LÚCIO AURÉLIO PURCINA

# TÉCNICAS DE OTIMIZAÇÃO EVOLUTIVA APLICADAS À SOLUÇÃO DE GRANDES SISTEMAS LINEARES

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA MECÂNICA

2010

LÚCIO AURÉLIO PURCINA

# TÉCNICAS DE OTIMIZAÇÃO EVOLUTIVA APLICADAS À SOLUÇÃO DE GRANDES SISTEMAS LINEARES

Tese de Doutorado intitulada **Técnicas de Otimização Evolutiva Aplicadas à Solução de Grandes Sistemas Lineares**. Parte integrante dos requisitos para obtenção do titulo de Doutor em Engenharia Mecânica, sob a orientação da Professora Dra. Sezimária de Fátima Pereira Saramago.

**Uberlândia, 2010**

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU , MG, Brasil

P985t Purcina, Lúcio Aurélio, 1973-
Técnicas de otimização evolutiva aplicadas à solução de grandes sistemas lineares [manuscrito] / Lúcio Aurélio Purcina. - 2010.
146 f. : il.

Orientadora: Sezimária de Fátima Pereira Saramago.

Tese (Doutorado) – Universidade Federal de Uberlândia, Programa de Pós-Graduação em Engenharia Mecânica.
Inclui bibliografia.

1. Engenharia mecânica - Teses. 2. Dinâmica - Teses. 3. Equações lineares - Teses. 4. Otimização - Teses. 5. Identificação de sistemas - Teses. I. Saramago, Sezimária de Fátima Pereira. II. Universidade Federal de Uberlândia. Programa de Pós-Graduação em Engenharia Mecânica. III. Título.

CDU: 621

# Dedicatória

À memória de meus Pais Verônica da Silva Purcina e Venerável Purcina.

# Agradecimentos

Agradeço a Deus, por ter me dado o dom de estudar, aos meus Pais Verônica da Silva Purcina e Venerável Purcina, por me darem a vida e me incentivarem a seguir este caminho e por que durante toda vida fizeram tudo para que este momento se concretizasse. À minha esposa Joelma Pereira Silva Purcina, por me dar muita força e aos meus filhos, Marco Aurélio Silva Purcina e Eduardo Augusto Silva Purcina, por suportarem minha ausência e ser a inspiração para o êxito.

Agradeço de forma especial à minha orientadora Professora Doutora Sezimária de Fátima Pereira Saramago, por ter caminhado comigo me indicando o melhor caminho nesta etapa de minha vida.

Agradeço a CAPES pelo suporte financeiro, que sem ele seria muito difícil realizar este trabalho.

PURCINA, L. A. **Técnicas de Otimização Evolutiva Aplicadas à Solução de Grandes Sistemas Lineares**. 2010. 146 f. Tese de Doutorado, Universidade Federal de Uberlândia, Uberlândia.

# Resumo

Muitos campos da engenharia e outras ciências aplicadas exigem a utilização da solução de sistemas lineares algébricos. Dependendo do modelo matemático usado para representar o fenômeno, os sistemas lineares são de elevada dimensão. Tradicionalmente, os grandes sistemas lineares são resolvidos através de métodos iterativos. A convergência destes métodos depende dos autovalores da matriz dos coeficientes. Assim, quando a matriz dos coeficientes perde uma das seguintes características como ser simétrica ou positiva definida, os métodos iterativos (estacionários e não estacionários) perdem a eficácia. Existem vários métodos para resolver os sistemas lineares. O objetivo é encontrar o método mais eficaz para um determinado problema. Infelizmente, um método que funciona bem para um tipo de problema pode não funcionar tão bem para outro. Na verdade, ele pode nem mesmo funcionar. Assim, várias pesquisas ainda estão sendo desenvolvidas e aprimoradas nesta área de conhecimento. O objetivo deste trabalho é propor a aplicação de algumas técnicas na solução de grandes sistemas lineares. A este propósito, são testados métodos iterativos não-estacionários clássicos (Gradientes Conjugados, Resíduos Mínimos, Gradientes BiConjugados) e comparados com os dois métodos de otimização evolutiva, Evolução Diferencial e Algoritmos Genéticos. Nesta pesquisa são resolvidos os problemas como a equação de Laplace bi-dimensional e identificação indireta de forças dinâmicas. A solução analítica é comparada com as soluções numéricas calculadas usando os métodos mencionados. Os resultados obtidos são apresentados, analisando os parâmetros mais importantes e sua influência sobre a convergência e eficiência dos métodos testados.

---

*Palavras Chave: Sistemas Lineares. Métodos de Otimização Evolutiva. Identificação de Forças.*

PURCINA, L. A. **Evolutionary Optimization Techniques Applied to Solution of Large Linear Systems.** 2010. 146 f. Ph. D. Thesis, Universidade Federal de Uberlândia, Uberlândia.

# Abstract

Many fields of engineering and other applied sciences demand the use of the solution of algebraic linear systems. Depending on the mathematical model used to represent the phenomenon, the linear systems will have high dimensions. Traditionally, large linear systems are resolved by using iterative methods. The convergence of these methods depends on the eigenvalues of the coefficients matrix. Thus, when the coefficients matrix loses one of the following characteristics as being symmetric or positive definite, the iterative methods (stationary and nonstationary) lose efficiency. Many methods exist for solving the linear systems. The aim is to find the most effective method for a particular problem. However, a method that works well for one type of problem might not work so well for others. Indeed, it may not even work. Thus, several researches are still being developed and improved in this area of knowledge. The aim of this works is to propose the application of some techniques in solving large linear systems. On this purpose, classic no stationary iterative methods are tested (Conjugate Gradients, Minimal Residual, Bi-Conjugate Gradients, Generalized Minimal Residual) and compared with two evolutionary optimization methods, Differential Evolution and Genetic Algorithms. In this research, problems like bi-dimensional Laplace equation and identification indirect of dynamical forces are solved. The analytical solution is compared with the numerical solutions calculated by using the mentioned methods. Results are presented, analyzing the most important parameters and their influence on the convergence and efficiency of the tested methods.

---

*Keywords: Linear Systems. Evolutionary Optimization Methods. Identification of Forces.*

# Lista de Figuras

# Lista de Tabelas

# Listas de Símbolos

A, B, C,... – Matriz

$A^T$, $B^T$, $C^T$,... – Matriz transposta

$A^H$ – Transposta conjugada de $A \in \mathbb{C}^{n\times n}$

$A^{-1}$ – Inversa da matriz $A$

$A^+$ – Pseudo-invesa da matriz $A$

AGs – Algoritmos Genéticos

$a_{ij}$ – Termo geral da matriz

$\{A^{(k)}\}$ – Sequência de matrizes

$b$ – Vetor dos termos independentes

BiCG – Gradientes Bi-Conjugados

$\mathbb{C}$ - Conjunto dos números complexos

$\mathbb{C}^n$ – Espaço vetorial complexo de dimensão $n$.

$\mathbb{C}^{m\times n}$ – Espaço vetorial complexo de todas as matrizes de ordem $m\times n$

CG – Gradientes Conjugados

$C^\infty(\mathbb{R}^n)$ – Espaço vetorial da funções infinitamente diferenciáveis

ED – Evolução Diferencial

F.R.I.s – Funções de resposta ao impulso

F.R.F.s – Funções de resposta em freqüência

$\{F(x)\}\in\mathbb{R}^N$ – Espectro das forças estimadas

$f(t)$ – Força excitadora em função do tempo $t$

GMRES – Resíduos Mínimos Generalizados

H(x) – Matriz das FRFs

$h(t)$ – FRIs

H($x$) – Matriz hessiana de uma função

K$(v, A, k) \in \boldsymbol{R}^{n\times k}$ – Matriz de Krilov

MINRES – Resíduos Mínimos

$\mathbb{N}$ - Conjunto dos números naturais

$Np$ – Número de indivíduos de uma população

ns – matriz não simétrica

npd – matriz não positiva definida

pd – matriz positiva definida

P(x) – Função de penalidade

*Pc* – Probabilidade de cruzamento

$p_c$ – Posição de cruzamento

$S^{\perp}$ – Complemento ortogonal de um subespaço vetorial

s – matriz simétrica

$\mathbb{R}$ - Conjunto dos números reais

$\mathbb{R}^n$ – Espaço vetorial real de dimensão *n*.

$\mathbb{R}^{m\times n}$ – Espaço vetorial real de todas as matrizes de ordem $m\times n$

$\{x^{(k)}\}$ – Sequência de vetores

$x$ – Vetor coluna das variáveis

$v_1, v_2, ..., v_m$ – Vetores

$x(t)$ – Amplitude da vibração em resposta ao tempo $t$

$\alpha_k$ , $\beta_k$,... – Escalar, tamanho do passo na direção de busca

$\beta_i$ – Parâmetro do critério de Sassenfeld

$\Delta x, \Delta y, \Delta t, ...$ – Discretização de uma variável

$\varepsilon$ – Parâmetro de parada

$\varepsilon(n)$ - Esparsidade de uma matriz de ordem $n$

$\phi$(x) - Função pseudo-objetivo de penalidade

$\lambda$ – Autovalor de uma matriz $A$

$\sigma(A)$ – Espectro de uma matriz $A$

$\rho(A)$ – Raio espectral da matriz $A$

$\kappa(A)$ – Fator-condição da matriz A

$\Sigma m_i$ – Somatório

$\varphi(x^{k-1})$ – Função de iteração

$\Omega$ – Espaço de busca

$\omega$ - Fator de extrapolação (relaxação)

$\varpi$ – Frequência

$\nabla F(x)$ – Vetor gradiente de uma função

$\frac{\partial F(x)}{\partial x}$ – Derivada parcial de uma função

# Sumário

# Capítulo 1: Introdução

Praticamente todos os campos da engenharia e também de ciências aplicadas, como física e química, utilizam a solução de sistemas lineares algébricos. Temas como: análise estrutural, escoamento de fluidos, transferência de calor, aerodinâmica, entre outros, utilizam fortemente a solução de tais sistemas como parte integrante da modelagem destes fenômenos. Dependendo do modelo matemático usado para representar o fenômeno, os sistemas lineares serão de elevada dimensão. As primeiras tentativas documentadas de solução de sistemas lineares aparecem no século XIX (GAUSS, 1823; LIOUVILLE, 1837; JACOBI, 1845; SEIDEL, 1874). Dada sua grande aplicação, este assunto foi ao longo do tempo envolvendo mais pesquisadores e as publicações foram então aumentando. O tema continua ainda extremamente rico, tanto que se pode destacar um bom número de publicações recentes, como em MEHMOOD *et al.* (2005), GRIGORI *et al.* (2005), FASANO (2005a), FASANO (2005b), ANSONI *et al.* (2007), HASHEMI e KARGARNOVIN (2007), MANGUOGLU *et al.* (2009) e BABU *et al.* (2010).

Na realidade, este estudo dificilmente se esgotará, porque sempre será possível criar novas maneiras de armazenar dados, novos métodos (ou variantes) de obtenção da solução. Também os recursos computacionais estão em constante evolução e permitindo que a solução eficiente de sistemas lineares também sofra um processo evolutivo. Portanto, a existência de algoritmos numéricos eficientes para solução de tais sistemas, é de extrema importância. Neste trabalho são estudados alguns algoritmos iterativos não-estacionários, tais como: Método dos Gradientes Conjugados, Método dos Resíduos Mínimos, Método dos Gradientes Bi-Conjugados, Método dos Resíduos Mínimos Generalizados. Além disso, são estudados dois métodos evolutivos: Algoritmos Genéticos e Evolução Diferencial. A seguir serão apresentados alguns trabalhos que se utilizam de métodos de otimização para solução de problemas envolvendo sistemas lineares e também aplicações em problema de identificação de parâmetros.

BENZI (2002) inspeciona técnicas de pré-condicionadores para a solução iterativa de grandes sistemas lineares, com um enfoque em métodos algébricos satisfatórios para matrizes esparsas. O artigo proporciona uma atualização de recentes algoritmos desenvolvidos usando pré-condicionadores para grandes sistemas de equações lineares esparsos. A atenção é quase

exclusivamente restrita ao objetivo principal, ou seja, apresentar técnicas algébricas que possam ser aplicadas a matrizes esparsas em geral. Uma atenção especial é voltada a assuntos práticos, como robustez e eficiência de implementação em computadores com arquitetura moderna. Não são discutidos com profundidade os aspectos teóricos.

CAMPOS-SILVA e APARECIDO (2003) implementam algoritmos para resolver grandes sistemas lineares esparsos resultantes da solução de problemas convectivos-difusivos através de métodos numéricos como diferenças finitas, volumes finitos ou elementos finitos, explorando e preservando a esparsidade inicial da matriz global. São armazenados só os coeficientes não-nulos do sistema, na forma vetorial. Os métodos implementados são iterativos estacionários. Alguns testes numéricos são implementados e os resultados obtidos utilizados para comparar o desempenho dos métodos e sua performance computacional.

MEHMOOD *et al*. (2005) consideram o problema de análise de estado fixo de Cadeias de Markov com Tempo Contínuo (CTMCs). CTMCs são formalismos amplamente usados para a análise de desempenho de sistemas de computadores e sistemas de comunicação. Uma grande variedade de aplicações do desempenho útil pode ser derivada de um CTMC pela computação de suas probabilidades de fixar estado. Um CTMC pode ser representado por um conjunto de estados e uma matriz que contém as taxas de transição de estado como coeficientes, e pode ser analisada usando um modelo de verificação probabilística. Assim, a modelagem CTMC para sistemas reais é muito grande. São tratados, no artigo, estes grandes problemas, considerando métodos simbólicos. O método iterativo de Jacobi é usado, sendo aplicado computação paralela para a solução do estado fixo CTMC.

Em GRIGORI *et al.* (2005) é empregado o método direto de Gauss para resolver um sistema não simétrico esparso de equações lineares do tipo $Ax = b$. A eliminação utilizada consiste em fatorar a matriz $A$ explicitamente no produto de $L$ e $U$, onde $L$ é uma matriz triangular inferior e $U$ é uma matriz triangular superior. A seguir resolve-se $(LU)x = b$, como um fator de cada vez. Uma das características principais da fatoração $LU$ esparso é a noção de enchimento (*fill-in*). Esta definição refere-se a um elemento que era zero na matriz original A, mas se torna não nulo durante a fatoração. Como estes enchimentos (*fill-ins*) podem ser computados sem referir-se aos valores numéricos da matriz, pode-se alocar memória para que sejam organizadas as computações antes de calcular os valores numéricos dos fatores. Assim, a resolução de um sistema esparso é dividida em várias fases. São apresentadas as fases específicas de SuperLU-DIST (LI *et al.*, 2003), um algoritmo largamente usado para resolver grandes sistemas lineares esparsos, não-simétricos, em computadores de memória distribuída.

FASANO (2005a) apresenta um método iterativo não estacionário, onde um novo algoritmo de gradientes conjugados (CG) é baseado em métodos planares para gradientes conjugados. Este método permite a solução de sistemas de equações lineares cujas matrizes dos coeficientes são não positiva-definidas e não singulares. Este é o caso onde a aplicação do algoritmo padrão CG apresentado por HESTENES e STIEFEL (1952) pode falhar, devido a uma possível divisão por zero. É dada uma prova completa da convergência global para o novo método planar. Além disso, é descrito uma importante característica do algoritmo planar que é usado dentro do otimizador.

Em FASANO (2005b) é descrito uma aplicação do método dos gradientes conjugados introduzido no artigo anterior, sendo resolvido um problema indefinido de equações lineares não singulares.

ANSONI *et al.* (2007) apresentam uma contribuição para o desenvolvimento de um método especializado para resolver sistemas lineares associados a problemas de tomografia. Problemas de fluxo multifásico em tomografia envolvem a solução de um grande sistema linear, cuja matriz obtida a partir da discretização em elementos finitos é esparsa, simétrica e positiva-definida, exigindo esforço computacional elevado tanto para o armazenamento de dados quanto para a resolução. O método de decomposição em valores singular (SVD - método direto) e o método do gradiente bi-conjugado pré-condicionado (CPNG - método iterativo) foram estudados com relação às características da matriz e do desempenho computacional. Dois tipos de problema foram estudados: a reconstrução do coeficiente de convecção, por sensoriamento térmico não-intrusivo (aplicação de um fluxo de calor na fronteira externa e medição da resposta em temperatura) e reconstrução de distribuição da permissividade do fluido, por sensoriamento elétrico não-intrusivo (aplicando uma distribuição de tensão na fronteira externa e medindo a resposta das correntes). Em ambos os casos, os resultados mostraram que o método CPNG obteve um maior desempenho computacional em comparação com o método de SVD. Mais especificamente, o método CPNG é mais rápido na obtenção da solução.

HASHEMI e KARGARNOVIN (2007) apresentam a identificação da força de impacto agindo sobre uma viga simplesmente apoiada. Trata-se de um problema inverso em que a resposta medida da estrutura é utilizada para determinar a força aplicada. O problema de identificação é formulado como um problema de otimização a partir de um sistema linear sendo utilizado algoritmos genéticos para a busca da solução ótima. A função objetivo é calculada pela diferença entre as respostas analíticas e valores medidos. As variáveis de

decisão são a localização e magnitude da força aplicada. Os resultados da simulação mostram a eficácia da abordagem, bem como sua robustez em relação ao ruído de medição e localização dos sensores.

MANGUOGLU *et al.* (2009) mostram que o surgimento de plataformas de computação paralela em grande escala, junto com o aumento do número de núcleos disponíveis em processadores convencionais, representam desafios significativos para o algoritmo e desenvolvimento de software, especialmente os que solucionam sistemas lineares esparsos. Estes solucionadores devem otimizar o desempenho paralelo, o desempenho do processador (em série), bem como os requisitos de memória, sendo robusto para uma grande quantidade de aplicações e sistemas. Neste trabalho, apresenta-se um novo solver paralelo que combina as características desejáveis de métodos diretos (solidez) e eficiência dos iterativos (baixo custo computacional) enquanto busca diminuir as suas desvantagens tais como requisitos de memória, falta de consistência, etc. Este solver híbrido proposto é baseado nos códigos gerais para sistemas lineares esparsos, "PARDISO" e na família "SPIKE" de "solver" híbridos. O algoritmo resultante, chamado "PSPIKE", é tão robusto quanto "solver" diretos e mais confiáveis do que os métodos clássicos pré-condicionados do subespaço de Krylov.

Em BABU *et al*. (2010) demonstra-se que os enigmas de Sudoku podem ser formulados e resolvidos como um sistema de equações lineares esparso. O trabalho começa mostrando que o conjunto de regras do enigma Sudoku pode ser expresso como um sistema linear indeterminado $Ax = b$, onde $A$ é uma matriz de dimensão $n$ x $m$ com $n > m$. Prova-se, então, que a solução de Sudoku é a solução mais dispersa do sistema, e que, portanto, pode ser obtida pela minimização de $min||x||_0$ sujeito a $Ax = b$, onde $||x||_0$ é a norma $L_0$ de $x$, ou seja, o número de componentes não-nulas. No entanto, a solução de um problema de minimização da norma $L_0$ é normalmente difícil. Ao invés disso, resolve-se um problema de programação linear muito mais simples, que consiste em minimizar $min||x||_1$ sujeito a $Ax = b$ em que $||x||_1$ é a norma $L_1$ de $x$, ou seja, a soma dos valores absolutos das componentes de $x$. Em seguida, numericamente, mostra-se que esta abordagem resolve enigmas de Sudoku.

O objetivo principal deste trabalho é estudar a aplicação de técnicas de otimização evolutivas na solução de grandes sistemas lineares. Para isto, problemas de otimização serão definidos com o propósito de obter a solução de sistemas lineares. O desempenho de duas técnicas de otimização evolutivas (Evolução Diferencial e Algoritmos Genéticos) será comparado com o de alguns métodos iterativos, tais como, Gradientes Conjugados (CG), Resíduos Mínimos (MINRES), Resíduos Mínimos Generalizados (GMRES), Gradientes Bi-

Conjugados (BiCG) em problemas considerados de difíceis soluções. A principal contribuição é disponibilizar um algoritmo que não dependa das características da matriz de coeficientes, podendo ser aplicado mesmo nos casos onde a matriz não seja simétrica e nem positiva-definida, sem a necessidade de trabalhar com pré-condicionadores para adequar o sistema aos métodos tradicionais.

A identificação indireta de forças externas em sistemas mecânicos é um problema complexo, tem como objetivo estimar as forças excitadoras (entrada) a partir das respostas medidas pelo comportamento estrutural (saída) e um modelo matemático dinâmico do sistema. Embora simples em sua essência, o problema de estimar forças a partir da resposta dinâmica é considerada um problema de difícil aplicação prática. Na verdade, a matriz resultante para o problema de identificação de forças tem-se revelado altamente mal condicionada do ponto de vista numérico.

Dada a importância crucial do tratamento do mal condicionamento numérico que caracteriza, em geral, os problemas inversos, um grande esforço vem sendo realizado visando o desenvolvimento de algoritmos numericamente estáveis. Na literatura, os problemas de identificação indireta de forças externas em sistemas mecânicos são resolvidos por métodos complexos, por exemplo, o método de valores singulares. Uma limitação óbvia das técnicas tradicionais de identificação de forças é que eles são incapazes de obter a solução de um sistema de grande porte, em que a matriz de coeficientes são não-simétricas, não positiva-definida e mal condicionadas. Assim, várias pesquisas ainda estão sendo feitas e aprimoradas nesta área de conhecimento. Neste trabalho duas técnicas evolutivas e uma iterativa não-estacionária foram testadas na solução de sistemas lineares, considerando-se problemas de identificação indireta de forças dinâmicas. A solução analítica é comparada com as soluções numéricas calculadas usando os métodos mencionados. Os resultados obtidos são apresentados, analisando os parâmetros mais importantes e sua influência sobre a convergência e eficiência dos métodos testados.

Os métodos naturais, dos quais os algoritmos evolucionários ou evolutivos fazem parte, se caracterizam pela busca da melhor solução através de regras de probabilidade, trabalhando de maneira “aleatória orientada”. Tais métodos utilizam apenas as informações da função de otimização, não requerendo informações sobre suas derivadas ou possíveis descontinuidades. Estas técnicas ganharam popularidade com a evolução dos computadores, já que requerem um número elevado de avaliações do problema. Isto é necessário para que se dê chance ao método de explorar devidamente toda a região do espaço de busca em que está

contida a solução ótima, resultando em grande número de avaliações da função, necessários para encontrar a solução.

Os Algoritmos Genéticos (AG) são métodos de busca semi-aleatória baseada na teoria de evolução de Charles Darwin. Esses métodos operam com conjuntos de candidatos, chamados de *população*, que são constantemente modificados utilizando dois princípios básicos da evolução natural das espécies: *seleção* e *variação*. Estes princípios tentam representar a competição onde os indivíduos mais aptos reproduzem e repassam seu material genético às gerações futuras, e os indivíduos menos aptos tendem a desaparecer da população (DEB, 2001). O método foi desenvolvido por HOLLAND (1975) na Universidade de Michigan e popularizado por um de seus alunos, GOLDBERG (1989), que apresentou a solução de problemas complexos de engenharia usando os algoritmos genéticos.

A idéia principal da Evolução Diferencial (ED), desenvolvido por STORN e PRICE (1995), é gerar novos indivíduos, denotados vetores doadores, pela adição da diferença vetorial ponderada entre dois indivíduos aleatórios da população a um terceiro indivíduo. Será mostrado que este princípio de usar diferenças de vetores para perturbar a população (indivíduos) resulta em um método de rápida convergência, de fácil implementação e bastante robusto.

Esta tese está dividida em 7 capítulos. O Capítulo 2 apresenta algumas definições fundamentais necessárias para o desenvolvimento do trabalho e relembra alguns conceitos fundamentais da Álgebra Linear. Nos Capítulos 3 e 4 é feita uma revisão dos métodos iterativos, termo este que se refere a um conjunto de técnicas que usam aproximações sucessivas para obter soluções mais precisas para um sistema linear, a cada passo. O Capítulo 5 traz uma fundamentação teórica dos métodos naturais de otimização: Algoritmos Genéticos e Evolução Diferencial.

No Capítulo 6, seção 6.1, é feita uma comparação entre os resultados obtidos com métodos iterativos e Evolução Diferencial (ED) por meio da utilização de sistemas testes. Na seção 6.2, utiliza-se um problema modelado por uma equação de Laplace, visando comparar a solução analítica com as soluções numéricas. Na seção 6.3 as técnicas utilizadas para a solução de grandes sistemas lineares são aplicadas a problemas de identificação indireta de forças dinâmicas. Inicialmente a metodologia é testada em um problema teórico, onde se faz a excitação por meio de uma força harmônica. A seguir, aplica-se um ruído branco e, de forma similar, são identificas as forças dinâmicas. Para verificar a eficácia da utilização de métodos evolutivos, conforme proposto nesta pesquisa, procurou-se identificar as forças dinâmicas

através dos dados obtidos experimentalmente. Os resultados e discussões desta aplicação estão apresentados na última subseção daquele capítulo. E, finalmente, no Capítulo 7 algumas discussões e conclusões são consideradas, bem como uma previsão de trabalhos futuros.

# Capítulo 2: Fundamentos Básicos

Neste capítulo, alguns conceitos fundamentais da Álgebra Linear serão apresentados através de algumas definições necessárias para o desenvolvimento do trabalho.

## 2.1 Definições Básicas

Seja $\mathbb{R}^n$ o espaço vetorial real. Define-se o vetor $x \in \mathbb{R}^n$ por:

$$x = \{x\} = \begin{Bmatrix} x_1 \\ \vdots \\ x_n \end{Bmatrix}, \quad x_i \in \mathbb{R} \tag{2.1}$$

Seja $\mathbb{R}^{m\times n}$ o espaço vetorial, real, de todas as matrizes de ordem $m\times n$, com $m, n \in \mathbb{N}^*$. Define-se a matriz $A \in \mathbb{R}^{m\times n}$ por:

$$A = (a_{ij}) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R} \tag{2.2}$$

Diz-se que uma matriz $A$ tem uma forma *Hessenberg superior* se $a_{ij} = 0$, sempre que $i > j+1$.

Uma matriz $A = (a_{ij})$, $i, j = 1, 2,..., n$, diz-se de diagonal estritamente dominante por linhas se

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}|, \; i = 1, 2, \ldots, n. \tag{2.3}$$

A norma de um vetor em $x \in \mathbb{R}^n$ é uma função $f: \mathbb{R}^n \to \mathbb{R}$ para a qual as seguintes propriedades são satisfeitas:

$$\begin{cases} f(x) \geq 0, \ \forall \ x \in \mathbb{R}^n, \ \ f(x) = 0 \ \text{ se, e somente se, } \ x = \mathbf{0} \ \ (\text{vetor nulo}) \\ f(x+y) \leq f(x) + f(y), \ \ \forall \ \ x, y \in \mathbb{R}^n \\ f(\alpha x) = |\alpha| \, f(x), \ \forall \ x \in \mathbb{R}^n \ \ \text{ e } \ \forall \alpha \in \mathbb{R} \end{cases} \tag{2.4}$$

A função norma $f(x)$ é denotada por $f(x) = ||x||$. Uma classe de normas muito útil e bastante utilizada é a *p-norma*, definida por:

$$\|x\|_p = \left( |x_1|^p + \cdots + |x_n|^p \right)^{1/p} = \left( \sum_{i=1}^{n} |x_i|^p \right)^{1/p}, \ \ p \geq 1 \tag{2.5}$$

Diz-se que uma seqüência de vetores de $\mathbb{R}^n$ $\{x^{(k)}\}$, com $k \in \mathbb{N}$ e $x^{(k)} \in \mathbb{R}^n$, converge para $x \in \mathbb{R}^n$ se:

$$\lim_{k \to \infty} \left\| x^{(k)} - x \right\| = 0 \tag{2.6}$$

A definição de norma de matriz é equivalente à definição de norma de um vetor. Em particular, a função $f$: $\mathbb{R}^{m \times n} \rightarrow \mathbb{R}$ é uma norma de matriz se satisfaz as seguintes condições:

$$\begin{cases} f(A) \geq 0, \ \forall A \in \mathbb{R}^{m \times n}, \ \ f(A) = 0 \text{ se, e somente se, } A = \mathbf{0} \ (\text{matriz nula}) \\ f(A+B) \leq f(A) + f(B), \ \forall A, B \in \mathbb{R}^{m \times n} \\ f(\alpha A) = |\alpha| \, f(A), \ \forall A \in \mathbb{R}^{m \times n} \text{ e } \forall \alpha \in \mathbb{R} \end{cases} \tag{2.7}$$

De forma análoga, a função norma $f(A)$ é denotada por $f(A) = ||A||$. Diz-se que uma sequência de matrizes de $\mathbb{R}^{m \times n}$, $\{A^{(k)}\}$, $k \in \mathbb{N}$, converge para $A \in \mathbb{R}^{m \times n}$ se:

$$\lim_{k \to \infty} \left\| A^{(k)} - A \right\| = 0 \tag{2.8}$$

Seja $A=(a_{ij})$ uma matriz quadrada $n \times n$, real. Denota-se por $A^T = (a_{ji})$ a matriz transposta de $A$. Diz-se que $A$ é uma matriz simétrica se $A^T = A$, ou seja, se $a_{ij} = a_{ji}$. Uma matriz quadrada é anti-simétrica se $A^T = - A$, ou $a_{ij} = - a_{ji}$. Obviamente todos os elementos

diagonais de uma matriz anti-simétrica devem ser nulos, pois $a_{ii} = - a_{ii} \Rightarrow a_{ii} = 0$. Diz-se que $A$ é positiva-definida se:

$$y^T Ay > 0, \text{ para qualquer } y \in \mathbb{R}^n \text{ e } y \neq \mathbf{0} \tag{2.9}$$

Se a matriz $A$ for positiva-definida e simétrica, pode-se afirmar que existe uma única matriz $L$, triangular inferior e uma única matriz $U$, triangular superior, tais que:

$$A = LL^T \text{ e } A = U^T U \tag{2.10}$$

Um subespaço de $\mathbb{R}^n$ é um subconjunto não-vazio $S \subset \mathbb{R}^n$ que é fechado para as operações de adição e multiplicação por escalar, ou seja, a soma de dois elementos de $S$ pertence a $S$ e a multiplicação de um elemento de $S$ por um escalar ainda é um elemento de $S$. Dados os vetores $v_1, v_2, ..., v_m \in \mathbb{R}^n$, uma *combinação linear* destes vetores é um vetor da forma $c_1v_1 + c_2v_2 + ... + c_mv_m$, onde as constantes $c_1, c_2, ..., c_m \in \mathbb{R}$. Os vetores $v_1, v_2, ..., v_m$ são *linearmente independentes* (*LI*) se a equação $c_1v_1 + c_2v_2 + ... + c_m v_m = 0$ tem apenas a solução nula, isto é, $c_1 = c_2 = ... = c_m = 0$. Se os vetores $v_1, v_2, ..., v_m$ não são linearmente independentes, então eles são chamados *linearmente dependentes* (*LD*).

O *posto* de uma matriz é o número de colunas linearmente independentes, que é o mesmo número de linhas linearmente independentes.

O *span* de $v_1, v_2, ..., v_m$ ou espaço gerado pelos vetores $v_1, v_2, ..., v_m$, denotado por $span\{v_1, v_2, ..., v_m\}$, é o conjunto de todas as combinações lineares de $v_1, v_2, ..., v_m$. É claro que $span\{v_1, v_2, ..., v_m\}$ é fechado para as operações de adição e multiplicação por escalar, logo $span\{v_1, v_2, ..., v_m\}$ é um subespaço vetorial de $\mathbb{R}^n$.

Dados um vetor $v \in \mathbb{R}^n$, uma matriz $A \in \mathbb{R}^{n \times n}$ e um número $k \in \mathbb{N}$, o subespaço $span\{v, Av, ..., A^{k-1}v\}$ é chamado o *k-ésimo subespaço de Krylov* denotado por:

$$K(v, A, k) \equiv span\{v, Av, ..., A^{k-1}v\} \tag{2.11}$$

Define-se a matriz de Krylov $K(v, A, k) \in \mathbb{R}^{n \times k}$ como sendo $K(v, A, k) = [v, Av, ..., A^{k-1}v]$.

Seja um subconjunto não-vazio $S \subset \mathbb{R}^n$. O *complemento ortogonal* de *S*, denotado por $S^{\perp}$ (lê-se *S-perp*), é definido como o conjunto de vetores de $\mathbb{R}^n$ que são ortogonais a todos os vetores de *S*. Isto é,

$$S^{\perp} \equiv \{u \in \mathbb{R}^n \mid \langle u, v \rangle = 0, \ \forall\, v \in S\} \tag{2.12}$$

O conjunto $S^{\perp}$ é não-vazio, uma vez que contém pelo menos o vetor nulo, é fácil ver que $S^{\perp}$ é fechado para as operações de adição e multiplicação por escalar, ou seja, $S^{\perp}$ é um subespaço vetorial de $\mathbb{R}^n$.

Seja $\mathbb{C}^{m\times n}$ o espaço vetorial complexo de todas as matrizes de ordem $m\times n$, $m$ *e* $n \in \mathbb{N}$. Define-se a matriz $A \in \mathbb{C}^{m\times n}$ por:

$$A = (a_{ij}) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix},\ a_{ij} \in \mathbb{C},\ 1 \le i \le m, 1 \le j \le n \tag{2.13}$$

A *transposta conjugada* de $A \in \mathbb{C}^{n\times n}$ é a matriz $A^H \in \mathbb{C}^{n\times n}$ cujas entradas são $\bar{a}_{ji}$ ($\bar{a}_{ji}$ denota o complexo conjugado de $a_{ji}$). Uma matriz *A* é dita ser *Hermitiana* se $A = A^H$, isto é, $a_{ij} = \bar{a}_{ji}$ para todo *i* e *j*. Uma matriz $A \in \mathbb{C}^{n\times n}$ *Hermitiana* que satisfaz:

$$x^H A x > 0, \text{ para qualquer } x \in \mathbb{C}^n \text{ e } x \neq \mathbf{0} \tag{2.14}$$

é dita positiva-definida.

Seja $A \in \mathbb{C}^{n\times n}$. Um vetor $v \in \mathbb{C}^n$ é chamado um autovetor de *A* se $v \neq \mathbf{0}$ e $Av$ é um múltiplo de *v*; isto é existe um $\lambda \in \mathbb{C}$ tal que $Av = \lambda v$. O escalar $\lambda$ é chamado o autovalor de *A* associado ao autovetor *v*. O conjunto de todos os autovalores de *A* é chamado o espectro de *A*, denotado por $\sigma(A)$. A máxima distância de um autovalor à origem é chamada raio espectral de *A*, denotado por $\rho\,(A)$.

## 2.2 Sistemas Lineares

Um dos problemas que aparece frequentemente em modelos matemáticos, que representam fenômenos físicos, é a resolução de sistemas de equações lineares. Um sistema de equações lineares, de ordem *n*, é uma coleção finita de *n equações lineares* todas com *n variáveis* (todas nas mesmas variáveis), consideradas em conjunto e normalmente apresentadas na forma:

$$\begin{cases} a_{11}\ x_1 + \cdots + a_{1n}\ x_n = b_1 \\ \quad \vdots \qquad \cdots \qquad \vdots \qquad \vdots \\ a_{n1}\ x_1 + \cdots + a_{nn}\ x_n = b_n \end{cases} \tag{2.15}$$

O sistema (2.15) também pode ser representado na forma matricial:

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \cdots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}, \text{ ou seja, } Ax = b \tag{2.16}$$

Sendo

$A$ = matriz dos coeficientes, ($A \in \mathbb{R}^{n\times n}$);

$x$ = vetor das variáveis (ou incógnitas), ($x \in \mathbb{R}^{n}$);

$b$ = vetor dos termos independentes ($b \in \mathbb{R}^{n}$).

Uma solução de um sistema de equações lineares nas variáveis $x_1, x_2, \ldots, x_n$ é uma sequência ordenada ($\alpha_1, \alpha_2, \ldots, \alpha_n$) de números tais que as substituições $x_i = \alpha_i$, $i = 1, \ldots, n$ transformam todas as equações do sistema em identidades verdadeiras. Existem vários estudos sobre métodos que sejam eficazes para a obtenção de solução de um sistema, sobretudo quando o sistema possui dimensão *n* grande. Este "problema" é de tal importância que incentivou os pesquisadores a criar e desenvolver computadores digitais que fossem capazes de resolver os sistemas de equações.

Muitas vezes a solução de um sistema linear é encontrada através da minimização de seu vetor resíduo. O vetor resíduo de um sistema linear é definido por:

$$r(x) = b - Ax \tag{2.17}$$

Os sistemas lineares podem, de forma geral, ser resolvidos pelos seguintes grupos de métodos (ACTON, 1990, 1996, BULIRSCH *et al.*,1992; PRESS *et al.*,1992): *métodos diretos e métodos indiretos*.

Métodos Diretos são aqueles que, na ausência de erros de arredondamento, permitem a obtenção da solução de um sistema com um número finito e bem determinado de operações aritméticas. Dentre os métodos diretos podem-se citar os métodos: *Gauss-Jordan, Decomposição LU* e *Cholesky* (PRESS *et al.*, 1992; ISAACSON e KELLER,1994).

Métodos Indiretos, ou Iterativos, são métodos que a partir de uma aproximação inicial procura-se a solução do sistema aplicando um processo iterativo. A idéia geral dos métodos iterativos é converter o sistema de equações lineares $Ax = b$ em um processo recursivo tal que $x^{(k)} = Cx^{(k-1)} + g = \varphi(x^{(k-1)}, A, b)$ para uma condição inicial $x^{(0)}$ conhecida. Estes métodos se dividem em Métodos Iterativos Estacionários e Não-Estacionários.

Métodos Iterativos Estacionários são aqueles cujos operadores matriciais que atuam sobre $x^{(k-1)}$ para produzir $x^{(k)}$ são constantes para todas as iterações. Os métodos iterativos estacionários mais conhecidos são: *Jacobi*, *Gauss-Seidel*, e *SOR* (*Successive Over Relaxation*). Estes métodos são mais antigos, de fácil entendimento e implementação, mas, normalmente, não são muito eficientes.

Métodos Iterativos Não-Estacionários (AXELSON, 1996; BARRET *et al.*, 1994; GOLUB e van LOAN, 1993). Estes métodos diferem dos estacionários devido ao fato de que os operadores matriciais que atuam sobre $x^{(k-1)}$ para produzir $x^{(k)}$ não são constantes ao longo do processo iterativo. Estes métodos se baseiam na teoria de otimização, ou seja, buscam transformar o sistema $Ax = b$ em um problema de minimização do resíduo $\|r_k\| = \min \|b - Ax^{(k)}\|$ e procuram o mínimo caminhando em uma direção de busca bem definida. A cada iteração tem-se uma nova direção de busca da solução, o que justifica o nome não-estacionário. Nesta categoria podem ser destacados: *Gradientes Conjugados* (*CG*), *Resíduos Mínimos* (*MINRES*), *Resíduos Mínimos Generalizados* (G*MRES*), *Resíduos Quase-Mínimos* (Q*MR*), *Gradientes Bi-Conjugados* (*BiCG*), *Gradientes Conjugados ao Quadrado* (*CGS*), *Gradientes Biconjugados Estabilizado* (*BiCG-STAB*), *Iteração de Chebyshev e Lanczos* (LANCZOS, 1950, 1952). Estes métodos são relativamente recentes, sua análise é normalmente mais complexa, podendo ser altamente eficientes.

## 2.3 Sistemas Lineares Esparsos

Um sistema linear $Ax = b$ é dito esparso quando a matriz de coeficientes $A$ possui uma grande parte de seus elementos iguais a zero. O armazenamento e a realização de operações matemáticas com elementos nulos normalmente se tornam desnecessários. Espaços de armazenamento e esforços computacionais podem ser perdidos. A idéia da esparsidade é armazenar e operar somente os elementos de uma matriz que não sejam nulos. Pode-se economizar um espaço significativo de memória se apenas os termos diferentes de zero forem armazenados. As operações usuais sobre estas matrizes (somar, multiplicar, inverter, pivotar) também podem ser feitas em tempo muito menor se não forem armazenadas as posições que contém zeros. Matrizes esparsas têm aplicações em problemas de engenharia, física e em computação, por exemplo, o método das malhas para resolução de circuitos elétricos ou sistemas de equações lineares, o armazenamento de dados e planilhas eletrônicas, etc.

A esparsidade $\varepsilon(n)$, em função da ordem de $A$, é definida como sendo o quociente entre a quantidade de elementos nulos de $A$ pela sua dimensão, como segue:

$$\varepsilon(n) \equiv \frac{quantidade \quad de \quad elementos \quad nulos \quad de \quad A}{n^2} = \\ = 1 - \frac{quantidade \quad de \quad elementos \quad não-nulos \quad de \quad A}{n^2} \tag{2.18}$$

Tabela 2.1 – Esparsidade para matrizes tri-diagonais em função da ordem n.

| $n$ | $\varepsilon(n)$ | $\varepsilon(n)$ % |
|---|---|---|
| 10 | 0.72 | 72 |
| 100 | 0.9702 | ≈97 |
| 1000 | 0.997002 | ≈99.7 |
| 10000 | 0.99970002 | ≈99.97 |
| 100000 | 0.9999700002 | ≈99.997 |
| 1000000 | 0.999997000002 | ≈99.9997 |

A Tab. 2.1 apresenta a esparsidade $\varepsilon(n)$, dada pela Eq. (2.18), para matrizes tridiagonais em função de sua ordem $n$, (conforme APARECIDO, 2004).

## 2.4 Métodos Diretos para Solução de Sistemas Lineares

Métodos diretos de solução transformam o sistema $Ax = b$ em outro, dito equivalente, cujas soluções são as mesmas do sistema original. A vantagem do sistema equivalente é que a sua matriz de coeficientes possui características que o tornam mais simples de ser resolvido do que o sistema original. A desvantagem é que, caso a matriz $A$ seja esparsa, estes métodos podem destruir sua esparsidade, no todo ou em parte. Os métodos diretos mais comuns são: *Gauss-Jordan, Decomposição LU e Cholesky*. Segundo alguns autores, a decomposição *LU* é uma consequência do método de Gauss-Jordan e a decomposição de Cholesky é um caso especial da decomposição *LU.*

### 2.4.1 Gauss-Jordan

O nome do método de Gauss-Jordan faz referência aos cientistas Carl Friedrich Gauss (1777-1855) e Wilhelm Jordan (1842-1899). Gauss popularizou este método, conhecido como método de Gauss ou eliminação gaussiana, que consiste na transformação de um sistema de equações lineares por meio de operações elementares em um sistema equivalente na forma escalonada. Apesar da denominação do método ser uma homenagem ao matemático alemão Carl Friedrich Gauss, referências sobre esse método já existiam em antigos textos indianos e chineses, de cerca de 2000 anos atrás. Entretanto, os europeus o desconheciam até que Gauss o utilizou enquanto trabalhava num sistema de equações que buscava determinar, por aproximação, a órbita do asteróide Ceres. Essa aproximação foi feita através do método dos quadrados mínimos, método também atribuído a Gauss. Outro matemático alemão, Wilhelm Jordan, fez uma modificação e popularizou o método no século XIX e esta modificação é denominada método de Gauss-Jordan ou eliminação de Gauss-Jordan.

Este método consiste na transformação da matriz de coeficiente em outra, pela aplicação de sucessivas operações elementares sobre as equações do sistema. Essas operações são: multiplicar uma equação por uma constante; adicionar um múltiplo de uma equação a outra equação e trocar a posição de duas equações. Tais operações são aplicadas ao sistema em uma ordem que, ao final, a matriz de coeficientes do sistema transformado seja diagonal, mais especificamente, o sistema original $Ax = b$ transforma-se em:

$$Dx = c \tag{2.19}$$

onde $D$ é uma matriz diagonal e $c$ é um vetor, ambos com as dimensões de $A$ e $b$, respectivamente. Com isso o sistema linear (2.19) é, então, resolvido diretamente.

Devem-se tomar alguns cuidados durante a aplicação do método de Gauss-Jordan, como garantir que os termos $a_{ii}$ não sejam nulos.

### 2.4.2 Decomposição *LU*

Em álgebra linear, a decomposição LU (em que *LU* vem do inglês *lower* e *upper*) é uma forma de fatoração de uma matriz não singular como o produto de uma matriz triangular inferior (*lower*) e uma matriz triangular superior (*upper*), ambas com a mesma dimensão de *A*. Às vezes se deve pré-multiplicar a matriz a ser decomposta por uma matriz de permutação. Esta decomposição se usa em análise numérica para resolver sistemas de equações (mais eficientemente) ou encontrar as matrizes inversas. Embora as idéias tenham sido conhecidas antes, muitas vezes o crédito pela popularização da decomposição *LU* é atribuída ao lógico e matemático britânico Alan Turing (precursor do computador), pelo seu trabalho de 1948, nesse assunto.

A decomposição *LU* é uma variante do método de Gauss-Jordan. É uma das técnicas mais usadas para resolver sistemas de equações algébricas. A decomposição LU é feita usando eliminação de Gauss, registrando em uma matriz diagonal unitária, os valores multiplicados pela linha pivô. Após a decomposição, resolver o sistema linear original $Ax = b$, torna-se o mesmo que resolver dois sistemas de equações lineares mais simples:

$$\begin{cases} Ly = b \\ Ux = y \end{cases} \tag{2.20}$$

Cuja solução pode ser obtida, facilmente, por substituição inversa e direta, respectivamente.

Como se percebe, a matriz $U$ é toda de zeros abaixo da diagonal principal e a matriz $L$ é toda de zeros acima da unitária diagonal principal. Pode-se, para economizar memória, armazenar as matrizes $L$ e $U$ em uma só matriz $A^*$ e um vetor $K$ com registro das trocas de linhas feitas durante a decomposição *LU*. Observe que se $A$ é originalmente esparsa, devido

ao processo de decomposição, $A^*$ pode se tornar não esparsa, aumentando, desta forma, tanto o espaço necessário para armazenar a matriz, como o esforço computacional para calcular a solução. Este é o principal problema que apresenta o método da decomposição *LU*.

### 2.4.3 Cholesky

No caso em que a matriz do sistema linear é simétrica pode-se simplificar os cálculos da decomposição *LU* significativamente, levando em conta a simetria. Esta é a estratégia do método de Cholesky. O método da fatoração de Cholesky (ou decomposição de Cholesky) foi assim denominado depois que André-Louis Cholesky estabeleceu que uma matriz simétrica e positiva-definida pode ser decomposta em uma matriz triangular inferior e sua transposta. Este método procura decompor uma matriz *A* na forma:

$$A = LL^T \tag{2.21}$$

sendo $L$ uma matriz triangular inferior com elementos da diagonal principal estritamente positivos. Para tanto, exige-se que esta matriz seja simétrica e positiva-definida.

Desta forma, substituindo (2.21) em $A\,x = b$ tem-se

$$(LL^T)\,x = b \tag{2.22}$$

o qual pode, agora, ser decomposto nos dois sistemas triangulares seguintes:

$$\begin{cases} L^T x = y \\ Ly = b \end{cases} \tag{2.23}$$

Que podem ser facilmente resolvidos, com no caso da decomposição *LU*.

Para os sistemas onde a matriz de coeficientes é simétrica, contudo, não positiva-definida, não se pode aplicar o algoritmo. Assim, pode-se utilizar o método de Cholesky para verificar se a matriz *A* simétrica é positiva-definida. Na literatura, como por exemplo WATKINS (2002), GOLUB e van LOAN(1993) ou SULLI e MAYRERS (2003), se encontra facilmente a demonstração do seguinte teorema:

***Teorema 2.1 (Cholesky)***: *Uma matriz simétrica A é positiva-definida se, e somente se, pode ser decomposta como* $A = LL^T$*, onde L é uma matriz triangular inferior com elementos estritamente positivos na diagonal.*

Para este tipo de sistema pode-se utilizar uma forma alternativa do método de Cholesky, o método de Cholesky modificado, fazendo:

$$A = LDL^T \tag{2.24}$$

onde $D$ é uma matriz diagonal que pode ser armazenada em um vetor. No método de Cholesky modificado, se $A$ é esparsa, a sua forma decomposta $L$, geralmente, não o será, o que representa o principal problema do método.

A decomposição de Cholesky requer aproximadamente a metade das operações exigidas pela decomposição *LU*. O algoritmo de Cholesky é incondicionalmente estável. Como $A$ é positiva-definida, não há necessidade de pivoteamento, pois neste caso ela sempre é diagonal dominante, veja por exemplo, SULLI e MAYRERS (2003), pag. 88 – 89.

Em relação às matrizes obtidas a partir da discretização de EDPs, os métodos diretos normalmente não apresentam boa eficiência computacional. A razão é que estes métodos não aproveitam a esparsidade da matriz de coeficientes para acelerar o processo de solução. Pelo contrário, durante a aplicação dos métodos, elementos da matriz de coeficientes que eram nulos deixam de sê-lo, não podendo mais ser ignorados. Isso aumenta tanto o espaço para armazenamento como o esforço computacional para calcular a solução numérica.

Além disso, durante a execução dos métodos diretos, pode haver um acumulo de erros de arredondamento, fazendo com que a solução calculada seja diferente da solução real do sistema linear. Portanto, métodos diretos de solução não são, em geral, empregados na resolução de problemas com grande número de incógnitas ou cujas matrizes de coeficientes sejam esparsas. Nesses problemas utilizam-se os métodos iterativos.

# Capítulo 3: Métodos Iterativos Estacionários

Neste capítulo é feita uma revisão dos métodos iterativos. Este termo refere-se a um conjunto de técnicas que usam aproximações sucessivas para obter soluções mais precisas para um sistema linear, a cada iteração do algoritmo. Nos métodos iterativos a solução é definida como o limite de uma sucessão infinita de vetores.

Em certos casos, tais métodos são particularmente mais eficientes (tempo e memória) do que os diretos, por exemplo, quando a matriz dos coeficientes de grande dimensão é uma matriz esparsa. Eles ainda são mais econômicos no sentido que utilizam menos memória do computador. Podem também, sob certas condições, ser aplicados para resolver um conjunto de equações não lineares.

A idéia geral dos métodos iterativos é converter o sistema de equações $Ax = b$ em um processo iterativo:

$$x = Cx + g = \varphi(x, A, b), \tag{3.1}$$

onde $C$ é uma matriz com dimensões $n$ x $n$, chamada matriz de iteração, $g$ é um vetor com dimensões $n$ x $1$, chamado vetor de iteração e $\varphi(x, A, b)$ é a função de iteração matricial

Partindo de um vetor aproximação inicial $x^{(0)}$, constrói-se uma seqüência iterativa de vetores:

$$\begin{aligned} x^{(1)} &= Cx^{(0)} + g = \varphi(x^{(0)}, A, b) \\ x^{(2)} &= Cx^{(1)} + g = \varphi(x^{(1)}, A, b) \\ &\vdots \\ x^{(k)} &= Cx^{(k-1)} + g = \varphi(x^{(k-1)}, A, b) \end{aligned} \tag{3.2}$$

Se a sequência de aproximação $x^{(0)}, x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$, converge para um valor $\alpha$, ou seja, é tal que $\lim_{k\to\infty} x^{(k)} = \alpha \implies \alpha = C\alpha + g$, então $\alpha$ é a solução do sistema $Ax = b$.

As matrizes $C$ e $g$ da Eq. (3.1) são construídas a partir da decomposição da matriz de coeficientes $A$. Diferentes decomposições originam métodos iterativos distintos. Os métodos iterativos se dividem em Estacionários e Não-Estacionários.

Um método iterativo é estacionário se cada aproximação é obtida da anterior sempre pelo mesmo processo. Ou seja, quando se expressa o processo iterativo da seguinte forma:

$$x^{(k)} = C\,x^{(k-1)} + g, \text{ com } k = 1, 2, 3, \ldots \tag{3.3}$$

tanto a matriz de iteração $C$ quanto o vetor de iteração g são constantes ao longo de todas as iterações.

No caso de métodos iterativos, torna-se necessário saber se a sequência que está sendo obtida está convergindo ou não para a solução desejada. O estudo da convergência destes métodos é dado no Teorema 3.1, cuja demonstração pode ser verificada em GOLUB e van LOAN (1993).

***Teorema 3.1*****:** *Seja* $\bar{x}$ *a solução única do sistema* $Ax = b$*. A sequência* $\{x^{(k)}\}_{k\in\mathbb{N}}$*, obtida do processo iterativo* (3.3)*, converge para a solução* $\bar{x}$ *se, e somente se,* $||C|| < 1$*, para alguma norma e para qualquer aproximação inicial* $x^{(0)} \in \mathbb{R}^n$ *considerada. Mais ainda, verifica-se que:*

$$\left\| x - x^{(k)} \right\| \leq \|C\|^k \left\| x - x^{(0)} \right\|. \tag{3.4}$$

Note que o critério de convergência não depende do vetor inicial $x^{(0)}$, sendo que este apenas influencia no número de iterações necessárias para atingir a precisão desejada.

Os métodos iterativos são ditos estacionários porque os operadores matriciais que atuam sobre $x^{(k-1)}$ para produzir $x^{(k)}$ são constantes para todas as iterações. Inspecionando-se estas equações percebe-se que não é necessário fazer transformações nos elementos de $A$, apenas efetuar multiplicações de $A$ (ou partes de $A$: $L$, $D$ e $U$) por algum vetor. Estes processos preservam a esparsidade de $A$.

A seguir serão apresentados alguns dos principais métodos iterativos estacionários: método de Jacobi, método de Gauss-Seidel, método SOR (Successive Over Relaxation) e método SSOR (Symmetric Successive Over relaxation).

## 3.1 Método de Jacobi

Trata-se de um algoritmo que para determinar a solução de um sistema de equações lineares trabalha com os maiores valores absoluto em cada linha e coluna dominadas pelo elemento da sua diagonal. O método tem o nome do matemático Alemão Carl Gustav Jakob Jacobi.

Considere o sistema original $Ax = b$, supondo que $a_{ii} \neq 0$ para todo $i = 1, 2, ..., n$. O método de Jacobi transforma o sistema linear $Ax = b$ em $x = Cx + g$, fazendo o isolamento do vetor $x$ mediante a separação pela diagonal, obtendo-se:

$$\begin{cases} x_1 = \dfrac{1}{a_{11}}(b_1 - a_{12}x_2 - \cdots - a_{1n}x_n) \\ x_2 = \dfrac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3 - \cdots - a_{2n}x_n) \\ \quad \vdots \\ x_n = \dfrac{1}{a_{nn}}(b_n - a_{n1}x_1 - a_{n2}x_2 - \cdots - a_{nn-1}x_{n-1}) \end{cases} . \tag{3.5}$$

Ou de forma simplificada:

$$x_i = \left( b_i - \sum_{j=1}^{i-1} a_{i,j}x_j - \sum_{j=i+1}^{n} a_{i,j}x_j \right) \Big/ a_{i,i} \tag{3.6}$$

Desta forma, tem-se $x = Cx + g$, onde

$$\mathrm{C} = \begin{bmatrix} 0 & -a_{12}/a_{11} & -a_{13}/a_{11} & \ldots -a_{1n}/a_{11} \\ -a_{21}/a_{22} & 0 & -a_{23}/a_{22} & \ldots -a_{2n}/a_{22} \\ \vdots & \vdots & \vdots & \ddots \quad \vdots \\ -a_{n1}/a_{nn} & -a_{n2}/a_{nn} & -a_{n3}/a_{nn} & \ldots \quad 0 \end{bmatrix} \quad \text{e} \quad g = \begin{bmatrix} b_1/a_{11} \\ b_2/a_{22} \\ \vdots \\ b_n/a_{nn} \end{bmatrix}. \tag{3.7}$$

Assim o método de Jacobi consiste em, dado uma aproximação inicial $x^{(0)}$, obter uma sequência $x^{(0)}, x^{(1)}, x^{(2)}, ..., x^{(k)}, \ldots$, através da relação recursiva $x^{(k)} = Cx^{(k-1)} + g$, ou seja:

$$x_i^{(k)} = \left( b_i - \sum_{j=1}^{i-1} a_{i,j} x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(k-1)} \right) \Big/ a_{i,i} \qquad (3.8)$$

Note que a ordem a qual as equações são examinadas é irrelevante, uma vez que o método de Jacobi as trata independentemente, por isto este método também é conhecido como o método de deslocamentos simultâneos, uma vez que poderiam ser feitas, a princípio, atualizações simultaneamente. Esta característica torna o Método de Jacobi adequado para utilizar a computação paralela na solução de grandes sistemas lineares.

Outra possibilidade de apresentar o método consiste em fazer a decomposição da matriz $A$ na forma $A = M_J - N_J = D - (L + U)$, onde $D$ é a diagonal de $A$, $L$ é a matriz triangular estritamente inferior e $U$ a matriz triangular estritamente superior obtidas de $A$. Assim o sistema linear $Ax = b$ pode ser escrito como:

$$(M_J - N_J)x = b \text{ ou } [D - (L + U)]x = b \qquad (3.9)$$

ou seja,

$$x = D^{-1}(L + U)x + D^{-1}b \qquad (3.10)$$

o qual sugere o esquema iterativo

$$x^{(k)} = D^{-1}(L + U)x^{(k-1)} + D^{-1}b \text{ ou } x^{(k)} = D^{-1}[(L+U)\, x^{(k-1)} + b], \quad k = 1, 2, 3,\ldots \qquad (3.11)$$

Comparando esta expressão com $x^{(k)} = Cx^{(k-1)} + g$ pode-se fazer a seguinte identificação:

$$C = D^{-1}(L + U) \text{ e } g = D^{-1}b \qquad (3.12)$$

Como critério de convergência, foi visto que a matriz de iteração $C$ deve satisfazer a condição $\|C\| < 1$. Usando a Norma do Máximo das Linhas sobre a matriz $C$ em (3.12) tem-se o seguinte critério de convergência para o Método de Jacobi:

***Teorema 3.2*** (Critério das Linhas): *Dado o sistema linear Ax = b. Seja os* $c_{ij}$ *de tal forma que*

$$c_{ij} = \frac{1}{|a_{ii}|}\sum_{\substack{j=1 \\ j\neq i}}^{n}|a_{ij}| < 1, \text{ para } i = 1, 2, ..., n \tag{3.13}$$

*Então o Método de Jacobi gera uma sequência convergente para a solução única do sistema, para qualquer escolha de* $x^{(0)}$.

**Demonstração**: Se for provado que $||C|| < 1$ então o Teorema 3.1 assegura imediatamente a convergência do método.

Como visto na Eq. (3.12), $C = D^{-1}(L + U)$, verifica-se, facilmente, que se $C = (c_{ij})$, para $i, j = 1, 2,...,n$, então

$$\begin{cases} c_{ii} = 0\,, \\ c_{ij} = -\dfrac{a_{ij}}{a_{ii}}, \quad i \neq j \end{cases} \tag{3.14}$$

pelo que, tomando a norma do máximo,

$$\|C\|_{\infty} = \max_i \sum_{j=1}^{n}|c_{ij}| = \max_i \sum_{\substack{j=1 \\ j\neq i}}^{n}\frac{|a_{ij}|}{|a_{ii}|} = \max_i \frac{\sum_{\substack{j=1 \\ j\neq i}}^{n}|a_{ij}|}{|a_{ii}|} < 1, \tag{3.15}$$

em que a última passagem se justifica pela definição de matriz com diagonal estritamente dominante por linhas. □

Este critério fornece uma condição suficiente, mas não necessária. Isto é, o critério pode não ser satisfeito e o método ainda pode convergir. Outros critérios podem ser obtidos usando outras normas.

## 3.2 Método de Gauss-Seidel

É um método iterativo para resolução de sistemas de equações lineares. O seu nome é uma homenagem aos matemáticos alemães Carl Friedrich Gauss e Philipp Ludwig von Seidel. É semelhante ao método de Jacobi. Da mesma forma que no método de Jacobi, no método de Gauss-Seidel o sistema linear $Ax = b$ é escrito na forma equivalente $x = Cx + g$, por separação da diagonal. O processo iterativo consiste em, sendo $x^{(0)}$, uma aproximação inicial, obter uma sequência $x^{(0)}, x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$, através de:

$$x_i^{(k)} = \left( b_i - \sum_{j=i}^{i-1} a_{i,j} x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(k-1)} \right) \Big/ a_{i,i} \tag{3.16}$$

O método de Gauss-Seidel difere do método de Jacobi apenas no fato de que os valores atualizados dos elementos $x_i^{(k)}$ são utilizados, prontamente, assim que estejam disponíveis. Portanto, no processo iterativo de Gauss-Seidel, no momento de se calcular $x_j^{(k)}$ utiliza-se todos os valores $x_1^{(k)}, x_2^{(k)}, \ldots, x_{j-1}^{(k)}$, que já foram calculados, bem como os restantes valores $x_{j+1}^{(k-1)}, x_{j+2}^{(k-1)}, \ldots, x_n^{(k-1)}$ da iteração anterior.

Em termos matriciais considere-se a decomposição da matriz da matriz de coeficientes $A = M_S - N_S = (D - L) - U$. O sistema $Ax = b$ pode escrever-se

$$(M_S - N_S)x = b \text{ ou } [(D - L) - U]x = b \tag{3.17}$$

Assim,

$$(D - L)x = Ux + b \text{ ou } x = (D - L)^{-1}Ux + (D - L)^{-1}b \tag{3.18}$$

que sugere o esquema iterativo

$$x^{(k)} = (D - L)^{-1}(U x^{(k-1)} + b), \quad k = 1, 2, 3, \ldots \tag{3.19}$$

Pode-se fazer a seguinte identificação, ao comparar esta expressão com $x^{(k)} = Cx^{(k-1)} + g$

$$C = (D - L)^{-1}U \text{ e } g = (D - L)^{-1} b \tag{3.20}$$

Tal como sucede com o método de Jacobi, o método de Gauss-Seidel é convergente para matrizes de diagonal estritamente dominante por linhas. Para o método de Gauss-Seidel, a análise da convergência pode ser feita pelo critério de Sassenfeld, cuja demonstração pode ser vista, por exemplo, em RUGGIERO e LOPES (1988), apresentado a seguir:

***Critério de Sassenfeld***: *Dado o sistema* $Ax = b$**.** *Seja* $\beta_i$ *tal que*

$$\beta_i = \frac{1}{|a_{i,i}|}\left(\sum_{j=1}^{i-1}|a_{i,j}|\,\beta_j + \sum_{j=i+1}^{n}|a_{i,j}|\right) < 1, \quad i = 1,2,3,... \tag{3.21}$$

*Então, o método de Gauss-Seidel gera uma seqüência* $\{x^{(k)}\}$, $k \in \mathbf{N}$, *que converge para a solução do sistema.*

Em geral se o método de Jacobi converge, o método de Gauss-Seidel convergirá mais rapidamente, embora ainda relativamente lento comparado com outras técnicas. Portanto, o método de Gauss-Seidel tem um desempenho superior ao de Jacobi.

Dois fatos importantes sobre o método de Gauss-Seidel devem ser observados. Primeiro cada componente da nova iteração depende das componentes calculadas previamente, as atualizações não podem ser feitas simultaneamente como pelo método de Jacobi. Segundo, a nova iteração de $x^{(k)}$ depende da ordem com que as equações são examinadas. O método de Gauss-Seidel pode ser chamado de método de deslocamentos sucessivos, para indicar a dependência da ordem das iterações. Se esta ordem é mudada, as componentes da nova iteração também podem mudar.

Estes dois pontos são importantes porque se $A$ é esparsa, a dependência de cada componente da nova iteração das componentes anteriores não é absoluta. A presença de zeros na matriz pode remover a influência de algumas das componentes anteriores. Pode se reduzir tal dependência usando um ordenamento inteligente das equações, assim restabelecendo a capacidade de utilizar computação paralela. Como sempre, reordenar as equações pode afetar

a taxa à qual o método de Gauss-Seidel converge. Uma escolha ruim da ordenação pode diminuir a taxa de convergência; uma boa escolha pode aumentar a taxa de convergência.

## 3.3 Método SOR (Successive Over Relaxation)

Em álgebra linear numérica, o método de sobre-relaxação sucessiva (SOR) é uma variante do método de Gauss-Seidel para a resolução de um sistema de equações lineares, resultando em uma convergência mais rápida. Foi concebido simultaneamente por David M. Young e H. Frankel em 1950, com a finalidade de resolver sistemas lineares automaticamente em computadores digitais.

Este método consiste em considerar um parâmetro real ω não nulo, chamado fator de relaxação ou extrapolação, e definir:

$$x_i^{(k)} = \omega \bar{x}_i^{(k)} + (1-\omega)\, x_i^{(k-1)}, \qquad k = 1, 2, 3, \ldots \tag{3.22}$$

onde $\bar{x}$ denota a aproximação obtida pelo método de Gauss-Seidel. Esta extrapolação leva a forma de uma média ponderada entre a iteração anterior do SOR e iteração seguinte calculada pelo método de Gauss-Seidel, para cada componente. A idéia é escolher um valor $\omega$ que acelera a taxa de convergência para a solução.

Como nos métodos anteriores, considerando a decomposição da matriz de coeficientes $A = M_\omega - N_\omega$. O sistema $Ax = b$ pode escrever-se

$$(M_\omega - N_\omega)x = b \text{ ou } M_\omega x = N_\omega + \omega b \tag{3.23}$$

Fazendo

$$M_\omega = D - \omega L \text{ e } N_\omega = (1-\omega)D + \omega U \tag{3.24}$$

Assim o sistema $M_\omega x = N_\omega x + \omega b$ toma a seguinte forma:

$$(D - \omega L)x = [(1 - \omega)D + \omega U]\}x + \omega b \tag{3.25}$$

Ou

$$x = (D - \omega L)^{-1}[\omega U + (1 - \omega)D]x + \omega (D - \omega L)^{-1}b \tag{3.26}$$

o que sugere o esquema iterativo

$$x^{(k)} = (D - \omega L)^{-1}[\omega U + (1-\omega)D]x^{(k-1)} + \omega(D - \omega L)^{-1}b, \;\; k = 1, 2, 3, \ldots \tag{3.27}$$

Comparando esta expressão com $x^{(k)} = Cx^{(k-1)} + g$, resulta em

$$C = (D - \omega L)^{-1}[\omega U + (1 - \omega)D] \;\text{ e }\; g = \omega (D - \omega L)^{-1} b \tag{3.28}$$

Se $\omega = 1$ o método SOR se reduz ao método de Gauss-Seidel. As condições necessárias e suficientes para a convergência do método SOR são dadas pelos seguintes teoremas, cujas demonstrações se encontram facilmente em qualquer livro de analise numérica, como por exemplo, ORTEGA (1972) pag. 123 - 133.

***Teorema 3.3*** (*Kahan - condição necessária*)*: Para que haja convergência do método SOR, qualquer que seja a aproximação inicial* $x^{(0)}$*, é necessário que* $\omega$ *pertença ao intervalo* (*0, 2*).

***Teorema 3.4*** (*Ostrowski-Reich - condição suficiente*)*: Se a matriz A for simétrica e positiva-definida, se* $\omega$ *pertencer ao intervalo* (*0, 2*)*, então o método SOR converge, para qualquer que seja a aproximação inicial* $x^{(0)}$.

Em geral, não é possível calcular com antecedência o valor ótimo de $\omega$ relacionado à taxa de convergência do SOR. Até mesmo quando é possível calcular o valor ótimo de $\omega$, o custo computacional é normalmente proibitivo. A escolha de $\omega$ pode afetar significativamente a taxa de convergência do SOR. Podem-se empregar algoritmos adaptáveis para determinar $\omega$, utilizando estimativas para a taxa na qual a iteração corrente está convergindo.

Para matrizes de coeficientes *A*, provenientes da discretização de EDPs elípticas, há uma relação direta entre o espectro da matriz de Jacobi e da matriz de iteração do método SOR. A princípio, determinado o raio espectral $\rho$ da matriz de iteração de Jacobi pode-se determinar a priori, teoricamente, o valor ótimo de $\omega$ para o SOR, da seguinte maneira:

$$\omega_{opt} = \frac{2}{1+\sqrt{1-\rho^2}} \tag{3.29}$$

Isto raramente ocorre, uma vez que calcular o raio espectral da matriz de Jacobi requer um custo computacional geralmente muito alto. Porém, estimativas aproximadas de $\rho$ podem gerar estimativas razoáveis, para o valor ótimo de $\omega$.

## 3.4 Método SSOR (Symmetric Successive Over Relaxation)

Assim como o SOR o método SSOR é creditado ao matemático e cientista da computação, americano, David M. Young. Este método consiste em, desde que a matriz de coeficientes *A* seja simétrica, aplicar simultaneamente dois métodos SOR, de tal modo que a matriz de iteração resultante seja semelhante a uma matriz simétrica. Especificamente, a primeira "varredura" de SOR é realizada conforme a Eq. (3.27), mas na segunda "varredura", as variáveis desconhecidas são atualizadas na ordem inversa. Isto é, SSOR é uma varredura para frente de SOR seguida por uma para trás, ou seja:

$$\begin{aligned} x_i^{\left(\mathrm{k}+\frac{1}{2}\right)} &= \omega\left( b_i - \sum_{j=1}^{i-1} a_{i,j} x_j^{\left(\mathrm{k}+\frac{1}{2}\right)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(\mathrm{k})} \right) \Big/ a_{i,i} + (1-\omega) x_j^{(\mathrm{k})} \ , \ i = 1, 2, \cdots, n-1, n \\ x_i^{(\mathrm{k}+1)} &= \omega\left( b_i - \sum_{j=1}^{i-1} a_{i,j} x_j^{(\mathrm{k}+1)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{\left(\mathrm{k}+\frac{1}{2}\right)} \right) \Big/ a_{i,i} + (1-\omega) x_j^{\left(\mathrm{k}+\frac{1}{2}\right)} \ , \ i = n,\ n-1, \cdots, 2, 1 \end{aligned} \tag{3.30}$$

Em que *n* é a dimensão da matriz de coeficientes.

Desta forma, pode-se escrever o sistema $Ax = b$, considerando a decomposição da matriz $A = M_\omega - N_\omega$ para a primeira varredura do SOR e $A = \tilde{M}_\omega - \tilde{N}_\omega$ para a segunda varredura, tem-se:

$$
\begin{aligned}
&M_\omega x = N_\omega x + \omega b, \qquad i = 1, 2, \cdots, n-1, n \\
&\tilde{M}_\omega x = \tilde{N}_\omega x + \omega b \qquad i = n,\ n-1, \cdots, 2, 1
\end{aligned}
\tag{3.31}
$$

Fazendo

$$
\tilde{M}_\omega = D - \omega U \text{ e } \tilde{N}_\omega = (1-\omega)D + \omega L \tag{3.32}
$$

Como *A* é simétrica, tem-se o esquema iterativo:

$$
\begin{aligned}
&M x_i^{\left(k-\frac{1}{2}\right)} = N x_i^{(k-1)} + \omega b, \qquad i = 1,\ 2, \cdots,\ n-1,\ n \\
&M^T x_i^{(k)} = N^T x_i^{\left(k-\frac{1}{2}\right)} + \omega b, \quad i = n,\ n-1, \cdots,\ 2,\ 1
\end{aligned}
\tag{3.33}
$$

Ou melhorando

$$
x_i^{(k)} = (M^{-T} N^T M^{-1} N) x_i^{(k-1)} + (M^{-T} N^T M^{-1} + M^{-T}) \omega b \tag{3.34}
$$

Da definição de $M_\omega$ e $N_\omega$ pode-se escrever:

$$
x^{(k)} = B_1 B_2\, x^{(k-1)} + \omega(2 - \omega)(D - \omega U)^{-1} D (D - \omega L)^{-1} b, \ \ k = 1, 2, 3, \ldots \tag{3.35}
$$

onde

$$
\begin{aligned}
&B_1 = (D - \omega U)^{-1}[\omega L + (1-\omega)D] \\
&\qquad\qquad \text{e} \\
&B_2 = (D - \omega L)^{-1}[\omega U + (1-\omega)D]
\end{aligned}
\tag{3.36}
$$

Note que $B_2$ simplesmente é a matriz de iteração para SOR da Eq. (3.27) e que em $B_1$ os papéis de *L* e *U* são invertidos.

Comparando esta expressão com $x^{(k)} = Cx^{(k-1)} + g$, pode-se fazer a seguinte identificação $C = M^{-T} N^T M^{-1} N = B_1 B_2$ e $g = M^{-T}(N^T M^{-1} + M M^{-1})\omega b$, ou melhor:

$$C = (D - \omega U)^{-1}[\omega L + (1-\omega)D](D - \omega L)^{-1}[\omega U + (1-\omega)D]$$

e (3.37)

$$g = (D - \omega U)^{-1}\left\{\left[\omega L + (1-\omega)D\right](D - \omega L)^{-1} + (D - \omega L)(D - \omega L)^{-1}\right\}\omega b$$

É possível provar que $C$ é semelhante a uma matriz simétrica (GOLUB e van LOAN, 1993).

A semelhança da matriz de iteração SSOR com uma matriz simétrica permite a aplicação do SSOR como um pré-condicionador para outros métodos iterativos para matrizes simétricas. Assim, este método iterativo não apresenta qualquer vantagem sobre o método SOR, sendo mais utilizado como pré-condicionador para os métodos não-estacionários, pois sua taxa de convergência é geralmente mais lenta do que a taxa de convergência do SOR com $\omega$ ideal.

# Capítulo 4: Métodos Iterativos Não-Estacionários

A teoria dos métodos iterativos estacionários foi solidamente estabelecida com o trabalho de D.M. Young na década de 1950. Os *métodos iterativos não-estacionários* são mais recentes e, embora a teoria por trás destes algoritmos seja mais complexa, são normalmente mais eficientes do que os métodos estacionários. São métodos diretos na sua essência, mas na prática são usados como métodos iterativos. Portanto, a solução aproximada depende de uma tolerância pré-fixada. Sua eficiência depende de critérios de convergência relacionados à matriz dos coeficientes. A matriz dos coeficientes $A$ e o vetor dos termos independentes $b$ não são alterados durante o processo iterativo, portanto, preserva as características da matriz de coeficientes, como a sua esparsidade. Os cálculos da passagem da aproximação atual à seguinte envolvem informações que mudam a cada iteração, isto é, possuem parâmetros de busca que variam a cada iteração, assim, não trabalham com uma matriz de iteração. Esta flexibilidade justifica o melhor desempenho desses métodos em relação aos estacionários.

A essência dos métodos iterativos não-estacionários é transformar o sistema $Ax = b$ em um problema de minimização do resíduo $||r_k|| = \min ||b-Ax^{(k)}||$, em seguida procura-se o mínimo, caminhando em uma direção de busca bem definida, e a cada iteração a direção de busca da solução é modificada, o que justifica o nome não-estacionário. Como estes métodos utilizam-se de problemas de otimização, são derivados diretamente do método dos gradientes conjugados desenvolvido originalmente por Hestenes e Stiefel em 1952. Os vários métodos iterativos não-estacionários, para solução iterativa de sistemas lineares, são formulados adaptando-se o método dos gradientes conjugados de várias formas, levando-se em consideração as características do sistema: se a matriz $A$ é positiva-definida, simétrica ou genérica (BARRET *et al.*, 1994). Dentre os métodos iterativos não-estacionários mais conhecidos, podem-se citar os métodos dos Gradientes Conjugados, Minimização de Resíduos, Minimização de Resíduos Generalizados, Quase-Minimização de Resíduos, Gradientes Biconjugados, Gradientes Biconjugados Estabilizado e Lanczos.

O Método do Gradiente Conjugado foi desenvolvido originalmente por Hestenes e Stiefel em 1952, para a solução em $n$ passos de um conjunto de equações lineares simultâneas. Em sua forma geral, pode ser aplicado para otimização de funções no $\mathbb{R}^n$ . No final da década de 1960 o método foi estendido para a solução de problemas de controle ótimo.

Tomando uma variação deste método, a partir do sistema linear $Ax = b$ define-se o seguinte funcional:

$$r = r(x) = b - Ax \tag{4.1}$$

sendo $x$ um vetor genérico. O vetor $r(x)$ é definido como o vetor resíduo, cuja norma mede a distância de $x$ à solução do sistema. Desta forma, se a norma de $r(x)$ for nula, tem-se que $r(x)$ é um vetor nulo e $x$ é a solução do sistema linear. Portanto, obter a solução $x$ do sistema linear $Ax = b$ é o mesmo que encontrar o zero (vetorial) de $r(x)$. Como o resíduo $r(x)$ é linear em $x$ e infinitamente diferenciável, ou seja, $r(x) \in C^{\infty}(\mathbb{R}^n)$, pode-se definir o seguinte funcional quadrático:

$$F(x) = \frac{1}{2} x^T Ax - x^T b \tag{4.2}$$

cujo vetor gradiente é dado por:

$$\nabla F(x) = \frac{\partial F(x)}{\partial x} = Ax - b \Rightarrow \nabla F(x) = -r(x) \tag{4.3}$$

e a matriz Hessina é calculada como:

$$H(x) = \frac{\partial^2 F(x)}{\partial x^2} = A \Rightarrow H(x) = A \tag{4.4}$$

Quando $x$ for uma solução do sistema linear, $r(x) = 0$, será também um ponto crítico de $F(x)$, $\nabla F(x)$. Portanto, minimizar $F(x)$ é o mesmo que resolver o sistema linear. Logo, pode-se obter a solução do sistema, utilizando algum método de minimização do funcional $F(x)$. Assim, já se tem a garantia de que, sendo $x$ uma solução do sistema, $x$ é um ponto crítico de $F(x)$. Para que $x$ seja um ponto de mínimo deve-se assegurar que a matriz Hessiana $H(x)$ seja *positiva-definida* e também *simétrica*, ou seja:

$$H_{ij}(x) = \frac{\partial^2 F(x)}{\partial x_i \partial x_j} = \frac{\partial^2 F(x)}{\partial x_j \partial x_i} = H_{ji}(x) \tag{4.5}$$

Mas a matriz Hessiana de $F(x)$ é exatamente a matriz de coeficientes ($H = A$), assim, para garantir que $x$ seja a solução do sistema deve-se impor que a matriz $A$ seja simétrica e positiva-definida. E quando a matriz de coeficientes não tiver estas características, deve-se adequar este método para a obtenção da solução.

Os métodos iterativos baseados em métodos de otimização tem como característica principal, em seu processo iterativo, atualizar o vetor $x^{(k)}$ a partir da solução inicial $x^{(0)}$ e do resíduo inicial $r_0 = b - Ax^{(0)}$. Além disso, são preservadas as características da matriz de coeficientes $A$ como, por exemplo, sua esparsidade, pois necessitam apenas de produto matriz-vetor, além das operações entre vetores.

## 4.1 Método da Descida Máxima (Steepest Descent)

O Método da Descida Máxima, também conhecido com método dos gradientes puros, foi proposto por Box e Wilson em 1951. É uma das estratégias mais simples, porém pouco eficiente, para se minimizar o funcional $F(x)$. Na formação da sequência de aproximações $x^{(0)}$, $x^{(1)}$, $x^{(2)}$,..., $x^{(k)}$,..., toma, a cada passo, o sentido contrário ao do gradiente do funcional $F(x)$, determinando-se o passo $\alpha_k$ de tal maneira que na direção $r_k$ o funcional seja minimizado, isto é, $F(x^{(k)}) < F(x^{(k-1)})$. Denominando-se:

$$r_k = -\nabla F\left(x^{(k)}\right) \equiv b - Ax^{(k)} = r\left(x^{(k)}\right) \tag{4.6}$$

Para determinar $\alpha_k$, calcula-se:

$$F\left(x^{(k)} + \alpha_k r_k\right) = \frac{1}{2}\left(x^{(k)} + \alpha_k r_k\right)^T A\left(x^{(k)} + \alpha_k r_k\right) - \left(x^{(k)} + \alpha_k r_k\right)^T b$$

ou

$$F\left(x^{(k)} + \alpha_k r_k\right) = F\left(x^{(k)}\right) - \alpha_k r_k^T r_k + \frac{1}{2}\alpha_k^2 r_k^T A r_k . \tag{4.7}$$

mimizando (4.7) em relação a $\alpha_k$, resulta em:

$$\frac{\partial F}{\partial \alpha} = -r_k^T r_k + \alpha_k r_k^T A r_k = 0 \Rightarrow \alpha_k = \frac{r_k^T r_k}{r_k^T A r_k} \tag{4.8}$$

Assim

$$x^{(k)} = x^{(k-1)} + \frac{r_{k-1}^T r_{k-1}}{r_{k-1}^T A r_{k-1}} (r_{k-1}) \tag{4.9}$$

A Fig. 4.1 apresenta um algoritmo do Método da Descida Máxima.

Escolha $x^0$ como vetor inicial e ε como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $||r_k|| > \varepsilon$
    $k = k+1$
    $r_k = b - Ax^k$
    $\alpha_k = \frac{r_k^T r_k}{r_k^T A r_k}$
    $x^k = x^{k-1} + \alpha_{k-1} r_{k-1}$
fim

Figura 4.1 - Algoritmo do Método da Descida Máxima

Através das grandezas de $\lambda_{max}$ e $\lambda_{min}$, maior e o menor autovalor da matriz $A$, respectivamente, define-se o fator-condição da matriz $A$, $\kappa(A)$, por:

$$\kappa(A) = \frac{\lambda_{max}}{\lambda_{min}} \tag{4.10}$$

Em NOCEDAL e WRIGHT (2006) é provado que:

$$\left[ F(x^{(k)}) + \frac{1}{2} b^T A^{-1} b \right] \leq \left[ 1 - \frac{1}{\kappa(A)} \right] \left[ F(x^{(k-1)}) + \frac{1}{2} b^T A^{-1} b \right], \tag{4.11}$$

O que implica na convergência global. Para sistemas mal condicionados, cuja matriz de coeficientes tem o fator condição grande, infelizmente, a taxa de convergência pode ser extremamente lenta quando o fator-condição não for suficientemente próximo da unidade. Neste caso, geometricamente, as curvas de nível de $F$ são hiper-elipsóides muito alongadas e o sentido contrário à direção do gradiente pode ser muito diferente da direção do mínimo de $F(x)$.

## 4.2 Métodos dos Gradientes Conjugados (CG)

Todos os métodos iterativos discutidos até este agora são limitados por sua falta de memória iterativa. Cada um usa apenas informações sobre $x^{(k-1)}$ para obter $x^{(k)}$. Todas as informações das iterações anteriores são esquecidas. O método do gradiente conjugado é uma simples variação na descida máxima, que funciona melhor porque possui uma memória iterativa.

O método dos Gradientes Conjugados foi desenvolvido independentemente, na década de 1950, por Eduard Stiefel, do Instituto de Matemática Aplicada em Zurique e por Magnus Hestenes com a cooperação do J.B. Rosser, G. Forsythe e L. Paige, do Instituto de Análise Numérica, Departamento Nacional de Standards. Cornelius Lanczos desenvolveu, em 1952, uma rotina intimamente relacionada com base em seu estudo sobre o problema de autovalor. Mas a sua natureza e possíveis aplicações eram incompreendidas naquela época. Somente na década de 1970 revelaram que esses métodos funcionam bem para resolver equações diferenciais parciais, especialmente do tipo elípticas.

Neste método, a cada passo, a direção de busca $p_k$ é calculada levando-se em conta não apenas o sentido contrário da direção do vetor gradiente de $F(x)$, $g_k = \nabla F(x^{(k)})$, como no método da descida máxima, mas também o conjunto de informações disponíveis até o $k$-ésimo passo. Ao atingir o $k$-ésimo passo, tem-se acumuladas as informações:

$$\{p_0, p_1, ..., p_k, g_0, g_1, ..., g_k\} \tag{4.12}$$

Sejam $S_k$ o subespaço de $\mathbb{R}^n$ gerado pelo conjunto de vetores (4.12) e $V_k$ o conjunto $\{x^{(k)}\} \cup S_k$, que contém o ponto $x^{(k)}$. O cálculo de $p_k$ é feito da seguinte forma: para $k = 0$ tome

$p_0 = - g_0$ e para $k > 0$, $p_k$ é tomado como um elemento do subespaço $S_k$. Então, $p_k$ é uma combinação linear dos gradientes $g_0, g_1,..., g_k$, ou seja, $S_k$ coincide com o subespaço gerado pelos gradientes $g_0, g_1,..., g_k$. A direção $p_k$, deve ser tal que, se $x^{(k)} = x^{(k-1)} + \alpha_k p_k$ for tomado como ponto ótimo na direção de $p_k$, esse ponto será também ótimo em todo o conjunto $V_k$. Em outras palavras, $p_k \in S_k$ deve ser escolhido de tal forma que a restrição de $F$ a $V_k$ seja mínima no ponto $x^{(k)}$.

Em sua essência, o método dos Gradientes Conjugados, utiliza em cada iteração, a aproximação da solução ($x^{(k)}$), o resíduo ($r_k$) correspondente a essa aproximação e a direção de busca ($p_k$) usada para atualizar a aproximação e o resíduo. Os escalares de atualização ($\alpha_k$) são calculados, em toda iteração do método, através de dois produtos internos. Estes escalares são definidos para garantir que a sequência de aproximações satisfaça certas condições de ortogonalidade. Tais condições implicam que, em um sistema linear positiva-definido e simétrico, a distância entre as aproximações e a solução real é mínima em alguma norma.

Atualiza-se as aproximações $x^{(k)}$, em cada iteração, através do escalar ($\alpha_k$) multiplicando-o pelo vetor de direção de busca ($p_k$):

$$x^{(k)} = x^{(k-1)} + \alpha_k p_k, \; k = 1, 2, 3,\ldots \tag{4.13}$$

Da mesma forma são atualizados os resíduos $r_k = b - A x^{(k)}$:

$$r_k = r_{k-1} - \alpha_k A p_k, \text{ para } k = 1, 2, 3,\ldots \tag{4.14}$$

O valor do passo $\alpha_k$ é calculado como no método da descida máxima, ou seja:

$$\alpha_k = \frac{r_{k-1}^T r_{k-1}}{p_k^T A p_k} \tag{4.15}$$

Da invariância e da ortogonalidade dos gradientes de $F$ tem-se que as direções de busca são atualizadas usando os resíduos:

$$p_k = r_k + \beta_{k-1} p_{k-1} \tag{4.16}$$

Onde $\beta_k$ é dado por:

$$\beta_k = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}} \tag{4.17}$$

Isto é assegurado pelo fato de que $p_k$ e $p_{k-1}$, ou equivalentemente, $r_k$ e $r_{k-1}$ são ortogonais.

Como se viu, o método da descida máxima apresenta uma baixa convergência. Para evitar isto, considera-se o problema de minimizar o funcional $F(x)$ ao longo de um conjunto de direções $\{p_1, p_2, \ldots\}$ que não correspondem necessariamente aos gradientes $\{r_0, r_1, r_2, \ldots\}$, este procedimento será feito nas próximas seções.

Considere a expansão em série de Taylor do funcional $F(x^{(k-1)} + \alpha_k p_k)$, a qual pode ser minimizada por:

$$\alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k} \tag{4.18}$$

Para $\alpha_k$ definido desta forma, tem-se que:

$$F\left(x^{(k-1)} + \alpha_k p_k\right) = F\left(x^{(k-1)}\right) - \frac{1}{2}\frac{\left(p_k^T r_{k-1}\right)^2}{p_k^T A p_k} \tag{4.19}$$

Para assegurar que $F(x^{(k)}) < F(x^{(k-1)})$ é necessário que não se tenha $p_k$ ortogonal a $r_{k-1}$. Neste caso o algoritmo do método é o mostrado na Fig. 4.2, conhecido como o Método das direções Gerais de Busca.

Escolha $x^0$ como vetor inicial e ε como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $||r_k|| > \varepsilon$
    $k = k+1$
    Escolha uma direção $p_k$, tal que $p^T_k\, r_{k-1} \neq 0$
    $\alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k}$
    $x^k = x^{k-1} + \alpha_k p_k$
    $r_k = b - Ax^k$
fim

Figura 4.2 - Algoritmo do Método com direções gerais de busca

É claro que:

$$x_k \in x^0 + \text{span}\{p_1, p_2, \ldots, p_k\} \equiv \{x^0 + \gamma_1 p_1 + \gamma_2 p_2 + \ldots + \gamma_k p_k : \gamma_i \in R\} \quad (4.20)$$

As direções de busca $\{p_1, p_2, \ldots, p_k\}$ devem ser escolhidas adequadamente para que se tenha a garantia da convergência sem as dificuldades apresentadas no método da Descida Máxima.

### 4.2.1 Direções de Busca *A*-Conjugadas

O método da descida máxima freqüentemente leva à execução de passos na mesma direção de passos anteriores. Para tentar evitar esta situação, são escolhidas direções de busca *A*-Conjugadas $\{p_1, p_2, \ldots, p_k\}$. Duas direções $p_i$ e $p_j$ são *A*-Conjugadas se:

$$p_i^T A p_j = 0 \quad \text{para todo } i \neq j \text{ e uma matriz positiva-definida } A. \quad (4.21)$$

Desta maneira, em cada direção será dado um único passo, e esse passo terá o comprimento correto para que seja alinhado exatamente com $x^{(k)}$. Ao fim de *n* passos, em precisão infinita, a solução será encontrada.

Em geral, pode-se provar que se as direções de busca são linearmente independentes e $x^{(k)}$ é solução do problema

$$x^{(k)} = \min_{x \in x^{(0)} + span\{p_1, p_2, \cdots, p_k\}} F(x) \quad \text{para } k = 1,2,3,\cdots \quad (4.22)$$

Então, a convergência é garantida em no máximo *n* passos, isto é se $x^{(n)}$ minimiza $F(x)$ sobre $\mathbb{R}^n$ então $x^{(n)}$ satisfaz $Ax^{(n)} = b$. Porém, para que esta abordagem seja viável, as direções de busca devem fornecer condições para se calcular facilmente o vetor $x^{(n)}$, sendo conhecido a priori o vetor $x^{(n-1)}$.

Uma maneira de se determinar adequadamente as direções de busca $p_k$ será apresentada a seguir. Se:

$$x^{(k)} = x^{(0)} + P_{k-1}\, y + \alpha_k\, p_k \tag{4.23}$$

Sendo $P_{k-1} = [p_1, p_2, \ldots, p_{k-1}]$, y $\in \mathbb{R}^{k-1}$ e $\alpha_k \in \mathbb{R}$, então pelo desenvolvimento de Taylor:

$$F\left(x^{(k)}\right) = F(x^{(0)} + P_{k-1}\gamma) + \alpha_k y^{T} P_{k-1}^{T} A p_k + \frac{\alpha^2}{2} p_k^T A p_k - \alpha_k p_k^T r_0 \,. \tag{4.24}$$

Tem-se que o termo $\alpha_k y^T\ P_{k-1}^T A p_k$ é nulo se $p_k \in \text{span}\{Ap_1, Ap_2, \ldots, Ap_{k-1}\}^{\perp}$, sendo assim, para minimizar o funcional dado em (4.22), basta dividir o problema em dois outros problemas de minimizações desacopladas, um para minimizar $y$ e outro que minimiza $\alpha_k$, da seguinte maneira:

$$\begin{aligned} \min_{x \in x^{(0)} + span\{p_1, p_2, \cdots, p_k\}} F(x) &= \min_{y,\alpha} \left\{ F\left(x^{(0)} + P_{k-1} y\right) + \frac{\alpha^2}{2} p_k^T A p_k - \alpha_k p_k^T r_0 \right\} = \\ &= \min_{y} \left\{ F\left(x^{(0)} + P_{k-1} y\right) \right\} + \min_{\alpha} \left\{ \frac{\alpha^2}{2} p_k^T A p_k - \alpha_k p_k^T r_0 \right\} \end{aligned} \tag{4.25}$$

Sendo $y_{k-1}$ a solução do primeiro problema de minimização, tem-se que $x^{(k-1)} = x^{(0)} + P_{k-1}^T y_{k-1}$ minimiza $F(x)$ sobre $x^{(0)}$ + span$\{p_1, p_2, \ldots, p_k\}$. A solução do segundo problema de minimização é dada por:

$$\alpha_k = \frac{p_{k-1}^T r_0}{p_k^T A p_k} \,. \tag{4.26}$$

Observe que, devido ao fato dos vetores serem $A$-conjugados, tem-se:

$$p_k^T r_{k-1} \;=\; p_k^T \left(b - A x^{(k-1)}\right) \;=\; p_k^T \left[b - A\left(x^{(0)} + P_{k-1}^T y_{k-1}\right)\right] \;=\; p_k^T r_0 \tag{4.27}$$

Pelos resultados obtidos segue que $x^{(k)} = x^{(k-1)} + \alpha_k p_k$, e obtemos o algoritmo do método com direções de busca *A*-conjugadas como descrito na Fig. 4.3:

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\|r_k\| > \varepsilon$
    $k = k+1$
    Escolha $p_k \in \text{span}\{Ap_1, Ap_2, \ldots, Ap_{k-1}\}^{\perp}$, tal que $p_k^T r_{k-1} \neq 0$
    $\alpha_k = \dfrac{p_k^T r_{k-1}}{p_k^T A p_k}$
    $x^k = x^{k-1} + \alpha_k p_k$
    $r_k = b - Ax^k$
fim

Figura 4.3 - Algoritmo do método com direções de busca *A*-conjugadas

## 4.2.2 Algumas versões para o Método dos Gradientes Conjugados

A seguir, mostra-se que, através do Lema 4.1, pode-se encontrar direções de busca que minimizam o funcional *F*.

**Lema 4.1** – *Dado* $r_{k-1} \neq 0$, *existe* $p_k \in \text{span}\{Ap_1, Ap_2, \ldots, Ap_{k-1}\}^{\perp}$ *tal que* $p_k^T r_{k-1} \neq 0$.

**Demonstração**: Para o caso $k = 1$, escolha $p_1 = r_0$, como na descida máxima. Para o caso de $k > 1$, sendo que $r_{k-1} \neq 0$, conclui-se que:

$$\begin{aligned} A^{-1}b \notin x^0 + span\{p_1, p_2, \cdots, p_{k-1}\} &\Rightarrow b \notin Ax^0 + span\{Ap_1, Ap_2, \cdots, Ap_{k-1}\} \\ &\Rightarrow r_0 \notin span\{Ap_1, Ap_2, \cdots, Ap_{k-1}\} \end{aligned} \tag{4.28}$$

Portanto, existe um $p \in \text{span}\{Ap_1, Ap_2, \ldots, Ap_{k-1}\}^{\perp}$ com a proriedade de que $p^T r_0 \neq 0$. Contudo, $x^{(k-1)} \in x^{(0)} + \text{span}\{p_1, p_2, \ldots, p_{k-1}\}$ e aí $r_{k-1} \in r_0 + \text{span}\{Ap_1, Ap_2, \ldots, Ap_{k-1}\}$. Donde $p_k^T r_{k-1} = p^T r_0 \neq 0$. □

Note que se $P_k = [p_1, p_2, \ldots, p_k]$ é a matriz dos vetores *A*-conjugados, então, devido ao fato de que a matriz *A* é positiva-definida e as direções de busca são não nulas, tem-se que a matriz diagonal:

$$P_k^T AP_k = \text{diag}(\, p_1^T Ap_1,\ p_2^T Ap_2, \ldots,\ p_k^T Ap_k\,) \tag{4.29}$$

é não singular. E isto segue que $P_k$ possui *posto* completo, *n*, garantindo que o Algoritmo da Fig. 4.3 irá convergir em no máximo *n* passos, porque o mínimo, se existir, de $F(x)$ sobre o espaço $\text{ran}(P_n) = \mathbb{R}^n$ será $x^{(n)}$.

Para aproveitar os aspectos positivos do método da descida máxima com o método das direções de busca *A*-conjugadas, deve-se escolher uma direção de busca $p_k$ o mais próximo possível do resíduo $r_{k-1}$, em outras palavras, deve-se escolher um vetor que seja *A*-conjugado de $p_1, p_2, \ldots, p_k$. Com esta escolha define-se a primeira versão do algoritmo do Método dos Gradientes Conjugados, apresentado pela Fig. 4.4.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\| r_k\| > \varepsilon$
  $k = k+1$
  se $k = 1$
    $p_1 = r_0$
  se não
    Tome $p_k$ como o mínimo de $||p - r_{k-1}||_2$ para todos os vetores $p_k \in \text{span}\{ Ap_1, Ap_2, \ldots, Ap_{k-1}\}^{\perp}$
  fim
  $\alpha_k = \dfrac{p_k^T r_{k-1}}{p_k^T Ap_k}$
  $x^k = x^{k-1} + \alpha_k p_k$
  $r_k = b - Ax^k$
fim

Figura 4.4 – Primeira versão do algoritmo do Método dos Gradientes Conjugados.

Para tornar este algoritmo um código eficaz para sistemas lineares esparsos $Ax = b$, precisa-se de um método eficiente para calcular o vetor $p_k$, dado no algoritmo da Fig. 4.4. Pois uma quantidade considerável de análises é necessária para desenvolver a recursão final. O primeiro passo é mostrar que $p_k$ é o mínimo residual de um problema de mínimos quadrados. O Lema 4.2 mostra como calcular o vetor $p_k$.

**Lema 4.2** - *Os vetores $p_k$ tomados no algoritmo da Fig. 4.4, com $k \geq 2$, satisfazem a seguinte equação*

$$p_k = r_{k-1} - A\,P_{k-1}\,z_{k-1}, \tag{4.30}$$

*onde* $P_{k-1} = [p_1, p_2, \ldots, p_{k-1}]$ *e* $z_{k-1}$ *é o minimo da seguinte função* $f(z) = \min_{z\in\mathbb{R}^{k-1}} \left\| r_{k-1} - AP_{k-1}z \right\|_2$.

**Demonstração:** Suponha que $z_{k-1}$ é o mínimo do problema de mínimos quadrados definido por $f(z)$ e seja $p$ o mínimo residual associado, definido por:

$$p = r_{k-1} - A\,P_{k-1}\,z_{k-1} \tag{4.31}$$

Desta forma, tem-se que $p$ é $A$-conjugado aos vetores de $P_{k-1}$, ou seja, $p^T AP_{k-1} = 0$. Além disso, tem-se que $p = [I - (AP_{k-1})(AP_{k-1})^{+}]r_{k-1}$, em que $(AP_{k-1})^{+}$ é a pseudo-inversa de $(AP_{k-1})$, é a projeção ortogonal de $r_{k-1}$ sobre o espaço $\text{ran}(AP_{k-1})^{\perp}$, portanto, $p$ é o vetor mais próximo do resíduo $r_{k-1}$, no espaço $\text{ran}(AP_{k-1})^{\perp}$. Sendo assim, basta tomar $p = p_k$. □

A partir dos resultados obtidos nos lemas anteriores, pode-se estabelecer importantes relações entre os resíduos $r_k$, as direções de busca $p_k$ e o sub-espaço de Krylov:

$$K(r_0, A, k) \equiv \text{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\} \tag{4.32}$$

**Teorema 4.1 -** *Após k passos no algoritmo do método gradiente conjugado* (*com* $r_k \neq 0$ *em cada etapa*), *tem-se:*

$$r_k = r_{k-1} - \alpha_k A\,p_k\,, \tag{4.33a}$$

$$P_k^T\, r_k = 0, \tag{4.33b}$$

$$span\{p_1, p_2, \ldots, p_k\} = span\{r_0, r_1, \ldots, r_{k-1}\} = K(r_0, A, k) \tag{4.33c}$$

**Demonstração:** Da relação de recorrência $x^{(k)} = x^{(k-1)} + \alpha_k p_k$ e da definição do resíduo $r_k = b - Ax^{(k)}$, segue que $r_k = b - A(x^{(k-1)} + \alpha_k p_k) \rightarrow r_k = [b - A\,x^{(k-1)}] - A\alpha_k p_k \rightarrow r_k = r_{k-1} - \alpha_k A p_k$, o que prova a Eq. (4.33a).

Para demonstrar que $P_k^T\, r_k = 0$, Eq. (4.33b), basta provar que todo e qualquer vetor de $P_k^T$ é ortogonal a $r_k$, isto é, deve-se mostrar que $p_k^T\, r_k = 0$ para qualquer $p_k \in P_k^T$. Da relação de recorrência para o residuo $r_k = r_{k-1} - \alpha_k A\,p_k$, tem-se:

$$p_k^T r_k = p_k^T (r_{k-1} - \alpha_k A p_k) = p_k^T r_{k-1} - \alpha_k p_k^T A p_k$$
$$\Rightarrow p_k^T r_k = p_k^T r_{k-1} - \frac{p_k^T r_{k-1}}{p_k^T A p_k} p_k^T A p_k = 0 \tag{4.34}$$

Para demonstrar que *span*{$p_1$, $p_2$, ..., $p_k$} = *span*{$r_0$, $r_1$, ..., $r_{k-1}$}, Eq. (4.33c), será usado indução sobre *k*. O teorema é trivial se $k = 1$, uma vez que $p_1 = r_0$ . Agora, suponha que os subespaços são iguais para todo $k = k$, e então prova-se que o teorema também é verdadeiro para $k = k+1$. O primeiro passo é mostrar que

$$span\{r_0, r_1, \cdots, r_k\} \subseteq K(r_0, A, k+1) \tag{4.35}$$

À luz da hipótese de indução, é suficiente mostra que $r_k \in K(r_0, A, k+1)$. Recordando a relação de recorrência dada pela Eq. (4.33a), $r_k = r_{k-1} - \alpha_k A p_k$, pode-se obter a condição de $Ap_k$. Usando o Lema-4.2, tem-se

$$p_{k-1} = r_{k-2} - [Ap_1, Ap_2, \cdots, Ap_k] z_{k-2} \in span\{r_0, r_1, \cdots, r_{k-1}\}. = K(r_0, A, k) \tag{4.36}$$

Segue disto que

$$Ap_{k-1} \in span\{Ar_0, A^2 r_0, \cdots, A^k r_0\}. \subseteq K(r_0, A, k+1) \tag{4.37}$$

Além disso, $r_{k-1} \in K(r_0, A, k). \subseteq K(r_0, A, k+1)$, assim,

$$r_k = r_{k-1} - \alpha_k A p_k \in K(r_0, A, k+1) \tag{4.38}$$

Isto prova (4.35).

O próximo passo é mostrar que

$$span\{p_1, p_2 \cdots, p_k\} \subseteq span\{r_0, r_1, \cdots, r_{k-1}\} \tag{4.39}$$

Por hipótese de indução,

$$p_i \in span\{r_0, r_1, \cdots, r_{k-1}\} \text{ para } i = 0, ..., k-1 \tag{4.40}$$

Assim é suficiente mostrar que $p_k \in span\{r_0, r_1, ..., r_k\}$. Mas isto segue imediatamente do Lema 4.2 seguido do Lema 4.1.

Combinando as equações (4.38) e (4.39), vê-se que os três subespaços de interesse são aninhados, ou seja, $span\{p_1, p_2, ..., p_k\} \subseteq span\{r_0, r_1, ..., r_{k-1}\} \subseteq K(r_0, A, k+1)$. Para mostrar que eles são iguais, basta mostrar que todos têm a mesma dimensão. Uma vez que $K(r_0, A, k+1)$ é gerado por um conjunto de $k + 1$ vetores, sua dimensão é no máximo $k + 1$. Mostrando que a dimensão do $span\{p_1, p_2, ..., p_k\}$ é exatamente $k + 1$, prova-se que os três espaços têm dimensão $k$ +1, e são, portanto, iguais. Mas, como se sabe que os $p_1, p_2, ..., p_k$ são todos diferente de zero e ortogonais e portanto, são linearmente independentes, formando uma base de $span\{p_1, p_2, ..., p_k\}$ cuja dimensão é $k + 1$. □

**Corolário 4.1**: *Os resíduos $r_0, r_1, ..., r_k$, produzidos pelo algoritmo do método dos gradientes conjugados são ortogonais com respeito ao produto interno padrão.*

**Demonstração:** Deve mostrar que $r_k \perp r_0, r_1, ..., r_{k-1}$ , para todo $k$. Mas isto segue imediatamente do fato de que $P_k^T r_k = 0$, Eq. (4.33b) e uma vez que $span\{p_1, p_2, ..., p_k\} = span\{r_0, r_1, ..., r_{k-1}\}$ conforme Eq. (4.33c). □

**Corolário 4.2 -** *As direções de busca $p_{k-1}$ e os resíduos $r_{k-1}$ geram um subspaço que contém a direção de busca $p_k$ para todo $k \geq 2$, ou seja, $p_k \in \text{span}\{p_{k-1}, r_{k-1}\}$ para $k \geq 2$.*

**Demonstração:** Se $k = 2$, então $p_2 \in span\{r_0, r_1\}$ pois da Eq. (4.33c) tem-se que $span\{p_1, p_2\} = span\{r_0, r_1\}$. Ora, como $p_1 = r_0$ conclui-se que $p_2$ é combinação linear de $p_1$ e $r_1$.

Para $k > 2$, basta que se decomponha o vetor $z_{k-1}$ do Lema 4.2 da seguinte forma:

$$z_{k-1} = \begin{bmatrix} \omega \\ \mu \end{bmatrix}, \quad \omega \in \mathbb{R}^{k-2}, \quad \mu \in \mathbb{R} \tag{4.41}$$

Da fórmula de atualização do resíduo, $r_{k-1} = r_{k-2} - \alpha_{k-1} A p_{k-1}$, segue -se que

$$Ap_{k-1} = \frac{r_{k-2} - r_{k-1}}{\alpha_{k-1}} \tag{4.42}$$

Portanto,

$$\begin{aligned} p_k &= r_{k-1} - AP_{k-1} z_{k-1} = r_{k-1} - AP_{k-1} \begin{pmatrix} \omega \\ \mu \end{pmatrix} = r_{k-1} - AP_{k-2}\omega - \mu A p_{k-1} = \\ & r_{k-1} - AP_{k-2}\omega - \mu A \left( \frac{r_{k-2} - r_{k-1}}{\alpha_{k-1}} \right) = \left( 1 + \frac{\mu}{\alpha_{k-1}} \right) r_{k-1} - \left( \frac{\mu}{\alpha_{k-1}} r_{k-2} + AP_{k-2}\omega \right) \end{aligned} \tag{4.43}$$

Fazendo

$$\begin{aligned} s_{k-1} &= -\frac{\mu}{\alpha_{k-1}} r_{k-2} - AP_{k-2}\omega \ \in span\left\{ r_{k-2}, AP_{k-2}\omega \right\} \\ &\subseteq span\left\{ r_{k-2}, Ap_1, \cdots, Ap_{k-2} \right\} \subseteq span\left\{ r_1, \cdots, r_{k-2} \right\} \end{aligned} \tag{4.44}$$

Pelo Corolario 4.1, $r_0$, $r_1, \ldots,$ $r_k$, são mutuamente ortogonais, segue que $s_{k-1} r_{k-1} = 0$. Desta forma, resolver o problema de mínimos quadrados no algoritmo da Fig. 4.4 se reduz a escolher $\omega$ e $\mu$ de tal que a norma de $p_k$ dada por:

$$\left\| p_k \right\|_2^2 = \left( 1 + \frac{\mu}{\alpha_{k-1}} \right)^2 \left\| r_{k-1} \right\|_2^2 + \left\| s_{k-1} \right\|_2^2 \tag{4.45}$$

seja mínima. Uma vez que $z_{k-2}$ minimiza a norma $L_2$ de $(r_{k-2} - AP_{k-2}z)$ fornecendo um resíduo $p_{k-1}$, segue que $s_{k-1}$ é múltiplo de $p_{k-1}$. Consequentemente, $p_k \in span\{p_{k-1}, r_{k-1}\}$. □

Com os resultados apresentados pelo Teorema 4.5 e os Corolários 4.1 e 4.2, mostra-se que a direção de busca $p_k$ é uma simples combinação linear de seu antecessor $p_{k-1}$ e do resíduo $r_{k-1}$.

Graças ao Corolário 4.2 a fórmula de atualização do resíduo $p_k$ se torna:

$$p_k = r_{k-1} + \beta_k p_{k-1} \tag{4.46}$$

A definição de que os vetores $p_k$ e $p_{k-1}$ são *A*-conjugados, ou seja, $p_{k-1}^T A p_k = 0$, é satisfeita se, e somente se:

$$\beta_k = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}} \tag{4.47}$$

Isso mostra que $p_k$ pode ser calculado com um custo computacional baixo e que este custo não cresce com *k*. Verifica-se também, que não há necessidade de salvar as antigas direções de busca $p_1$, $p_2$,..., $p_{k-1}$, portanto, o requisito de armazenamento não cresce com *k*, pois toda a "história iterativa" necessária é concentrada em $p_k$.

Isto leva à uma segunda versão do Método dos Gradientes Conjugados, apresentada na Fig. 4.5.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\| r_k \| > \varepsilon$
  $k = k+1$
  se $k = 1$
    $p_1 = r_0$
  se não
    $\beta_k = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}}$
    $p_k = r_{k-1} + \beta_k p_{k-1}$
  fim
  $\alpha_k = \frac{p_k^T r_{k-1}}{p_k^T A p_k}$
  $x^k = x^{k-1} + \alpha_k p_k$
  $r_k = b - Ax^k$
fim

Figura 4.5 - Segunda versão do algoritmo do Método dos Gradientes Conjugados.

Na implementação do algoritmo da Fig. 4.5, nota-se que o método exige três multiplicações do tipo matriz-vetor separadas em cada etapa. Entretanto, isto pode ser simplificado através da relação recursiva dos resíduos, $r_k = r_{k-1} - \alpha_{k-1} A p_k$ , lembrando-se que os resíduos são mutuamente ortogonais e multiplicando $r_{k-1} = r_{k-2} - \alpha_{k-1} A p_{k-1}$ por $r_{k-1}^T$, tem-se:

$$r_{k-1}^T r_{k-1} = r_{k-1}^T r_{k-2} - \alpha_{k-1} r_{k-1}^T A\, p_{k-1} \Rightarrow r_{k-1}^T r_{k-1} = -\alpha_{k-1} r_{k-1}^T A\, p_{k-1} \Rightarrow$$

$$\left(r_{k-1}^T r_{k-1}\right)^T = \left(-\alpha_{k-1} r_{k-1}^T A\, p_{k-1}\right)^T = -\alpha_{k-1} p_{k-1}^T A\, r_{k-1} \;\Rightarrow\; p_{k-1}^T A\, r_{k-1} = -\frac{r_{k-1}^T r_{k-1}}{\alpha_{k-1}} \tag{4.48}$$

Agora multiplicando $r_{k-1} = r_{k-2} - \alpha_{k-1} A\, p_{k-1}$ por $p_{k-1}^T$:

$$p_{r-1}^T r_{k-1} = p_{r-1}^T r_{k-2} - \alpha_{k-1} p_{r-1}^T A\, p_{k-1} \;\Rightarrow p_{k-1}^T r_{k-2} = \alpha_{k-1} p_{k-1}^T A\, p_{k-1} \Rightarrow$$

$$p_{k-1}^T A\, p_{k-1} = \frac{p_{k-1}^T r_{k-2}}{\alpha_{k-1}} \Rightarrow p_{k-1}^T A\, p_{k-1} = \frac{r_{k-2}^T r_{k-2}}{\alpha_{k-1}} \tag{4.49}$$

Substituindo as Eq. (4.48) e (4.49) na Eq. (4.47) tem-se calculados os valores de $\beta_k$:

$$\beta_k = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}} = \frac{r_{k-1}^T r_{k-1}}{r_{k-2}^T r_{k-2}} \tag{4.50}$$

Desta forma o algoritmo dos Gradientes Conjugados pode ser rescrito em sua terceira versão, como apresentado na Fig. 4.6.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\| r_k \| > \varepsilon$
  $k = k+1$
  se $k = 1$
    $p_1 = r_0$
  se não
    $\beta_k = \frac{r_{k-1}^T r_{k-1}}{r_{k-2}^T r_{k-2}}$
    $p_k = r_{k-1} + \beta_k p_{k-1}$
  fim
  $\alpha_k = \frac{r_{k-1}^T r_{k-1}}{p_k^T A p_k}$
  $x^k = x^{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k A p_k$
fim

Figura 4.6 – Terceira versão do Algoritmo do Método dos Gradientes Conjugados.

Este algoritmo pode ser ainda mais refinado, fazendo $w_k = Ap_k$, o que reduz o custo computacional, pois elimina o produto matriz-vetor $Ap_k$ que aparece em dois locais. Também,

para que o algoritmo fique mais elegante, define-se quadrado da norma do resíduo da seguinte maneira:

$$\rho_k = \|r_k\|_2^2 = r_k^T r_k \tag{4.51}$$

Com esta nova terminologia, o algoritmo requer apenas uma multiplicação matriz-vetor por iteração. Observe que apenas quatro vetores *n*-dimensionais de armazenamento são essenciais: *x, r, p* e *w*. Desta forma, o algoritmo para a implementação do Método dos Gradientes Conjugados é dado na Fig. 4.7.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\| r_k\| > \varepsilon$
  $k = k+1$
  se $k = 1$
    $p_1 = r_0$
  se não
    $\beta_k = -\frac{\rho_{k-1}}{\rho_{k-2}}$
    $p_k = r_{k-1} + \beta_k p_{k-1}$
  fim
  $w_k = Ap_k$
  $\alpha_k = \frac{\rho_{k-1}}{p_k^T w_k}$
  $x^k = x^{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k w$
  $\rho_k = r_k^T r_k$
Fim

Figura 4.7 - Algoritmo do Método dos Gradientes Conjugados (CG).

A metodologia apresentada pela formulação do método dos gradientes conjugados, no algoritmo da Fig. 4.7, é semelhante a que Hestenes e Stiefel apresentaram em 1952, é muito adequada, quando se quer solucinar grandes sistemas lineares esparsos. Este algoritmo procura sempre evitar cálculos com a matriz de coeficientes, o que evita o fenômeno do enchimento *(fill-in)* comum nos métodos diretos.

Inicialmente, o método foi concebido como um método direto uma vez que, sob uma aritmética exata, o algoritmo converge para a solução exata. No entanto, erros de arredondamento levam a uma perda de ortogonalidade entre os resíduos e a obtenção finita da

solução em *n* passos não é garantida. Além disso, quando o método dos gradientes conjugados é aplicado a sistemas lineares grandes, o número *n* de iterações é geralmente tão grande que representa uma quantidade inaceitável de trabalho. Como consequência dessas observações, é habitual considerar o método como uma técnica genuinamente iterativa cuja finalização é baseada em um número máximo de iteração e na norma residual.

### 4.2.3 Estudo da Convergência do Método dos Gradientes Conjugados

O método dos gradientes conjugados passou a ser considerado como um método iterativo a partir da publicação do trabalho realizado por REID (1971). Tata-se de um algorítmo muito útil, no entanto, para que esta técnica tenha sucesso, torna-se necessário conhecer sua taxa de convergência. Nesta seção, serão consideradas as propriedades de convergência do algoritmo CG através do estudo da convergência da sequência de aproximações, $\{x^{(0)}, x^{(1)}, \ldots, x^{(k)}\}$, gerada pelo processo iterativo deste método. Os resultados aqui apresentados mostram que o método funciona bem quando *A* está próxima da identidade, quer no sentido de uma perturbação com *posto* baixo ou no sentido da norma.

**Teorema 4.2 –** *O algoritmo do método dos gradientes conjugados, aplicado a um sistema linear Ax =b de ordem n, positivo-definido, converge para a solução exata em no máximo n iterações.*

**Demonstração:** Se ao efetuar *n-1* iterações a sequência não convergir para a solução *x*, os resíduos não nulos $r_0, r_1, \ldots, r_{(n-1)}$ formam uma base ortogonal de $\mathbb{R}^n$, pelo corolário 4.1. Após as etapas *n*, $r_n$ é ortogonal a $r_i$ para $i = 0, \ldots, n-1$. Uma vez que $r_n$ é ortogonal a uma base inteira de $\mathbb{R}^n$, deve se ter $r_n = 0$, daí $x_n = x$. □

O Teorema 4.2 também pode ser enunciado da seguinte forma: *Se A = I + B com A, B,* $I \in \mathbb{R}^{n \times n}$*, é uma matriz simétrica e positiva-definida e o posto*(*B*) *= r, então o Algoritmo da Fig. 4.7 converge em no máximo* (*r+1*) *iterações.*

O Teorema 4.2 vale para qualquer método que se utiliza de direções de buscas conjugadas. Em cada iteração a dimensão do espaço em que a procura está ocorrendo é reduzida por um, porque cada nova direção da busca é conjugada a todos as anteriores. Este Teorema apresenta uma propriedade interessante, mas de pouca importância prática. Quando

a dimensão *n* da matriz de coeficientes é muito grande, o Teorema (ignorando erros de arredondamento) garante a obtenção da solução exata com *n* ou menos iterações. O que não representa uma grande vantagem, pois dependendo do valor de *n*, o tempo computacional será muito alto. Na verdade, é desejável chegar à solução com um número menor de iterações, e existem exemplos que mostram que se pode obter isto.

Ignorando a propriedade de número finito de passos e continuando a ver o método dos gradientes conjugados como um método iterativo, deseja-se saber a rapidez com que a sequência se aproxima da solução *x*. O teorema, a seguir, fornece algumas informações nesse sentido, sua demonstração se encontra em WATKINS (2002, p.594) e também em LUENBERGER (1973, p.187).

**Teorema 4.3 –** *Seja* $A \in \mathbb{R}^{n\times n}$ *uma matriz simétrica e positiva-definida. O algoritmo do método dos gradientes conjugados produz uma sequência* $\{x^{(k)}\}_{k\in\mathbb{N}}$, *satisfazendo:*

$$\left\|x - x^{(k)}\right\|_A \leq 2\left\|x - x^{(0)}\right\|_A \left[\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right]^k \tag{4.52}$$

*onde* $\kappa(A)$ *é fator-condição de A.*

Desta forma, verifica-se que o método dos gradientes conjugados converge rapidamente se o fator-condição tende a um

$$\left(\kappa(A) = \frac{\lambda_{máx}}{\lambda_{\min}} \to 1\right) \tag{4.53}$$

e converge lentamente se o fator-condição tende ao infinito

$$\left(\kappa(A) = \frac{\lambda_{máx}}{\lambda_{\min}} \to \infty\right). \tag{4.54}$$

Todos os resultados aqui apresentados são validos para o algoritmo dos gradientes conjugados sem pré-condicionamento. Resultados semelhantes para o método pré-condicionado são obtidos apenas substituindo $A$ pela matriz transformada $\hat{A}=C^{-1}AC^{-T}$.

## 4.3 Método dos Gradientes Conjugados Pré-Condicionado (PCG)

Pelo Teorema 4.3, o método dos Gradientes Conjugados converge mais rápido quanto mais próximo da unidade for o fator-condição. No entanto, matrizes positiva-definidas e simétricas podem ser mal condicionadas, ou seja, nem sempre possuem o fator-condição com valores tendendo a um. Nestes casos, o pré-condicionamento é necessário para garantir a rápida convergência do método dos gradientes conjugados.

A idéia geral do Método dos Gradientes Conjugados Pré-condicionado, que será apresentado nesta seção, é *pré-condicionar* a *matriz dos coeficientes* $A$ de forma que seu *fator-condição* se aproxime da unidade. Assim, se o fator-condição de $A$ é grande, pode-se tentar reescrever o sistema $Ax = b$ na forma:

$$M^{-1}Ax = M^{-1}b \tag{4.55}$$

de modo que os autovalores de $M^{-1}A$ estejam mais próximos uns dos outros que os de $A$. Assim, pode-se obter convergência mais rápido.

O problema, agora, se resume em encontrar de forma correta a matriz pré-condicionadora $M$. Se $M= A$, então $M^{-1}A = I$, e teria convergência instantânea. Porém, naturalmente, o trabalho gasto para se obter $A^{-1}$ é maior que o necessário para resolver $Ax = b$. Daí deve-se obter $M$ de outra forma. Acontece que, mesmo que as matrizes $A$ e $M$ sejam simétricas e positiva-definidas o produto $M^{-1}A$ pode não ser. Para contornar este problema escreve-se

$$M = CC^{T} \tag{4.56}$$

Desta forma os autovalores de $M^{-1}A$ e de $C^{-1}AC^{-T}$ são iguais, o mesmo ocorrendo com o fator-condição.

Agora, fazendo a seguinte mudança de variável:

$$\hat{x} = C^T x \Leftrightarrow x = C^{-T}\hat{x} \tag{4.57}$$

e trocando

$$\begin{aligned} \hat{A} &= C^{-1}AC^{-T} \Leftrightarrow A = C\hat{A}C^T \\ \hat{b} &= C^{-1}b \Leftrightarrow b = C\hat{b} \end{aligned} \tag{4.58}$$

O sistema $Ax = b$ é transformado em:

$$\hat{A}\hat{x} = \hat{b} \tag{4.59}$$

A idéia mais simples é utilizar o algoritmo do Método dos Gradientes Conjugados dado pela Fig. 4.7, na solução do sistema pré-condicionado $\hat{A}\hat{x} = \hat{b}$ e a seguir usar a relação $x = C^{-T}\hat{x}$ da Eq. (4.57) para obter a solução do sistema original.

Uma maneira de evitar uma referência explicita à matriz $C^{-1}$ e simplificar o algoritmo é usar somente a matriz pré-condicionadora $M$ e não $C$. Além disso, deve-se definir o sistema linear auxiliar:

$$Mz_k = r_k \Leftrightarrow z_k = M^{-1}r_k \tag{4.60}$$

Desta forma, o Método dos Gradientes Conjugados Pré-condicionado é dado pelo algoritmo mostrado na Fig. 4.8.

Para que o algoritmo da Fig. 4.8 seja eficiente na solução de grandes sistemas lineares esparsos, o sistema linear $Mz = r$ deve ser facilmente resolvido. A escolha adequada da matriz pré-condicionadora $M$ é fundamental para se obter o método com boa taxa de convergência. A dificuldade reside no fato de que para cada problema físico deve-se procurar a matriz pré-condicionadora mais adequada. Uma alternativa que pode ser adotada é definir esta matriz como o fator triangular inferior resultante da decomposição de Cholesky.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $|| r_k|| > \varepsilon$
resolva $Mz_k = r_k \rightarrow z_k$
$k = k+1$
se $k = 1$
$p_1 = z_0$
se não
$\beta_k = \frac{r_{k-1}^T z_{k-1}}{r_{k-2}^T z_{k-2}}$
$p_k = z_{k-1} + \beta_k p_{k-1}$
fim
$w_k = Ap_k$
$\alpha_k = \frac{r_{k-1}^T z_{k-1}}{p_k^T w_k}$
$x_k = x^{k-1} + \alpha_k p_k$
$r_k = r_{k-1} - \alpha_k w_k$
fim

Figura 4.8 - Algoritmo do Método dos Gradientes Conjugados Pré-condicionado (PCG).

**Exemplo Ilustrativo:** Seja o sistema $Ax = b$, sendo:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 3 \end{pmatrix} \quad e \quad b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Aplicando o algoritmo da Fig. 4.8 considerando a matriz A obtida pelo método de Cholesky, tem-se:

Condição inicial:

$$x_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad r_0 = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, \; L = chol(A) = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}, \; M = L * L^T = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \; z_0 = \begin{bmatrix} -1 \\ 0 \\ 4 \end{bmatrix}$$

k=1, primeira iteração

$$z_1 = \begin{bmatrix} 0{,}0732 \\ -0{,}000 \\ -0{,}024 \end{bmatrix}, \; p_1 = \begin{bmatrix} -1 \\ 0 \\ 4 \end{bmatrix}, \; \alpha_1 = 0{,}2683, \quad x_1 = \begin{bmatrix} -0{,}2683 \\ 0 \\ 1{,}0732 \end{bmatrix}, \; r_1 = \begin{bmatrix} 0{,}1951 \\ 0{,}1220 \\ 0{,}0488 \end{bmatrix}$$

k = 2 , segunda iteração:

$$z_2 = 10^{-14}\begin{bmatrix} 0{,}0056 \\ -0{,}0520 \\ -0{,}1457 \end{bmatrix}, \quad p_2 = \begin{bmatrix} 0{,}07200 \\ -0{,}0000 \\ -0{,}0196 \end{bmatrix}, \quad \beta_2 = 0{,}0012, \quad \alpha_2 = 3{,}7273,$$

$$x_2 = \begin{bmatrix} 0{,}0000 \\ -0{,}000 \\ 1{,}0000 \end{bmatrix}, \quad r_2 = 10^{-15}\begin{bmatrix} -0{,}2498 \\ -0{,}3053 \\ -0{,}8812 \end{bmatrix}$$

k = 3 , terceira iteração:

$$z_3 = 10^{-15}\begin{bmatrix} -0{,}2201 \\ 0{,}3673 \\ -0{,}0649 \end{bmatrix}, \quad p_3 = 10^{-14}\begin{bmatrix} 0{,}0056 \\ 0{,}0520 \\ -0{,}1457 \end{bmatrix}, \quad \beta_3 = 8{,}4918 * 10^{-12}, \quad \alpha_3 = 0{,}2943,$$

$$x_3 = \begin{bmatrix} 0{,}0000 \\ -0{,}000 \\ 1{,}0000 \end{bmatrix}, \quad r_3 = 10^{-15}\begin{bmatrix} 0{,}0095 \\ 0{,}2296 \\ 0{,}0823 \end{bmatrix}$$

Segue que a solução do sistema é: $x = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$, com resíduo $r = 10^{-15}\begin{bmatrix} 0{,}0095 \\ 0{,}2296 \\ 0{,}0823 \end{bmatrix}$ cuja norma é $2{,}4407 * 10^{-16}$, após serem feitas 4 iterações. Todos os valores de $z_k$, $p_k$, $b_k$, $\alpha_k$, $x_k$ *e* $r_k$, são obtidos pelo algoritmo da Fig. 4.8.

## 4.4 Método dos Resíduos Mínimos (MINRES)

O algoritmo dos gradientes conjugados faz parte de uma ampla classe de algoritmos conhecidos como métodos do subespaço de Krylov, porque as sequências de aproximações geram o subespaço de Krylov. O método dos gradientes conjugados, apresentado ao longo das seções anteriores, é aplicável aos sistemas simétricos positiva-definidos. Para outros tipos de sistemas, outros métodos do subespaço de Krylov vêm sendo desenvolvidos. Nestas próximas seções será dada uma breve revisão de alguns dos métodos mais populares.

A eventual falha do método dos gradientes conjugados em problemas envolvendo matrizes simétricas não positiva-definidas leva a necessidade de um método numericamente estável, baseado nos vetores de Lanczos. Vários métodos são possíveis, mas nesta seção sugere-se um método que é numericamente satisfatório, trata-se de uma variante do método dos gradientes conjugados, denominado MINRES. Este método foi desenvolvido por Chris Paige e Michael Saunders em 1975, sendo baseado em uma fatoração ortogonal da matriz de coeficientes.

A sequência de vetores, gerada pelo método dos gradientes conjugados, corresponde a uma fatoração da matriz tri-diagonal semelhante à matriz dos coeficientes. O método MINRES evita a fatoração *LU*. Quando *A* é não positiva-definida, mas simétrica, ainda pode-se construir uma base ortogonal para o subespaço de Krylov com três termos da relação de recorrência. Introduzindo a seguinte fatoração da matriz *A*,

$$\begin{cases} AU = UT \equiv \left(u_1, u_2, \cdots, u_n\right) \begin{pmatrix} \gamma_1 & \beta_1 & \cdots & 0 & 0 \\ \beta_1 & \gamma_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \gamma_{n-1} & \beta_{n-1} \\ 0 & 0 & \cdots & \beta_{n-1} & \gamma_n \end{pmatrix} \\ U^T U = I \equiv \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{cases} \tag{4.61}$$

Sendo *U* ortogonal tem-se que:

$$U^T A U = U^T U T = T \tag{4.62}$$

Utilizando esta equação, pode-se escrever a seguinte relação de recorrência:

$$\begin{aligned} & u_k^T A u_k = \gamma_k \\ & u_k^T A u_{k+1} = \beta_k \\ & A u_k = \beta_k u_{k+1} + \gamma_k u_k + \beta_{k-1} u_{k-1} \end{aligned} \tag{4.63}$$

Considerando na fatoração apresentada em (4.61), $U = R$, como a matriz dos vetores resíduos, tem-se:

$$Ar_k = \beta_k r_{k+1} + \gamma_k r_k + \beta_{k-1} r_{k-1} \text{ ou } AR_k = R_{k+1}T_k \tag{4.64}$$

Onde todos os vetores $r_k$ são ortogonais. Se $\beta_k = 0$ o algoritmo termina e $R_k$ é uma base ortogonal para o subespaço de Krylov $K(r_0, A, k)$ associado. Portanto, os autovalores de $T_k$ são os autovalores de $A$. O método MINRES se inicia impondo:

$$u_1 = \frac{r_0}{\|r_0\|} \tag{4.65}$$

e escolhendo a aproximação sucessiva como:

$$x^{(k)} \in S_k = x^{(0)} + \text{span}\{u_1, u_2, \ldots, u_k\},\ x^{(k)} = R_k y\ ,\ y \in \mathbb{R}^k \tag{4.66}$$

impondo a condição de ortogonalidade ao resíduo da seguinte forma:

$$u_j^T\, r_i = 0,\ j \le i \tag{4.67}$$

Que minimiza:

$$\left\|Ax^{(k)} - b\right\|_2 = \left\|AR_k\, y - b\right\|_2 = \left\|R_{k+1}\, T_k\, y - b\right\|_2 \tag{4.68}$$

Considerando o fato que, se $D_{k+1} \equiv \text{diag}(||r_0||_2,\ ||r_1||_2,\ \ldots, ||r_k||_2)$, então $R_{k+1}D_{k+1}^{-1}$ é uma transformação ortonormal em relação ao atual subespaço de Krylov. Assim, resolver a Eq. (4.68) se reduz a resolver o problema de norma mínima:

$$\left\|Ax^{(k)} - b\right\|_2 = \left\|D_{k+1}T_k y - \|r_0\|_2\, e_1\right\|_2 \tag{4.69}$$

Outra forma para obter a solução do problema é resolver o sistema $T_k\, y = ||r_0||_2 e_1$, através do método de CG. Ou seja, a procura da solução, segundo esta estratégia, resume-se a encontrar $y_k \in \mathbb{R}^k$ tal que:

$$0 = \|r_0\| E_1^k - T_k y_k \tag{4.70}$$

Todavia a matriz $T_k$ poderá ser singular. Portanto, procura-se a nova solução impondo que a norma do resíduo correspondente seja minimizada. Tem-se, então, que:

$$x^{(k)} = x^{(0)} + U_k y_k \tag{4.71}$$

onde $y_k$ é um vetor tal que

$$y_k = \min \left\| b - A\left(x^{(0)} + U_k\, y\right) \right\| \tag{4.72}$$

Escolha vetores $x^0$ e $y^0$ como vetores iniciais e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$r_0(y) = b - Ay^0$
$k = 1$
    $u_1 = r_0/||r_0||$
enquanto $||r_k|| > \varepsilon$ e $k>kmax$
    $k = k+1$
    para $i=1, \ldots, n$
    $\gamma_k(i) = u^T_k A u_k$
    $\beta_k(i) = \sqrt{u_k^T A(Au_k - \gamma_k(i)u_k)}$
    $u_k = \beta k(i) uk_{+1} + \gamma_k u_k + \beta_{k-1}(i) u_{k-1}$
    fim
    $k = k+1$
    $y = T^{-1}||r_k|| E_1$
    $x^k = x^{k-1} + U_k y_k$
    $r_k = b - Ax^k$
fim

Figura 4.9 - Algoritmo do Método dos Resíduos Mínimos (MINRES).

Este problema de minimização quadrática pode ser resolvido com o seguinte procedimento: introduz-se uma fatoração *QR* na matriz $T_k$ considerando somente os elementos que tenham interesse, veja GLUB e van LOAN (1993).

Essa fatoração *QR* faz com que o elemento na posição ($k+1$, $k$) de $T_k$ seja eliminado por uma simples rotação de Givens e o sistema bi-diagonal superior resultante pode ser facilmente resolvido (os outros elementos da sub-diagonal podem ter sido removidos em passos anteriores da iteração). Este procedimento conduz ao método dos Resíduos Mínimos

(MINRES), (BARRET *et al*, 1994). O Algoritmo para o método MINRES é apresentado na Fig. 4.9.

**Exemplo Ilustrativo:** Seja o sistema $Ax = b$, sendo:

$$A = \begin{pmatrix} 3 & 2 & 4 \\ 2 & 1 & 3 \\ 4 & 3 & 2 \end{pmatrix} \quad \text{e} \quad b = \begin{bmatrix} 3 \\ 0 \\ 6 \end{bmatrix}$$

Considere a condição inicial:

$$x_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},\ r_0 = \begin{bmatrix} 3 \\ 0 \\ 6 \end{bmatrix}$$

Aplicando o algoritmo da MINRES, conforme Fig. 4.9, tem-se:

*k=1*, primeira iteração

para *i = 1, 2, 3*

$$u_{11} = \begin{bmatrix} 0.4472 \\ 0 \\ 0.8944 \end{bmatrix},\quad u_{12} = \begin{bmatrix} 0.5512 \\ 0.7975 \\ -0.2756 \end{bmatrix},\ u_{13} = \begin{bmatrix} -07044 \\ 0{,}6163 \\ 0{,}3522 \end{bmatrix},\ U_1 = \left[u_{11},\ u_{12},\ u_{13}\right]$$

$$U_1 = \begin{bmatrix} 0{,}4472 & 0{,}5512 & -0{,}7044 \\ 0 & 0{,}7875 & 0{,}6163 \\ 0{,}8944 & -0{,}7256 & 0{,}3522 \end{bmatrix}$$

$$\begin{cases} \gamma_{11} = 5{,}4000;\ \gamma_{12} = 0{,}9023;\ \gamma_{13} = 0{,}03023 \\ \beta_{11} = 4{,}5431;\ \beta_{12} = 0{,}5720 \end{cases};\ T_1 = \begin{bmatrix} \gamma_{11} & \beta_{11} & 0 \\ \beta_{11} & \gamma_{12} & \beta_{12} \\ 0 & \beta_{12} & \gamma_{13} \end{bmatrix}$$

$$T_1 = \begin{bmatrix} 5{,}400 & 4{,}531 & 0 \\ 4{,}5431 & 0{,}9023 & 0{,}5720 \\ 0 & 0{,}5720 & -0{,}3023 \end{bmatrix} \quad y_1 = \begin{bmatrix} -1{,}3416 \\ 3{,}0712 \\ 5{,}8110 \end{bmatrix},\ x_1 = \begin{bmatrix} -3{,}000 \\ 6{,}000 \\ 0{,}000 \end{bmatrix},\quad r_1 = 10^{-13}\begin{bmatrix} 0{,}3375 \\ 0{,}7449 \\ 0{,}1510 \end{bmatrix}$$

A solução deste sistema, usando este algoritmo, é $x = \begin{bmatrix} -3 \\ 6 \\ 0 \end{bmatrix}$, com resíduo $r = 10^{-13}\begin{bmatrix} 0.3375 \\ 0.7949 \\ 0.1510 \end{bmatrix}$

cuja norma é $8{,}7670*10^{-14}$, após serem feitas 2 iterações. Os valores de $u_{ki}$, $\gamma_{ki}$ e $\beta_{ki}$ são obtidos como no algoritmo da Fig. 4.9.

## 4.5 Método dos Resíduos Mínimos Generalizado (GMRES)

O algoritmo chamado MINRES formulado por Paige e Saunders utiliza os vetores de Lanczos como base para calcular uma solução aproximada $x^{(k)}$ que minimiza a norma residual sobre o subespaço de Krylov $K(A, r_0, k)$. Será apresentada nesta seção, uma generalização do algoritmo MINRES para resolver sistemas lineares não simétricos. Esta generalização é baseada no processo de Arnoldi, que é um análogo do algoritmo de Lanczos para matrizes não simétricas. O método GMRES foi desenvolvido por Yousef Saad e H. Martin Schultz, em 1986.

No lugar de trabalhar com uma matriz tridiagonal representada por $T_k$, como é produzido pelo método de Lanczos para matrizes simétricas, o método de Arnoldi produz uma matriz Hessenberg superior. Usando uma base ortornormal gerada pelo processo de Arnoldi, mostra-se que a solução aproximada, que minimiza a norma residual sobre $K(A, r_0, k)$, é facilmente calculada por uma técnica semelhante à do método MINRES.

O processo de Lanczos, o processo de Arnoldi e o processo de Gram-Schmidt modificado estão intimamente relacionados. Dado $k$ vetores linearmente independentes $u_1,...,u_k \in \mathbb{R}^n$ o processo de Gram-Schmidt modificado gera $k$ vetores ortonormais $v_1,..., v_k$ , onde cada $v_i \in$ span $(u_1,...,u_i)$. Dado $A \in \mathbb{R}^{n\times n}$ e $b \in \mathbb{R}^n$ o processo de Gram-Schmidt modificado em $(b, Ab,..., A^{k-1}b)$ é chamado o processo de Arnoldi, e quando $A$ é simétrica, é equivalente ao processo de Lanczos.

Dado $A \in \mathbb{R}^{n\times n}$ e $b \in \mathbb{R}^n$, o processo de Arnoldi calcula vetores $v_k$ como segue:

$$\begin{aligned} &\beta_1 v_1 = b, \quad \text{onde} \quad \beta_1 = \|b\|_2 \quad \text{minimiza} \quad v_1 \\ &w_k = Av_k, \; h_{i,k} = w_k^T v_i \qquad\qquad i = 1, 2, \cdots, k \\ &h_{k+1,k} v_{k+1} = w_k - h_{1,k} v_1 - \cdots - h_{k,k} v_k, \end{aligned} \tag{4.73}$$

sendo que $h_{k+1,k}$ minimiza $v_{k+1}$. Na forma matricial:

$$AV_k = V_{k+1}\underline{H_k}\text{, onde } V_k = [v_1 \cdots v_k],\ \underline{H_k} = \begin{bmatrix} H_k \\ h_{k+1,k}e_k^T \end{bmatrix},\ H_k = \left[h_{i,j}\right]_{j=1,.\,i=1}^{k,\,j+1} \tag{4.74}$$

Note que $H_k$ é uma matriz Hessenberg superior. Na aritmética exata as colunas de $V_k$ são ortonormais, e o processo é interrompido quando $h_{k+1,k} = 0$ ($k \leq n$). Donde se obtém $AV_k = V_kH_k$.

Para resolver um sistema $Ax = b$, onde $A$ é uma matriz quadrada e não simétrica, o método GMRES constrói aproximações da solução dadas por:

$$x^{(k)} = x^{(0)} + y_1v_1 + \ldots + y_kv_k \tag{4.75}$$

onde os coeficientes $y_k$ foram escolhidos de forma que a cada iteração do processo de Arnoldi, GMRES resolve um subproblema de mínimos quadrados:

$$y_k = \min_{y \in R^k} \left\| H_k y - \beta_1 v_1 \right\| \tag{4.76}$$

Todos os vetores $v_1,\ldots, v_k$ são salvos, e somente no final é que $y_k$ e $x_k$ precisam ser calculados, utilizando a fatoração $QR$ de [ $\underline{H_k}$ $\beta_1v_1$]. Isso é uma propriedade importante do algoritmo do GMRES, ou seja, a norma de resíduo pode ser computada sem que a aproximação tenha sido formada. Tornando o custo computacional menor, pois, a ação cara de formar a aproximação pode ser adiada até que a norma residual seja suficientemente pequena.

Quando $A = A^T$, GMRES é matematicamente equivalente a MINRES, mas não goza da rapidez da convergência. Quando $k$ é grande, $V_k$ e $\underline{H_k}$ consomem muita memória. O principal problema com "GMRES ilimitado" é que na $k$-ésima iteração o processo de Arnoldi requererá o armazenamento de $k$ vetores de dimensão $k$, além de uma matriz de dimensão ($k$+1) × k, o que levará a uma complexidade espacial de O($k^2$) e exigirá $k$ produtos matriz-vetor. Em outras palavras, envolve uma quantidade excessiva de cálculos e de tráfego de memória. Para amenizar esta dificuldade, usa-se o algoritmo iterativamente, isto é, pode-se reiniciar o algoritmo em todos os passos $m$, onde $m$ é um parâmetro inteiro fixo. Por exemplo, se no

máximo $m$ passos são toleráveis, então $x_m$ pode ser usada como o vetor inicial para a sequência GMRES seguinte. Desta forma, depois de um determinado número de iterações, pré-escolhido, os dados acumulados são eliminados e os resultados intermediários são usados como os dados iniciais para as próximas iterações. Este procedimento é repetido até que a convergência seja alcançada. No entanto, as propriedades de convergência são, então, imprevisíveis, exceto em casos especiais, e a estagnação (falta de progresso) pode ocorrer para alguns valores de $m$. A dificuldade está em escolher um valor apropriado para $m$. Se $m$ é muito pequeno, GMRES(m) pode convergir lentamente, ou não convergir. Um valor de $m$ maior que o necessário envolve trabalho excessivo (e usa mais armazenamento). Infelizmente, não há nenhuma regra definida que governa a escolha de $m$; escolher quando reiniciar é uma questão de experiência. A Fig. 4.10, mostra o algoritmo para o GMRES.

Escolha $x^0$ como vetor inicial e $\varepsilon$ como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$h_{10} = ||r_0||$
$k = 0$
enquanto $h_{k+1,k} > \varepsilon$
    $k = k+1$
    $r_k = Aq_k$
    para $i = 1, \ ,k$
        $h_{ik} = q^T_i r_k$
        $r_k = r_k - h_{ik}q_i$
    fim
    $h_{k+1,k} = ||r_k||$
    $x_k = x^{k-1} + Q_k y_{kk}$
fim

Figura 4.10 - Algoritmos dos Métodos GMRES.

## 4.6 Método dos Gradientes Bi-Conjugados (BiCG e PBiCG)

O Método dos Gradientes Conjugados, com ou sem pré-condicionamento, é um método que só se aplica a sistemas lineares cujas matrizes de coeficientes são simétricas e positiva-definidas, isto devido ao fato de que este método constrói uma sequência ortogonal de vetores dos resíduos $r_1, ..., r_k$ e uma sequência ortogonal de vetores de direções de busca $p_1, ..., p_k$. Agora, quando se trata de sistemas lineares, cujas matrizes de coeficientes são positiva-definidas, mas não-simétricas, é necessário que se tenha uma alternativa computacional para resolver este tipo de problema. O Método dos Gradientes Bi-Conjugados é uma opção para estes casos, sua fundamentação é baseada na construção de duas sequências

ortogonais de vetores dos resíduos $r_1,\ldots, r_k$ e $s_1,\ldots,s_k$ e duas sequências ortogonais de vetores de direções de busca $p_1,\ldots, p_k$ e $q_1,\ldots, q_k$, com a desvantagem de não mais garantir a minimização do resíduo. O método dos gradientes bi-conjugados, e suas derivações, não se baseiam na teoria de minimização e sim na teoria de Lanczos, com isso a minimização do resíduo não é garantida.

As duas sequências ortogonais de vetores dos resíduos e direções de busca no Método dos Gradientes Bi-Conjugados são atualizadas, baseadas em $A^T$ ao invés de $A$, segundo as seguintes fórmulas:

$$\begin{aligned} r_k &= r_{k-1} - \alpha_k A p_k, \quad s_k = s_{k-1} - \alpha_k A^T q_k \\ p_k &= r_{k-1} + \beta_k p_{k-1}, \quad q_k = s_{k-1} + \beta_k q_{k-1} \end{aligned} \tag{4.77}$$

Sendo que

$$\alpha_k = \frac{s_{k-1}^T r_{k-1}}{q_k^T A p_k} \quad \text{e} \quad \beta_k = \frac{s_K^T r_k}{s_{k-1}^T r_{k-1}} \tag{4.78}$$

garantem a relação de bi-ortogonalidade:

$$s_i^T r_j = q_i^T A p_j = 0 \qquad se \qquad i \neq j \tag{4.79}$$

Assim como o GMRES é desenvolvido com base no processo de Arnoldi, o método dos Gradientes Bi-Conjugados (BiCG) é desenvolvido segundo o processo de Lanczos não-simétrico. O ponto de partida para o desenvolvimento de BiCG é utilizar a derivação do método do gradiente conjugado através do processo de Lanczos. Em termos de vetores de Lanczos, a $k$-ésima iteração do CG é dada por $x^{(k)} = x^{(0)} + Q_k y_k$, onde $Q_k$ é a matriz de vetores de Lanczos, $T_k = Q_k^T A Q_k$ é tridiagonal, e $y_k$ e resolve $T_k y_k = Q_k^T r_0$. Note que

$$Q_k^T (b - A x_k) = Q_k^T (r_0 - A Q_k y_k) = 0 \tag{4.80}$$

Assim, pode-se caracterizar $x^{(k)}$, insistindo que $x^{(k)} \in x^{(0)} + K(A, r_0, k)$ e que produz um resíduo ortogonal a um subespaço determinado, digamos $K(A, r_0, k)$.

No caso não-simétrico pode-se estender esta noção produzindo a sequência de aproximações $\{x^{(k)}\}$ com a propriedade de que $x^{(k)} \in x^{(0)} + K(A, r_0, k)$ e produz um resíduo que é ortogonal a $K(A^T, s_0, k)$ para algum $s_0 \in \mathbb{R}^n$. Simplificações ocorrem se o processo de Lanczos não-simétrico é usado para gerar bases para os dois espaços de Krylov. Em particular, após $k$ passos do algoritmo de Lanczos não-simétrico tenha $Q_k$, $P_k \in \mathbb{R}^{n\times k}$ tal que $P_k^T Q_k = I_k$ e uma matriz tridigiagonal $T_k = P_k^T A Q_k$ tal que:

$$\begin{aligned} AQ_k &= Q_k T_k + r_k e_k^T, \text{ com } \quad P_k^T r_k = 0 \\ A^T P_k &= P_k T_k^T + s_k e_k^T, \text{ com } \quad Q_k^T s_k = 0 \end{aligned} \tag{4.81}$$

Então, pelo BiCG, a aproximação $x^{(k)}$ é atualizada por $x^{(k)} = x^{(0)} + Q_k y_k$,, onde $T_k y_k = Q_k^T r_0$.

Como era de se esperar, é possível desenvolver a recursividade de modo que se pode calcular $x^{(k)}$ como uma simples combinação de $x^{(k-1)}$ e $q_{k-1}$, em vez de uma combinação linear de todos os vetores $q$ anteriores.

Escolha $x^0$ como vetor inicial e ε como parâmetro de parada (positivo e suficientemente pequeno, $10^{-6}$)
$r_0 = b - Ax^0$
$k = 0$
enquanto $\| r_k \| > \varepsilon$
  $k = k+1$
  resolva $Mz_{k-1} = r_{k-1} \rightarrow z_{k-1}$ , $Mv_{k-1} = s_{k-1} \rightarrow v_{k-1}$
  se $z_{k-1}^T s_{k-1} = 0$, o método falha
  se $k = 1$
    $p_1 = z_0$ , $q_1 = v_0$
  se não $\beta_k = \dfrac{z_{k-1}^T s_{k-1}}{z_{k-2}^T s_{k-2}}$
    $p_k = z_{k-1} + \beta_{k-1} p_{k-1}$ , $q_k = v_{k-1} + \beta_{k-1} q_{k-1}$
  fim
  $w_k = A p_k$ , $\varpi_k = A^T q_k$
  $\alpha_k = \dfrac{z_{k-1}^T s_{k-1}}{q_k^T w_k}$
  $x^k = x^{k-1} + \alpha_k p_k$
  $r_k = r_{k-1} - \alpha_k p_k$, , $s_k = s_{k-1} - \alpha_k \varpi_k$
fim

Figura 4.11 - Algoritmo do Método dos Gradientes Bi-Conjugados Pré-condicionado (PBiCG).

Da mesma forma que existe o Método dos Gradientes Conjugados Pré-Condicionado (PCG), também é possível formular o método bi-conjugado na forma pré-condicionada, que poderia ser chamado de Método dos Gradientes Bi-Conjugados Pré-Condicionado (PBiCG). A Fig. 4.11 apresenta o algoritmo para o método PBiCG.

Pouco se sabe a respeito da convergência do Método PBiCG. Quando a matriz de coeficientes de um sistema linear é positiva-definida e simétrica o método PBiCG produz resultados semelhantes ao apresentados pelo método dos gradientes conjugados PCG, com a desvantagem de necessitar do dobro do custo computacional em cada iteração. Para matrizes positiva-definidas e não-simétricas, o método apresenta uma convergência razoável, mas também pode-se observar um comportamento de convergência muito irregular, e o método pode até mesmo falhar totalmente. Estes comportamentos estão relacionados ao fato de que o método BICG está sujeito a danos graves por causa de sua dependência em relação ao processo de Lanczos não-simétrico.

# Capítulo 5: Métodos Evolutivos de Otimização

Os métodos naturais são procedimentos iterativos que tentam simular os processos usados na natureza para resolver problemas difíceis. Estes algoritmos também são referidos na literatura como algoritmos estocásticos, entre os mais conhecidos pode-se citar Simulated Annealing, Busca Tabu e um grupo de métodos baseados em população. Neste último grupo destacam-se os Algoritmos Evolutivos ou Evolucionários (Algoritmos Genéticos, Estratégias de Evolução, Evolução Diferencial, etc.) e os algoritmos baseados na inteligência coletiva (otimização por enxame de partículas, colônia de formigas, etc.). Diversas técnicas vêm sendo desenvolvidas e aprimoradas, dentre elas, neste capítulo serão considerados os Algoritmos Genéticos (AGs) e a Evolução Diferencial (ED).

## 5.1- Definições Básicas

Os métodos de otimização consideram vários conceitos importantes que serão apresentados a seguir.

Sejam dados um conjunto $\Omega \subset \mathbb{R}^n$ e uma função $f : \Omega \longrightarrow \mathbb{R}$. Um problema de otimização visa determinar um minimizador de $f$ no conjunto $\Omega$, sendo escrito como:

$$\min f(X) \text{ sujeito a } X \in \Omega \subset \mathbb{R}^n \tag{5.1}$$

O conjunto $\Omega$ é chamado *conjunto viável* e define a região onde estão as possíveis soluções do problema. Também recebe o nome de *região viável*, *espaço de busca*, ou ainda *espaço de projeto*. Para limitar certo espaço, utilizam-se as funções de restrição. Os pontos de $\Omega$ são chamados *pontos viáveis*.

A função $f(X)$ é a função que atinge um valor extremo, ou seja, permite avaliar o grau de otimalidade de cada elemento do espaço de busca. É chamada *função objetivo* ou *função custo*, ou ainda *função de adaptação* (*fitness*). A função objetivo pode ser unidimensional, ou seja, possui apenas uma variável de projeto, ou multidimensional, isto é, possui mais de uma variável de projeto.

*Variável de projeto* ou *de decisão* são variáveis que se modificam durante o processo de otimização, ou seja, são quantidades que alteram o valor da função objetivo, sendo representadas por $x_i$, para $i = 1,..., n$.

Um ponto $X^* \in \Omega$ é minimizador global de (5.1) se $f(X^*) \leq f(X)$, para todo $X \in \Omega$ e é minimizador local de (5.1) se existe uma vizinhança $U$ de $X^*$ tal que $f(X^*) \leq f(X)$, para todo $X \in \Omega \cap U$ , como pode ser observado na Fig. 5.1

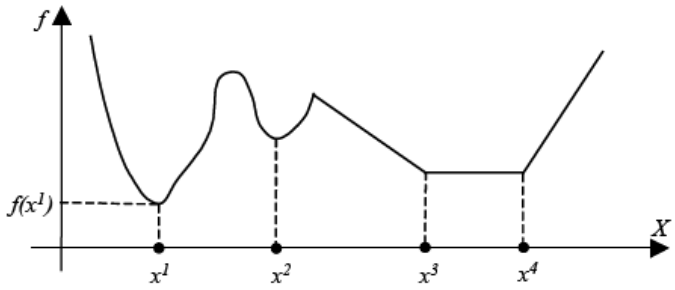



Figura 5.1 – Minimizador Global

Na Fig. 5.1 $x^1$ é o minimizador global, $f(x^1)$ é o valor ótimo, $x^2$ é um minimizador local estrito, $[x^3, x^4]$ é um conjunto de minimizadores locais não estritos.

*Restrições* são funções que delimitam o espaço de busca, através de igualdades ou desigualdades matemáticas, que estabelecem exigências para a solução ótima, $g : \mathbb{R}^n \longrightarrow \mathbb{R}^n$.

*Restrições laterais* são funções que servem para limitar os valores das variáveis de projeto, sendo definidas como: limite inferior $x^{inf} = (x_1^{inf}, x_2^{inf},..., x_n^{inf})^T$ e limite superior $x^{sup} = (x_1^{sup}, x_2^{sup},..., x_n^{sup})^T$, onde $T$ representa a transposição do vetor.

*Restrições ativas* são as restrições de desigualdade que são satisfeitas por $X^*$ na igualdade, ou seja, $X^*$ pertencente ao conjunto viável tal que $g_j(X^*)=0$, $j = 1,..., J$ (MARTINEZ & SANTOS,1995), caso contrário, a restrição é dita *inativa*.

A Fig. 5.2 ilustra um exemplo onde para $j = 1$ e para $j = 2$ as restrições são ativas e para $j = 3$ a restrição é inativa, pois, em torno do ponto ótimo $X^*$ ela não interfere na topologia de $\Omega$.

*Ponto ótimo* é o vetor das variáveis de projeto que satisfaz as restrições, além de corresponder a um extremo da função objetivo. Normalmente é denotado por $X^*$. *Valor ótimo* é o valor da função objetivo no ponto ótimo. *Solução ótima* é o par formado pelo ponto ótimo e pelo valor ótimo, $(X^*, f(X^*))$, podendo ser local ou global.

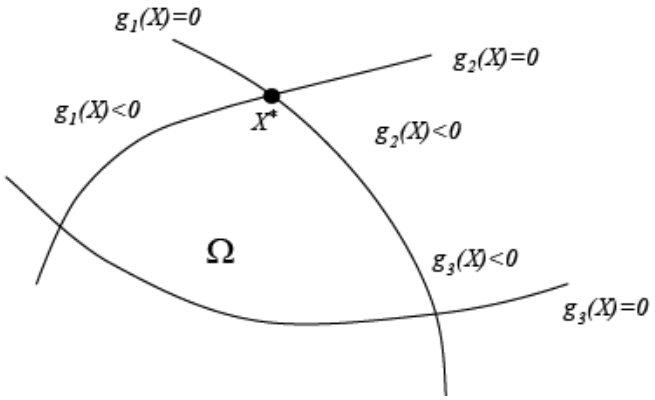



Figura 5.2 – Restrições Ativas e Inativas

O problema geral de otimização consiste em minimizar ou maximizar uma função objetivo, sujeita ou não a restrições de igualdade, desigualdade e restrições laterais. A função objetivo e as funções de restrições podem ser funções lineares ou não lineares em relação às variáveis de projeto, implícitas ou explícitas, descontínuas, não-diferenciáveis, não convexas, calculadas por técnicas analíticas ou numéricas.

Um problema de otimização pode ser formulado como o de encontrar um vetor de $n$ variáveis de projeto $X = (x_1, x_2,\ldots, x_n)^T$ que otimize uma função objetivo $f(X)$, e satisfaça as restrições de igualdade, desigualdade e laterais. O problema geral pode ser escrito como:

$$\text{Minimizar: } f(X),\ X \in \mathbb{R}^n \tag{5.2}$$

$$\text{Sujeito a } \begin{cases} h_l(X) = 0, & l = 1,\ldots,L \\ g_j(X) \le 0, & j = 1,\ldots,J \\ X_i^{\inf} \le X_i \le X_i^{\sup} \end{cases} \tag{5.3}$$

Sendo que $g_j(X)$ representa as restrições de desigualdade e $h_l(X)$ as restrições de igualdade, se houver.

Caso o problema seja maximizar a função objetivo, basta transformar o problema multiplicando a função por –1, *i.e.*, minimiza-se a função $-f(X)$, pois as soluções locais e globais de ambos os problemas são as mesmas, com sinais opostos para os valores ótimos, conforme a Fig. 5.3. Por isso, do ponto de vista matemático, não existe nenhuma diferença relevante entre os problemas de minimização e de maximização: todos os resultados obtidos

para uma classe de problemas podem ser estendidos para a outra classe sem dificuldade (IZMAILOV e SOLODOV, 2005).

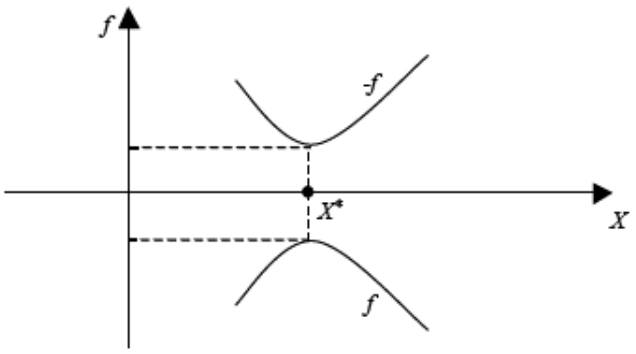



Figura 5.3 - max $f(X)$ = min – $f(X)$

Considere, agora, a solução de um sistema linear usando métodos de otimização. Neste caso, o funcional $f(X)$ deve ser escrito de forma que o seu ponto de mínimo $X^*$ corresponda à solução do sistema linear $Ax = b$. Por exemplo, pode-se propor a seguinte função objetivo:

$$mim\ r(x) = b - Ax \tag{5.4}$$

ou, ainda, propor um funcional quadrático definido por:

$$mim\ f(x) = \frac{1}{2} x^T Ax - x^T b \tag{5.5}$$

observe que, usando $f(x)$ definido na Eq. (5.5), o gradiente é dado por:

$$\nabla f(x) = Ax - b \tag{5.6}$$

assim, para o vetor solução $x^*$, o resíduo $r(x^*) = 0$, corresponde ao $\nabla f(x^*) = 0$.

A seguir será apresentada a revisão de dois métodos de otimização natural, Algoritmos Genéticos (AGs) e a Evolução Diferencial (ED), que podem ser usados para a solução do problema de otimização (5.1).

## 5.2- Algoritmos Genéticos (AGs)

Os algoritmos genéticos (AGs) são abordagens de busca probabilística semi-aleatória baseada na teoria de evolução de Charles Darwin. Assim, é inspirado pelo mecanismo de seleção natural, um processo biológico no qual os indivíduos mais fortes possuem maior chance de sobreviver em um ambiente competitivo. Os indivíduos selecionados produzem uma nova geração e, conseqüentemente, as características hereditárias da espécie são repassadas para essa nova geração, visando-se a preservação das qualidades adquiridas. Esses métodos operam com conjuntos de candidatos, chamados de *população*. Cada indivíduo na população, denominado cromossomo, representa uma solução do problema codificado. Os valores que compõem um indivíduo são denominados genes. Estes genes são constantemente modificados utilizando princípios básicos da evolução natural das espécies: *seleção*, *cruzamento* (*crossover*) *e mutação,* fazendo com que estes sofram alterações de uma geração para outra. Estes princípios tentam representar a competição onde os indivíduos mais aptos reproduzem e repassam seu material genético às gerações futuras, e os indivíduos menos aptos tendem a desaparecer da população (DEB, 2001).

O método foi desenvolvido por HOLLAND (1975) na Universidade de Michigan e popularizado por um de seus alunos, GOLDBERG (1989), que apresentou a solução de problemas complexos de engenharia usando os algoritmos genéticos.

Fazendo uma analogia com o processo da evolução natural, as soluções candidatas são denominadas *indivíduos*, também referidos na literatura como *cromossomos*, *cadeia de valores de parâmetros* ou *cadeias binárias* (HAUPT; HAUPT, 1998; DEB, 2001).

Resumidamente, o procedimento é o seguinte: define-se os parâmetros de otimização e a função objetivo, em seguida faz-se codificação dos parâmetros, efetua-se uma seqüência repetitiva de procedimentos que consistem em avaliar, selecionar, recombinar e modificar os indivíduos, gerando, assim, novas populações. Esses procedimentos são efetuados com base na *aptidão* de cada solução, em que indivíduos com melhores aptidões terão uma maior probabilidade de serem selecionados para o processo de cruzamento, passando assim parte

dos seus genes a seus descendentes. Cada iteração desse processo recebe o nome de *geração*. A finalização é realizada através de testes de convergência, sendo que um número máximo de gerações deve sempre ser estabelecido. O melhor indivíduo será adotado como solução do problema.

A codificação dos parâmetros da função objetivo pode ser expressa por *bits* (binária) ou por *parâmetros contínuos* (*números reais* ou *pontos flutuantes*). Nesta seção, é apresentado o algoritmo genético binário, a fim de facilitar a compreensão dos algoritmos genéticos clássicos, sendo citados alguns aspectos que os diferem dos algoritmos com parâmetros contínuos. Na codificação real, a representação de um cromossomo *d* com *n* parâmetros, dados por $p_1, p_2,..., p_n$, é escrito como um vetor com $1 \times n$ elementos, tal que:

$$cromossomo_d = [p_1, p_2, ..., p_n] \tag{5.7}$$

Na Eq.(5.7) cada parâmetro é representado por um número real. No algoritmo genético binário, a codificação de um cromossomo *d* que possui *n* parâmetros com, por exemplo, 5 bits cada é apresentada na Fig. 5.4.

$$cromossomo_d = \left[ \overbrace{\underbrace{11010}_{gene_1}}^{x_1} \quad \overbrace{\underbrace{01100}_{gene_1}}^{x_1} \quad \cdots \quad \overbrace{\underbrace{01010}_{gene_1}}^{x_1} \quad \cdots \quad \overbrace{\underbrace{00101}_{gene_1}}^{x_1} \right]$$

Figura 5.4 - Representação de um cromossomo com *n* genes de 5 bits cada

Cada *gene* representa uma variável da função objetivo. Os bits *zeros* e *uns* da cadeia binária correspondem aos *alelos* da genética. O número total de bits de um cromossomo é dado por $\Sigma m_i$, $i$=1,..., $n$, onde $m_i$ é o comprimento do gene $i$. Este comprimento depende da precisão requerida para o problema e da amplitude do intervalo definido pelas restrições laterais, podendo ser obtido por:

$$m_i = log_2 \left( \frac{x_i^{sup} - x_i^{inf}}{precisão} \right) \tag{5.8}$$

A decodificação pode ser feita conforme a Eq.(5.9) generalizada. Os tamanhos dos genes podem ser diferentes, pois dependem das restrições laterais.

$$x_i = x_i^{\inf} + decimal\left(gene_i\right)_2 \times \frac{x_i^{\sup} - x_i^{\inf}}{2^{m_i} - 1},\ i = 1, ..., \tag{5.9}$$

onde $decimal\left(gene_i\right)_2 = \sum bit_{p_i} \times 2^{p_i}$ é o valor decimal da variável, sendo $bit_{pi}$ o bit correspondente à posição $p_i$ do gene $i$.

Tanto a evolução biológica quanto os algoritmos genéticos começam com uma população inicial, constituída por *Np* indivíduos, gerada aleatoriamente. Com a população inicial definida, calcula-se a função objetivo, que é geralmente referida na literatura, como *função de avaliação* (*fitness*) ou *função custo*. Esta função associa um valor numérico, conhecido como grau de adaptação, a cada indivíduo da população. Quando possível esta função deve ser de cálculo rápido, uma vez que ela deve ser avaliada para todos os indivíduos da população, o que tende a elevar o esforço computacional.

Os procedimentos básicos que transformam a população ao longo das gerações em busca da solução do problema de otimização caracterizam os operadores genéticos: *seleção*, *cruzamento* e *mutação*. Na prática, tem-se verificado que a consistência de um AG é garantida por estes três operadores (HAUPT; HAUPT, 1998). Uma descrição destes operadores é dada a seguir.

## 5.2.1- Seleção (Reprodução)

Este operador determina quais indivíduos serão escolhidos para o cruzamento. O grau de adaptação de cada cromossomo é caracterizado de acordo com o valor da função custo. Em um problema de maximização, por exemplo, quanto maior este valor, maior a probabilidade de contribuir à geração seguinte. Existem vários mecanismos para executar o operador seleção (DeJONG, 1975; GOLDBERG, 1989).

Na seleção elitista os melhores indivíduos de uma população intermediária são escolhidos. Para obter a população intermediaria escolhe-se a metade da população de uma geração que corresponde aos indivíduos mais aptos, sendo a outra metade eliminada. Desse

modo o cruzamento é realizado com os melhores indivíduos e espera-se que a convergência do algoritmo se torne mais rápida.

Na seleção aleatória, como indica o próprio nome, os indivíduos de uma população intermediária são selecionados aleatoriamente para o posterior cruzamento.

Na seleção por torneio dois indivíduos são escolhidos aleatoriamente, a seguir gera-se um número aleatório *rand* ∈ [0, 1]. Se *rand* for menor que um parâmetro previamente definido, que determina o quanto este operador será elitista, o melhor indivíduo é escolhido, senão o outro sobreviverá.

Outra forma de fazer a seleção é através do método da Roleta (*roulette wheel*), no qual a probabilidade de seleção de um indivíduo é diretamente proporcional ao valor da função custo. Em seguida, a roleta, devidamente dividida, é girada e seleciona-se os indivíduos através deste sorteio. Logicamente, os indivíduos com maior participação na roleta apresentam maiores chances de serem selecionados. A cada rodada da roleta, um novo subconjunto é formado. Segundo SOARES (1997) o método da roleta possibilita a ocorrência de uma convergência prematura. GOLDBERG (1989) cita também a tentativa de redução dos erros estocásticos fornecidos pelo método da roleta com a utilização de métodos alternativos, como os citados anteriormente. Copiar um indivíduo conforme sua aptidão significa que indivíduos com valores maiores têm uma probabilidade maior de contribuir com um ou mais descendentes na próxima geração.

Para a representação gráfica na roleta, conforme Fig. 5.5, multiplica-se esta razão por 360 graus.

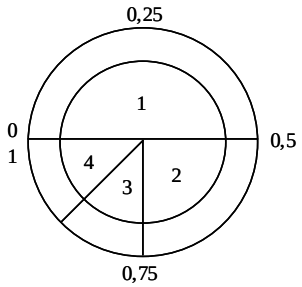



Figura 5.5 - Exemplo da seleção por roleta

A seguir, devem-se selecionar as cadeias que irão contribuir para a geração seguinte. Gera-se um conjunto de números aleatórios *rand* ∈ [0, 1] em quantidade igual ao número de indivíduos da população. Por exemplo, se *rand* < $Pa_1$, seleciona-se o primeiro cromossomo, senão, passar para o indivíduo subseqüente e fazer a análise novamente.

No processo de seleção, observa-se que alguns cromossomos podem ser selecionados mais de uma vez, ou seja, os melhores serão copiados mais vezes, enquanto que os piores não sobreviverão.

### 5.2.2- Cruzamento (*Crossover*)

Nos sistemas biológicos o cruzamento pode ocorrer durante a reprodução sexuada permitindo a troca de material genético entre dois indivíduos. O algoritmo genético utiliza os cromossomos determinados no processo de seleção para gerarem novos descendentes. Este operador é a primeira forma do algoritmo explorar o espaço de busca e evitar uma convergência prematura, ou seja, evitar ótimos locais (HAUPT; HAUPT, 1998; BRAGA, 1998).

A forma mais simples de cruzamento envolve dois pais que irão produzir dois descendentes. A quantidade de cromossomos da população a ser submetida ao cruzamento é definida através da *probabilidade de cruzamento Pc*, fornecida pelo usuário. É bastante usual adotar para esta probabilidade $Pc = 60\%$.

O processo de escolha dos indivíduos que serão cruzados deve ser feito em pares, gerando números aleatórios $rand_d \in [0, 1]$, $d = 1,..., Np$. Por exemplo, se $rand_1$ for menor que a probabilidade $Pc$, então o primeiro cromossomo da população será selecionado. Desta forma, um cromossomo $d$ é selecionado para o cruzamento se $rand_d < Pc$. A posição de cruzamento é selecionada entre o primeiro e o último bit dos cromossomos paternos. Uma maneira de selecionar esta posição é descrita a seguir:

Seja $p_c$ a posição de cruzamento na cadeia binária de cada cromossomo, dada conforme a Eq.(5.10).

$$p_c = 1 + rand\,[(\Sigma m_i - 1) - 1] \tag{5.10}$$

onde *rand* é um número aleatório no intervalo [0, 1] e $\Sigma m_i$, $i = 1,..., n$, é a quantidade de bits do cromossomo, que também representa a posição do último bit no cromossomo.

Cada cadeia é quebrada na posição $p_c$ e todas as informações de um cromossomo, compreendidas entre as posições $p_c + 1$ e $\Sigma m_i$, são copiadas para o outro cromossomo e vice-versa, como ilustrado na Fig. 5.6.

Figura 5.6 - Representação do operador cruzamento simples entre dois indivíduos

No caso contínuo, o modo mais simples é escolher um ou mais pontos no cromossomo e indicá-los como posições de cruzamento. As posições onde deve ocorrer o cruzamento são selecionadas aleatoriamente e conforme a Fig. 5.7, percebe-se nos descendentes uma combinação dos parâmetros de ambos os pais.

Figura 5.7 - Representação do operador cruzamento uniforme com parâmetros contínuos

Existem outros tipos de cruzamento que podem ser vistos na literatura (HAUPT; HAUPT, 1998; MICHALEWICZ, 1996; ESHELMAN e SCHAFFER, 1993).

### 5.2.3- Mutação

O próximo estágio de um AG é a operação de mutação (operador genético secundário) que modifica aleatoriamente os genes de um indivíduo através de uma taxa (ou probabilidade) de mutação. É uma modificação aleatória do material genético dos indivíduos, ou seja, é a alteração de pequenas percentagens nos bits dos cromossomos. Sendo outra forma do algoritmo genético explorar a região de busca. Este operador introduz maior aleatoriedade dentro da população para esta escapar de mínimos (ou máximos) locais, ou seja, introduz características que não dependem da população original aumentando a diversidade da população, podendo evitar que algoritmo convirja prematuramente.

Uma estratégia usual é a mutação de *ponto único*, que modifica o bit "1" para "0" e vice versa em determinadas posições do cromossomo, conforme Fig. 5.8. Os pontos de mutação são selecionados aleatoriamente em uma matriz $Np \times \Sigma m_i$, i = 1,..., $n$, que representa o número total de bits da população. Aumentar o número de mutações aumenta a liberdade do algoritmo em buscar soluções fora do espaço de busca, para evitar isso, usualmente faz-se mutação de 1% a 5% do total de bits por iteração. Esta mutação não se aplica aos melhores indivíduos, pois são considerados de *elite*.

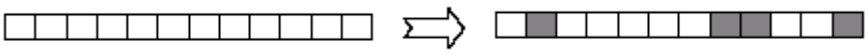


Figura 5.8 - Representação do operador mutação

Uma forma de realizar a mutação é gerar pares aleatórios (*A*, *B*), onde *A* representa o cromossomo a sofrer a mutação e *B* representa a posição do bit a ser mudado. Outra forma é selecionar aleatoriamente a posição em um cromossomo, obedecendo a uma *probabilidade de mutação* $P_m$ (também conhecida como *taxa de mutação*), e mudar o valor do bit. Neste caso, é necessário gerar números randômicos *rand* no intervalo [0, 1] em mesma quantidade de bits total da população, representando a posição seqüencial de cada bit na matriz populacional. Para os casos onde *rand* for menor que a probabilidade $P_m$ serão feitas as mutações nos bits correspondentes. Geralmente, recomenda-se usar $P_m$ igual a 1%. (HAUPT; HAUPT, 1998).

Na representação real, a probabilidade de mutação adequada está entre 1% e 20%. O parâmetro que sofre mutação é substituído por um novo parâmetro gerado aleatoriamente. Depois de realizadas as mutações, os custos associados aos descendentes e aos cromossomos que sofreram mutações são calculados, compondo a próxima geração.

Para problemas que envolvem ordenação e permutação, o operador mutação deve evitar que haja repetição de variáveis num mesmo indivíduo. Uma troca que ocorre com certa probabilidade pode ser aplicada e, a partir da escolha de duas variáveis, troca-se as posições de ambas.

De acordo com FILHO *et al.* (1994), tem sido introduzido um número de diferentes operadores genéticos desde que Holland propôs seu modelo básico. Eles são, em geral, versões do cruzamento e alteração dos processos genéticos adaptados às necessidades de problemas específicos. Alguns exemplos são: inversão, dominância e "*genetic edge recombination*".

### 5.2.4- Considerações finais

Como o algoritmo é iterativo, o procedimento acima é repetido até obter uma solução desejável. Usualmente é usado como critério de parada o número máximo de gerações, porém existem outros critérios tais como: estagnação da população (os cromossomos e seus

respectivos custos se repetem); existência de um indivíduo com qualidades satisfatórias, entre outros.

A Fig. 5.9 mostra um fluxograma básico de um algoritmo genético binário. O melhor indivíduo, ou seja, aquele que possuir a melhor aptidão será adotado como solução do problema de otimização.

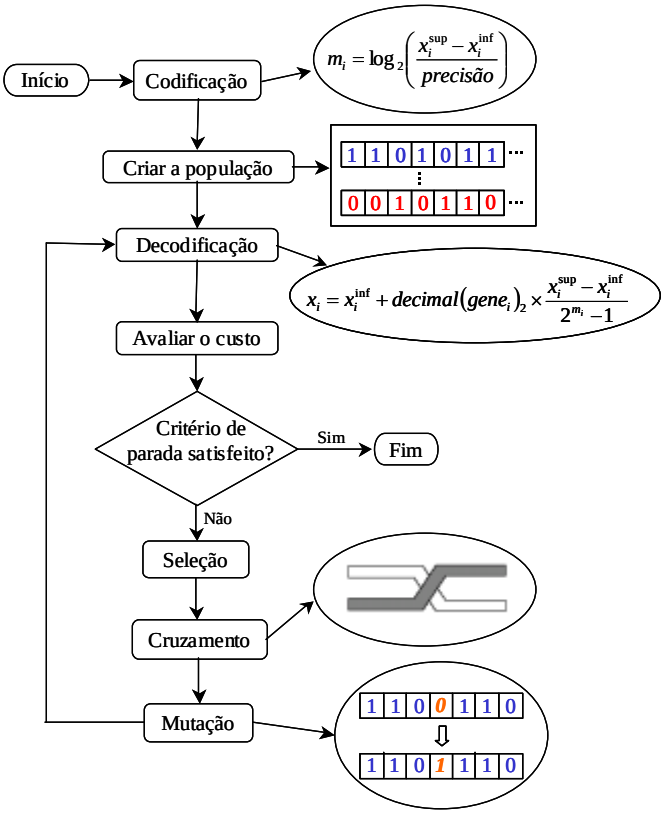



Figura 5.9 - Fluxograma básico de um algoritmo genético binário

Os algoritmos genéticos apresentam algumas vantagens quando comparados aos métodos de otimização determinísticos, podendo citar: facilidade em sua implementação; eficiência ao evitarem ótimos locais; não requerem o cálculo de derivadas, o que favorece a otimização de funções complicadas; são robustos e apresentam desempenho satisfatório na resolução de vários tipos de problemas, entre outras. Como desvantagem do método, tem-se o elevado custo computacional, e, além disto, nem sempre a solução ótima global é garantida (HAUPT e HAUPT, 1998).

## 5.3- Evolução Diferencial (ED)

Dentre os algoritmos evolucionários, destaca-se o algoritmo de Evolução Diferencial (Differential Evolution), desenvolvido por Storn e Price em 1995, visando a busca por melhores resultados com uma abordagem um pouco diferente da utilizada nos algoritmos genéticos e nas estratégias de evolução. Trata-se de um método de busca direta estocástica que surgiu de tentativas de resolver o problema de ajuste polinomial de Chebychev. Kenneth Price introduziu a idéia de usar diferenças de vetores para perturbar a população de vetores (indivíduos) resultando em um método que requer poucas variáveis de controle, é de rápida convergência, fácil de usar e robusto (STORN, 1999). Segundo STORN e PRICE (1995) a evolução diferencial é uma nova heurística para a minimização de funções não-lineares e não-diferenciáveis no espaço contínuo

A escolha do algoritmo de Evolução Diferencial para otimização numérica, conforme CHENG e HWANG (2001), está baseada nas seguintes características:

- É um algoritmo de busca estocástica, originado dos mecanismos de seleção natural;
- Dificilmente torna-se preso em ótimos locais, pois busca a solução ótima global manipulando uma população de soluções, ou seja, buscando simultaneamente a solução em diferentes regiões do espaço de busca;
- É muito eficaz para resolver problemas de otimização com função objetivo descontínua, pois não requer informação sobre suas derivadas;
- Permite que os parâmetros de entrada e saída sejam manipulados como números ordinários reais (pontos flutuantes) sem processamento extra, e, portanto, utiliza eficientemente os recursos do computador;
- Trabalha bem como otimizador local porque os diferenciais gerados por uma população convergente eventualmente tornam-se infinitesimais;
- É eficaz trabalhando com uma população pequena.

O seguinte conjunto de regras tem surgido como orientações para a escolha das variáveis de controle $F_P$ (fator de pertubação), $P_C$ (probabilidade de cruzamento) e $N_P$ (tamanho da população), conforme STORN (1996):

- A população inicial deve ser gerada o mais próximo possível da superfície da função objetivo;

- Freqüentemente a probabilidade de cruzamento $P_C \in [0,1]$ deve ser considerada menor do que um (por exemplo, 0,3). Caso não ocorra convergência, uma $P_C \in [0,8;1]$ pode ajudar;
- Para muitas aplicações $N_P$ de 10**D* (onde *D* é igual à dimensão ou ao número de variáveis) é uma boa escolha. $F_P$ é, normalmente, escolhido no intervalo [0.5, 1.0];
- Quanto maior for o tamanho da população escolhida, menor é o valor de $F_P$;
- Tem-se um bom sinal de convergência quando os parâmetros do melhor membro da população variam muito de geração para geração, especialmente durante o início do processo de minimização. Mesmo se o seu valor da função objetivo decrescer lentamente;
- Não há necessariamente um desempenho inapropriado, quando o valor da função objetivo do melhor membro da população apresentar platôs (áreas no espaço de busca em que todos os pontos têm o mesmo valor) durante o processo de minimização. Entretanto, isto indica que a minimização levará um tempo maior até encontrar o mínimo global (ou próximo dele) ou que o aumento do tamanho da população poderá ser benéfico para a convergência;
- O valor da função objetivo do melhor membro da população não pode cair de forma brusca. Caso aconteça, a otimização está em um mínimo local;
- A escolha apropriada da função objetivo é crucial. Quanto maior for a inclusão do conhecimento sobre o problema na função objetivo, maior será a possibilidade de uma convergência suave e adequada.

A idéia principal da evolução diferencial é gerar novos indivíduos, denotados vetores modificados ou doadores, pela adição da diferença vetorial ponderada entre dois indivíduos aleatórios da população a um terceiro indivíduo. Esta operação é chamada *mutação*.

As componentes deste novo indivíduo doador são misturadas com as componentes de um indivíduo escolhido aleatoriamente (denotado vetor alvo ou vetor a ser substituído), para resultar o chamado vetor tentativa, ou vetor experimental. O processo de misturar os parâmetros é referido freqüentemente como *cruzamento* na comunidade dos algoritmos evolutivos.

Se o custo do vetor experimental for menor que o custo do vetor alvo, então o vetor experimental será o vetor alvo da geração seguinte. Esta última operação é chamada *seleção*. O procedimento é finalizado através de algum critério de parada.

Os operadores da evolução diferencial se baseiam no princípio da evolução natural cujos objetivos são manter a diversidade da população, evitar convergências prematuras e obter a melhor solução.

### 5.3.1- Mutação

Para a obtenção do vetor doador $V^{(q+1)}$, considere os vetores $X_\alpha$, $X_\beta$ e $X_\gamma$ distintos entre si e escolhidos aleatoriamente em uma população com $Np$ indivíduos. $Np$ deve ser maior ou igual a 4 para garantir uma quantidade suficiente de indivíduos para a execução do método. Os índices aleatórios $\alpha$, $\beta$, $\gamma \in \{1,..., Np\}$ são inteiros distintos entre si. Utilizando o par de vetores ($X_\beta$, $X_\gamma$) da $q$-ésima geração define-se o vetor diferença ($X_\beta - X_\gamma$). Esta diferença é multiplicada por $F_p$, sendo denotada diferença vetorial ponderada ou apenas diferença ponderada e será usada para perturbar o terceiro vetor $X_\alpha$. Onde $F_p$ é o fator de perturbação que é um número real, positivo pertencente ao intervalo [0, 2] e que controla a amplitude do vetor diferença.

O processo de mutação pode ser escrito como:

$$V^{(q+1)} = X_\alpha^{(q)} + F_p(X_\beta^{(q)} - X_\gamma^{(q)}) \tag{5.11}$$

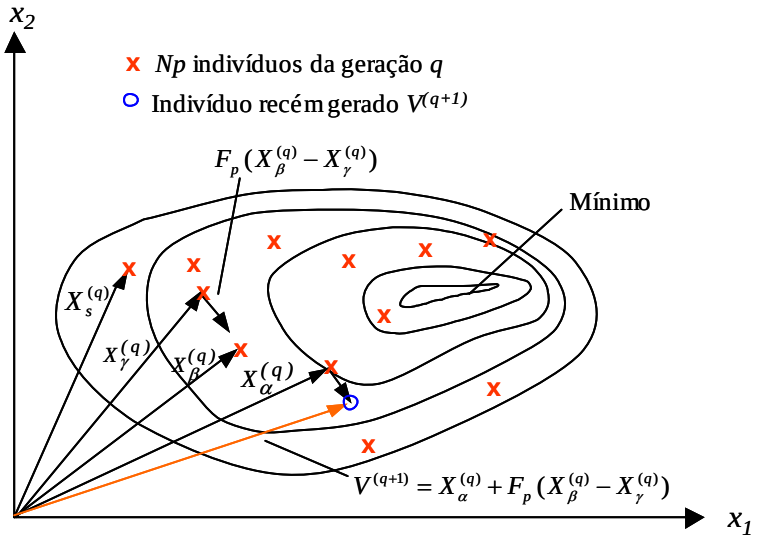



Figura 5.10 - Processo para gerar o vetor doador $V^{(q+1)}$ de uma função bidimensional

A Fig. 5.10 mostra um exemplo bidimensional que ilustra os diferentes vetores que geram o vetor doador $V^{(q+1)}$.

## 5.3.2- Cruzamento

Considere que para cada vetor alvo $X_s^{(q)}$, $s \in \{1,..., Np\}$, e diferente dos índices $\alpha$, $\beta$ e $\gamma$, foi gerado um vetor doador. O cruzamento é introduzido para aumentar a diversidade dos indivíduos que sofreram a mutação. Assim, utilizando o vetor doador e o vetor alvo, as componentes do vetor experimental $U^{(q+1)}$ são escolhidas pela seguinte comparação:

$$u(i)^{(q+1)} = \begin{cases} v(i)^{(q+1)}, \ se\ rand_i \le Pc \\ x_s(i)^{(q)}, \ se\ rand_i > Pc, \ i = 1,...,n \end{cases} \tag{5.12}$$

sendo $rand_i$ um número gerado aleatoriamente no intervalo [0, 1], $Pc \in [0,1]$ é a probabilidade do cruzamento e representa a probabilidade do vetor experimental herdar os valores das variáveis do vetor doador, devendo ser fornecida pelo usuário. Quando $Pc$ = 1, por exemplo, todas as componentes do vetor experimental virão do vetor doador $V^{(q+1)}$. Por outro lado, se $Pc$ = 0, todas as componentes do vetor experimental virão do vetor alvo $X_s^{(q)}$.

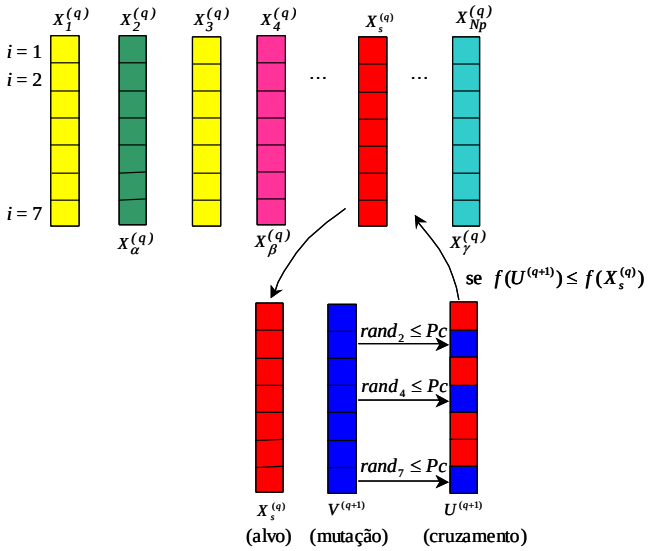



Figura 5.11 - Ilustração do processo de cruzamento binomial para $\alpha = 2$, $\beta = 4$ e $\gamma = Np$

Este tipo de cruzamento, apresentado por STORN e PRICE (1995) é denominado operador cruzamento binomial (devido aos experimentos binomiais independentes), sendo executado em cada variável sempre que um número aleatório *rand* ∈ [0, 1] for menor que a probabilidade de cruzamento *Pc*. A Fig 5.11 mostra o processo de cruzamento binomial para uma função de 7 variáveis.

Alguns anos mais tarde, STORN e PRICE (1997) desenvolveram o operador cruzamento exponencial, em que o cruzamento é executado nas variáveis enquanto o número aleatório *rand* ∈ [0,1] for menor que a probabilidade de cruzamento *Pc*. A primeira vez que este número aleatório ultrapassar o valor de *Pc*, nenhum cruzamento é executado e as variáveis restantes são deixadas intactas, ou seja:

Enquanto $rand_i \leq Pc$, $u(i)^{(q+1)} = v(i)^{(q+1)}$,

Se $rand_i > Pc$, $u(j)^{(q+1)} = x_s(j)^{(q)}$, $j = (i+1),..., n$ (5.13)

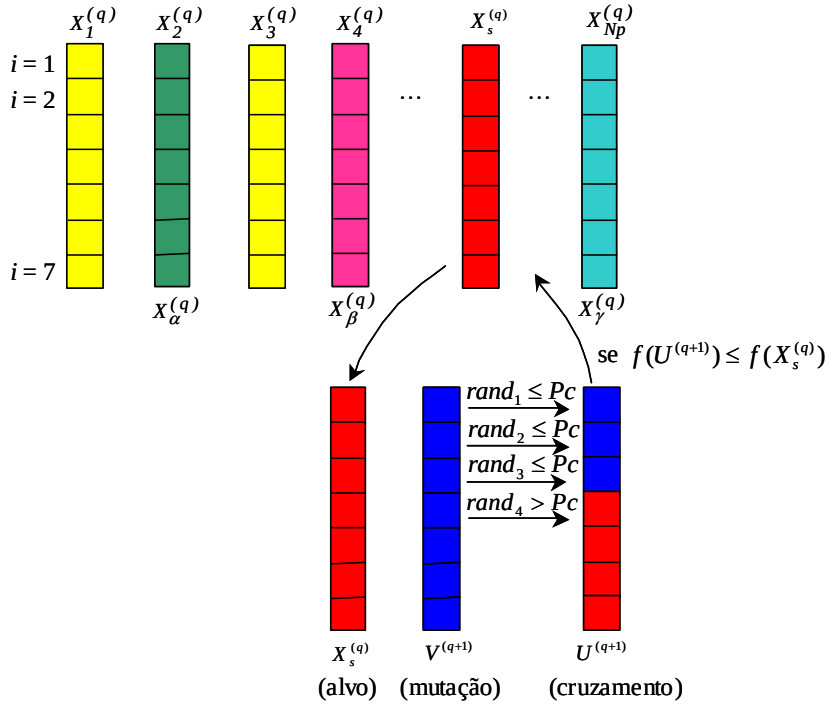



Figura 5.12 - Ilustração do processo de cruzamento exponencial para $\alpha = 2$, $\beta = 4$ e $\gamma = Np$

A Fig. 5.12 mostra o processo de cruzamento exponencial para uma função de 7 variáveis.

Se após o cruzamento uma ou mais componentes do vetor experimental estiver fora da região de busca, definida pelas restrições laterais das variáveis de projeto, as seguintes correções devem ser feitas:

$$\begin{cases} Se\ u(i) < x(i)^{inf}\ \ então\ u(i) = x(i)^{inf} \\ \\ Se\ u(i) > x(i)^{sup}\ \ então\ u(i) = x(i)^{sup} \end{cases} \qquad i = 1, ..., n \tag{5.14}$$

### 5.3.3- Seleção

A seleção é o processo de produzir filhos melhores. Diferentemente de outros algoritmos evolutivos, a evolução diferencial não usa hierarquia (elitismo) nem seleção proporcional. Em vez disso, o custo do vetor experimental $U^{(q+1)}$ é calculado e comparado com o custo do vetor alvo $X_s^{(q)}$. Se o custo do vetor experimental for menor que o custo do vetor alvo, o vetor alvo da próxima geração será o vetor experimental. Caso contrário, o vetor alvo da próxima geração será o vetor alvo da geração atual.

Em outras palavras, este processo pode ser escrito como:

$$\begin{cases} Se\ \ f(U^{(q+1)}) \leq f(X_s^{(q)})\ então\ X_s^{(q+1)} = U^{(q+1)} \\ \\ Se\ \ f(U^{(q+1)}) > f(X_s^{(q)})\ então\ X_s^{(q+1)} = X_s^{(q)} \end{cases} \tag{5.15}$$

O procedimento acima é finalizado através de algum critério de parada, sendo que o número máximo de gerações deve ser estabelecido.

Usualmente, o desempenho do algoritmo de ED depende principalmente do tamanho da população $N_P$, da região de busca, da taxa de cruzamento e também do fator de perturbação $F_P$.

## 5.3.4- Algoritmo da Evolução Diferencial

Nesta seção, apresenta-se de forma resumida, um algoritmo do método da Evolução Diferencial. Note que para uma melhor utilização deste algoritmo é necessário que se defina de forma precisa o espaço de busca $\Omega \subset \mathbb{R}^n$.

(i) Ecolher o fator de perturbação $F_p$ e a probabilidade de cruzamento *Pc*;

(ii) (**inicialização**) Gerar uma população inicial aleatória, com distribuição uniforme, como segue:

$$x(i)_d^{(q)} = x(i)^{inf} + rand\left(x(i)^{sup} - x(i)^{inf}\right),\ i = 1,..., n;\ \ d = 1,..., Np$$

onde $x(i)^{inf} \le x(i) \le x(i)^{sup}$ são as restrições laterais e $rand \in [0, 1]$.

(iii) Escolher um indivíduo aleatório $X_s^{(q)}$ a ser substituído (alvo);

(iv) Escolher outros três indivíduos $X_\alpha^{(q)}$, $X_\beta^{(q)}$ e $X_\gamma^{(q)}$, $\alpha \neq \beta \neq \gamma \neq s$ ou cinco indivíduos distintos, dependendo da estratégia adotada em (i);

(v) (**mutação**) Gerar um indivíduo doador $V^{(q+1)}$ de acordo com a estratégia escolhida, por exemplo:

$$V^{(q+1)} = X_\alpha^{(q)} + F_p(X_\beta^{(q)} - X_\gamma^{(q)})$$

(vi) (**cruzamento**) Gerar um indivíduo $U^{(q+1)}$ a ser comparado com $X_s^{(q)}$, através da equação dada por:

$$u(i)^{(q+1)} = \begin{cases} v(i)^{(q+1)}, \text{se } rand_i \leq Pc \\ x_s(i)^{(q)}, \text{se } rand_i > Pc \end{cases}, i = 1,...,n$$

Se após o cruzamento uma ou mais componentes de $U^{(q+1)}$ estiver fora da região de busca, faz-se a seguinte correção:

Se $u(i) < x(i)^{inf}$, então faz-se $u(i) = x(i)^{inf}$,
Se $u(i) > x(i)^{sup}$, então faz-se $u(i) = x(i)^{sup}$, $i = 1,..., n$.

(vii) (**seleção**) Escolher o melhor indivíduo analisando a função objetivo:

Se $f(U^{(q+1)}) \leq f(X_s^{(q)})$ então $X_s^{(q+1)} = U^{(q+1)}$ ;

Se $f(U^{(q+1)}) > f(X_s^{(q)})$ então $X_s^{(q+1)} = X_s^{(q)}$ .

(viii) (**critério de parada**) Se um critério de parada é satisfeito, fim. Senão, passar para a próxima geração ($q = q+1$) e voltar ao passo (iii).

# Capítulo 6: Simulações Numéricas – Aplicações

Neste capítulo, inicialmente, utilizam-se sistemas lineares dados por matrizes testes com o objetivo de que as soluções obtidas usando Evolução diferencial sejam comparadas com os resultados encontrados por alguns métodos iterativos não-estacionários: Gradientes Conjugados (CG), Resíduos Mínimos (MINRES), Resíduos Mínimos Generalizados (GMRES) e Gradientes Bi-Conjugados (BiCG). Em seguida, utiliza-se de um problema modelado por uma equação de Laplace visando comparar a solução analítica com as soluções numéricas. A utilização de técnicas evolutivas de otimização na solução de grandes sistemas lineares é aplicada ao problema de identificação indireta de forças dinâmicas, em um primeiro momento é testada a metodologia através de um problema teórico, onde se faz a excitação por uma força harmônica e os resultados obtidos por ED, AG e por GMRES são comparados com a solução analítica. Em seguida aplica-se um ruído branco e, de forma similar, são identificas as forças dinâmicas. Para verificar a eficácia da utilização de métodos evolutivos, conforme proposto nesta pesquisa, procurou-se identificar as forças dinâmicas através dos dados obtidos análise modal experimental. Os resultados e discussões desta aplicação serão apresentados na última seção deste capítulo.

As técnicas aqui apresentadas foram executadas utilizando códigos computacionais implementados em MATLAB®7, sendo utilizado um microprocessador Intel(R) Pentium; (R) 4 CPU 3.06 GHz, 1536 MB RAM.

A resolução dos sistemas utilizando os métodos iterativos, CG, MINRES, GMRES e BiCG, foi feita através de sub-rotinas do MATLAB®7 (PALM, 2005). Já na resolução utilizando o algoritmo ED, utilizou-se o código computacional desenvolvido por OLIVEIRA, (2006). Para a aplicação do método AG, utilizou-se as sub-rotinas GAOT do MATLAB®7.

Em todos os casos estudados, os parâmetros utilizados na Evolução Diferencial (ED) foram: número de indivíduos da população $N_p$ = 100, multiplicador da diferença ponderada $F_p$ = 0,8 e probabilidade de cruzamento $P_c$ = 0,5. Para a aplicação da Evolução Diferencial (ED), inicialmente, fez-se um teste comparativo entre os valores de algumas funções objetivo a fim de analisar qual a função apresenta-se mais adequada.

O algoritmo ED é inicializado com a geração de uma população de $N_P$ vetores.

$$x_j = [x_{j,1}, x_{j,2}, \cdots, x_{j,n}]^T,\ j = 1, 2, \cdots,\ N_p \tag{6.1}$$

Cujos valores iniciais são escolhidos aleatoriamente dentro de limites definidos pelo usuário,

$$x_{j,\,k} \in [x_k^{inf},\ x_k^{sup}],\ k = 1, 2, \ldots, n \tag{6.2}$$

Depois que a população inicial é gerada, o vetor custo de cada população é avaliado e armazenado para referência futura.

Os parâmetros utilizados nos Algoritmos Genéticos (AG) foram: número de indivíduos da população $N_p$ = 100, probabilidade de cruzamento $P_c$ = 0,6 e probabilidade de mutação $P_m$ = 0,08.

## 6.1. Comparação entre os Métodos, Iterativos Não-Estacionários e Heurísticos, Aplicados à Solução de Grandes Sistemas Lineares

Nesta seção testou-se a eficiência dos métodos Gradientes Conjugados (CG), Resíduos Mínimos (MINRES), Resíduos Mínimos Generalizados (GMRES), Gradientes Bi-Conjugados (BiCG) e Evolução Diferencial (ED), na resolução do sistema $Ax=b$ para quatro tipos de matrizes testes $A$, tri-diagonais, com dimensão $n \times n$, definidas como:

$$A1(ns, npd) \equiv A(i, j) = \begin{cases} \pi / n & se \quad i = j+1 \\ 4 / n & se \quad i = j \\ (\pi - 1) / n & se \quad j = i+1 \end{cases} \qquad \begin{matrix} i = 1, \cdots, n \\ j = 1, \cdots, n \end{matrix}$$

$$A2(ns, pd) \equiv A(i, j) = \begin{cases} 2 / n & se \quad i = j+1 \\ 4 / n & se \quad i = j \\ 1 / n & se \quad j = i+1 \end{cases} \qquad \begin{matrix} i = 1, \cdots, n \\ j = 1, \cdots, n \end{matrix} \tag{6.3}$$

$$A3(s, npd) \equiv A(i, j) = \begin{cases} (1+\pi) / n & se \quad i = j+1 \\ 4 / n & se \quad i = j \\ (1+\pi) / n & se \quad j = i+1 \end{cases} \qquad \begin{matrix} i = 1, \cdots, n \\ j = 1, \cdots, n \end{matrix}$$

$$A4(s, pd) \equiv A(i, j) = \begin{cases} 1 / n & se \quad i = j+1 \\ 4 / n & se \quad i = j \\ 1 / n & se \quad j = i+1 \end{cases} \qquad \begin{matrix} i = 1, \cdots, n \\ j = 1, \cdots, n \end{matrix}$$

Considerando o vetor *b*, calculado como:

$$b_i = \sum_{j=1}^{n} a_{ij}\,,\; i = 1,\, 2, \cdots, n \tag{6.4}$$

Estas matrizes esparsas foram compactadas usando a função SPDIAGS, também do MATLAB®7. As Tabelas 6.2 a 6.5 apresentam os resultados para cada uma das matrizes: a Tabela 6.2 mostra os resultados do sistema para a matriz *A1*(ns,npd), não simétrica e não positiva-definida. Os resultados para a matriz *A2*(ns,pd), não simétrica e positiva-definida, podem ser verificados na Tabela 6.3. Na Tabela 6.4 estão as soluções para a matriz *A3*(s,npd), simétrica e não positiva-definida e, por fim, na Tabela 6.5 estão apresentados o resultados do sistema para a matriz *A4*(s,pd), simétrica e positiva-definida.

Para a aplicação das técnicas evolutivas, fez-se um teste comparativo entre os valores de algumas funções objetivo a fim de analisar qual a função que se apresenta mais adequada. Considerando o sistema $Ax = b$ e o resíduo $r(x)$, algumas formas para a função objetivo foram estudadas:

$$\begin{aligned}
&F_1 = F(x) = 0.5\left(x^T Ax - x^T b\right);\\
&F_2 = F(x) = \left\|r(x)\right\|_2;\\
&F_3 = F(x) = \max_{i=1,\cdots,n}\left(\left|r_i(x)\right|\right);\\
&F_4 = F(x) = \left\|r(x)^T r(x)\right\|_2;\\
&F_5 = F(x) = \max\left(r(x)^T r(x)\right);\\
&F_6 = F(x) = \frac{\left\|r(x)\right\|_2}{\left\|b\right\|_2}.
\end{aligned} \tag{6.5}$$

A seguir serão apresentadas tabelas que mostram uma comparação entre as funções definidas pela Eq.(6.5), para o sistema linear $Ax = b$ onde a matriz de coeficientes *A* é a matriz *A1* definida na Eq. 6.3 e o vetor *b* é o definido pela Eq. 6.4. A comparação é feita tendo como

parâmetros o tempo computacional e o erro, medido através da norma euclidiana do resíduo, ou seja, $||r^T r||$.

Tabela 6.1 – Valores da função, do erro e do tempo computacional, para diferentes funções $F(x)$, aplicando ED.

| $n$ | $F_1$ | Erro $\|\|r^T r\|\|$ | t (s) | $F_2$ | Erro $\|\|r^T r\|\|$ | t (s) |
|---|---|---|---|---|---|---|
| $1*10^1$ | -5.2365 | $3.94*10^{-1}$ | 2.312 | $7.78*10^{-2}$ | $5.94*10^{-3}$ | 3.469 |
| $1*10^2$ | -5.4653 | $3.47*10^{-2}$ | 3.766 | $1.62*10^{-1}$ | $2.58*10^{-2}$ | 6.953 |
| $1*10^3$ | -5.4847 | $3.94*10^{-3}$ | 3.094 | $6.35*10^{-2}$ | $4.17*10^{-3}$ | 34.219 |
| $1*10^4$ | -5.4774 | $4.24*10^{-4}$ | 9.250 | $2.06*10^{-2}$ | $4.22*10^{-4}$ | 483.219 |
| $5*10^4$ | -5.4755 | $8.58*10^{-5}$ | 148.797 | $3.64*10^{-3}$ | $8.62*10^{-5}$ | 3571.031 |
| $1*10^5$ | -5.4756 | $4.29*10^{-5}$ | 917.578 | $6.56*10^{-3}$ | $4.31*10^{-5}$ | 14168.516 |

Tabela 6.2 – Valores da função, do erro e do tempo computacional, para diferentes funções $F(x)$, aplicando ED.

| $n$ | $F_3$ | Erro $\|\|r^T r\|\|$ | t(s) | $F_4$ | Erro $\|\|r^T r\|\|$ | t(s) |
|---|---|---|---|---|---|---|
| $1*10^1$ | $6.22*10^{-2}$ | $1.82*10^{-2}$ | 8.047 | $4.89*10^{-3}$ | $4.89*10^{-3}$ | 4.48 |
| $1*10^2$ | $3.69*10^{-2}$ | $3.43*10^{-2}$ | 6.094 | $2.74*10^{-2}$ | $2.74*10^{-2}$ | 7.41 |
| $1*10^3$ | $5.53*10^{-3}$ | $4.46*10^{-3}$ | 35.000 | $3.82*10^{-3}$ | $3.82*10^{-3}$ | 30.75 |
| $1*10^4$ | $6.42*10^{-4}$ | $4.34*10^{-4}$ | 426.562 | $4.21*10^{-4}$ | $4.21*10^{-4}$ | 460.50 |
| $5*10^4$ | $1.34*10^{-4}$ | $8.71*10^{-5}$ | 2399.766 | $1.46*10^{-4}$ | $1.46*10^{-4}$ | 3126.12 |
| $1*10^5$ | $6.89*10^{-5}$ | $4.35*10^{-5}$ | 5611.391 | $4.23*10^{-5}$ | $4.23*10^{-5}$ | 6322.77 |

Conforme mostrado nas Tabelas 6.1, 6.2 e 6.3, para a solução do sistema linear dado pela matriz $A = A1$, a Evolução Diferencial apresentou melhor performance com a utilização da função $F_1$ dada na Eq. (6.5).

Tabela 6.3 – Valores da função, do erro e do tempo computacional, para diferentes funções $F(x)$, aplicando ED.

| $n$ | $F_5$ | Erro $\|\|r^T r\|\|$ | t(s) | $F_6$ | Erro $\|\|r^T r\|\|$ | t(s) |
|---|---|---|---|---|---|---|
| $1*10^1$ | $7.31*10^{-3}$ | $7.31*10^{-3}$ | 4.61 | 0.0190 | $6.22*10^{-3}$ | 4.218 |
| $1*10^2$ | $2.93*10^{-2}$ | $2.93*10^{-2}$ | 9.27 | 0.1438 | $2.49*10^{-2}$ | 6.328 |
| $1*10^3$ | $4.01*10^{-3}$ | $4.01*10^{-3}$ | 32.66 | 0.1757 | $3.92*10^{-3}$ | 37.671 |
| $1*10^4$ | $4.23*10^{-4}$ | $4.23*10^{-4}$ | 424.58 | 0.1825 | $4.24*10^{-4}$ | 417.515 |
| $5*10^4$ | $2.36*10^{-4}$ | $2.36*10^{-4}$ | 3031.46 | 0.1834 | $8.56*10^{-5}$ | 2400.406 |
| $1*10^5$ | $4.31*10^{-5}$ | $4.31*10^{-5}$ | 5532.563 | 0.1840 | $4.31*10^{-5}$ | 5579.218 |

As tabelas 6.4 a 6.7 apresentam alguns resultados do sistema $Ax = b$, utilizando as diferentes matrizes de coeficientes dadas em (6.3), comparando a eficiência dos métodos, em relação ao erro e tempo computacional. Em todos os casos utilizou-se a função objetivo $F_1$, dada na Eq. (6.5), considerando o erro, também, como sendo a norma euclidiana do resíduo.

Tabela 6.4 – Comparação entre os métodos GMRES e ED para A1(ns, npd).

| $n$ | Erro $\|\|r^T r\|\|$ | | Tempo computacional t(s) | |
|---|---|---|---|---|
| | GMRES | ED | GMRES | ED |
| $1*10^1$ | $9.54*10^{-31}$ | $3.94*10^{-1}$ | 0.109 | 2.312 |
| $1*10^2$ | $2.13*10^{-5}$ | $3.47*10^{-2}$ | 0.111 | 3.766 |
| $1*10^3$ | $2.14*10^{-7}$ | $3.94*10^{-3}$ | 0.125 | 3.094 |
| $1*10^4$ | $2.14*10^{-9}$ | $4.24*10^{-4}$ | 0.250 | 9.250 |
| $5*10^4$ | $8.54*10^{-11}$ | $8.58*10^{-5}$ | 1.219 | 148.797 |
| $1*10^5$ | $2.14*10^{-11}$ | $4.29*10^{-5}$ | 3.375 | 917.578 |
| $2*10^5$ | $5.34*10^{-12}$ | $2.15*10^{-5}$ | 6.391 | 4657.156 |
| $4*10^5$ | $1.34*10^{-12}$ | $1.08*10^{-5}$ | 12.891 | 15211.063 |
| $6*10^5$ | $5.93*10^{-13}$ | $7.21*10^{-6}$ | 54.266 | 31405.359 |
| $8*10^5$ | $3.34*10^{-13}$ | $5.42*10^{-6}$ | 91.187 | 51181.328 |
| $1*10^6$ | $3.14*10^{-13}$ | $4.33*10^{-6}$ | 499.094 | 76232.433 |

Foram considerados valores de $n$, número de variáveis do sistema, variando de 10 a $10^6$, conforme apresentado nas Tabelas 6.4 a 6.7. Estas tabelas apresentam os resultados do erro e o tempo computacional, para cada matriz, e os métodos adequados a serem utilizados

em cada tipo. Observa-se que, em todos os casos, o método da Evolução Diferencial tem eficiência razoável, e como esperada, um tempo computacional bastante elevado. Vale a pena observar que, para as matrizes consideradas na Eq. (6.3), à medida que *n* cresce ocorre a diminuição do fator-condição $\kappa(A)=\lambda_{max}/\lambda_{min}$, dado na Eq. (4.11), o que justifica a diminuição do erro dos métodos com o crescimento da ordem das matrizes.

Tabela 6.5 – Comparação entre os métodos GMRES, BiCG e ED para A2(ns,pd)

| *n* | Erro $\|\|r^T r\|\|$ | | | Tempo computacional (s) | | |
|---|---|---|---|---|---|---|
| | GMRES | BiCG | ED | GMRES | BiCG | ED |
| $1*10^1$ | $1.58*10^{-30}$ | $1.87*10^{-14}$ | $3.32*10^{-2}$ | 0.094 | 0.047 | 1.968 |
| $1*10^2$ | $8.38*10^{-10}$ | $3.28*10^{-7}$ | $1.05*10^{-2}$ | 0.109 | 0.063 | 2.188 |
| $1*10^3$ | $8.43*10^{-12}$ | $2.16*10^{-7}$ | $1.58*10^{-3}$ | 0.125 | 0.047 | 2.390 |
| $1*10^4$ | $8.43*10^{-14}$ | $4.54*10^{-8}$ | $1.69*10^{-4}$ | 0.266 | 0.156 | 10.969 |
| $5*10^4$ | $4.36*10^{-15}$ | $2.58*10^{-8}$ | $3.46*10^{-5}$ | 1.187 | 0.469 | 170.485 |
| $1*10^5$ | $8.47*10^{-16}$ | $1.35*10^{-8}$ | $1.74*10^{-5}$ | 3.187 | 1.110 | 790.750 |
| $2*10^5$ | $2.10*10^{-16}$ | $1.33*10^{-8}$ | $8.69*10^{-6}$ | 7.032 | 2.218 | 1336.578 |
| $4*10^5$ | $5.25*10^{-17}$ | $6.39*10^{-9}$ | $4.35*10^{-6}$ | 13.906 | 4.297 | 4198.703 |
| $6*10^5$ | $6.83*10^{-17}$ | $4.28*10^{-9}$ | $2.91*10^{-6}$ | 24.984 | 7.110 | 17853.437 |
| $8*10^5$ | $3.84*10^{-17}$ | $7.82*10^{-9}$ | $2.18*10^{-6}$ | 44.391 | 12.562 | 49460.203 |
| $1*10^6$ | $2.45*10^{-17}$ | $6.24*10^{-9}$ | $1.74*10^{-6}$ | 248.860 | 19.656 | 80241.250 |

Tabela 6.6 – Comparação entre os métodos GMRES, MINRES e ED para *A3*(s,npd).

| *n* | Erro $\|\|r^T r\|\|$ | | | Tempo computacional (s) | | |
|---|---|---|---|---|---|---|
| | GMRES | MINRES | ED | GMRES | MINRES | ED |
| $1*10^1$ | $2.24*10^{-28}$ | $2.42*10^{-15}$ | $1.72*10^{-1}$ | 0.109 | 0.078 | 2.094 |
| $1*10^2$ | $3.93*10^{-6}$ | $1.86*10^{-14}$ | $2.25*10^{-2}$ | 0.141 | 0.078 | 2.063 |
| $1*10^3$ | $3.94*10^{-8}$ | $8.93*10^{-8}$ | $3.69*10^{-3}$ | 0.110 | 0.265 | 2.156 |
| $1*10^4$ | $3.94*10^{-10}$ | $1.27*10^{-7}$ | $4.05*10^{-4}$ | 0.265 | 1.610 | 10.515 |
| $5*10^4$ | $1.57*10^{-11}$ | $5.48*10^{-8}$ | $8.27*10^{-5}$ | 1.218 | 4.625 | 191.657 |
| $1*10^5$ | $3.94*10^{-12}$ | $3.94*10^{-8}$ | $4.14*10^{-5}$ | 3.187 | 9.563 | 823.985 |
| $2*10^5$ | $9.87*10^{-13}$ | $2.74*10^{-8}$ | $2.07*10^{-5}$ | 6.938 | 15.875 | 3302.875 |
| $4*10^5$ | $2.47*10^{-13}$ | $1.92*10^{-8}$ | $1.04*10^{-5}$ | 13.563 | 25.218 | 12376.875 |
| $6*10^5$ | $1.09*10^{-13}$ | $1.56*10^{-8}$ | $6.95*10^{-6}$ | 37.062 | 43.157 | 28334.953 |
| $8*10^5$ | $6.17*10^{-14}$ | $1.34*10^{-8}$ | $5.22*10^{-6}$ | 49.781 | 51.812 | 54265.188 |
| $1*10^6$ | $3.95*10^{-14}$ | $1.21*10^{-8}$ | $4.17*10^{-6}$ | 211.875 | 123.437 | 78226.375 |

Tabela 6.7 – Comparação entre os métodos GMRES, BiCG, MINRES, CG e ED com A4(s,pd).

| $n$ | Erro $\|\|r^T r\|\|$ | | | | |
|---|---|---|---|---|---|
| | GMRES | BiCG | MINRES | CG | ED |
| $1*10^1$ | $2.19*10^{-30}$ | 0 | $8.54*10^{-16}$ | 0 | $3.97*10^{-2}$ |
| $1*10^2$ | $3.19*10^{-14}$ | $1.91*10^{-7}$ | $1.83*10^{-7}$ | $1.94*10^{-7}$ | $1.23*10^{-2}$ |
| $1*10^3$ | $4.51*10^{-15}$ | $7.37*10^{-8}$ | $6.75*10^{-8}$ | $7.37*10^{-7}$ | $1.35*10^{-3}$ |
| $1*10^4$ | $6.29*10^{-16}$ | $2.58*10^{-8}$ | $2.52*10^{-8}$ | $2.63*10^{-8}$ | $1.47*10^{-4}$ |
| $5*10^4$ | $3.52*10^{-16}$ | $1.97*10^{-8}$ | $1.97*10^{-8}$ | $1.97*10^{-8}$ | $2.97*10^{-5}$ |
| $1*10^5$ | $8.79*10^{-17}$ | $9.75*10^{-9}$ | $9.49*10^{-9}$ | $9.75*10^{-9}$ | $1.49*10^{-5}$ |
| $2*10^5$ | $2.19*10^{-17}$ | $4.86*10^{-9}$ | $4.68*10^{-9}$ | $4.86*10^{-9}$ | $7.47*10^{-6}$ |
| $4*10^5$ | $7.63*10^{-17}$ | $9.07*10^{-9}$ | $8.74*10^{-9}$ | $9.07*10^{-9}$ | $3.74*10^{-6}$ |
| $6*10^5$ | $3.39*10^{-17}$ | $6.04*10^{-9}$ | $5.82*10^{-9}$ | $6.04*10^{-9}$ | $2.49*10^{-6}$ |
| $8*10^5$ | $1.91*10^{-17}$ | $4.53*10^{-9}$ | $4.37*10^{-9}$ | $4.53*10^{-9}$ | $1.87*10^{-6}$ |
| $1*10^6$ | $1.22*10^{-17}$ | $3.63*10^{-9}$ | $3.49*10^{-9}$ | $3.63*10^{-9}$ | $1.49*10^{-6}$ |
| $n$ | Tempo computacional (s) | | | | |
| | GMRES | BiCG | MINRES | CG | ED |
| $1*10^1$ | 0.110 | 0.046 | 0.063 | 0.031 | 0.984 |
| $1*10^2$ | 0.109 | 0.047 | 0.062 | 0.046 | 2.438 |
| $1*10^3$ | 0.110 | 0.062 | 0.063 | 0.047 | 3.187 |
| $1*10^4$ | 0.203 | 0.094 | 0.094 | 0.094 | 9.906 |
| $5*10^4$ | 0.657 | 0.265 | 0.235 | 0.218 | 179.500 |
| $1*10^5$ | 2.062 | 0.625 | 0.469 | 0.453 | 816.282 |
| $2*10^5$ | 3.407 | 1.218 | 0.907 | 0.953 | 3319.640 |
| $4*10^5$ | 5.110 | 2.109 | 1.500 | 1.750 | 18202.375 |
| $6*10^5$ | 7.610 | 3.093 | 2.297 | 2.516 | 31462.141 |
| $8*10^5$ | 21.969 | 8.657 | 3.203 | 3.125 | 50796.735 |
| $1*10^6$ | 67.563 | 54.016 | 4.563 | 3.968 | 93461.687 |

Os resultados mostram que a Evolução Diferencial pode ser considerada como uma opção viável, pois, apesar de ser um processo randômico, com elevado custo computacional, pode ser aplicada a qualquer tipo de matriz e sempre procura o mínimo global dos problemas de otimização. Analisando a Tab. 6.4 observa-se que a ED passa a ser uma nova ferramenta para o caso de matrizes não simétricas e não positiva-definidas, situação em que quase todos os outros métodos testados falharam.

## 6.2 Solução da Equação de Laplace Bi-dimensional

Nesta seção, apresenta-se o problema envolvendo uma equação de Laplace, visando comparar a solução analítica com as soluções numéricas.

Considere uma placa quadrada fina metálica de 1 metro de comprimento, e encontre a temperatura de equilíbrio $u(x,y)$ (após um tempo muito grande), onde $u(x,y)$ é a temperatura da barra na posição $x$ e $y$ como na Fig. 6.1:

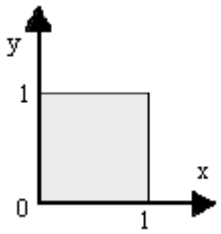



Figura 6.1: Placa Quadrada

Este problema é regido pela equação de Laplace e as condições de contorno a seguir:

$$\text{Equação de Laplace}: \ \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

$$\text{Condições de Dirichlet}: \begin{cases} u(x,0) = 100^0 C \\ u(0,y) = 100^0 C \\ u(x,1) = 0^0 C \\ u(1,y) = 0^0 C \end{cases} \tag{6.6}$$

### 6.1.1 Solução Analítica

Para a solução analítica do problema considerou-se o método da separação de variáveis e a obtenção das autofunções $X_n(x)$ do problema de Sturm-Liouville associado. Em seguida foi feita uma expansão da solução em série de autofunções.

Serão procuradas, para esta EDP, soluções da forma

$$u(x,y) = X(x)Y(y) \tag{6.7}$$

onde $X(x)$ e $Y(y)$ são funções de uma variável independente, a determinar.

A solução do problema (6.6) é decomposta na soma de duas funções soluções de problemas simplificados. Assim,

$$u(x,y) = u_1(x,y) + u_2(x,y) \tag{6.8}$$

onde as funções $u_1$ e $u_2$ são soluções, respectivamente, dos seguintes problemas:

$$\begin{cases} \dfrac{\partial^2 u_1}{\partial x^2} + \dfrac{\partial^2 u_1}{\partial y^2} = 0 \\ u_1(x,0) = 100^0 C \\ u_1(0,y) = 0^0 C \\ u_1(x,1) = 0^0 C \\ u_1(1,y) = 0^0 C \end{cases} \tag{6.9}$$

$$\begin{cases} \dfrac{\partial^2 u_2}{\partial x^2} + \dfrac{\partial^2 u_2}{\partial y^2} = 0 \\ u_2(x,0) = 0^0 C \\ u_2(0,y) = 100^0 C \\ u_2(x,1) = 0^0 C \\ u_2(1,y) = 0^0 C \end{cases} \tag{6.10}$$

Aplicando o método da separação de variáveis, em cada um dos problemas acima, determina-se funções da forma $X(x)Y(y)$, soluções do problema linear homogêneo.

Substituindo-se a expressão (6.6) na EDP (6.9), obtém-se:

$$X''(x)Y(x) + X(x)Y''(y) = 0, \tag{6.11}$$

A equação (6.11), após a divisão pelo produto $X(x)Y(y)$, conduz à Eq. (6.12), obtendo-se a separação de variáveis, uma em cada membro. Segue-se que esta equação só pode ser verdadeira se ambos os membros forem iguais a uma constante:

$$\frac{X''(x)}{X(x)} = -\frac{Y''(y)}{Y(y)} = \eta\,. \tag{6.12}$$

A expressão (6.12) representa duas equações, ambas dependendo da constante de separação η, que é ainda desconhecida. Resolvendo, primeiramente, a equação em *X(x),*

$$X''(x) - \eta\, X(x) = 0. \tag{6.13}$$

Exigindo, que a função $u_1(x,y) = X(x)Y(y)$ satisfaça, no intervalo $0 < x < 1$, as condições de contorno, a saber

$$u_1(0,y) = 0 \;\; e \;\; u_2(1,y) = 0, \;\; 0 < y < 1 \;\Rightarrow\; X(0) = 0 \;\; e \;\; X(1) = 0. \tag{6.14}$$

As equações (6.13) e (6.14) constituem um problema de Sturm-Liouville regular. Para o problema

$$X''(x) - \eta\, X(x) = 0; \qquad X(0) = 0; \qquad X(1) = 0. \tag{6.15}$$

Para calcular os autovalores e as autofunções correspondentes faz-se $X(x) = e^{ax}$, obtendo

$$\begin{aligned} &X''(x) - \eta\, X(x) = 0 \Rightarrow (a^2 - \eta)e^{ax} \Rightarrow a = \pm\sqrt{\eta}, \quad \mu < 0 \;\Rightarrow\; a = \pm i\mu; \\ &X(x) = C_1\cos(\mu x) + C_2\mathrm{sen}(\mu x) \end{aligned} \tag{6.16}$$

Das condições de contorno (6.12), obtém-se

$$\begin{aligned} &X(0) = C_1\cos(\mu 0) + C_2\mathrm{sen}(\mu 0) = 0 \;\Rightarrow\; C_1 = 0; \\ &X(1) = C_2\mathrm{sen}(\mu) = 0 \;\Rightarrow\; \mu = k\pi, \qquad\qquad k = 0,1,2,... \end{aligned} \tag{6.17}$$

Logo tem-se

$$\mu_n = n\pi \quad \Rightarrow \quad X_n(x) = C_n\mathrm{sen}(n\pi x) \quad \text{com} \quad n = 0,1,2,... \tag{6.18}$$

Fazendo $Y(y) = e^{ay}$, na equação $Y''(y) + \eta Y(y) = 0$, obtém-se

$$Y_n(y) = C_3 e^{-n\pi y} + C_4 e^{n\pi y}. \tag{6.19}$$

Usando a condição $Y(1) = 0$, resultando em:

$$C_3 e^{-n\pi} + C_4 e^{n\pi} = 0 \quad \Rightarrow \quad C_3 = Ce^{n\pi} \quad e \quad C_4 = -Ce^{-n\pi}. \tag{6.20}$$

Para alguma constante $C$, assim:

$$\begin{aligned} &Y_n(y) = Ce^{n\pi} e^{-n\pi y} - Ce^{-n\pi} e^{n\pi y} \quad \Rightarrow \quad Y_n(y) = Ce^{n\pi(1-y)} - Ce^{n\pi(y-1)} \Rightarrow \\ &\Rightarrow Y_n(y) = C\left( e^{n\pi(1-y)} - e^{-n\pi(1-y)} \right) \quad \Rightarrow \quad Y_n(y) = 2C \operatorname{senh}\left(n\pi(1-y)\right). \end{aligned} \tag{6.21}$$

Logo a solução particular é dada por:

$$u_1(x, y) = \operatorname{sen}\left(n\pi x\right) \operatorname{senh}\left(n\pi(1-y)\right). \tag{6.22}$$

Pelo principio da superposição, a solução geral é dada por:

$$u_1(x, y) = \sum_{n=1}^{\infty} \left[ K_{n,1} \operatorname{sen}\left(n\pi x\right) \operatorname{senh}\left(n\pi(1-y)\right) \right]. \tag{6.23}$$

Deseja-se agora determinar os coeficientes $K_{n,1}$ de modo a obter-se uma solução que satisfaça a condição de contorno $u_1(x,0) = 100^oC$, tem-se:

$$u_1(x, 0) = \sum_{n=1}^{\infty} \left[ K_{n,1} \operatorname{sen}\left(n\pi x\right) \operatorname{senh}\left(n\pi\right) \right] = 100. \tag{6.24}$$

Através da determinação dos coeficientes da série de Fourier de senos desta função, apresentada na Eq. (6.24), tem-se:

$$K_{n,1}\text{senh}(n\pi) = 2\int_0^1 100\ \text{sen}\left(n\pi x\right)dx \Rightarrow K_{n,1} = \frac{200\left(1\text{-cos}(n\pi)\right)}{n\pi\,\text{senh}(n\pi)}. \tag{6.25}$$

Portanto, a solução para $u_1(x,y)$ é dada por:

$$u_1(x,y) = 200\sum_{n=1}^{\infty}\left[\frac{1-\cos(n\pi)}{n\pi.\text{senh}(n\pi)}\ \text{sen}\left(n\pi x\right)\ \text{senh}\left(n\pi(1-y)\right)\right]. \tag{6.26}$$

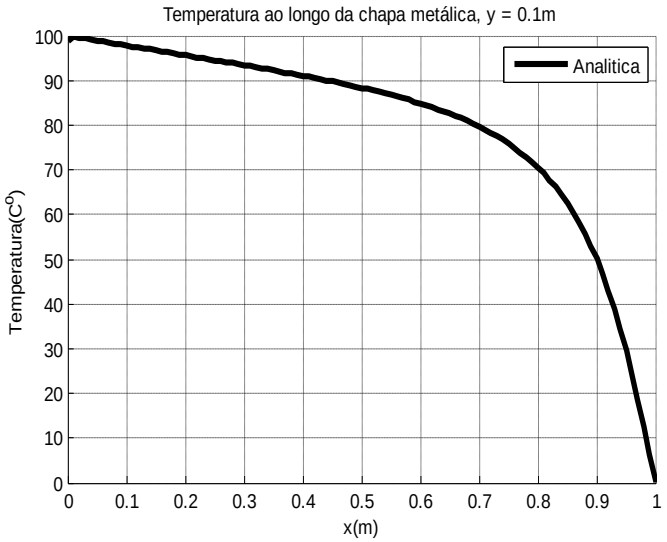
O mesmo procedimento é utilizado para obter a solução do problema (6.10), resultando em:

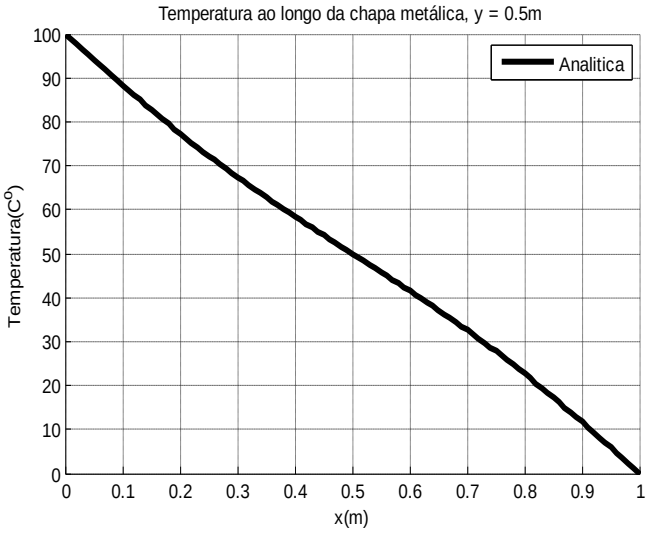
$$u_2(x,y) = 200\sum_{n=1}^{\infty}\frac{1-\cos(n\pi)}{n\pi.\text{senh}(n\pi)}\ \text{sen}\left(n\pi y\right)\ \text{senh}\left(n\pi(1-x)\right). \tag{6.27}$$

Logo a solução geral é:

$$u(x,y) = 200\sum_{n=1}^{\infty}\frac{1\text{-cos}(n\pi)}{n\pi\,\text{senh}(n\pi)}\left[\text{sen}\left(n\pi x\right)\ \text{senh}\left(n\pi(1-y)\right)+\text{sen}\left(n\pi y\right)\ \text{senh}\left(n\pi(1-x)\right)\right]. \tag{6.28}$$

Apresentam-se, na Fig. 6.2, gráficos dos valores de $u(x,y)$ para alguns valores fixos de $y$, início, meio e fim. A Fig. 6.3 apresenta gráficos dos valores de $u(x,y)$ para alguns valores fixos de $x$.



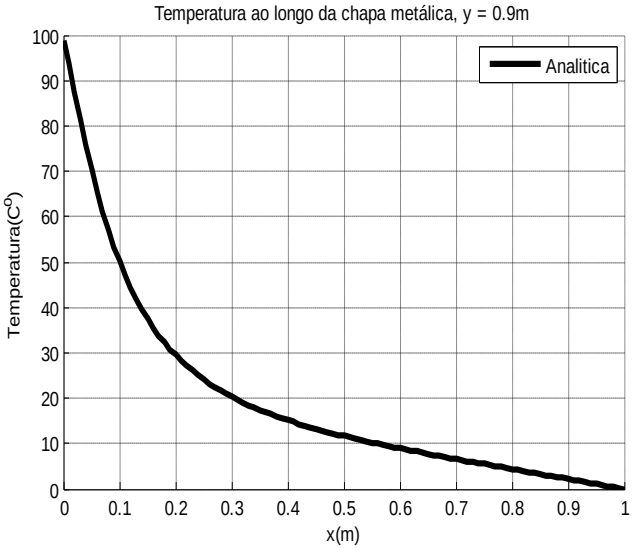




Figura 6.2: Temperatura da chapa para valores fixos de *y*

Nota-se, devido à simetria do problema, que não há diferença entre fixar *x* ou *y*.

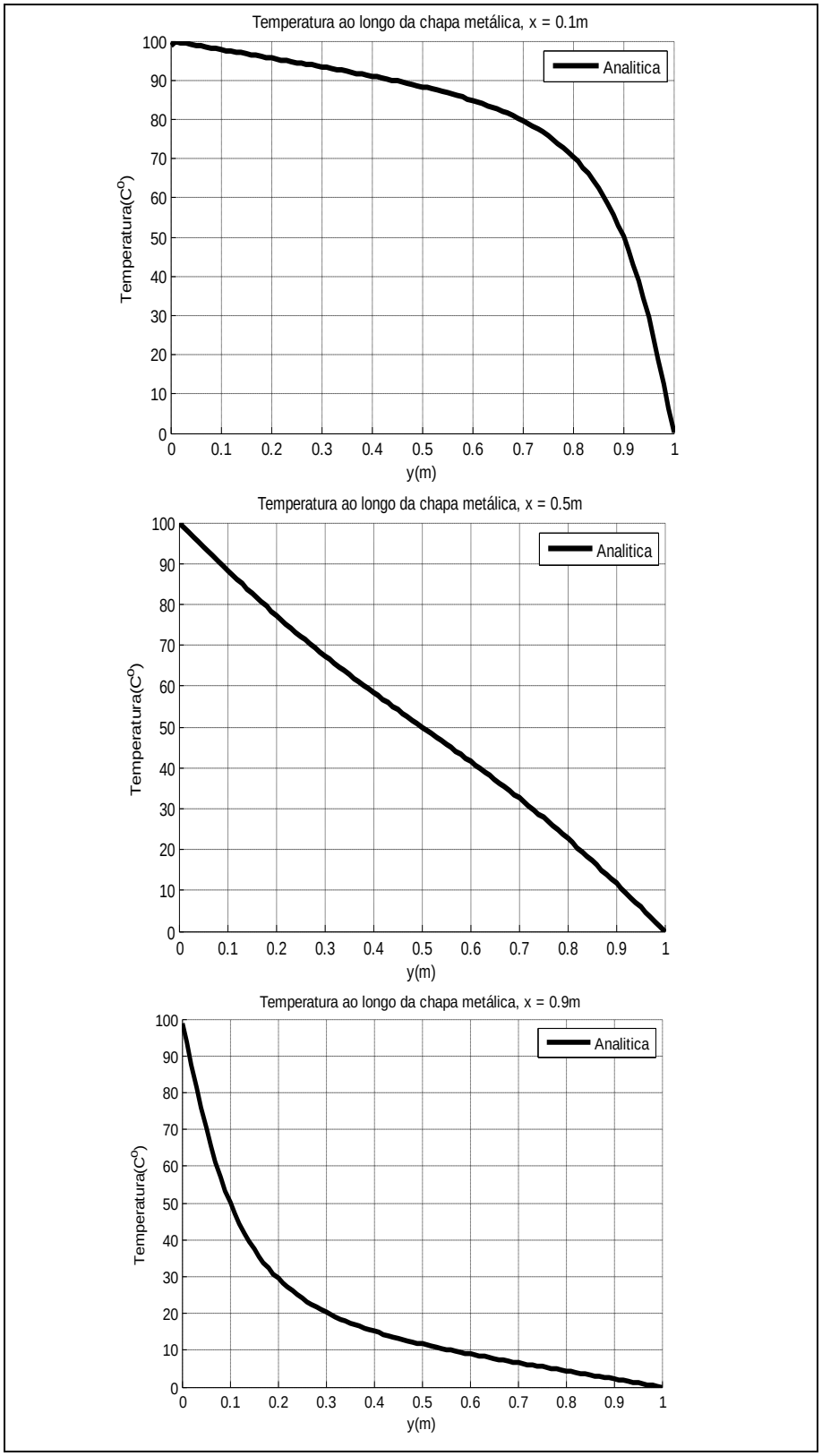



Figura 6.3: Temperatura da chapa para valores fixos de *x*

### 6.1.2 Solução Numérica:

A seguir apresenta-se a solução numérica deste problema. Aqui se utilizou, para discretização da placa, o método das diferenças finitas, devido ao fato de com este método a matriz que se obtém, para o sistema linear, é uma matriz esparsa, penta-diagonal, simétrica e positiva-definida, o que torna a aplicação dos métodos estudados mais eficiente.

Aplicando, na equação diferencial, as aproximações de diferenças finitas com passos: $\Delta x = \Delta y = 0{,}25$ metros, e trocando o índice *i* por *j* para a derivada em relação à *y*, tem-se:

$$\begin{aligned}\frac{\partial^2 u_{i,j}}{\partial x^2} &= \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} \\ \frac{\partial^2 u_{i,j}}{\partial y^2} &= \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{k^2} = \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2}\end{aligned} \tag{6.29}$$

Ou seja,

$$\frac{\partial^2 u_{i,j}}{\partial x^2} + \frac{\partial^2 u_{i,j}}{\partial y^2} = 0 \;\Rightarrow\; \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} = 0 \tag{6.30}$$

Considerando os incrementos $\Delta x = \Delta y$, e isolando o termo $u_{i,j}$, tem-se:

$$u_{i,j} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}}{4} \tag{6.31}$$

A Eq. 6.31 fornece a temperatura na posição (*i, j*), em função da média aritmética das temperaturas na borda inferior, superior, direita e esquerda da placa.

Equações deste tipo, onde o novo valor é calculado em função de valores desconhecidos, são denominadas implícitas. Assim, escreve-se um sistema linear, onde a Eq.(6.31) é expandida para cada ponto do interior da placa, seguindo a numeração dada na Fig. 6.4.

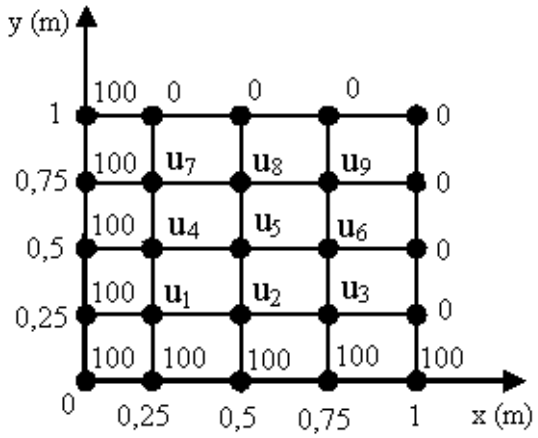



Figura 6.4: Malha da discretização da placa.

Expandindo a Eq. (6.31) para cada ponto do interior, de $u_1$ a $u_9$, obtêm-se as seguintes equações:

$$\begin{cases} u_1 = \dfrac{u_2 + 100 + u_4 + 100}{4};\ u_2 = \dfrac{u_3 + u_1 + u_5 + 100}{4};\ u_3 = \dfrac{0 + u_2 + u_6 + 100}{4} \\ u_4 = \dfrac{u_5 + 100 + u_7 + u_1}{4};\ u_5 = \dfrac{u_6 + u_4 + u_8 + u_2}{4};\ u_6 = \dfrac{0 + u_5 + u_9 + u_3}{4} \\ u_7 = \dfrac{u_8 + 100 + 0 + u_4}{4};\ u_8 = \dfrac{u_9 + u_7 + 0 + u_5}{4};\ u_9 = \dfrac{0 + u_8 + 0 + u_6}{4} \end{cases} \tag{6.32}$$

O que conduz ao seguinte sistema linear:

$$\begin{cases} 4u_1 - u_2 - u_4 = 200 \\ -u_1 + 4u_2 - u_3 - u_5 = 100 \\ -u_2 + 4u_3 - u_6 = 100 \\ -u_1 + 4u_4 - u_5 - u_7 = 100 \\ -u_2 - u_4 + 4u_5 - u_6 - u_8 = 0 \\ -u_3 - u_5 + 4u_6 - u_9 = 0 \\ -u_4 + 4u_7 - u_8 = 100 \\ -u_5 - u_7 + 4u_8 - u_9 = 0 \\ -u_6 - u_8 + 4u_9 = 0 \end{cases} \tag{6.33}$$

Escrevendo o sistema linear (6.33), na sua forma matricial $Ax = b$, resulta:

$$A=\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix}; \quad x=\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \end{bmatrix}; \quad b=\begin{bmatrix} 200 \\ 100 \\ 100 \\ 100 \\ 0 \\ 0 \\ 100 \\ 0 \\ 0 \end{bmatrix} \tag{6.34}$$

Cuja expressão geral de *A* é dada por:

$$A(i,j)=\begin{cases} -1 & se \quad i=j+(nx-1) \\ -1 & se \quad i=j+1 \\ 4 & se \quad i=j \\ -1 & se \quad j=i+1 \\ -1 & se \quad j=i+(nx-1) \end{cases} \quad \begin{array}{l} \text{para} \quad i=1,\cdots,n \quad e \quad j=1,\cdots,n; \\ n=(n-1)*(nx-1) \end{array} \tag{6.35}$$

$$A(j*i,\ j*i+1)=0 \quad \text{para} \quad j=1,\cdots,nx-2 \quad e \quad i=nx-1$$

Onde *nx* é o número de subdivisões do intervalo [0, 1].

Resolvendo o sistema pelos métodos ED, PCG, MINRES, BiCG, obtém-se os valores apresentados na Tabela 6.8.

Tabela 6.8- Valores numéricos da temperatura obtidos pelos Métodos ED, PCG, MINRES e BiCG, para *nx* = 4

| | ED | PCG | MINRES | BiCG | Analítica |
|---|---|---|---|---|---|
| $u_1$ | 85.7138 | 85.7143 | 85.7143 | 85.7143 | 85.7143 |
| $u_2$ | 71.4277 | 71.4286 | 71.4286 | 71.4286 | 71.4286 |
| $u_3$ | 49.9980 | 50.0000 | 50.0000 | 50.0000 | 50.0000 |
| $u_4$ | 71.4261 | 71.4286 | 71.4286 | 71.4286 | 71.4286 |
| $u_5$ | 49.9982 | 50.0000 | 50.0000 | 50.0000 | 50.0000 |
| $u_6$ | 28.5738 | 28.5714 | 28.5714 | 28.5714 | 28.5714 |
| $u_7$ | 49.9991 | 50.0000 | 50.0000 | 50.0000 | 50.0000 |
| $u_8$ | 28.5710 | 28.5714 | 28.5714 | 28.5714 | 28.5714 |
| $u_9$ | 14.2869 | 14.2857 | 14.2857 | 14.2857 | 14.2857 |

Na Fig. 6.5, apresenta-se um gráfico comparativo dos valores numéricos da temperatura obtidos pelos métodos ED, PCG, MINRES e BiCG, dados na Tabela 6.8.

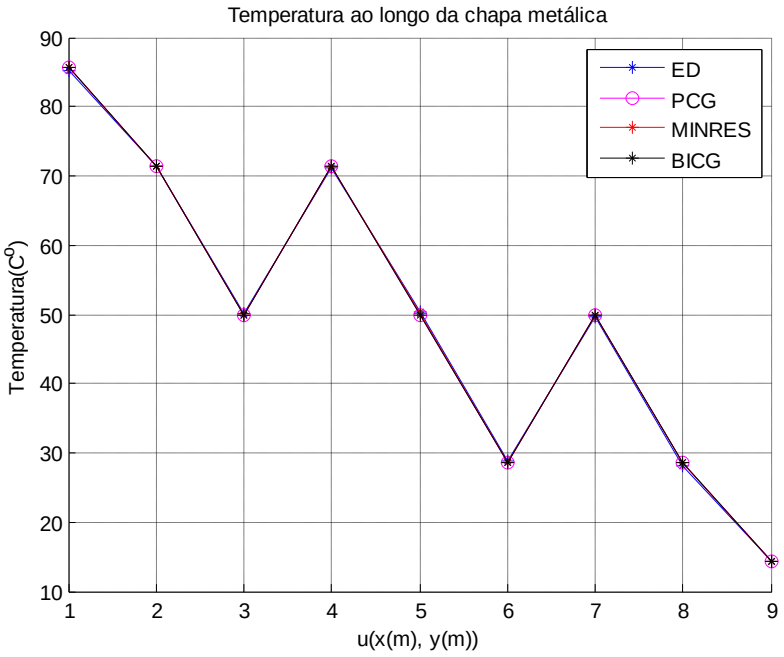



Figura 6.5: Gráfico dos valores numéricos da temperatura

Observa-se a simetria em relação à diagonal principal da placa, e constata-se que os valores da temperatura, nos vértices da placa, não foram considerados nesta aproximação.

Apresentam-se, a seguir, os gráficos da solução analítica e soluções numéricas usando o Método da Evolução Diferencial (ED), Método dos Gradientes Conjugados Pré-Condicionado (PCG), Método dos Resíduos Mínimos (MINRES) e o Método dos Gradientes Bi-Conjugados (BiCG). Foi adotado o número de subintervalos variando de $nx = 11$ a $nx = 72$, implicando que a dimensão da matriz varia de $n = 100$ a $n = 5041$.

Nas Fig. 6.6 a 6.9, pode-se observar que as soluções numéricas apresentam pouca diferença entre os métodos utilizados. Além disso, nota-se que quanto mais subintervalos são considerados, mais as soluções numéricas se aproximam da solução analítica.

As soluções serão sempre consideradas em três posições distintas da placa: duas próximas às extremidades e uma na posição central.

No caso da Fig. 6.6, quando se considera um pequeno número de nós, as soluções numéricas apresentadas pelos quatro métodos estudados aparentemente diferem da solução

analítica. Nas figuras seguintes, à medida que se aumenta o número de nós, as soluções numéricas tornam-se similares à solução analítica.

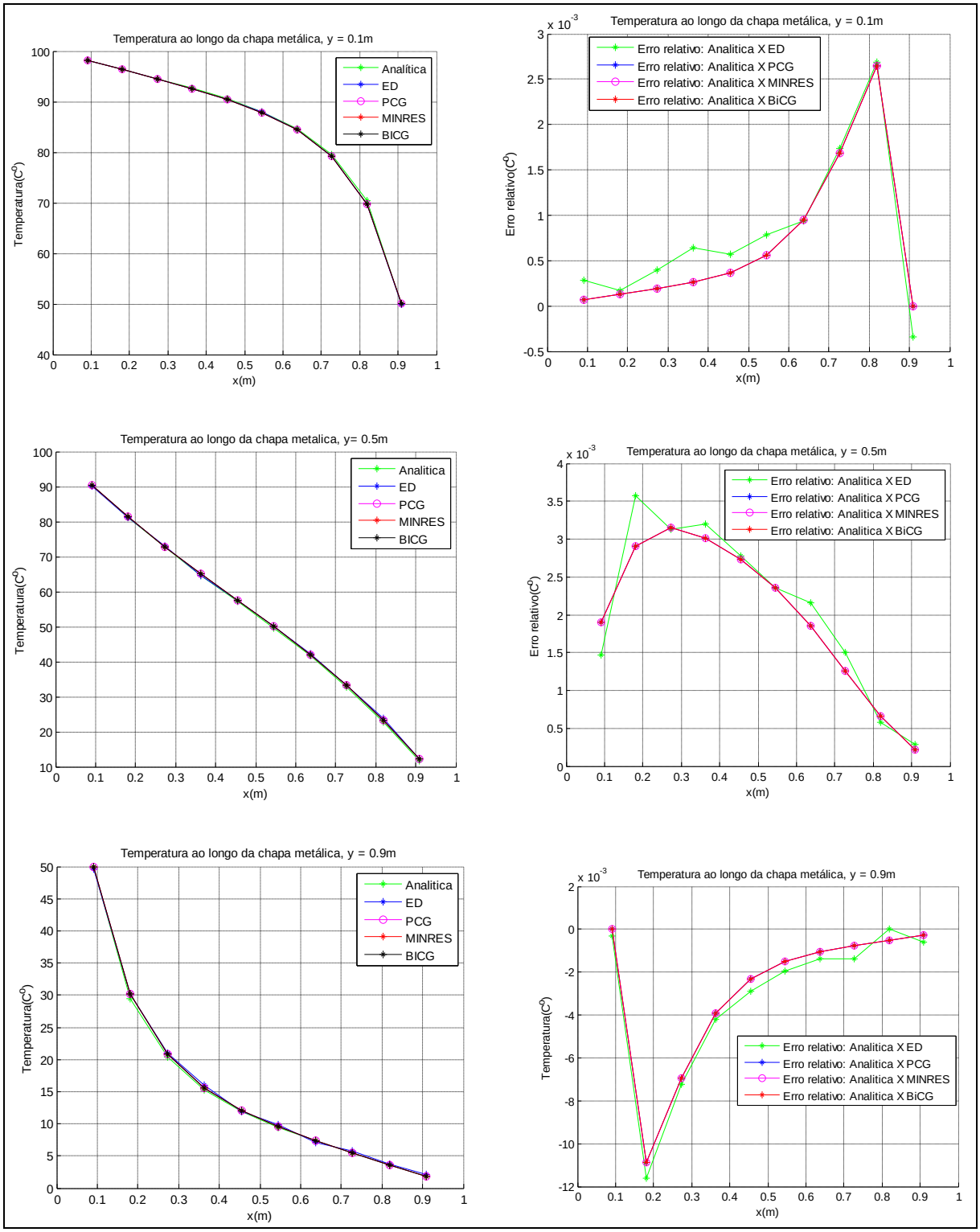



Figura 6.6: Gráfico comparativo entre as soluções analítica e numéricas obtidas pelos métodos ED, PCG, MINRES e BiCG, com n = 100 nós, considerando y = 0.1m, y = 0.5m e y = 0.9m, apresentando o erro relativo entre elas.

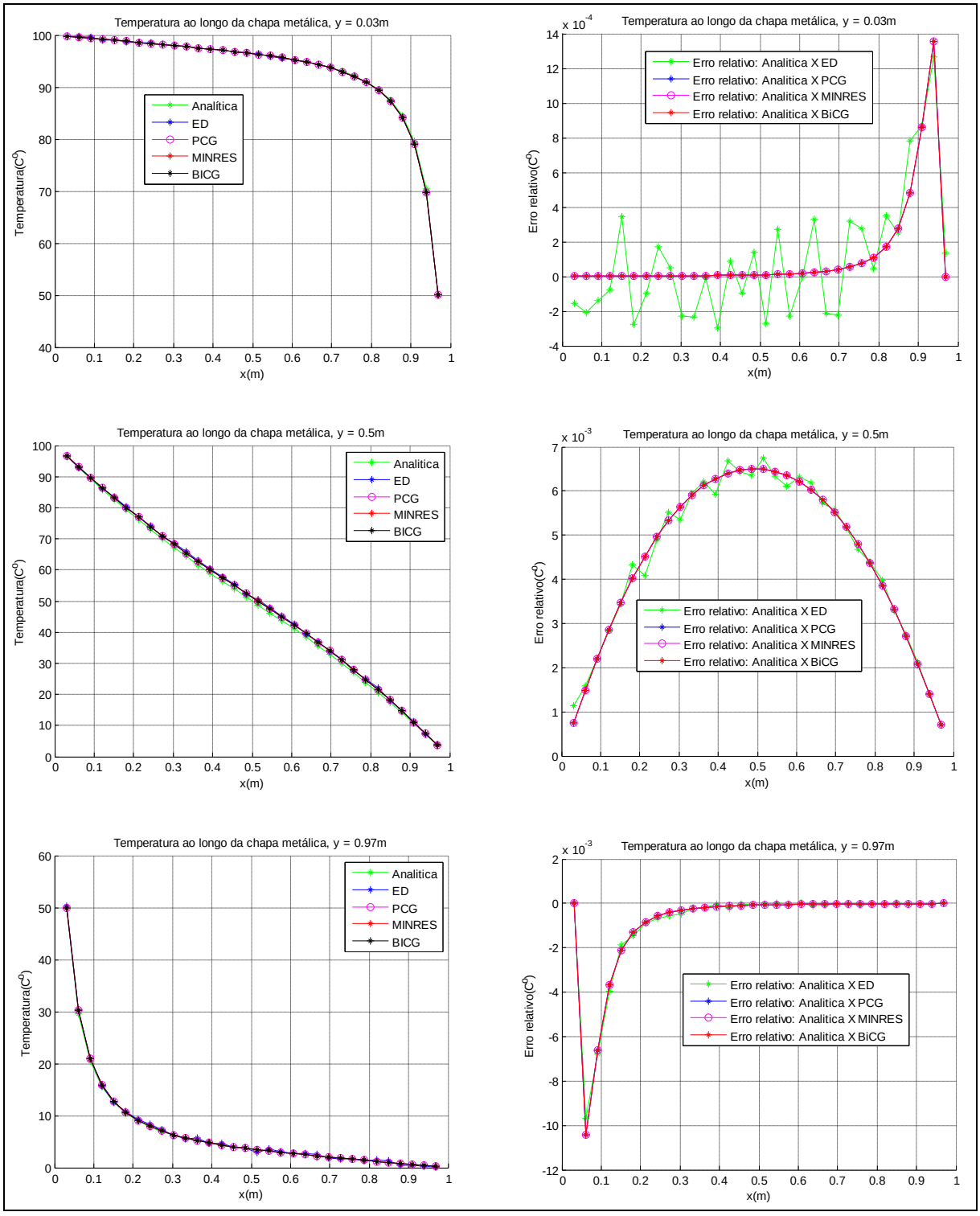



Figura 6.7: Gráfico comparativo entre as soluções analítica e numéricas obtidas pelos métodos ED, PCG, MINRES e BiCG, com n = 1024 nós, considerando y = 0.03m, y = 0.5m e y= 0.97m, apresentando o erro relativo entre elas.

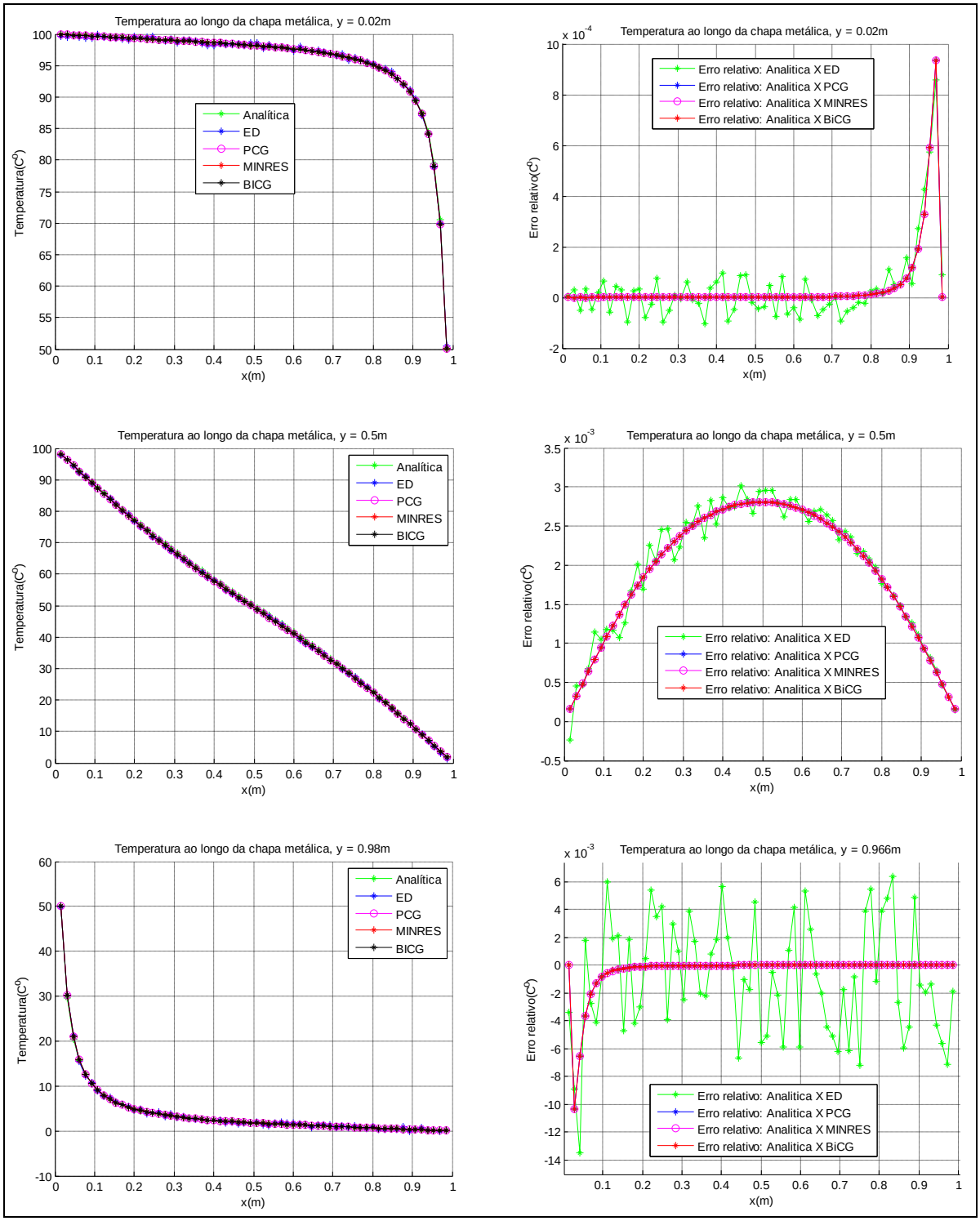



Figura 6.8: Gráfico comparativo entre as soluções analítica e numéricas obtidas pelos métodos ED, PCG, MINRES e BiCG, com n = 4096 nós, considerando y = 0.02m, y = 0.5m e y = 0.98m, apresentando o erro relativo entre elas.

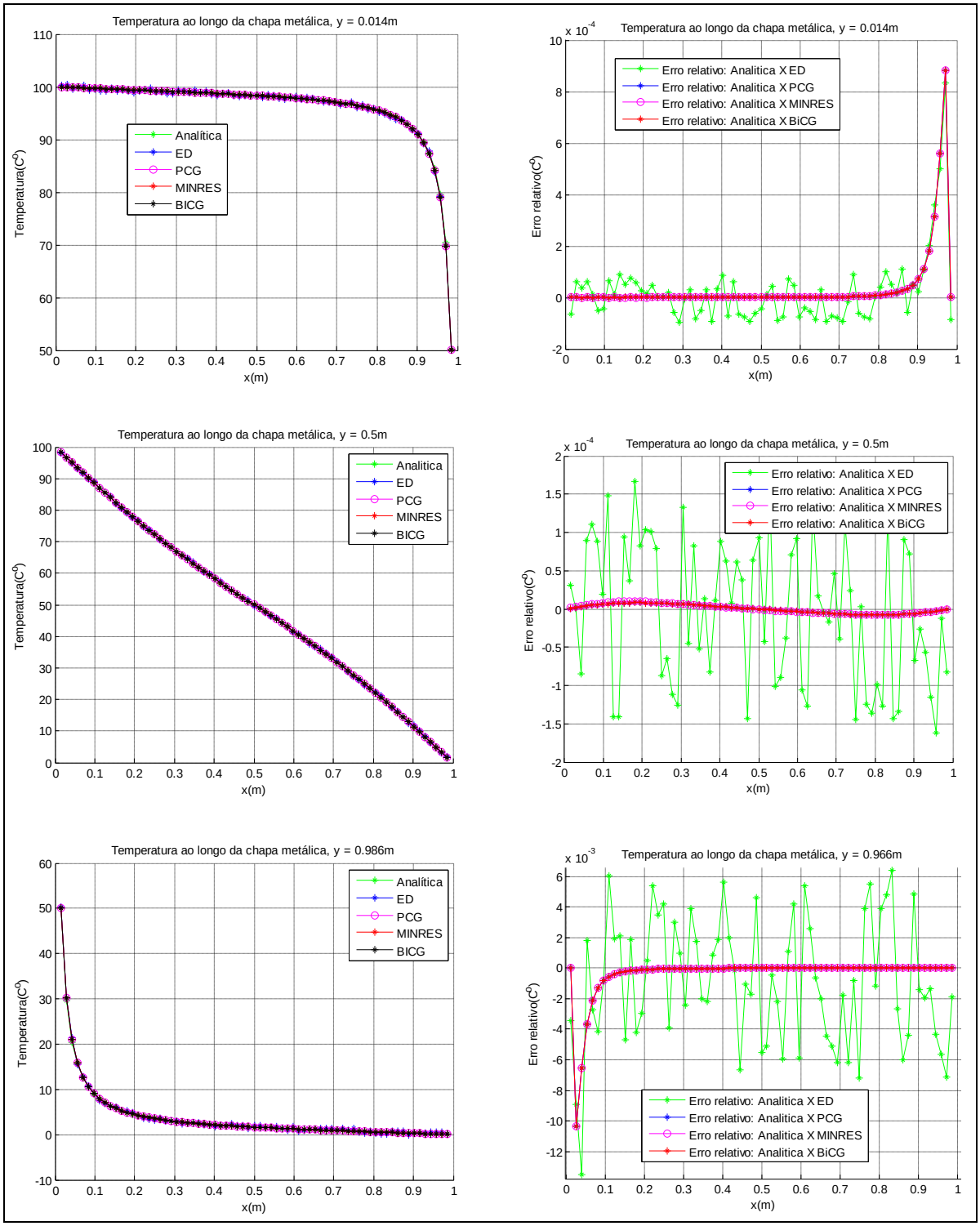



Figura 6.9: Gráfico comparativo entre as soluções analítica e numéricas obtidas pelos métodos ED, PCG, MINRES e BiCG, com n = 5041 nós, considerando y = 0.014m, y = 0.5m e y= 0.986m, apresentando o erro relativo entre elas.

Ao se considerar as Figs. 6.8 e 6.9, observa-se que as soluções numéricas obtidas pelos métodos PCG, MINRES, BiCG e ED estão muito próximas da solução analítica, apresentam

um erro relativo na casa de $10^{-4}$. Isto permite concluir que estes métodos são todos muito eficientes para o problema em estudo. Percebe-se que o método da Evolução Diferencial, além de poder ser aplicado a todos os tipos de matrizes de coeficientes de um sistema linear, é também eficiente quando se trata de resolver um problema físico.

A Tabela 6.9 apresenta uma comparação do tempo computacional necessário para que os métodos considerados calculem a solução numérica, do problema estudado nesta seção. Nota-se que o tempo computacional exigido em cada um dos métodos iterativos se diferencia bastante. Para este problema em estudo, o método da Evolução Diferencial apresentou bom resultado e um tempo computacional competitivo em relação aos demais.

Tabela 6.9 – Comparação entre os métodos BiCG, MINRES, PCG e ED.

| *n* | Tempo Computacional em segundos | | | |
|---|---|---|---|---|
| | BiCG | MINRES | PCG | ED |
| 100 | 0.078 | 0.047 | 0.093 | 0.094 |
| 1024 | 1.375 | 0.500 | 0.906 | 0.500 |
| 4096 | 56.359 | 18.000 | 36.016 | 9.031 |
| 5041 | 78.468 | 26.329 | 52.641 | 11.469 |

É conveniente observar que, para este problema, tomaram-se números pequenos de nós, pois os resultados já estavam próximos da solução analítica. Além disso, não foi utilizado nenhum algoritmo para compactar a matriz dos coeficientes. Desta forma, exigiu-se muita memória para armazenar a matriz. Este fato pode justificar a baixa performance dos métodos iterativos.

## 6.3 O Problema de Identificação Indireta de Forças Dinâmicas

No âmbito da dinâmica estrutural, o conhecimento preciso da distribuição e magnitude das forças externas é de primordial importância, podendo proporcionar maior confiança nas simulações numéricas baseadas em modelos analíticos. Além disso, o conhecimento das forças excitadoras é imprescindível no projeto de sistemas de isolamento e proteção contra danos provocados por carregamentos dinâmicos.

Em certas situações, as forças excitadoras podem ser medidas diretamente com o auxílio de transdutores de força (células de carga). Entretanto, nos casos em que são tratadas estruturas leves, de pequeno porte, a introdução destes dispositivos pode alterar as características estruturais do sistema, resultando em predições incorretas das forças. Em outras situações, os locais em que as forças são aplicadas podem não ser acessíveis à instrumentação. Dificuldades para a medição direta também existem quando as forças são distribuídas em diversas localizações espaciais, sendo necessário, neste caso, considerável montante de instrumentação. Em tais casos, o uso de técnicas de identificação indireta de forças (ou técnicas de reconstrução de forças) apresenta-se como uma alternativa interessante.

A identificação indireta de forças busca estimar as forças excitadoras (entradas), a partir das respostas dinâmicas medidas da estrutura (saída) e de um modelo matemático do sistema. Em essência, a estrutura instrumentada torna-se ela mesma um transdutor de carga. Vale lembrar que as respostas dinâmicas podem ser facilmente obtidas através de procedimentos usuais de ensaios dinâmicos. Diversos tipos de modelos - obtidos tanto por técnicas de modelagem analítica, quanto experimental - podem ser empregados, tais como: funções de resposta em freqüência (F.R.F.s), funções de resposta ao impulso (F.R.I.s), auto-soluções (freqüências naturais, fatores de amortecimento modais, massas generalizadas e componentes de modos de vibração) ou ainda diversos tipos de modelos estruturais representados por matrizes de inércia, rigidez e amortecimento.

Existem dois grupos principais nos quais os métodos de identificação de forças estão inseridos: os métodos que operam no domínio das freqüências, que buscam determinar os espectros das forcas excitadoras e; o os métodos fundamentados no domínio do tempo, que objetivam caracterizar as variações das excitações externas em função do tempo. Uma revisão abrangente de alguns dos métodos de identificação de forças documentados na literatura é apresentada no trabalho de STEVENS (1987), notando-se que a maioria dos estudos realizados enfoca os métodos no domínio da freqüência, ao passo que os métodos operando no domínio do tempo têm merecido menor atenção. A escolha dentre os dois tipos de métodos deve ser feita levando-se em conta a natureza dos dados experimentais disponíveis e as especificidades da aplicação pretendida. Os diversos estudos já realizados revelam que, em geral, ambos os tipos de métodos apresentam suas vantagens e inconvenientes.

Conforme ilustrado na Fig. 6.10, em princípio, o procedimento de identificação indireta de forças é simples, visto que basta inverter o problema direto, que consiste em calcular a resposta (saída) do sistema a partir do conhecimento da força excitadora (entrada) e

da função de transferência (modelo), que relaciona a excitação e a resposta. Portanto, mediante a resolução de um problema inverso, as forças são estimadas a partir do conhecimento prévio da resposta do sistema e de sua função de transferência.

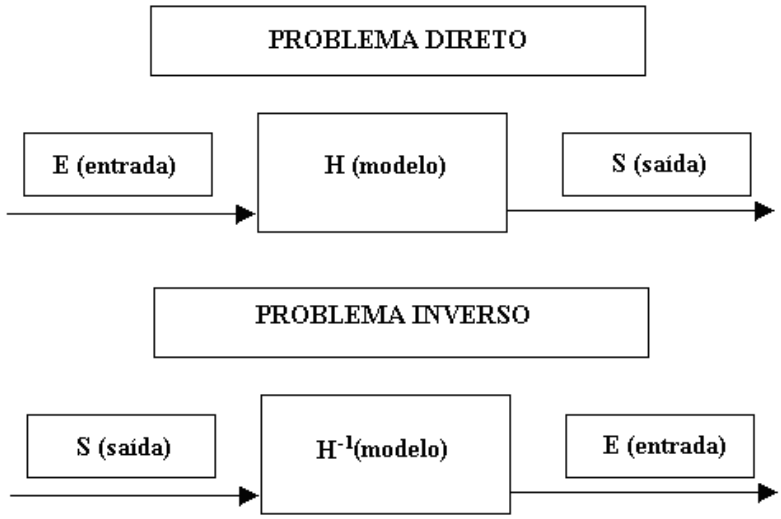



Figura 6.10 - Representação esquemática dos problemas direto e inverso da dinâmica estrutural.

Embora simples na sua essência, o problema de identificação de forças a partir das respostas dinâmicas é considerado um problema de difícil implementação prática, estando as dificuldades ligadas aos seguintes fatores principais:

- Na grande maioria das vezes o problema de identificação revela-se altamente mal condicionado sob ponto de vista numérico. Este mal condicionamento é traduzido pela quase singularidade do operador H, que deve ser invertido para a resolução do problema inverso (conforme Figura 6.10). Do ponto de vista computacional, o mal condicionamento numérico se revela através de uma extrema sensibilidade dos resultados de identificação à perturbação presente nos dados utilizados (HUNT, 1972).
- Na grande maioria dos casos práticos, o modelo H deve ser construído a partir de dados analíticos ou experimentais incompletos, que são inevitavelmente afetados por incertezas de modelagem e ruídos de medição. Além disso, por limitações de natureza prática, apenas um número reduzido de coordenadas são geralmente instrumentadas para aquisição das respostas dinâmicas.

Se várias forças devem ser identificadas simultaneamente, o problema de mal condicionamento mostra-se ainda mais severo. O sucesso da técnica de identificação fica, neste caso, condicionado pela observabilidade (possibilidade de estimar as variáveis de estado que não podem ser medidas, através do conhecimento das variáveis de entrada e saída do sistema em qualquer instante) das respostas causadas por cada uma das forças separadamente. Com efeito, em algumas situações, as contribuições às respostas dinâmicas de cada uma das forças podem ter distribuições espaciais bastante semelhantes, resultando na quase singularidade do operador que representa as funções de transferência. Quando isto acontece, é muito difícil separar as contribuições causadas por cada força, assim, pequenas variações (erros) na medição das respostas tendem a causar grandes variações nas amplitudes das forças identificadas. Foi sugerido por HILLARY e EWINS (1984) que, em certas situações, dificuldades deste tipo podem ser reduzidas pela inclusão de posições de medição adicionais ou ainda movendo estas posições de modo a promover um maior enriquecimento da base de dados experimentais utilizada.

Dada a grande importância do tratamento do mal condicionamento numérico que caracteriza, de modo geral, os problemas inversos, um grande esforço tem sido empreendido visando o desenvolvimento de algoritmos de estimação numericamente estáveis. Algumas estratégias têm sido propostas, baseadas na formulação de um novo problema associado ao problema original, sendo, porém, bem condicionado. Este procedimento é conhecido como regularização do problema mal condicionado. As mais conhecidas estratégias de regularização utilizadas são: regularização baseada na Decomposição em Valores Singulares (DVS) (MAIA, 1989), o método de regularização de Tikhonov (TIKHONOV e ARSENIN, 1977) e as técnicas baseadas na aplicação de restrições determinísticas a priori, utilizando funções de penalidade e multiplicadores de Lagrange (THOMAS e PROST, 1991).

O objetivo desta seção é aplicar as técnicas de otimização evolutivas, Evolução Diferencial e Algoritmos Genéticos na solução do sistema linear obtido através do problema de identificação indireta de forças dinâmicas. Para isso, inicialmente, são considerados problemas teóricos de identificação de forças. No primeiro caso, o sistema é excitado por uma força harmônica. São feitos vários testes usando a ED, o AG e a técnica iterativa GMRES; as respostas obtidas são comparadas. No segundo caso, o mesmo sistema é excitado através da aplicação de um ruído branco. Na última subseção é resolvido um problema real de identificação indireta de forças, a partir de dados experimentais.

### 6.3.1 Sistema excitado por uma força harmônica.

A metodologia da estimativa indireta de forças no domínio de freqüências se baseia na inversão direta da matriz FRF e seu respectivo produto com o espectro da vibração medida. Devido o mal condicionamento da matriz FRF, normalmente é utilizada sua pseudo-inversa de acordo com a Eq. (6.35).

$$\{F(\varpi)\} = [H(\varpi)]^{+} \{X(\varpi)\} \tag{6.35}$$

Na Eq. (6.35), $\{X(\varpi)\} \in \mathbb{R}^n$ é o vetor amplitude da vibração medida na resposta, $[H(\varpi)]^{+} \in \mathbb{R}^{n\times n}$ é a pseudo-inversa da matriz FRFs e $\{F(\varpi)\} \in \mathbb{R}^n$ é o espectro das forças estimadas.

A pseudo-inversa, pela abordagem dos quadrados mínimos, é dada pela Eq.(6.36).

$$[H(\varpi)]^{+} = ([H(\varpi)]^{T} [H(\varpi)])^{-1} [H(\varpi)]^{T} \tag{6.36}$$

Dois problemas ocorrem quando a freqüência abordada é usada:

- Grandes erros podem ser observados em regiões de ressonância (SILVA, 2000).
- O método não se pode ser utilizado em aplicações em tempo real.

A técnica do tempo é menos sensível a problemas de ressonâncias (SILVA, 2000) e pode ser usada em tempo real também. Baseia-se na integral de convolução de Dhuramel dada pela Eq. (6.37), onde $h(t)$ é a FRI, $f(t)$ é a força excitadora em função do tempo $t$ e $x(t)$ é a amplitude da vibração em resposta no tempo $t$. Se a $h(t)$ é a FRI da aceleração, então a resposta em aceleração pode ser calculada por:

$$x(t) = \int_{-\infty}^{\infty} h(t-\tau) f(\tau) d\tau \tag{6.37}$$

Para sistemas mecânicos fisicamente realizáveis a FRI é nula para todos os valores de $t$ menores que *0* e a Eq. (6.37) que pode ser reescrita como:

$$x(t) = \int_0^t h(t-\tau) f(\tau) d\tau \tag{6.38}$$

Considerando a discretização $\Delta t$ do tempo de amostragem da Eq. (6.38) pode ser aproximada por:

$$x(k\Delta t) = \sum_{i=0}^{k} h[(k-i)\Delta t] f(i\Delta t)\Delta t \text{ , com k = 0, 1, 2, …, p-1} \tag{6.39}$$

Em notação vetorial, a Eq. (6.39) pode ser reescrita como:

$$x_k = \sum_{i=0}^{k} h_{k-i} f_i \text{ , com k = 0, 1, 2, …, p-1} \tag{6.40}$$

A expansão da Eq. (6.40) resulta em:

$$\begin{aligned}
&Para\ k = 0: \ x_0 = h_0 f_0 \\
&Para\ k = 1: \ x_1 = h_1 f_0 + h_0 f_1 \\
&Para\ k = 2: \ x_2 = h_1 f_0 + h_1 f_1 + h_0 f_2 \\
&\vdots \\
&Para\ k = p-1: \ x_{p-1} = h_{p-1} f_0 + h_{p-2} f_1 + \cdots + h_0 f_{p-1}
\end{aligned} \tag{6.41}$$

Em notação matricial, a Eq. (6.41) é dada por:

$$\{X\} = [H]\ \{F\} \tag{6.42}$$

Onde

$$\{X\} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{Bmatrix} \quad ; \quad \{F\} = \begin{Bmatrix} f_1 \\ f_2 \\ \vdots \\ f_p \end{Bmatrix} \quad e \quad [H] = \begin{bmatrix} h_0 & 0 & \cdots & 0 \\ h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ h_{p-1} & h_{p-2} & \cdots & h_0 \end{bmatrix}$$

A partir daí resolve-se este sistema de equações lineares, para a identificação do valor da força excitadora.

Neste trabalho, o objetivo é obter a solução usando alguma técnica que não exija mudanças na matriz *H*, a fim de melhor condicioná-la. Conforme descrito na literatura em geral, o pré-condicionamento da matriz *H*, que é um procedimento usual, mas bastante oneroso. Para testar as técnicas de otimização na identificação de forças dinâmicas, utilizando a formulação dada pelas Eqs. (6.41) e (6.42), construiu-se o seguinte problema teórico:

$$h(i,j) = A_0\left(e^{-\xi\varpi t(j+1)}\right)\sin\left(\sqrt{1-\xi^2}\varpi t(j+1)\right);$$
$$X = HF \qquad i, j = 1, 2, ..., N-1, \text{ com } j \le i. \tag{6.43}$$

Considerando $A_0 = 1e^{-3}$; $\xi = 0.08$; $\varpi = 2\pi 500$; $dt = 1/2048$; $t \in [0, 1]$ e N=comprimento(t). Neste problema a força *F* é dada por:

$$F = 5\sin(2\pi 300t) + 1.5\cos(2\pi 300t) \tag{6.44}$$

Conhecendo-se a matriz [*H*] e o vetor {*X*}, resolve-se o sistema linear *HF=X*, usando Evolução Diferencial (ED), Algoritmos Genéticos (AG) e a técnica não-estacionária GMRES. A seguir, é feita uma comparação entre a solução analítica (dada por *F*) e as soluções numéricas.

A Fig. 6.11 apresenta a comparação entre as soluções analíticas e numéricas obtidas pelos métodos ED e GMRES. Na Fig. 6.12 é apresentada a comparação entre a solução analítica e a solução numérica obtida pelo AG. A solução analítica é facilmente obtida uma vez que a força excitadora é fornecida pela Eq. (6.44).

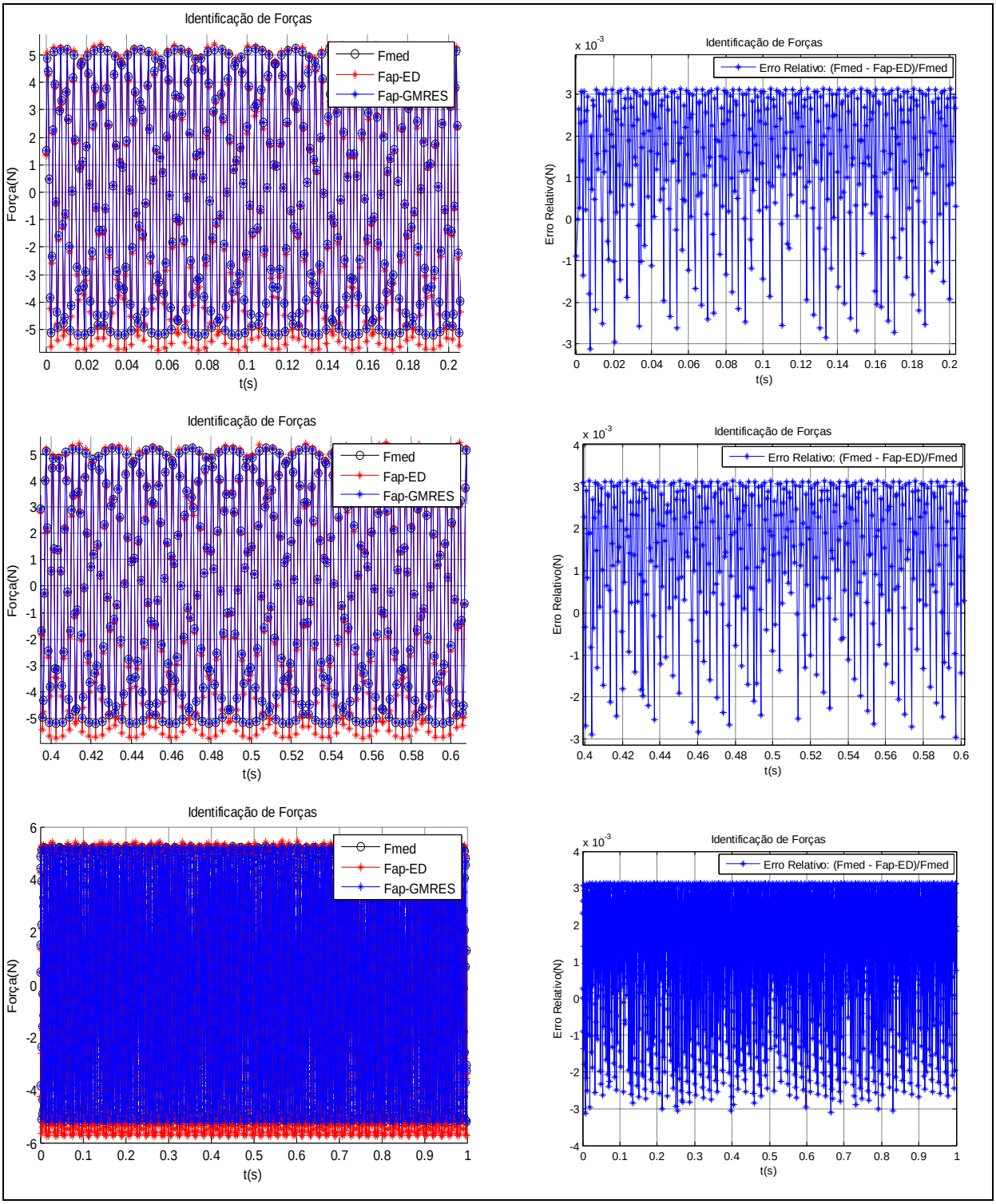



Figura 6.11- Solução analítica e numérica pelos métodos ED e GMRES, para t ∈ [0, 0.2], t∈[0.4, 0.6] e t ∈[0, 1] em segundos, apresentando o erro relativo entre elas.

A análise das Fig. 6.11 mostra que as soluções obtidas pelos métodos numéricos cometem um erro relativo de apenas $10^{-3}$, em problemas de identificação dinâmica, este erro pode ser considerado aceitável.

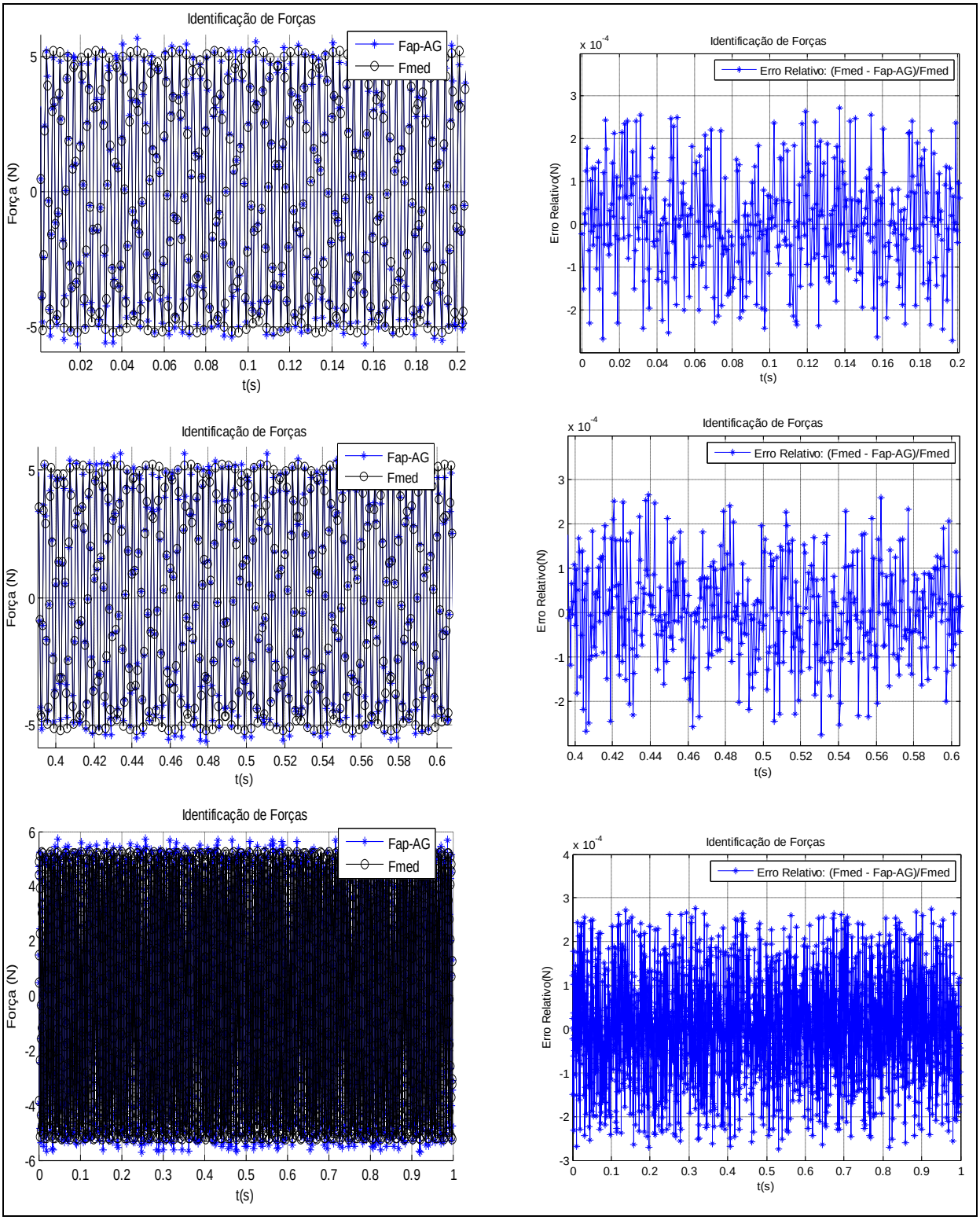



Figura 6.12- Solução analítica e numérica pelo método AG, para t∈[0, 0.2], t∈[0.4, 0.6] e t∈[0, 1] em segundos, apresentando o erro relativo entre elas.

Também, analisando a Fig. 6.12, pode-se observar que a solução obtida pelo método dos algoritmos genéticos apresenta um erro relativo de $10^{-4}$, assim, concluí-se que os métodos

testados foram eficazes para este tipo de problema. Nota-se, também, que para este problema a ED obteve resultados com erros relativos, considerando a solução numérica e a solução analítica, um pouco maiores do que os valores obtidos com AG.

Portanto, para resolver este tipo de problema, os métodos evolutivos poderão ser utilizados. Pois estas metodologias irão resolver o problema de forma eficiente com a vantagem de não se preocuparem com a quantidade de pontos e nem a técnica utilizada na discretização do tempo. Observa-se, aqui, que o método GMRES também resolveu este problema, de forma até mais eficiente do que os evolutivos, pois resultou um erro de $10^{-7}$, conforme Fig. 6.13. Vale ressaltar que, o método GMRES só pode ser utilizado para este caso, onde a matriz de coeficientes não era de dimensão elevada (n = 2048), pois quando se fez o teste com uma matriz de dimensão mais elevada, o método não pode ser utilizado por falta de memória do computador.

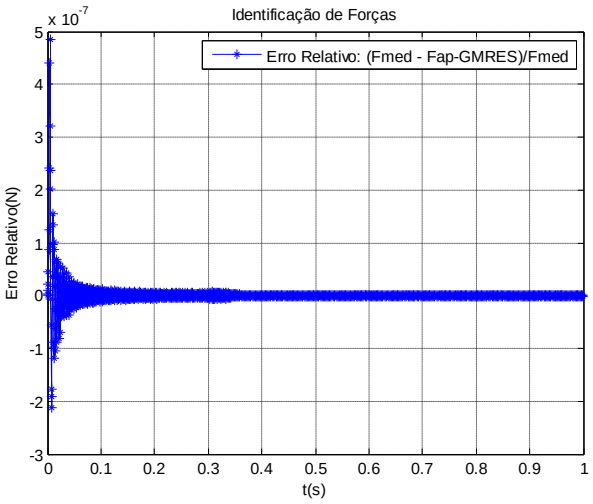



Figura 6.13- Erro relativo entre a solução analítica e numérica pelo método GMRES.

## 6.3.2 Sistema excitado por um ruído branco.

A seguir, serão comparadas as soluções obtidas pelas técnicas ED, AG e GMRES com a solução analítica, para o mesmo problema, mas neste caso a força excitadora é dada por um ruído branco:

$$F = randn(1, N\text{-}1) \tag{6.45}$$

Os resultados são apresentados a seguir, comparando a solução analítica com a solução numérica. A Fig. 6.14 apresenta uma comparação entre as soluções analítica e numérica, quando os métodos ED e GMRES são utilizados, para casos onde *F* é dada pela Eq. (6.45). Na Fig. 6.15 é feita a mesma comparação, agora, com o método AG sendo utilizado.

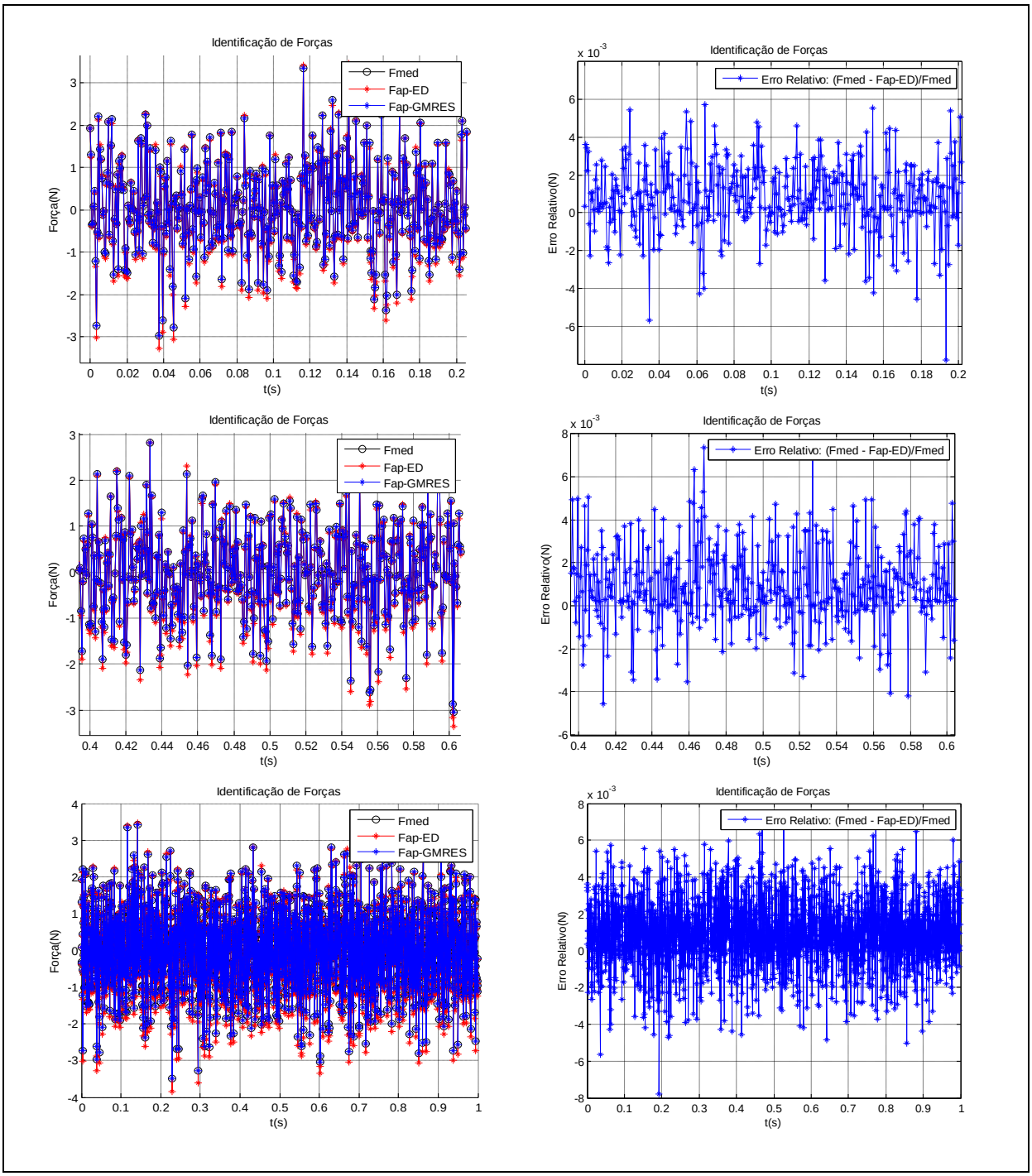



Figura 6.14- Solução analítica e numérica pelos métodos ED e GMRES, para t∈[0, 0.2], t∈[0.4, 0.6] e t∈[0, 1] em segundos, apresentando o erro relativo entre elas.

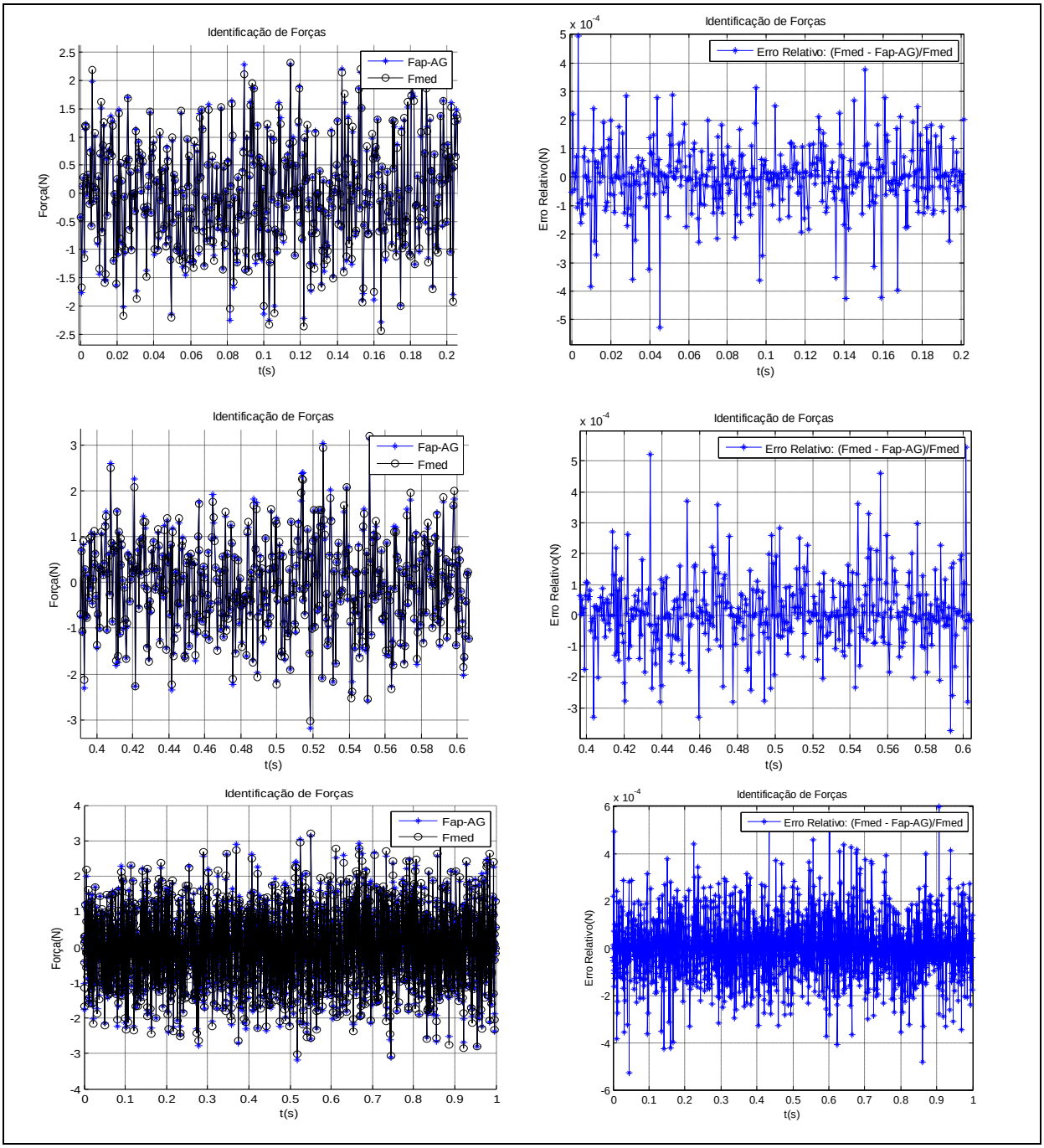



Figura 6.15- Solução analítica e numérica pelo método AG, para t∈[0, 0.2], t∈[0.4, 0.6] e t∈[0, 1] em segundos, apresentando o erro relativo entre elas.

As Fig. 6.14 e 6.15 mostram que a solução obtida pelos métodos numéricos, para este problema, apresenta um erro relativo semelhante ao erro apresentado para o problema anterior. Sendo assim, pode-se concluir que os métodos evolutivos também são viáveis para este tipo de problema.

### 6.3.3 Sistema dinâmico usando dados experimentais.

Nesta seção serão avaliados os métodos evolutivos para resolver um sistema linear proveniente de um problema de identificação indireta de forças. Este problema consiste em determinar a força aplicada a um sistema formado por três placas conectadas por três conjuntos de quatro lâminas de aço dispostas em paralelo. No domínio de baixas freqüências, o sistema pode ser modelado como um sistema de três graus de liberdade pouco amortecido. A Fig. (6.16) ilustra a estrutura utilizada para a avaliação dos métodos evolutivos na identificação da força.

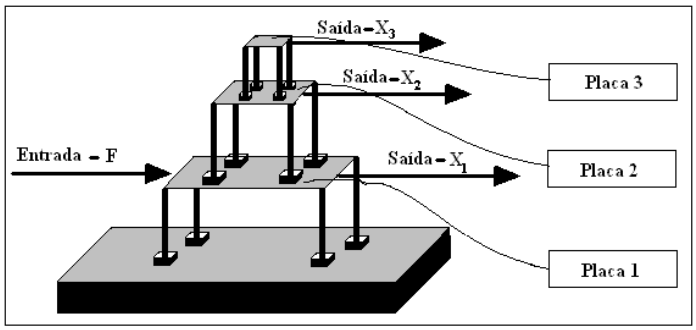



Figura 6.16- Ilustração do Sistema mecânico utilizado nos experimentos

No experimento realizado uma força de excitação por impacto *F* foi aplicada na coordenada 1 (Placa 1). As correspondentes respostas temporais ($X_1$, $X_2$ e $X_3$) foram medidas nas coordenadas 1, 2 e 3. Tanto a força como as respostas temporais foram medidas somente na direção horizontal. Para se realizar o experimento foram utilizados os seguintes equipamentos:

- Um excitador dinâmico B&K;
- Uma célula de carga B&K do tipo 8200;
- Um gerador de sinais B&K do tipo 1049;
- Um amplificador de potência B&K do tipo 2712;
- Uma placa de aquisição A/D da National Instruments, USB 9233;
- Três acelerômetros PCB do tipo 35 20C33;
- Um condicionador de sinais PCB do tipo 482 A 20.

Os dados de aquisição são:

- Taxa de aquisição - 2048 Hz;

- Número de pontos - 16384;
- Número de médias para as FRFs - 800;
- No gerador foi colocada um ruído branco de 2 a 100 Hz e amplitude 1.25 V;
- No amplificador, 7 A RMS de corrente e ganho 2.

Os testes foram feitos com:

- Teste 1: ganho 2 e amplitude 1.25 V;
- Teste 2: ganho 4 e amplitude 1.25 V;
- Teste 3: ganho 4 e amplitude 0.313V.

No experimento têm-se três localizações de medição e uma força de excitação, a Eq.(6.37) pode ser desenvolvida como segue:

$$\begin{aligned} x_1(t) &= \int_0^t [h_{11}(t-\tau)]\{f(t)\}d\tau \\ x_2(t) &= \int_0^t [h_{21}(t-\tau)]\{f(t)\}d\tau \\ x_2(t) &= \int_0^t [h_{31}(t-\tau)]\{f(t)\}d\tau \end{aligned} \tag{6.46}$$

Onde a F.R.I. $h_{i1}(t)$ representa a respostas na coordenada $i$, devido a excitação aplicada na placa 1.

A discretização $\Delta t$ do tempo de amostragem da Eq. (6.46) é dada por:

$$\begin{aligned} x_1(k\Delta t) &= \sum_{i=0}^{k} h_{11}[(k-i)\Delta t]f(i\Delta t)\Delta t \\ x_2(k\Delta t) &= \sum_{i=0}^{k} h_{21}[(k-i)\Delta t]f(i\Delta t)\Delta t \\ x_3(k\Delta t) &= \sum_{i=0}^{k} h_{31}[(k-i)\Delta t]f(i\Delta t)\Delta t \end{aligned} \quad , \quad \text{com k = 0, 1, 2, …, p-1} \tag{6.47}$$

A Eq. (6.47) pode ser rescrita como:

$$x_1(k) = \sum_{i=0}^{k} h_{11}[(k-i)] f(i) \Delta t$$
$$x_2(k) = \sum_{i=0}^{k} h_{21}[(k-i)] f(i) \Delta t \quad , \qquad \text{com k = 0, 1, 2, …, p-1} \tag{6.48}$$
$$x_3(k) = \sum_{i=0}^{k} h_{31}[(k-i)] f(i) \Delta t$$

As Eqs. (6.48) são agrupadas em um sistema de (*3*x*p*) equações e (p) incógnitas, da seguinte forma matricial:

$$\{X\} = [h]\{F\} \tag{6.49}$$

Onde

$$\{X\} = \begin{Bmatrix} \{X(0)\} \\ \{X(1)\} \\ \vdots \\ \{X(p-1)\} \end{Bmatrix}, \quad [h] = \begin{bmatrix} [h(0)] & [0] & \cdots & [0] \\ [h(1)] & [h(0)] & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [h(p-1)] & [h(p-2)] & \cdots & [h(0)] \end{bmatrix} \quad e \quad \{F\} = \begin{Bmatrix} F(0) \\ F(1) \\ \vdots \\ F(p-1) \end{Bmatrix} \tag{6.50}$$

Sendo

$$\{X(i)\} = \begin{Bmatrix} x_1(i) \\ x_2(i) \\ x_3(i) \end{Bmatrix}, \quad [h(i)] = \Delta t \begin{bmatrix} h_{11}(i) \\ h_{21}(i) \\ h_{31}(i) \end{bmatrix} \quad e \quad [0] = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad \text{com} \quad i = 0,1,..., p-1 \tag{6.51}$$

Diferentes algoritmos podem ser utilizados para calcular o vetor das forças desconhecidas $\{F\}$ a partir da Eq. (6.49). Desde que o número de equações seja maior do que o número de incógnitas e que $[h]$ tenha posto completo, a solução normal pelo método dos mínimos quadrados para a Eq. (6.52) pode ser calculado segundo:

$$\{F\} = ([h]^T [h])^{-1} [h]^T \{X\}. \tag{6.52}$$

O sistema de equações (6.49) é geralmente mal condicionado, ou seja, pequenos erros afetando $\{X\}$ e ou $[h]$ podem conduzir a grandes erros no vetor da força estimada $\{F\}$. Neste trabalho, os métodos da Evolução Diferencial e dos Algoritmos Genéticos são utilizados para resolver as equações de estimação.

Para se obter as aproximações, minimiza-se o funcional quadrático dado por:

$$f(R) = \|R\|^2 = \left\|\{X\} - [h]\{F\}\right\|^2 \tag{6.53}$$

Os sinais de força (sinal de entrada) e de acelerações (sinal de saída) foram amplificados usando amplificadores de sinais, e enviados para a placa de aquisição e posteriormente armazenados em um microcomputador. Cada sinal adquirido foi observado no intervalo de *0* a *13,67s* discretizado em 3500 pontos igualmente espaçados ($\Delta t = 3{,}9 \times 10^{-3} s$). Desta forma, a Eq. (6.49) foi resolvida para as F.R.I.s $h_{11}(t)$, $h_{21}(t)$ e $h_{31}(t)$, utilizando os métodos evolutivos. A título de ilustração, as Figs. (6.17) a (6.19), fornecem a aceleração e a F.R.I. para as placas 1, 2 e 3, respectivamente.

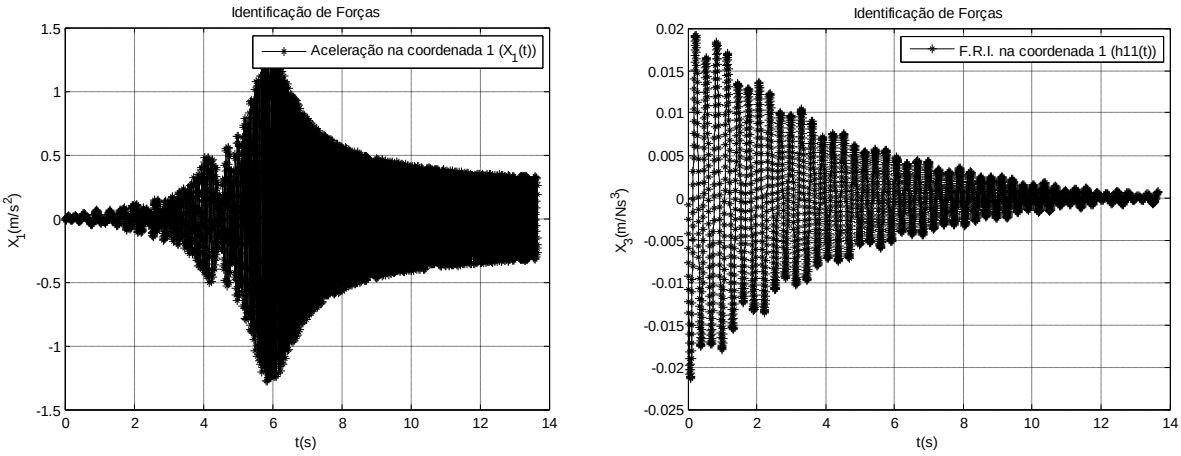



Figura 6.17- Resposta em aceleração $X_1(t)$ e F.R.I. $h_{11}(t)$ para placa 1.

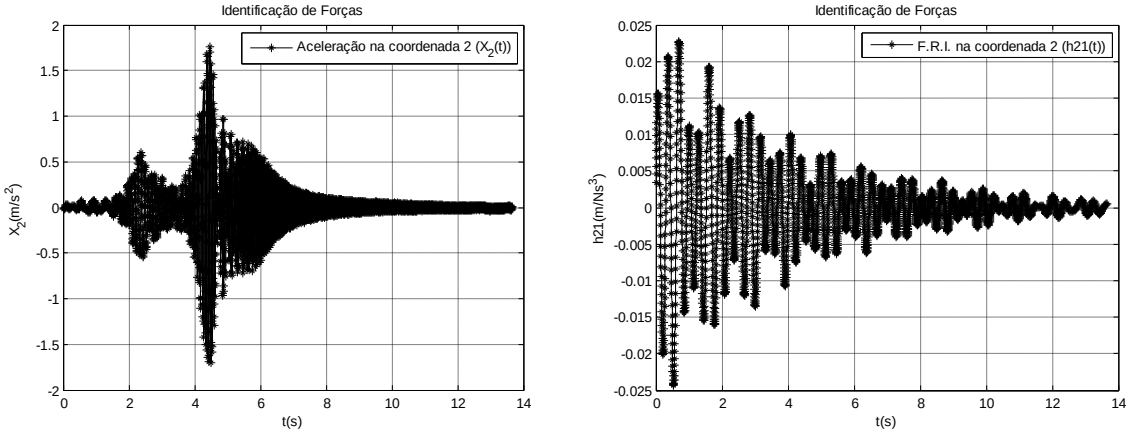



Figura 6.18- Resposta em aceleração $X_2(t)$ e F.R.I. $h_{21}(t)$ para placa 2.

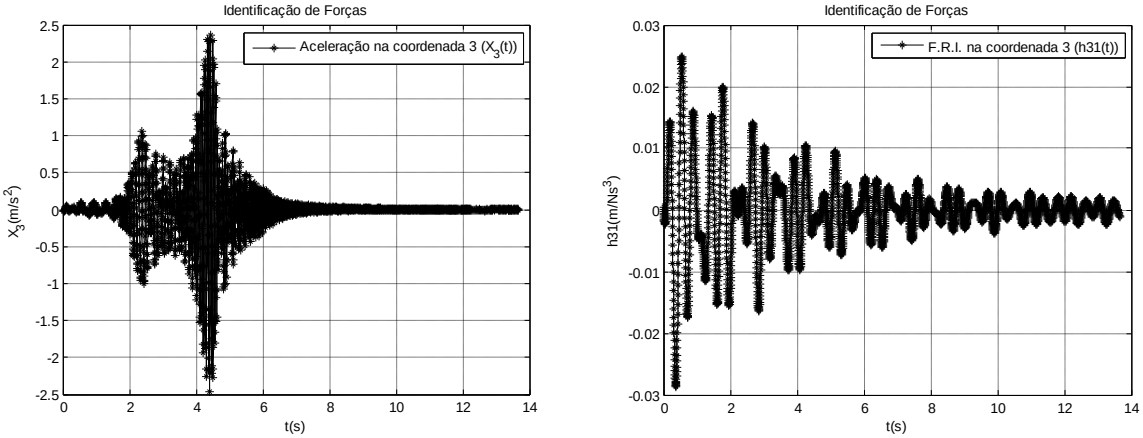
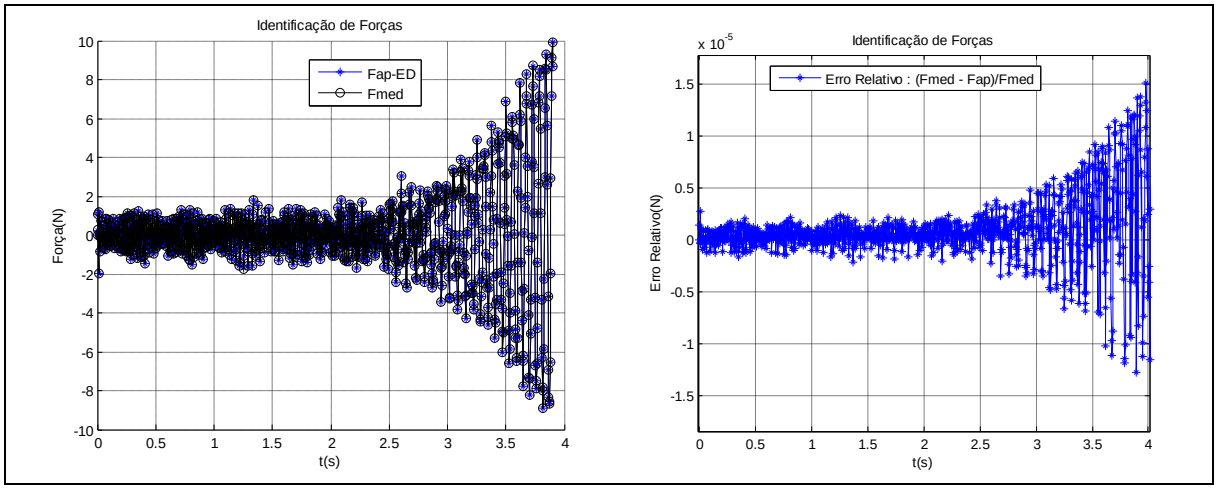



Figura 6.19- Resposta em aceleração $X_3(t)$ e F.R.I. $h_{31}(t)$ para placa 3.

Assumindo que a força de excitação era desconhecida e usando as F.R.I.s previamente identificadas via análise modal experimental, o método de identificação de forças baseado na integral de convolução no domínio do tempo, combinado com os métodos evolutivos, foram empregados para encontrar aproximações da força de entrada. Uma vez que a força exata era disponível, ela pode ser comparada à força identificada, com a finalidade de avaliar a precisão do procedimento de reconstrução de forças.

A Fig. 6.20 apresenta a força de excitação aproximada, obtida pela aplicação da ED, em comparação com a força exata. Além disso, no ultimo gráfico desta figura pode ser visto o erro relativo entre estes dois valores.



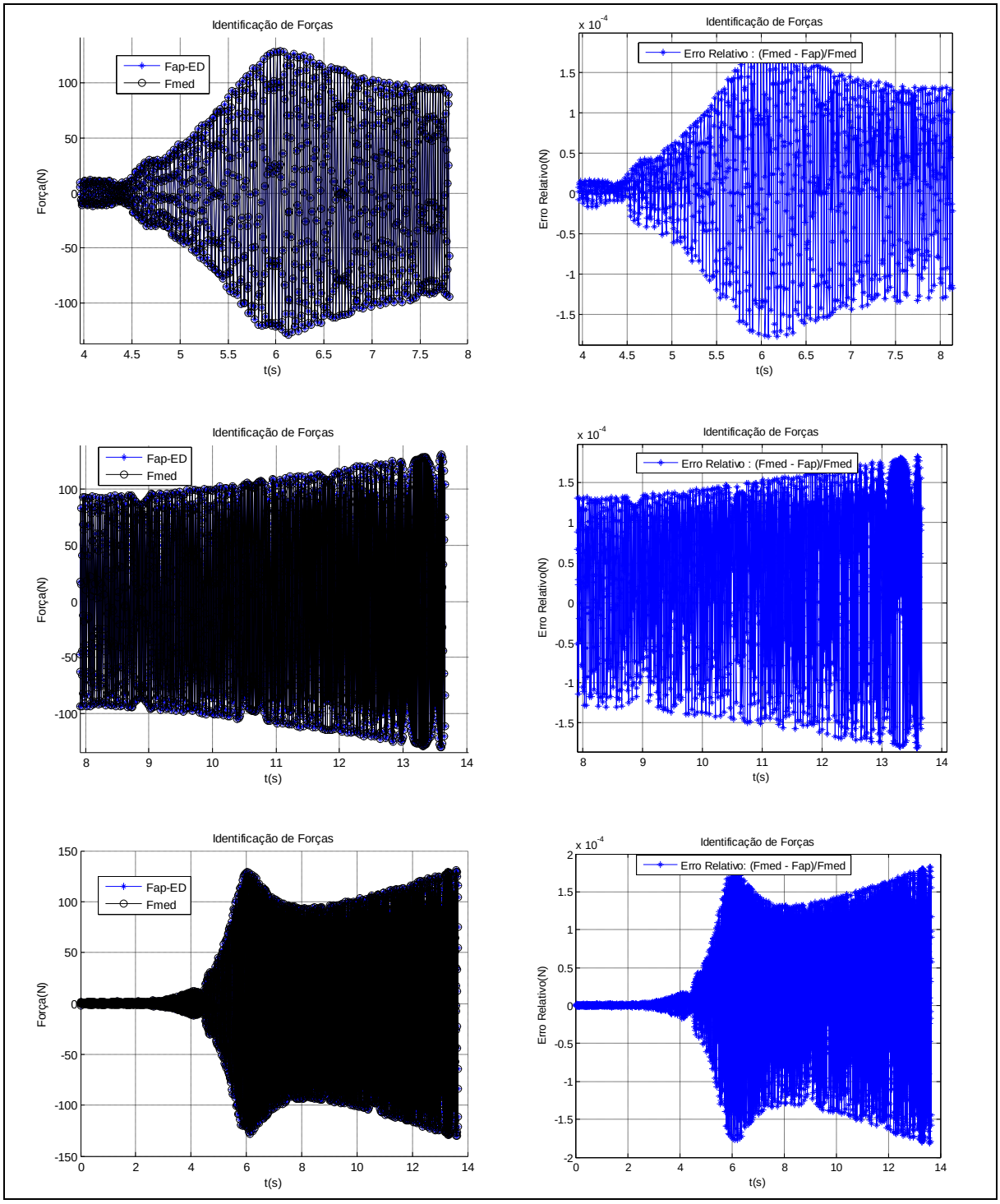



Figura 6.20- Comparativo entre a Força aproximada por ED e os valores medidos, com apresentação do erro relativo entre eles.

Nas Figs. 6.21 a 6.23 são apresentados os gráficos das respostas, em aceleração, comparando as respostas medidas com os valores aproximados pela ED, para cada uma das placas, considerando os respectivos erros relativos.

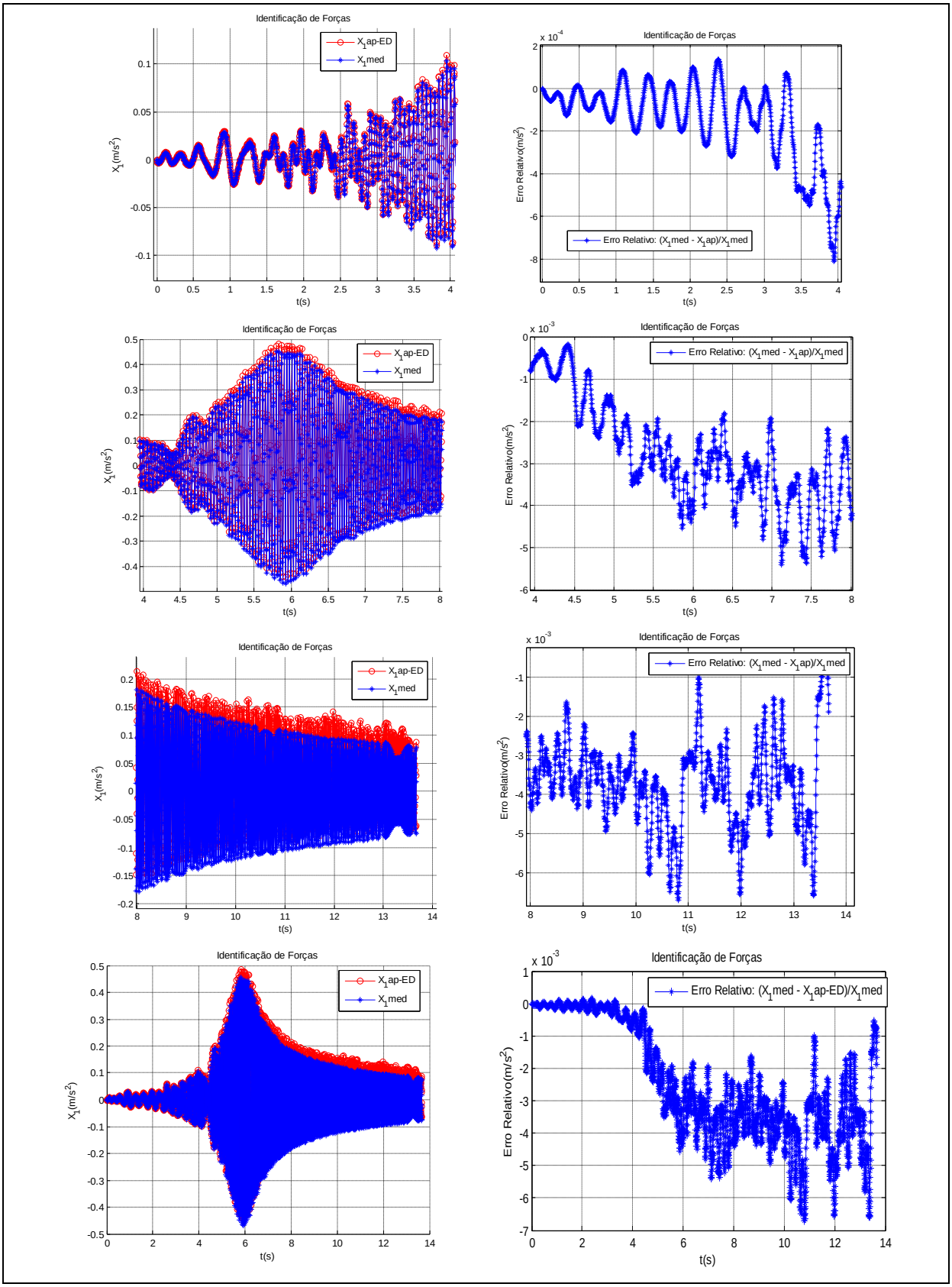



Figura 6.21- Comparativo entre a resposta $X_1$ aproximada por ED e os valores medidos, com apresentação do erro relativo entre elas.

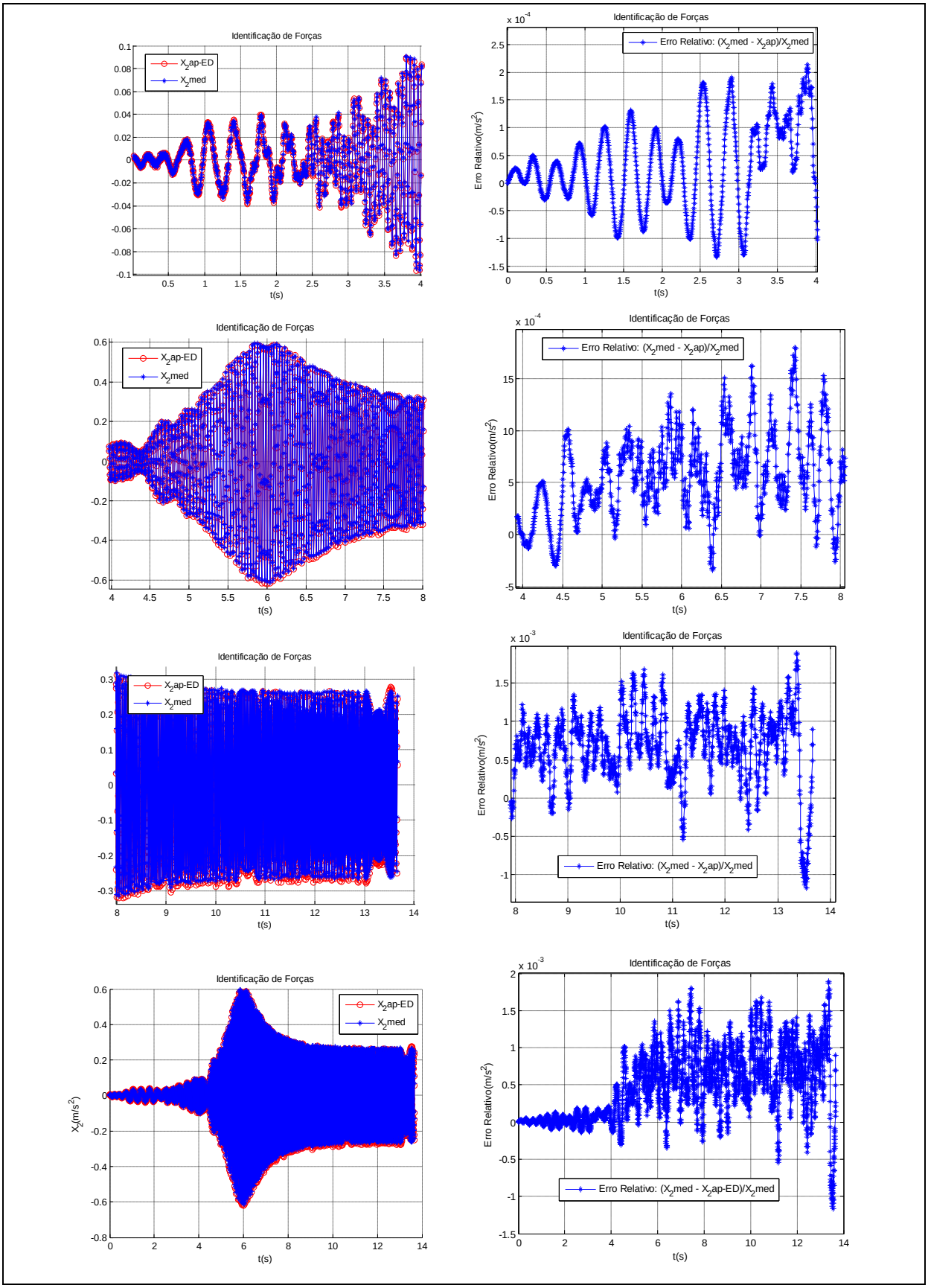



Figura 6.22- Comparativo entre a resposta $X_2$ aproximada por ED e os valores medidos, com apresentação do erro relativo entre elas.

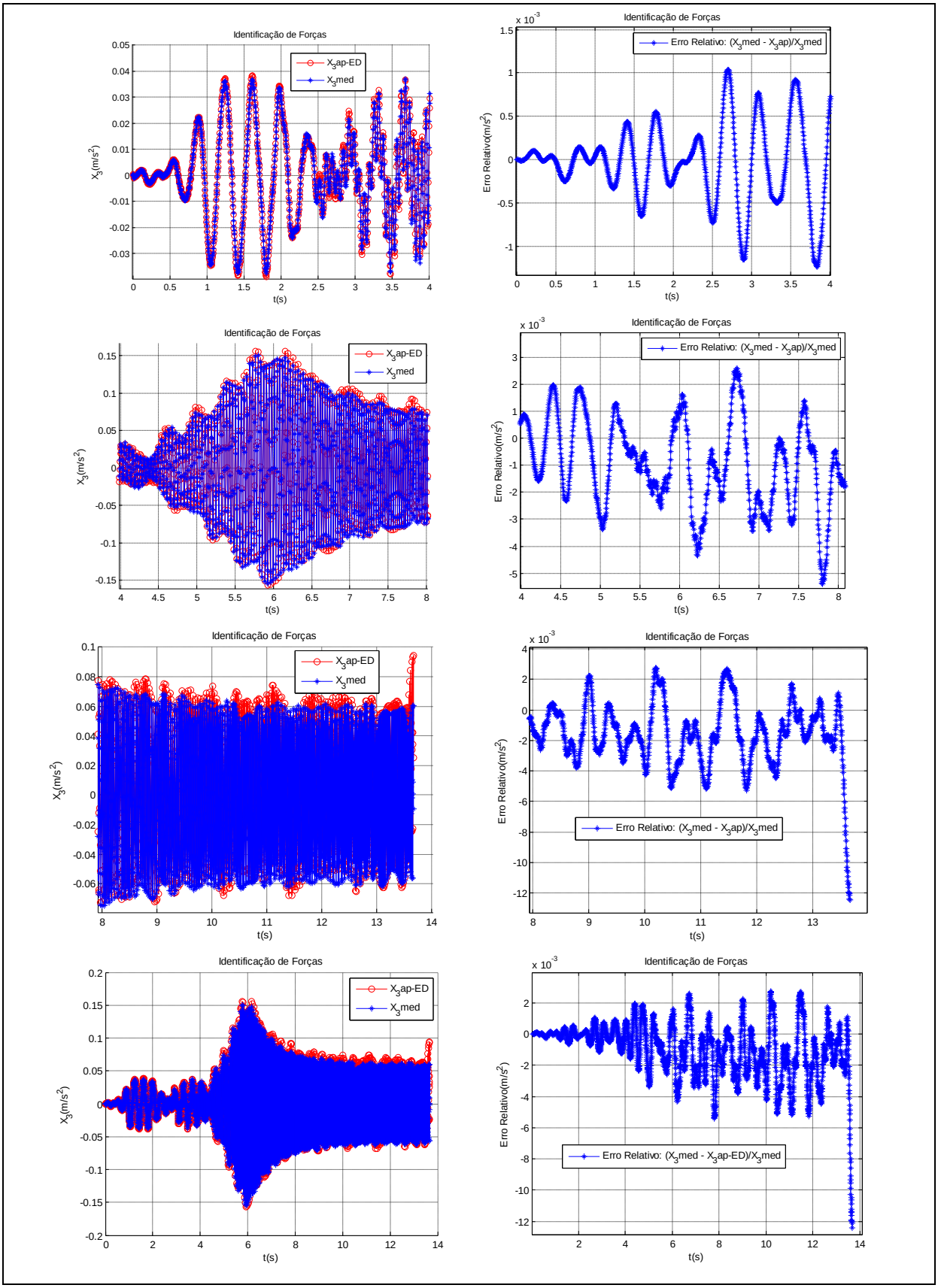
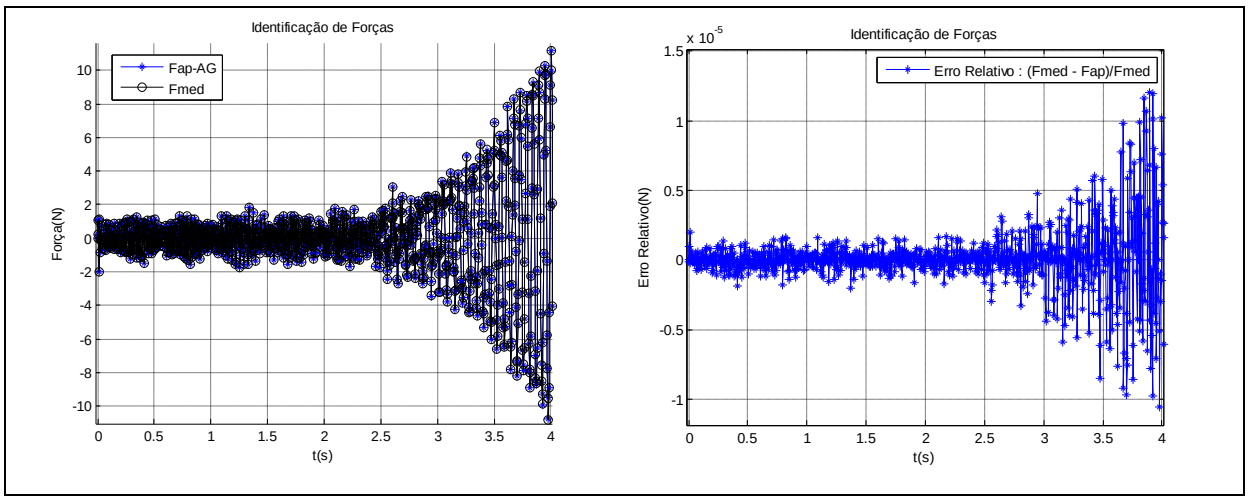



Figura 6.23- Comparativo entre a resposta $X_3$ aproximada por ED e os valores medidos, com apresentação do erro relativo entre elas.

Observando os resultados apresentados pelas figuras anteriores percebe-se que o método da Evolução Diferencial é um método bastante confiável, pois apresenta uma aproximação da força de entrada muito próxima da força medida. Dá para se observar, também que o erro relativo entre a força medida e a aproximada é um erro aceitável, pois fica na casa de $10^{-4}$ e isso é muito razoável, tratando-se de um problema físico como o analisado. A principal vantagem deste método, é que se pode trabalhar com a matriz da F.R.I.s na forma com as quais se apresenta, não sendo necessário nenhum tipo de condicionamento, nem tão pouco a preocupação pelo fato da mesma não ser uma matriz quadrada, exigência apresentada pela maioria dos métodos tradicionais. Por se tratar de um método de otimização não-determinístico, tem se pouca influência de erros numéricos, pois só se trabalha com produto de matriz por vetor. Por isso, pode-se concluir que este método, além de ser um método robusto, é um método que pode ser muito indicado para resolver este tipo de problema.

A seguir, apresentam-se também os resultados obtidos pelo método dos Algoritmos Genéticos (AG), dados pelas Figs. 6.24 a 6.27. Observando os gráficos, verifica-se que este método também pode ser indicado, com a ressalva de que se necessita um pouco mais de tempo computacional, pois requer um número maior de gerações para se obter resultados como os obtidos pelo método da Evolução Diferencial (ED).



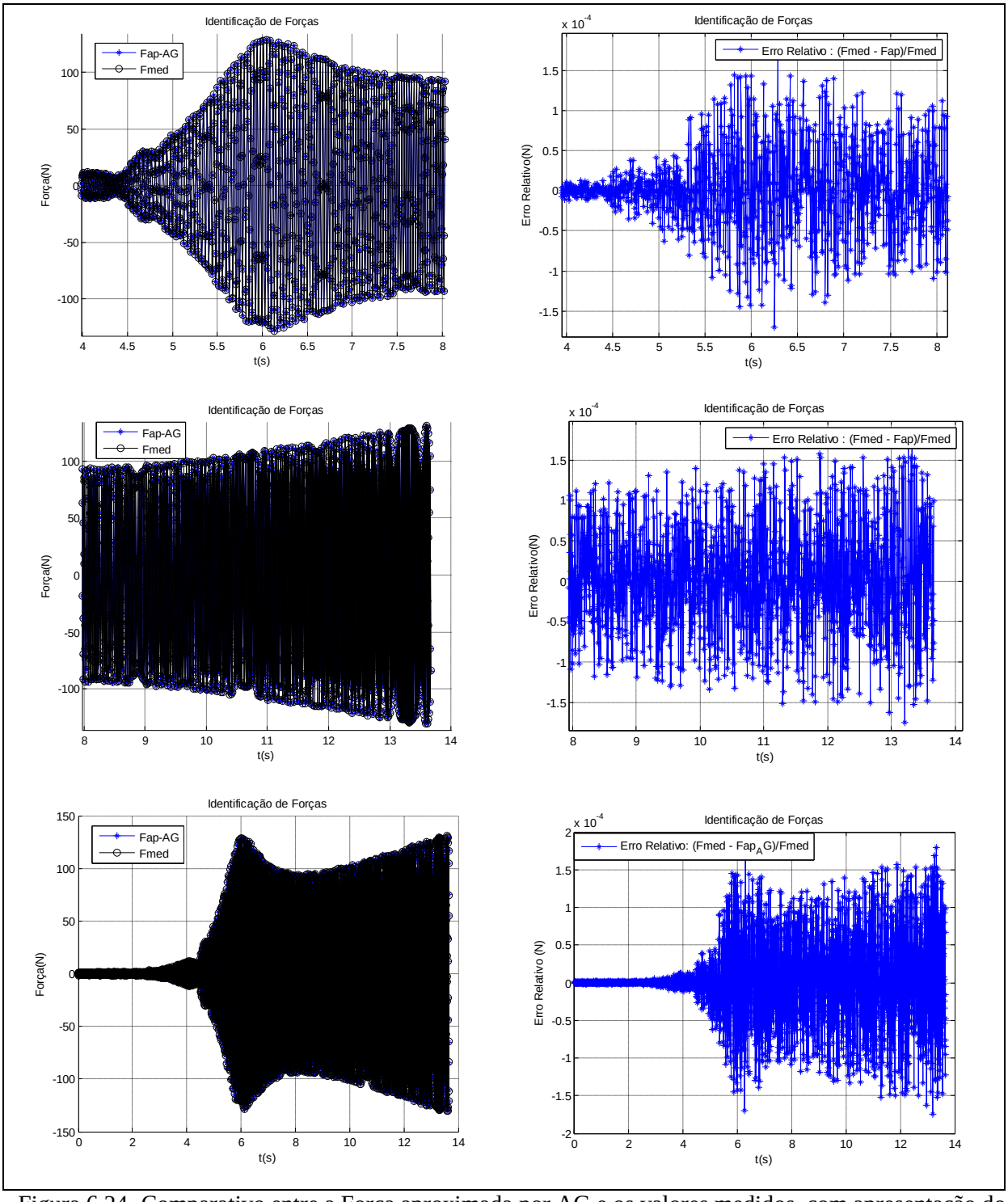



Figura 6.24- Comparativo entre a Força aproximada por AG e os valores medidos, com apresentação do erro relativo entres elas.

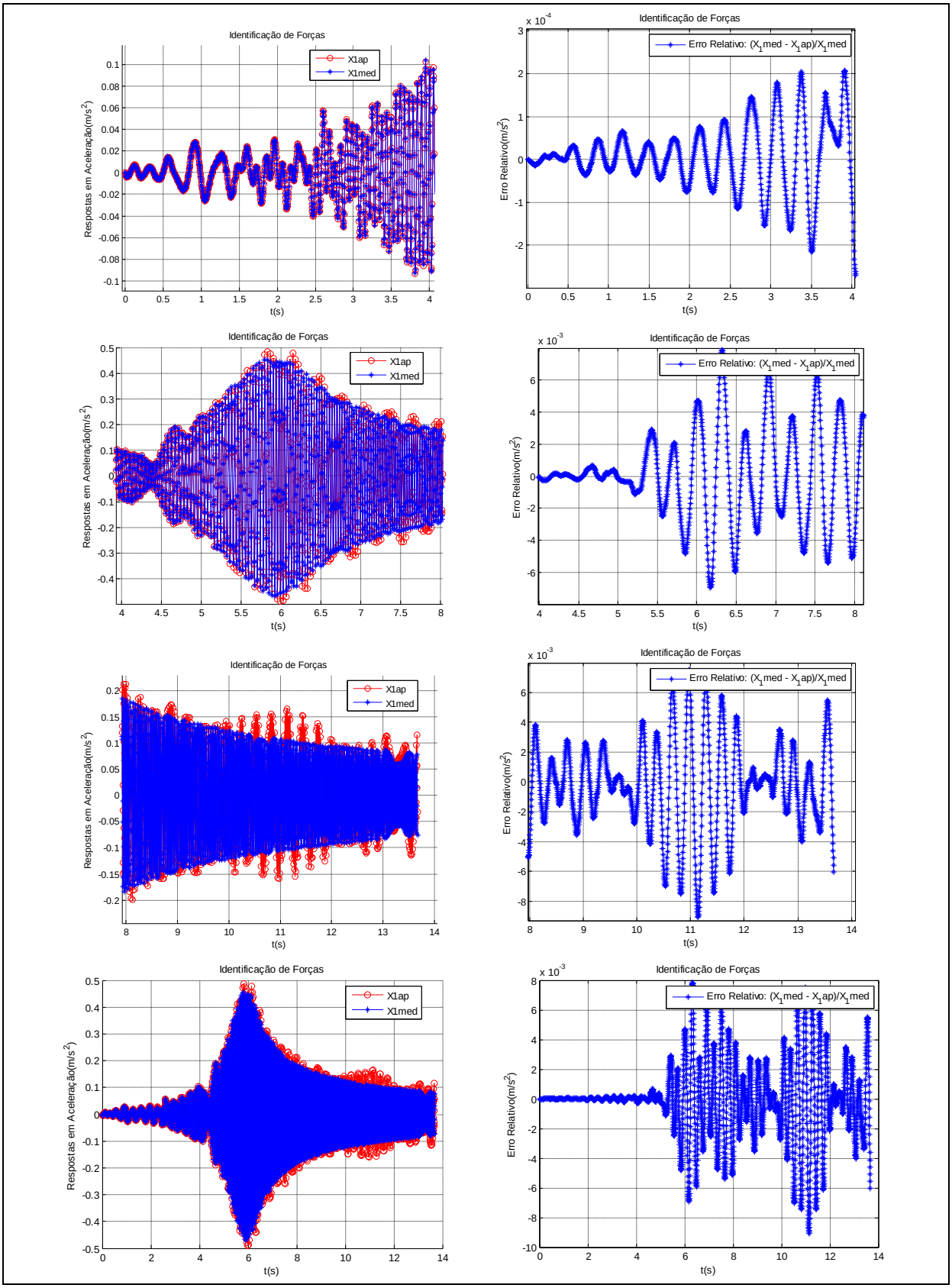



Figura 6.25- Comparativo entre a resposta $X_1$ aproximada por AG e os valores medidos, com apresentação do erro relativo entre elas.

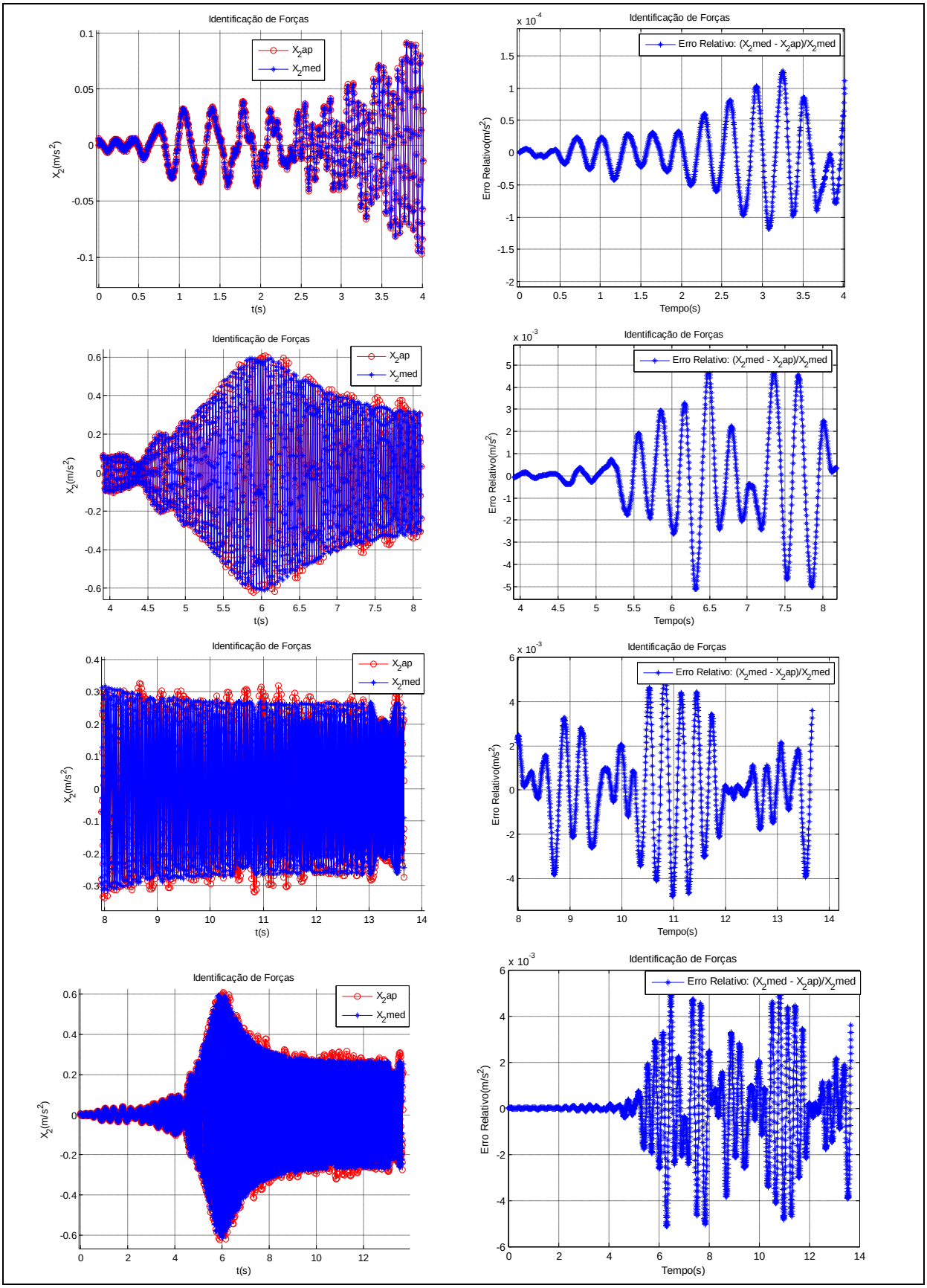



Figura 6.26- Comparativo entre a resposta $X_2$ aproximada por AG e os valores medidos, com apresentação do erro relativo entre elas.

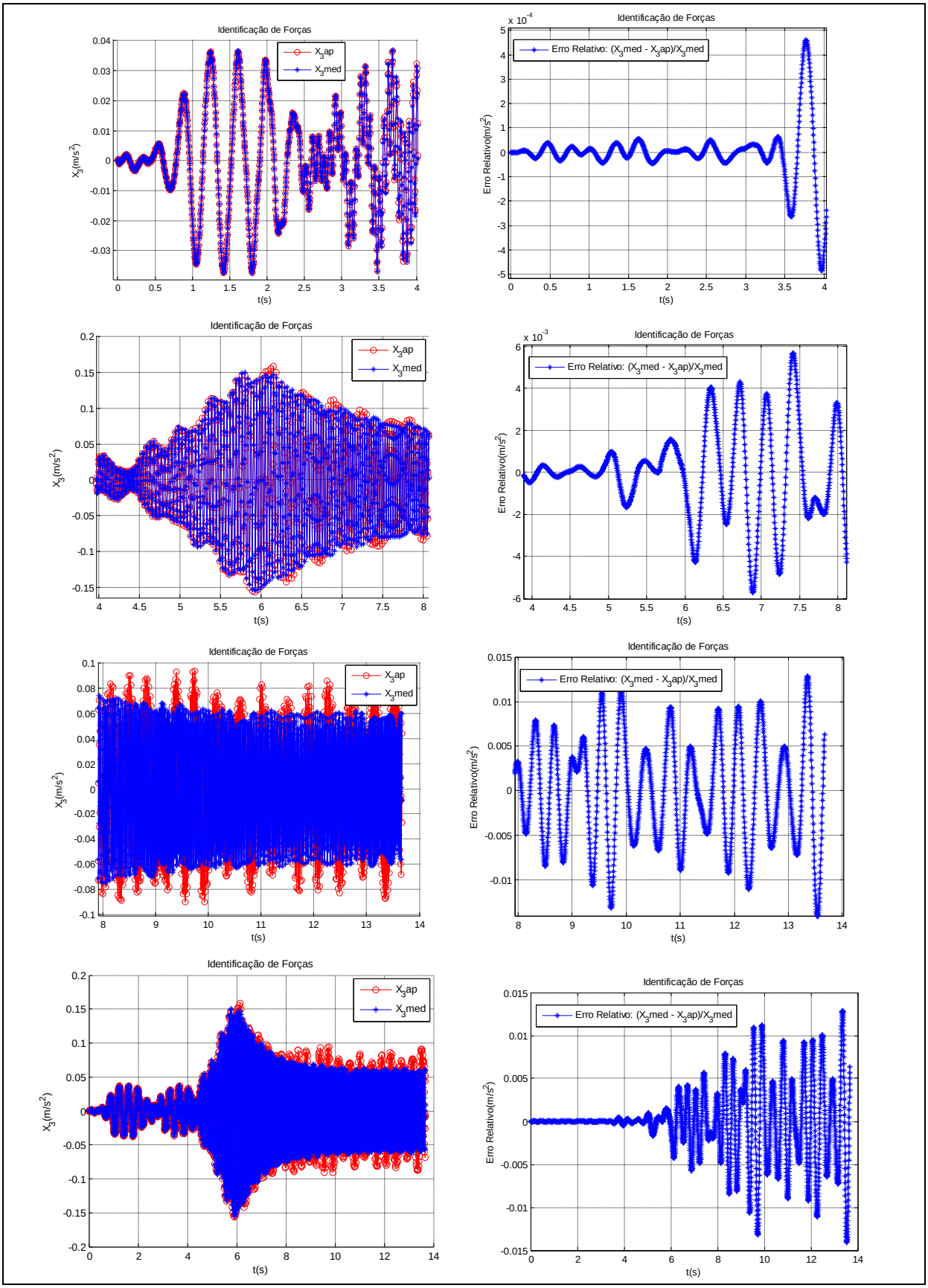



Figura 6.27- Comparativo entre a resposta $X_3$ aproximada por ED e os valores medidos, com apresentação do erro relativo entre elas.

Pode se concluir, a partir da análise dos resultados apresentados pelas Figs. 6.24 a 6.27, que o método dos Algoritmos Genéticos apresenta resultados menos precisos do que os obtidos pela Evolução Diferencial. O erro relativo apresentado por AG, correspondente as respostas em aceleração, é maior do que o erro encontrado pela utilização da ED.

Em geral, este tipo de problema, é resolvido utilizando técnicas que dependem da matriz de coeficientes, e uma delas faz com que a matriz seja multiplicada por sua transposta, para que a matriz resultante seja quadrada. Esta multiplicação implica em um custo computacional alto, pois a multiplicação de matrizes exige muita memória, além do fato de que podem ocorrer erros de arredondamento que vai acarretar imprecisões na resolução do sistema.

Desta forma, a proposta deste trabalho, que pretende disponibilizar uma metodologia que não dependa das características da matriz de coeficientes, pôde ser comprovada nesta aplicação. Vale ressaltar que, neste caso, trabalhou-se com uma matriz de coeficientes retangular, formada por três blocos triangulares, ou seja, sem as características exigidas pelos métodos determinísticos. Assim, este problema de identificação de forças dinâmicas foi resolvido, sem usar nenhuma técnica de regularização da matriz de coeficientes, aplicando apenas os métodos evolutivos. Caso fosse usar métodos determinísticos, como por exemplo, o GMRES, a matriz deveria ser pelo menos quadrada. Portanto, a metodologia proposta se mostrou eficiente.

# Capítulo 7: Conclusões e Trabalhos Futuros

Neste trabalho, um problema de otimização foi definido com o propósito de obter a solução de sistemas lineares. Foram utilizadas quatro técnicas iterativas, Método dos Gradientes Conjugados (CG), Método dos Gradientes Bi-Conjugados (BiCG), Método dos resíduos Mínimos (MINRES) e o Método dos Resíduos Mínimos Generalizados (GMRES), e duas evolutivas, Método da Evolução Diferencial (ED) e o Método dos Algoritmos Genéticos (AG). Todas as técnicas testadas apresentaram bom desempenho na obtenção da solução do problema. Ressaltando que, algumas técnicas iterativas não podem ser aplicadas a qualquer tipo de problema, uma vez que, dependem das características da matriz de coeficientes. O método dos Resíduos Mínimos Generalizado é um método que pode ser usado para todos os tipos de matrizes, mas perde a eficiência com o aumento do numero de variáveis e apresenta grande exigência de memória para o armazenamento dos dados.

As técnicas de otimização evolutivas exigem um esforço computacional maior quando comparada com os outros métodos, porém podem ser utilizadas para qualquer tipo de matriz, assim como o GMRES, mostrando serem algoritmos de grande potencial para serem aplicados a problemas complexos. Porém, ressalta-se que elas, ao contrário do GMRES, não exigem o uso de técnicas de regularização da matriz de coeficientes .

Durante as pesquisas verificou-se que os métodos de otimização estudados, com enfoque na Evolução Diferencial e nos Algoritmos Genéticos, são ferramentas eficientes para a obtenção de soluções numéricas de sistemas lineares de grandes dimensões. Os resultados obtidos mostraram que os métodos evolutivos são confiáveis, pois suas convergências não dependem das características das matrizes de coeficientes, como por exemplo, o fator condição. Além disso, pode-se observar uma melhor performance da Evolução Diferencial, que se apresentou muito eficiente quando se trata de resolver um problema de identificação dinâmica, pois, proporciona uma solução numérica próxima da solução analítica, com um erro considerado nulo em dinâmica.

O estudo foi focado essencialmente em sistemas lineares onde as matrizes dos coeficientes são não simétricas e não positiva-definidas, pois são poucos os métodos disponíveis para a solução numérica destes sistemas. Foram utilizados métodos evolutivos, que possuem maior chance de obter mínimos globais, enquanto que os métodos

convencionais podem ficar presos nos mínimos locais. Os métodos evolutivos estudados apresentam uma eficiência grande, porém necessitam de um elevado tempo computacional.

Foram estudadas a formulação de cada um dos métodos, a avaliação de suas características operacionais e desempenho, mediante a realização de algumas aplicações em problemas de dinâmica, tanto em estruturas simuladas numericamente, quanto em uma estrutura real ensaiada em laboratório. Em todas estas aplicações, buscou-se caracterizar as dificuldades presentes nas situações práticas com ênfase no uso de dados incompletos, na presença de ruídos de medida e na ocorrência de mal condicionamento numérico.

Uma aplicação interessante foi o problema de identificação indireta de forças dinâmicas, no qual se trabalhou com um experimento constituído por uma estrutura formada por um pórtico tri-dimensional, ou seja, um sistema de três graus de liberdade pouco amortecido. Este problema foi resolvido com eficiência pelos métodos de otimização heurística, tanto a Evolução Diferencial quanto os Algoritmos Genéticos.

Nesta aplicação, algumas técnicas baseadas nos métodos clássicos de otimização não puderam ser utilizadas, devido ao fato da matriz ser retangular e estes exigem que ela seja quadrada. Logo, a menos que se faça algum tipo de condicionamento da matriz dos coeficientes do sistema linear estes não podem ser utilizados, enquanto que a técnica proposta neste trabalho foi utilizada com muito êxito, mostrando se assim uma ótima alternativa computacional na resolução de problemas de Identificação Indireta de Forças Dinâmicas.

Em todas as aplicações pode-se concluir que os métodos de otimização evolutivos funcionam muito bem, e pode ser uma opção em relação aos métodos clássicos para se resolver problemas envolvendo grandes sistemas lineares. Estes métodos não fazem nenhuma exigência sobre a apresentação da matriz dos coeficientes, enquanto que aqueles exigem que esta tenha características especiais, para que possam ser aplicados.

De modo geral, a técnica e os métodos estudados demonstraram ser bem adaptados às aplicações práticas de engenharia.

Uma atenção especial na utilização das técnicas heurísticas deve ser a definição do espaço de busca (ou região viável). Para que se possa usá-los com eficiência é necessário saber definir bem o espaço de busca. Para se definir este espaço de busca é necessário que se tenha uma idéia da solução do problema, pois quanto menor esta região menor será o tempo computacional e melhor será o desempenho destas técnicas.

Vale ressaltar que os resultados obtidos nesta pesquisa foram apresentados em artigos publicados em congressos internacionais: "***Comparação entre Métodos Iterativos não***

***Estacionários e Métodos Heurísticos Aplicados à Solução de grandes Sistemas Lineares",*** 8º congresso Iberoamericano de Ingeneria Mecânica (CIBIM8); "***Differential Evolution Applied to the Solution of Large Linear Systems***", International Conference on Engineering Optimization (EngOpt 2008); "***A new approach to indirect dynamics forces identification*",** XIII International Symposium on Dynamic Problems of Mechanics (DINAME 2009); "***Differential Evolution applied to solve problems of indirect identification of external forces"***, IV European Congress on Computational Mechanics: Solids, Structures and Coupled Problems in Engineering (ECCM IV).

Alguns aspectos relativos às técnicas estudadas não foram abordados neste trabalho e poderão ser objeto de investigação futura. Dentre eles podem ser citados:

- A paralelização dos métodos evolutivos, visando o ganho no tempo computacional;
- Utilização de algoritmos híbridos, técnicas clássicas e evolutivas trabalhando em conjunto, fazendo com que este tenha uma convergência mais rápida;
- Avaliação de desempenho do método quando aplicado a sistemas estruturais mais complexos, sujeitos a maior número de forças excitadoras;
- A aplicação da metodologia a problemas de dinâmica dos fluídos.

# REFERÊNCIAS


ACTON, F. S. (1990), ***Numerical Methods That Work,*** Mathematical Assn. of Amer.; ISBN: 0883854503.

ACTON, F. S. (1996), ***Real Computing Made Real :*** *Preventing* ***Errors in Scientjfic and Engineering Calculations,*** 259 pages, Princeton Univ. Pr; ISBN: 0691036632

ALTHAUS, G. W., SPEDICATO, E. (Eds.) (1998), ***Algorithms for Large Scale Linear Algebraic Systems: - Applications in Science and Engineering,*** Kluwer Academic Publishers; ISBN: 079234975X.

AMES, W. F. (1992), ***Numerical Methods for Partial Diferential Equations,*** Third Edition, Academic Press, Boston.

ANSONI, J., BRANDI A. C., CAROSIO, G.L.C., SELGHIM Jr., P.,(2007) ***Study of methods for the resolution of large sparse linear systems associated with multiphase flow tomography problem,*** XXX Congresso Nacional de Matemática Aplicada e Computacional, Florianópolis – SC.

APARECIDO, J. B., (2004). ***Solução de Grandes Sistemas Lineares Esparsos,*** Ilha Solteira.

ARMSTRONG, B., LAUKO, I., WADE, B. (2005), ***Optimal NPID Stabilizations of Linear Systems,*** Journal of Optimization Theory and Applications, Vol. 124, N$^{o}$ 2, pp. 307-322.

ASHBY, S. (1985), ***CHEYCODE: A Fortran implementations of Manteuffel's adaptative Chebyshev algorithm,*** Tech. Rep. UIUCDCS-R-85-1203, University of Illinois.

AXELSSON, O. (1996), ***Iterative Solution Methods,*** 654 pages, 2nd edition, Cambridge University Press, ISBN: 0-52 1 -55569-8.

BABU, B. V. & MUNAWAR, S. A., (2001). ***Optimal Design of Shell-and-Tube Heat Exchangers by Different Strategies of Differential Evolution***, The Faculty Lounge, Article nº 003873.

BABU, Prabhu, PELCKMANS, Kristiaan, STOICA, Petre and LI, Jian (2010). ***Linear Systems, Sparse Solutions, and Sudoku***. Signal Processing Letters, IEE, vol. 17, issue 1, pp. 40-42, Jan. 2010.

BÄCK, T., (1992), ***Self-adaptation in genetic algorithms.*** In: Varela and Bourgine, pp. 263-271.

BARRET, R. BERRY, M. CHAN, T. F. DEMMEL, J. DONATO, J. DONGARRA, J. EIJKHOUT, V. POZO, R. ROMINE, C. , VORST, H. van der (1994), Templates for the

Solution of Linear Systems: Building Blocks for Iterative Methods, 2a ed. Philadelphia, PA: SIAM.

BENZI, M. (2002), ***Preconditioning Techniques for Large Linear Systems: A Survey***, Jornal of Computational Physics 182, pp. 418-477.

BERGAMASCHI, P.R., SARAMAGO, S.F.P. & COELHO, L.S, (2005). **Comparative Study of SQP and Metaheuristics for Robotic Manipulator**, submitted to Journal of Optimization Theory and Applications.

BEYER, H.G., (1995)a, ***Toward a theory of evolution strategies: On the benefit of sex – (μ/μ, λ)-theory.*** Evolutionary Computation Journal 3(1), pp. 81-111.

BEYER, H.G., (1995)b, ***Toward a theory of evolution strategies: Self-adaptation.*** Evolutionary Computation Journal 3(3), pp. 311-347.

BJÖRCK, A., ELFVING, T. (1979). ***Accelerated projection methods for computing pseudo-inverse solutions of systems of linear equations***. BIT, 19, pp. 145-163.

BOX, G. E.; P. e WILSON, K. B. (1951), ***On the Experimental Attainment of Optimum Conditions***, Journal of the Royal Statistical Society. *Series B Metho.* v. 13, p. 1- 45**.**

BRAGA, C. G. (1998) ***O Uso de Algoritmos Genéticos para Aplicação em Problemas de Otimização de Sistemas Mecânicos***. Dissertação de Mestrado – Universidade Federal de Uberlândia, Uberlândia, MG, Brasil.

BULIRSCH, R. (1992), et al., ***Introduction to Numerical Analysis*** (Texts in Applied Mathematics, N$^{o}$ 12, 660 pages, 2nd edition, Springer Verlag; ISBN: 038797878X.

CAMPOS-SILVA, J. B., APARECIDO, J. B., (2003), ***Data Structure and Stationary Iterative Solutions Methods for Large Sparse Linear systems,*** In: COBEM-2003.

CAMPOS-SILVA, J. B., APARECIDO, J. B., MOURA, L. F. M., (1999), ***A Control Volume-Finite Element Method (CVFEM) for Unsteady Fluid Flows***, In: **COBEM-99,** Águas de Lindoya.

CHARALAMPAKIS, A.E e KOUMOUSIS, V.K., (2008) ***Identification of Bouc–Wen hysteretic systems by a hybrid evolutionary algorithm.*** Journal of Sound and Vibration vol. 314, pp. 571–585.

CHENG, S.L., HWANG, C. (2001), ***Optimal Approximation of Linear Systems by a Differential Evolution Algorithm.*** IEEE Transactions on Systems, vol.31, no. 6, November 2001.

COELHO, L. S., WENTZ, D. A., DOMANSKI, D., (2004), ***Abordagem de Evolução Diferencial implementada em FPGA e aplicada à otimização de um projeto mecânico com restrições.*** SBRN.

CONTE, S. D. (1972) ***Elementos de Análise Numérica***. Editora Globo.

COSTA, D. R., APARECIDO, J. B. (1991), ***Analysis of Duct Flow using Finite Element Method***, In: Idelson, 5. R. (Ed.), Computational Mechanics, Santa Fé, Argentina, AMCA - Associacion Argentina de Mecanica Computacional, vol. II, pp. 303-312 (in Portuguese).

CRUZ, I.L.L.; WILLIGENBURG, L.G.V.; STRATEN, G. V., (2003), ***Efficient Differential Evolution algorithms for multimodal optimal control problems,*** Applied Soft Computing Journal, vol. 3, i. 2, pp. 97-122.

DEB, K., (1995), ***Otimization for Engineering Design: Algorithms and Examples*****.** New Delhi: Prentice-Hall.

DEB, K., (2001), ***Multi-objetive optimization using Evolutionary Algorithms.*** John Wiley&Sons, pp. 77-80 e 129-135.

De JONG, K. (1975). ***An analysis of the behaviour of a class of genetic adaptive systems***. PhD thesis, University of Michigan.

DUFF, L., GRIMES, R., LEWIS, J. (1989), ***Sparse Matrix Test Problems,*** ACM Trans. Math. Sofi**.,** 15, pp. 1-14.

ESHELMAN, L.J., SCHAFFER, J.D. (1993) ***Crossover's niche***, Proc. 5th Intern. Conf. on Genetic Algorithms, S. Forrest (Ed.), Morgan Kaufmann, San Mateo, CA, pp 9-14.

ESHELMAN, L.; SCHAFFER. (1993) ***Real Coded Genetic Algorithms and Interval-Schemata.*** In Foundation of Genetic Algorithms 2, pp. 187-202.

FASANO, G., (2005)a, ***Planar Conjugate Gradient Algorithm for Large-Scale Unconstrained Optimization, Pat 1: Theory,*** Journal of Optimization Theory and Applications, Vol. 125, N$^o$ 3, pp. 523-541.

FASANO, G., (2005)b, ***Planar Conjugate Gradient Algorithm for Large-Scale Unconstrained Optimization, Pat 2: Aplications,*** Journal of Optimization Theory and Applications, Vol. 125, N$^o$ 3, pp. 543-558.

FILHO, J. L. R.; TRELEAVEN, P. C.; ALIPPI, C., (1994), ***Genetic-Algorithm Programming Environments***, IEEE Computer, vol. 27, n. 6, pp. 28 – 43, jun. 1994.

FLETCHER, R. (1975), ***Conjugate Gradient Methods for Indefinite Systems***, in Numerical Analysis Dundee, G. Watson (Ed.), Berlin,Springer-Verlag, pp.73-89.

FOGEL, L. J., ANGELINE, P. J., FOGEL, D. B., (1995), ***An evolutionary programming approach to self-adaptation on finite state machines.*** In: Proceedings of the Fourth International Conference on Evolutionary Programming, pp. 355-365.

FOX, R. L., (1971), ***Optimization Methods for Engineering Design*.** Reading, MA: Addison-Wesley.

FREUND, R., NACHTIGAL, N. (1991). ***QMR: A quasi-minimal residual method for non-Hermitian linear systems***. Math., 60, pp. 315-339.

FREUND, R., NACHTIGAL, N. (1994). ***An implementation of the QMR method based on cupled two-term recurrences***. SIAM J. Sci. Statist. Comput., 15, pp. 313-337.

GAUSS, C. F. (1823). ***Brief an Gerlinf vom.*** Werke Vol.9. pp. 278-281, translated by G. E. Forsythe (1950), in MTAC 5, 255-258.

GOLDBERG, D. E. (1989). ***Genetic Algorithms in Search, Optimization, and Machine Learning***. Reading, MA: Addison-Wesley.

GOLUB, G. H., van LOAN, C. F. (1993), ***Matrix Computations,*** 694 pages, 3rd edition, Johns Hopkins Univ. Pr; ISBN: 0801854148.

GRAVVANIS, G. A. (2000), ***Explicit Preconditioned Conjugate Gradient Schemes for Solving Biharmonic Equations***, Engineering Computations**,** Vol. 17, N. 2, pp.l 54-165.

GRIGORI, L., DEMMEL, J.W., LI, X.S. (2005), ***Parallel Symbolic Factorization for Sparse LU Factorization with Static Pivoting.*** .....

GUTKNECHT, M. H. (1992), ***A Complete Theory of Unsymmetric Lanczos Process and Related Algorithms, Part 1***, SIAM J. Matrix Anal. Appl., 13, pp. 594-639.

GUTKNECHT, M. H. (1994), ***A Complete Theory of Unsymmetric Lanczos Process and Related Algorithms, Part II***, SIAM J. Matrix Anal. Appl., 15, pp. 15-58.

HACKBUSCH, W. (1994), ***Iterative Solution of Large Sparse Systems of Equations*** (Applied Mathematical Sciences, V. 95), Springer Verlag; ISBN: 0387940642.

HAGEMAN, L., YOUNG, D. (1981), Applied Iterative Methods. New York: Academic Press.

HANSEN, N., OSTERMEIER, A., (1996), ***Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation.*** In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 312-317.

HARTMANN, S., TEBBENS, J.D., QUINT, K.J. and MEISTER, A., (2009), ***Iterative solvers within sequences of large linear systems in non-linear structural mechanics,*** ZAMM Z. Angew. Math. Mech. 89, No. 9, 711 – 728.

HASHEMI, R. and KARGARNOVIN, M.H., (2007), ***Vibration Base Identification of Impact Force Using Genetic Algorithm,*** World Academy of Science, Engineering and Technology 36.

HAUG, E. J., ARORA, J. S., (1989), ***Introduction to Optimal Design.*** New York: McGraw Hill.

HAUPT, R. L., HAUPT, S. E. (1998), ***Pratical Genetic Algorithms,*** Wiley-Interscience.

HESTENES, M. R., STIEFEL, E. (1952), ***Methods of Conjugate Gradients for Solving Linear Systems***, J. Res. Nat. Bureau ofStandards**,** 49, pp. 409-436.

HILLARY, B., EWINS, D. J. (1984). ***The Use of Strain Gages in Force Determination and Frequency. Response Function Measurements,*** Proceeding of 2nd IMAC, Orlando, pp. 627-634.

HIMMELBLAU, D. M., (1972), ***Applied Nonlinear Programming*****.** New York: McGraw Hill. http://www.netlib.org/linalg/html_templates/Templates.html.

HOLLAND, J.H. (1975). ***Adaptation in natural and artificial systems***. MIT Press, Illinois Genetic Algorithm Laboratory. IlliGAL, University of Illinois at Urbana-Champaign, pp. 11-147.

HUANG, Z. WANG C. and MA, M., (2009), ***A Robust Archived Differential Evolution Algorithm for Global Optimization Problems,*** Jounal of Computers, vol. 4, N$^{o}$. 2.

HUNT, B. R., (1972), ***A Theorem on the Difficulty of Numerical Deconvolution***, IEEE Transactions on Audio and Electro acoustics, pp. 94-95.

ISAACSON, E., KELLER, H. B. (Contributor) (1994), ***Analysis ofNumerical Methods,*** 541 pages, reprint edition, Dover Pubs.; ISBN: 0486680290.

IZMAILOV, A., SOLODOV, M. (*2005)* ***Otimização, volume 1, Condições de Otimalidade, Elementos de Análise Convexa e de Dualidade***. Rio de Janeiro, IMPA.

JACOBI, C. G. J. (1845), ***Uber eine neue Auflosungsart.*** Astr. Nachr., 22, No. 523, pp. 297-3O3.

JERONIMO, C.E.; CAMPOS SILVA, J.B., MOURA, L.F.M., (1994), ***Numerical Solution of Heat Problems Using the Finite Element Method Based on Control Volume***. Revista Brasileira de Ciências Mecânicas**,** vol. XVI, n°1 - pp. 35-50.

KAHAN, W. (1958), ***Gauss-Seidel Methods of Solving Large Systems of Linear Equations.*** Ph.D. thesis. Toronto, Canada, University of Toronto.

KRINK, T. and PATERLINI, S. (2008), ***Differential evolution for multiobjective portfolio optimization***, Working Paper Series, RECent-Center for Economic Research, Univ. of Modena and Reggio E., Italy.

LANCZOS, C. (1950), ***An Iteration Method for the Solution of the Eigenvalue Problem of Differential and Integral Operators***, J. Res. Nat. Bureau ofStandards**,** 45, pp**.** 25 5-282.

LANCZOS, C. (1952), ***Solution of Linear Equations by Minimized Iterations***, J. Res. Nat. Bureau ofStandards, 49, pp. 33-53.

LI, X., DEMMEL, J. (2003), ***SuperLU_DIST: A Scalable Distributed-memory Sparse Direct Solver for Unsymmetric linear systems.*** ACM Transactions on mathematical software, 29(2).

LIOUVILLE, J. (1837), ***Sur le développement des fonctions en series***..., II. J. Math. Fures Appl. (1), 2, pp. 16-37.

LUENBERGER, D.G. (1973), **Introduction to Linear and Non-Linear Programming**, Addison-Wesley, New York.

MAIA, N. M. M., (1989), ***An Introduction to the Singular Value Decomposition Thecnicque* (*SVD*)**, Proceedings of the 7th International Modal Analysis Conference, Las Vegas, USA, pp. 335-339.

MAIA, N., SILVA, J. *et al*, (1996), ***Theoretical and Experimental Modal Analysis*** Research studies Press Ltd., Baldock, Hertfordshire, England

MANGUOGLU, Murat, SAMEH, Ahmed H. and SCHENK, Olaf (2009)**.** ***PSPIKE: A Parallel Hybrid Sparse Linear System Solver***, Lecture Notes in Computer Science, ISSN 0302-9743, Volume 5704/2009, p.p. 797-808, Springer Berlin / Heidelberg.

MANTEUFFEL, T. (1977). **The Tchebychev iteration for nonsymmetric linear systems,** Numer. Mth., 28, pp. 307-327.

MARTINEZ, J. M.; SANTOS, S. A. (1995). ***Métodos computacionais de otimização***. XX COLÓQUIO BRASILEIRO DE MATEMÁTICA - IMPA, Rio de Janeiro (ISBN 85-244-0092-7, 256 p).

MEHMOOD, R., CROWCROFT, J. (2005), ***Parallel iterative Solutions Method for Large Sparse Linear Equations System.*** Technical Report, UCAM-CL-TR-650, ISSN 1476-2986.

MEIER-YANG, U. (1992), ***Preconditioned Conjugate Gradient-Like Methods for Linear Systems***, Tech. Rep., CSRD, University of Illinois, Urbana, IL.

MICHALEWICZ, Z. (1996). ***Genetic Algorithms + Data Structures = Evolution Programs***, 3.ed. New York: Springer-Verlag.

NOCEDAL, Jorge e WRIGHT, Stephen J. (2006). ***Numerical Optimization***. Segunda Edição. Springer Verlag, New York.

OLIVEIRA, G.T.S., (2006). ***Estudo e aplicações da evolução diferencial***. Dissertação de Mestrado – Universidade Federal de Uberlândia, Uberlândia, MG, Brasil.

ORTEGA, J. M, (1972), ***Numerical Analysis: a second curse,*** Academic Press, New Iork.

PAIGE, C. C., SAUNDERS, M. A. (1975). Solution of sparse indefinite systems of linear equations**,** SIAM J. Num. Anal., Vol. 12, No. 4, pp. 617-629.

PAIGE, C. C., SAUNDERS, M. A. (1982). ***LSQR: An algorithm for sparse linear equations and sparse least squares***. ACM Trans. Math. Soft., 8, pp.43-71.

PALM, William J., (2005). ***Introduction to MATLAB 7 for engineers***, McGraw- Hill International Edition New York.

PARLETT, B.N., TAYLOR, D.R., LIU, Z.A.(1985), ***A look-ahead Lanczos algotithm for unsymmetric matrices***, Mathematics of Computations, 44, pp. 105-124.

PRESS, W. H.; FLANNERY, B. P.; TEUKOLSKY, S. A.; VETTERLING, W. T. (1992), ***Successive Overrelaxation (SOR).*** Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed. Cambridge, England: Cambridge University Press, pp. 866-869.

RECHENBERG, I., (1973), ***Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution.*** Stuttgart: Frommann-Holzboog.

REID, J. (1971), ***On the Method of Conjugate Gradients for the Solution ofLarge Sparse Systems of Linear Equations, in Large Sparse Sets of Linear Equations***, J. Reid (Ed.), Academid Press, London, pp.23l-254.

REKLAITIS, G. V., RAVINDRAN, A., RAGSDELL, K. M., (1983), ***Engineering Optimization - Methods and Applications***. John Wiley and Sons, USA.

RUGGIERO, Márcia A. G. e LOPES, Vera L. R. (1988). - ***Cálculo numérico: Aspectos Teóricos e Computacionais***. McGraw-Hill.

SAAD, Y., SCHULTZ, M. (1986). ***GMRES: A generalized minimal residual algorithm for solving nonsymemetric linear systems.*** SIAM J. Sci. Statist. Comput., 7, pp. 856-869.

SARVANAN, N. N., FOGEL, D. B., NELSON, K. M., (1995), ***A comparison of methods for self-adaptation in evolutionary algorithms*.** BioSystems 36(2), pp. 157-166.

SCHWEFEL, H.P., (1981), ***Numerical Optimization of Computer Models.*** Chichester, UK: Wiley.

SCHWEFEL, H.P., (1987), ***Collective intelligence in evolving systems.*** In W. Wolff, C. J. Soeder and F. Drepper (Eds), Ecodynamics – Contributions to Theoretical Ecology, Berlin: Springer, pp. 95-100.

SCHWEFEL, H.P., (1995), ***Evolution and Optimun Seeking.*** John Wiley&Sons, pp. 105-124.

SCHWEFEL, H.P., BÄCK, T., (1998), ***Artificial evolution: How and why?.*** In D. Quagliarella, J. Périaux, C. Poloni and G. Winter (Eds), Genetic Algorithms and Evolution Strategies in Engineering and Computer Science: Recent Advances and Industrial Applications, Chichester, UK: Wiley, pp. 1-19.

SEIDEL, L., (1874), ***Über ein Verfahren die Gleichungen, auf welche die Methode des kleinsten Quadrate führt, sowie lineäre Gleichungen überhaupt, durch successive Annäherung aufzulösen***, Communication à la section math-physique de l'Académie Royale des Sciences de Berlin, séance du 7 fév.

SHYY, W., OUYANG, H., BLOSCH, E., THAKUR, S. S., LIU, J. (1997), ***Computational Techniques for Complex Transport Phenomena,*** Cambridge Univ. Press, New York.

SILVA, L. A., (2000), ***Técnicas de Identificação de Forças no Domínio do Tempo***. Universidade Federal de Uberlândia, Uberlândia, MG.

SLEIJPEN, G. L. G., FOKKEMA, D. R. (1993), ***Bi-CGSTAB(l) for Linear Equations Involving Unsymmetric Matrix with Complex Spectrum***, Elec. Trans. Numer. Anal. 1, pp. 11-32.

SOARES, G. L., (1997), ***Algoritmos Genéticos: Estudo, Novas Técnicas e Aplicações***. Belo Horizonte, 137p. Monografia (Especialização em Engenharia Elétrica) – Escola de Engenharia, UFMG.

SOUTHWELL, R. (1946), ***Relaxation Methods in Theoretical Physics.*** Clarendon Press, Oxford.

STEVENS, K.K. (1987), ***Force identification problem—an overview***, Proceedings of SEM Spring Conference on Experimental Mechanics, Florida, USA, pp. 838–844.

STORN, R., PRICE, K., (1995), ***Differential Evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces*.** Technical Report TR-95-012, International Computer Science Institute, Berkeley, March 1995.

STORN, R. (1996). ***On the usage of differential evolution for function optimization***. Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS). pp. 519-523.

STORN, R.; PRICE, K. (1997). ***Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces***. Journal of Global Optimization 11: 341-359.

STORN, R. (1999), ***System Design by Constraint Adaptation and Differential Evolution***. IEEE Transactions on Evolutionary Computation, v. 3, n. 1, p. 22–34.

STORN, R.M. PRICE, K.; LAMPINEN, J.A. (2005). ***Differential Evolution: A Practical Approach to Global optimization***. Springer. ISBN 978-3-540-20950-8.

SULLI, Endre, MAYERS, David (2003), ***An introduction to Numerical Analysis,*** Cambridge University Press, ISBN 0-521-00794-1.

THOMAS, G., PROST, R., (1991), ***Iterative Constrained Deconvolution***, Signal Processing vol. 23, no 1, April.

TIKHONOV, A. N., ARSENIN, V. A. (1977). ***Solutions of Ill-posed Problems***. Wiley, New York.

TONG, C. (1992), ***A Comparative Study of Preconditioned Lanczos Methods for NonSymetric Linear Systems,*** Tech. Rep. SAND91-8240, Sandia Nat. Lab., Livermore.

VAN DER VORST, H. (1992), ***Bi-CGSTAB: A Fast and Smoothly Convergeing variant of Bi-CG for the Solution of Nonsymmetric Linear Systems,*** SIAM J, Sci Statist. Comput., 13, pp 631-644.

VANDERPLAATS, G. N, (1995), ***DOT - Design Optimization Tools Program – Users anual***, Vanderplaats Research & Development, Inc, Colorado Springs.

VANDERPLAATS, G. N. (1999), ***Numerical Optimization Techniques for Engineering Design*****.** Vanderplaats Research and Development, Inc., 3rd ed.

VARGA, R. (1962), Matrix Iterative Analysis. Englewood Cliffs, NJ: Prentice-Hall.

WANG, Mingliang, KLIE, Hector, PARASHAR, Manish and SUDAN, Hari (2009). ***Solving Sparse Linear Systems on NVIDIA Tesla GPUs***. Lecture Notes in Computer Science Springer Berlin / Heidelberg ISSN0302-9743 Volume 5544/2009 pp 864-873.

WATKINS, David S. (2002). ***Fundamentals of Matrix Computations.*** J. Wiley and Sons, segunda edição, New York 2002.

WEISSTEIN, E. W. et al. ***"Conjugate Gradient Method."*** From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/ConjugateGradientMethod.html

WEISSTEIN, E. W. et al. ***"Gauss-Seidel Method."*** From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/Gauss-SeidelMethod.html

WEISSTEIN, E. W. et al. ***"Jacobi Method."*** From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/JacobiMethod.html

WEISSTEIN, E. W. et al. ***"Successive Overrelaxation Method."*** From MathWorld -A Wolfram Web Resource. http://mathworld.wolfram.com/SuccessiveOverrelaxationMethod.html

YOUNG, D.M. (1971) Iterative Solutions of Large Linear Systems. New York: Academic Press.

YOUSEFI, H, HANDROOS, H. and SOLEYMANI, A., (2008), ***Application of Differential Evolution in system identification of a servo-hydraulic system with a flexible load,*** Journal of Mechatronics, vol. 18. pp 513–528.

ZIATEV, Z. (1991), ***Computational Methods for General Sparse Matrices*** (Mathematics and Its Appiications, Vol. 65, Kiuwer Academic Pubiishers; ISBN: 0792311 54X.

ZIATEV, Z. (1981), ***Solution of Large and Sparse Systems of Linear Algebraic Equations*** (Lecture Notes in Computer Science, Vol. 121, Springer-Verlag; ISBN: 0-387-10874-2.

# Livros Grátis

( http://www.livrosgratis.com.br )

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo