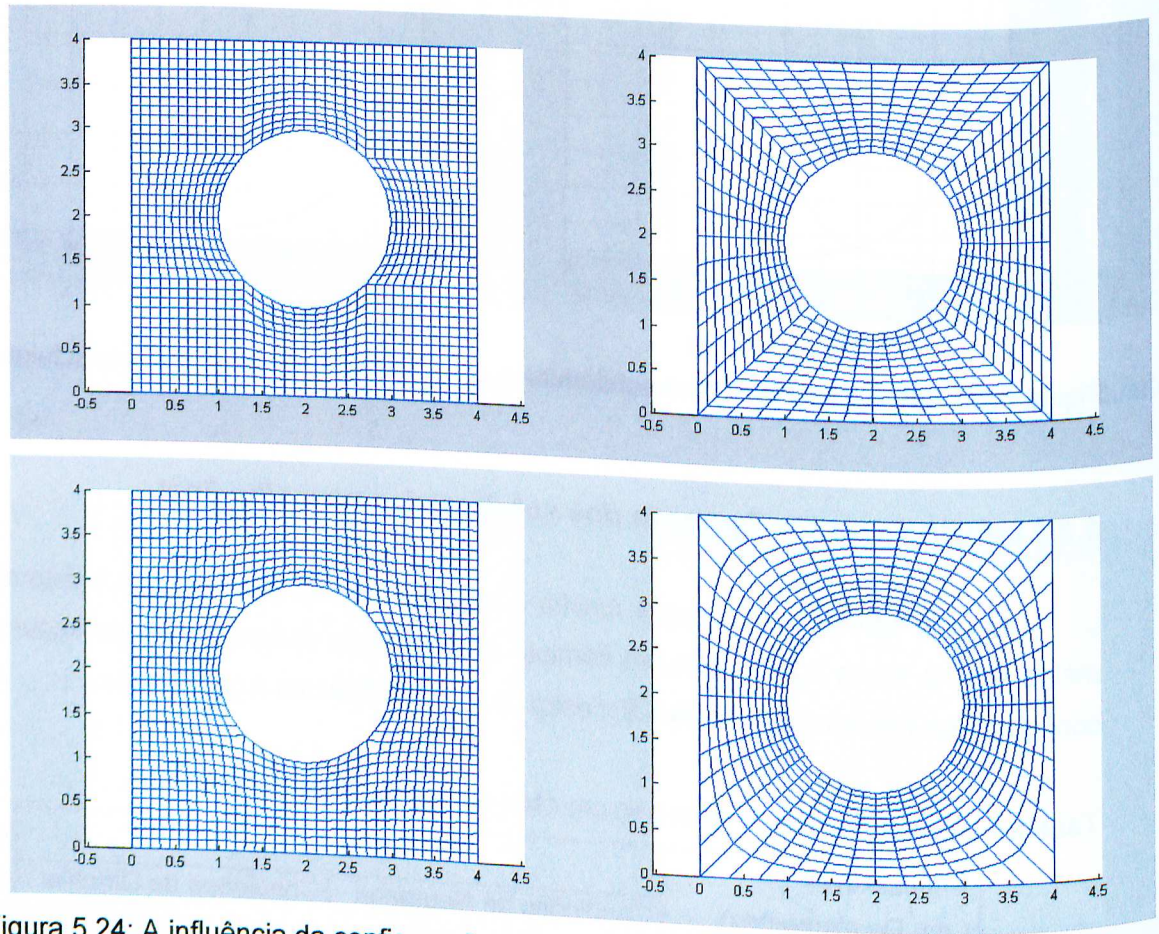


Figura 5.24: A influência da configuração dos subdomínios no formato da malha.

5.4 Análise dos Resultados

Os resultados mostram que os dois métodos de otimização empregados são robustos quanto ao tratamento de geometrias complexas. O algoritmo genético, porém, apresentou um custo computacional muito alto, inviabilizando sua aplicação a geometrias que necessitam de decomposição em subdomínios. Quanto ao emprego do multi grelha, este só se mostrou eficiente quando aplicado ao AG. O método de decomposição da geometria em subdomínios provou ser bastante versátil, mas também pode aumentar o custo computacional. Porém, se em alguns casos a otimização de faces internas ao domínio eleva o tempo de cálculo, em outros mostrou-se extremamente interessante, principalmente quando aplicada a geometrias típicas de problemas de CFD, que exigem uma discretização cuidadosa ao redor dos corpos

submersos no fluido. Estes tópicos serão analisados agora separadamente e depois comparados entre si.

5.4.1 Robustez na Discretização

Como o método de geração empregado é uma técnica de discretização coincidente com a fronteira, a malha se adaptou bem aos domínios em praticamente todos os casos. Entretanto, devido a uma limitação inerente ao uso de equações de Laplace na geração de malhas estruturadas, há o problema da discretização de cantos e quinas pronunciados, o que em alguns casos gera elementos distorcidos, resultando em formas quase triangulares ou muito alongados, com alta razão de aspecto. Por isso, via de regra, nas proximidades de cantos os elementos são bem maiores que no interior da malha, embora geralmente tenham forma aproximadamente retangular. Este fato pode ser notado tanto nos algoritmos genéticos como no algoritmo de busca por gradiente, por ser uma característica intrínseca ao método de geração da malha.

Quanto à continuidade da malha entre áreas adjacentes, há dois aspectos a considerar. O método das condições de Dirichlet apresenta um custo computacional significativamente menor que o método das condições de Neumann, mas como não altera a configuração dos subdomínios, requer maior cuidado por parte do usuário. A redefinição dos subdomínios nem sempre minimiza a distorção da malha, mas é necessária em certos casos, como em problemas de CFD, por exemplo, pois fornece uma malha melhor ajustada à fronteira. A este método, porém, falta generalização suficiente para realocar os vértices dos subdomínios, que se localizam sobre a fronteira externa, mas não contribuem para a forma final do domínio. Além disso, seria interessante que a geração dos subdomínios fosse automática. Uma possibilidade neste caso seria a utilização de um método algébrico para geração de malhas não estruturadas, para gerar os subdomínios como uma malha altamente grosseira formada por elementos quadrilaterais. O algoritmo seria então aplicado ao problema usando estes elementos como subdomínios.

Outro aspecto a se analisar é o da garantia de simetria da malha. No primeiro e no terceiro casos de análise de subdomínios distorcidos, principalmente, obteve-se uma malha totalmente simétrica após a aplicação do algoritmo. Ou seja, mesmo de um ponto de vista puramente geométrico, o algoritmo busca a configuração de mínima distorção dos subdomínios, provavelmente a um custo menor do que se calculasse a energia de distorção da malha.

Neste tópico é importante analisar o desempenho do algoritmo no terceiro caso, onde a distorção da malha aumentou após a aplicação do algoritmo. A diferença entre este e os outros casos de domínios distorcidos analisados é que não haviam, neste caso, vértices internos ao domínio. Isto limita a eficácia do algoritmo e faz com que ele não minimize as distorções, mas apenas redistribua-as, por falta de liberdade para realocar os nós de quina localizados na fronteira externa.

5.4.2 Custo Computacional

Este parâmetro foi o que eliminou o algoritmo genético nas etapas iniciais do trabalho, pois apesar de sua robustez, mostrou-se muito lento. Esta lentidão, entretanto, é uma característica intrínseca dos métodos de busca pseudo-randômicos, devido ao alto número de avaliações da função objetivo nestes algoritmos. Por outro lado, como uma das grandes vantagens destes métodos é que eles executam uma busca global no espaço de projeto, deve-se avaliar a possibilidade da aplicação destes métodos futuramente em conjunto com métodos de busca determinísticos. Esta característica pode ser usada futuramente em próximas etapas para automatizar a configuração dos subdomínios, gerando-os de forma otimizada.

Já o método de busca gradiente deve ser analisado em função do método empregado para se obter a continuidade da malha. No caso da otimização das faces (aplicação de condições de Neumann), o custo mostrou-se diretamente proporcional ao número de faces otimizáveis e à sua forma inicial, de modo que quanto maior a distorção inicial dos subdomínios, maior o tempo para convergência. Para continuidade por condições de Dirichlet, os casos convergiram com 4 iterações, o que sugere que, se os subdomínios forem bem definidos, poucas iterações podem levar a uma malha ótima.

Outro aspecto interessante quanto ao custo computacional é a relação entre o tempo gasto para gerar a malha inicial e para obter a malha ótima pelo método das condições de Dirichlet. Observa-se que, na maioria dos casos, quase metade do tempo gasto – e em alguns casos, mais da metade do tempo – para se obter a malha ótima foi gasto na geração da malha inicial. Mais uma vez, a configuração inicial dos subdomínios mostra-se de grande importância. Este teste foi feito tendo em vista uma das próximas etapas deste trabalho, que é a aplicação do método a problemas de interação fluido – estrutura. Espera-se que, uma vez gerada a malha ótima, as pequenas alterações ocorridas entre algumas iterações não sejam significativas, de modo que se possa gerar uma nova malha a partir da atual, otimizando apenas a região que teve sua forma modificada. Além disso, se for possível o emprego de condições de Dirichlet na continuidade da malha, o custo computacional será ainda menor. Ou

seja, espera-se que pequenos movimentos do sistema tenham efeitos localizados sobre a malha, fazendo com que a remalhagem seja necessária não após cada iteração, mas apenas após a malha atingir certos níveis de deformação. Com isso seria possível resolver problemas de fronteiras móveis sem o emprego de malhas não estruturadas ou malhas de quimera, que elevam o custo computacional do programa, bem como sua dificuldade de implementação.

Quanto ao desempenho do multigrelha, o fator determinante do baixo desempenho na busca por gradiente talvez tenha sido o critério de parada adotado, uma vez que sua principal aplicação é nos casos onde ocorre estagnação do erro. Isto não acontece neste algoritmo porque quando o valor da função objetivo é aproximadamente constante nas duas últimas iterações, o processo iterativo é terminado. Com a aplicação do multigrelha, esta convergência fica prejudicada, uma vez que seu objetivo é justamente re-introduzir as oscilações presentes nas primeiras etapas da solução. Como o critério de parada no algoritmo genético é o número máximo de iterações, o multigrelha tornou-se bastante eficiente quando o número de variáveis de projeto aumentou. Seu baixo desempenho para problemas de poucas variáveis de projeto se deve provavelmente ao seu próprio custo computacional, uma vez que requer a resolução do problema em diferentes malhas.

Capítulo 6

Conclusões

O algoritmo apresentado nesta dissertação foi proposto como uma metodologia para discretização de domínios arbitrários através de malhas estruturadas, tendo em vista sua futura aplicação em problemas de fronteira móvel. Neste sentido, o uso de um gerador elíptico de malhas, governado por equações de Laplace, mostrou-se adequado ao tema, embora tenha suas próprias limitações. Para contorná-las, pode-se sugerir o uso de equações de Poisson em substituição às equações de Laplace, buscando pontos ótimos para a inserção dos termos fonte. Outro caminho seria empregar o método proposto por Berle (Berle, 1999), que utiliza algoritmos genéticos na otimização local do posicionamento dos nós: ao invés de analisar toda a malha, o problema é reduzido a alguns nós que terão liberdade de movimento. Como isso pode aumentar o custo computacional do algoritmo, deve-se também estudar um modo de otimizar os subdomínios de maneira que se eliminem as quinas e cantos do problema, buscando uma transição mais suave entre faces adjacentes, ou ainda um método híbrido de continuidade da malha, que combine a suavidade da otimização de faces com a precisão da imposição das condições de contorno.

Quanto à representação do domínio, a precisão da spline cúbica a um baixo custo computacional faz deste método uma poderosa ferramenta computacional e de fácil implementação. Assim, nas próximas etapas deste trabalho pode-se sugerir a criação de rotinas que gerem primitivas geométricas como polígonos, círculos, retas e curvas genéricas, de modo a facilitar a interface do programa com o usuário. Além disso, pode-se estudar também seu emprego num método algébrico de geração de malhas. Uma outra alternativa seria implementar uma interface com programas de CAD comerciais, para facilitar o trabalho do usuário.

A composição do domínio pela união de subdomínios quadrilaterais permite que o método seja aplicado praticamente a qualquer domínio bidimensional. Entretanto, um algoritmo para geração automática de subdomínios tornaria o método ainda mais genérico e, provavelmente, mais eficiente, uma vez que sua forma e posicionamento já seriam otimizados durante a geração. Espera-se que, automatizando esta etapa, seja possível obter malhas com transição mais suave entre subdomínios adjacentes e, principalmente, seja possível amenizar os problemas na discretização de cantos e quinas inerentes ao uso de equações de Laplace na

geração de malhas. De qualquer modo, mesmo no estágio atual, a aplicação da decomposição por subdomínios, se bem aplicada, leva a resultados bastante interessantes.

O algoritmo genético, apesar de apresentar bons resultados quanto à distribuição da malha sobre a superfície exige um tempo computacional muito alto para se tornar competitivo quando a remalhagem do domínio é necessária. Apesar disso, não deve ser descartado quando se trata de problemas em regime permanente e merece estudos posteriores. Uma sugestão seria a diminuição do tamanho da população e a associação com um método de busca por gradiente a ele, alternando os métodos de busca ao longo do processo iterativo.

O método de busca por gradiente mostrou-se eficiente e permitiu a aplicação do algoritmo a geometrias complexas, modeladas por subdomínios multiplamente conexos.

As maiores dificuldades encontradas durante o desenvolvimento do algoritmo surgiram durante a criação dos métodos de continuidade da malha, principalmente para otimização das faces. Foi necessário desenvolver um critério suficientemente genérico que identificasse qualquer vértice interno ao domínio e que portanto pudesse ser realocado. O critério inicial, que dizia que cada vértice interno deveria ser compartilhado por 4 áreas, logo foi descartado, chegando-se então à condição necessária e suficiente para que um ponto seja um vértice interno: ele deve ser compartilhado por um conjunto de no mínimo 3 áreas e cada uma destas deve ter duas vizinhas no conjunto.

Para as próximas etapas deste trabalho sugere-se o desenvolvimento de algoritmos que flexibilizem a interface com o usuário, o que pode ser feito através de um programa CAD comercial ou pelo desenvolvimento de rotinas que trabalhem com formas geométricas genéricas. Outro ponto a desenvolver é a generalização do método da otimização dos subdomínios, permitindo que vértices sobre a fronteira externa também sejam reposicionados, desde que não modifiquem a forma do domínio. Isto deve aumentar o custo computacional.

Capítulo 7

Referências Bibliográficas

Batina, J. T.,1989. "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," AIAA Paper 89-0115-CP.

Berle, A.W.,1998. "Implementação de um Sistema Experimental de Algoritmos Genéticos em C++ para Geração e Otimização de Malhas Subjacentes ao Método de Elementos Finitos." Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia, 22 de dezembro de 1998.

Bern, M., Eppstein, D., 1997. "Quadrilateral Meshing by Circle Packing". Proceedings of the second CGC Workshop, Computational Geometry, 1997

Briggs, W.L., 1987. "A Multi Grid Tutorial", Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania

Cheng, W., 1993. "Adaptative Grid Optimization for Structural Analysis – A Geometry-based Approach". Computer Methods in Applied Mechanics and Engineering.

Crumpton, P. I., Giles, M. B., 1997. "Implicit Time-Accurate Solutions on Unstructured Dynamic Grids," International Journal for Numerical Methods in Fluids, Vol. 25, No. 11, pp. 1285–1300.

Frank, J Bossen and Paul S. Heckbert (1996) "A Pliant Method for Anisotropic Mesh Generation", Proceedings, 5th International Meshing Roundtable.

Goldberg, D.E., 1989, Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, Reading, MA.

Jones, W.T., Samareh, J.A., 1995. "A Grid Generation System for Multidisciplinary Design Optimization", Proceedings of the Workshop on Surface Modeling, Grid Generation, and Related Issues in CFD Solutions, NASA Lewis Research Center, Cleveland, Ohio.

Koza, J. R., 1992. "Genetic Programming: On the Programming of Computers by Means of Natural Selection". MIT Press, Cambridge, Mass.

Koza, J. R., 1994. "Genetic Programming II: Automatic Discovery of Reusable Programs.". MIT Press, Cambridge, Mass.

Lebrón, S.C., Ribeiro, C.A., 2000. "Método de Multi Grelha na Resolução de Sistemas Lineares de Grande Porte com Aplicação em Dinâmica dos Fluidos". VI POSMEC, Universidade Federal de Uberlândia, Uberlândia - MG.

Lohner, R, Morgan, K. e Zienkiewicz, O. C. (1986) "Adaptive Grid Refinement for Compressible Euler Equations", Accuracy Estimates and Adaptive refinements in Finite Element Computations, I. Babuska et. al. eds., Wiley.

Maliska C. R.,1995, "Transferência de Calor e Mecânica dos Fluidos Computacional – Fundamentos e Coordenadas Generalizadas", LTC - Livros Técnicos e Científicos Editora S. A., Rio de Janeiro.

Mastin, C.W., 1995. "Multilevel Elliptic Smoothing of Large Three-Dimensional Grids, Workshop on Surface Modeling, Grid Generation, and Related Issues in CFD Solutions". NASA Lewis Research Center, Cleveland, Ohio, NASA CP 3291.

Michalewicz, Z., 1994, "Genetic Algorithms+Data Structure=Evolution Programs", Spring-Verlag, New York, 2ª ed.

Milioli, F.E., 1985. "Solução Numérica de Problemas Bidimensionais de Convecção Natural em Cavidades Arbitrárias", Dissertação de Mestrado, Universidade Federal de Santa Catarina, Florianópolis, Brasil.

Owen, S., 1997. "A Survey of Unstructured Mesh Generation Technology". URL: http://www.andrew.cmu.edu/user/sowen/survey/index.html.

Ribeiro, C.R., 2000 "Otimização de Malhas de Elementos Finitos via Algoritmos Genéticos". IV SIMMEC, Uberlândia - MG.

Roger, D. F., Adams J. A., 1990, "Mathematical Elements for Computer Graphics", McGraw-Hill,Inc, New York.

Samareh, J.A.,2000."Grid Generation for Multidisciplinary Design and Optimization of an Aerospace Vehicle – Issues and Chalenges. Proceedings of 12th AIAA Computational Fluid Dynamics Conference, AIAA – 95 – 1689, San Diego, California.

Shanmugasundaram, R., Garriz, J.A., Samareh, J.A.,1997."The Development of a Tool for Semi-Automated Generation of Structured and Unstructured Grids About Isolated Rotorcraft Blades". Technical Specialists' Meeting for Rotorcraft Acoustics and Aerodynamics". Williamsburg, Virginia.

Shimada, K., Liao, J.H., Itoh, T.,1998. "Quadrilateral Meshing with Directionality Control trough the Packing of Square Cells.

Shimada, K., Yamada, A., Itoh, T. (1997) "Anisotropic Triangular Meshing of Parametric Surfaces via Close Packing of Ellipsoidal Bubbles", Proceedings, 6th International Meshing Roundtable.

Teixeira, R.L., 2001. "Uma Metodologia de projeto de controladores híbridos inteligentes com aplicações no controle ativo de vibrações mecânicas". Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia, MG.

Thompson, J.F., Warsi, Z.U. e Mastin, C.W. 1985. "Numerical Grid Generation – Foundations and Applications", Elsevier Science Publishing Co., EUA.

Vanderplaats, G.N., 1998."Numerical Optimization Techniques for Engineering Design: With Applications". McGrawHill Book Company.

Wagner,C.,1998. "Introduction to Algebraic Multigrid – course notes". Universität Heidelberg, Heidelberg, Germany.

Whitley, Darrell, 1993."A Genetic Algorithm Tutorial". Computer Science Department, Colorado State University.

William H. Press, W.H., Teukolsky, S.A., Vetterling, W.A., , Flannery, B.P., 1992. "Numerical Recipes in C". Cambridge Press.

Zhou, T., Shimada, K, 2000. "An Angle-Based Approach to Two-Dimensional Mesh Smoothing". Proceedings, 9th International Meshing Roundtable, Sandia National Laboratories. New Orleans, Louisiana. (www.andrew.cmu.edu).