CNPq – Conselho Nacional de Desenvolvimento Científico

# Relatório do Projeto

OPTIMIZE-ELTE

Desenvolvimento de Sistema Integrado para
Projeto Ótimo de Estruturas para Linhas de Transmissão
de Energia Elétrica

Coordenador do Projeto
José Herskovits Norman, Dr. Ing.

Instituição Executora
PEM-COPPE-UFRJ

Edital
CT-Energ/MCT/CNPq n° 017/2005

N° do Processo
55.0187/2005-7 – EDITAL 017/2005

Agosto de 2008

# INDICE

# RELATÓRIO DO PROJETO

## 1. INTRODUÇÃO

Nos próximos anos serão feitos no Brasil volumosos investimentos em Linhas de Transmissão de Energia Elétrica (LTE). Portanto, a utilização de ferramentas computacionais específicas para o projeto, análise estrutural e otimização das estruturas metálicas autoportantes das linhas de transmissão elétrica será sumamente conveniente e terá alto retorno financeiro. Cabe destacar que a otimização de estruturas treliçadas é uma área clássica da Otimização Estrutural que trata do projeto ótimo de elementos estruturais e de sistemas utilizados em diversos campos da Engenharia. A mesma foi inicialmente aplicada na indústria aeroespacial, onde a redução do peso estrutural é de capital importância.

Este documento descreve o sistema OTIMIZE-ELTE, um sistema integrado de análise e otimização estrutural para o projeto de estruturas treliçadas para linhas de transmissão de energia elétrica. Mediante OPTIMIZE-ELTE é possível determinar as dimensões e a forma geométrica da estrutura de "custo" mínimo que verifica um conjunto de restrições de projeto, isto é, restrições geométricas e mecânicas. O mesmo poderá ser utilizado por Empresas de Consultoria ou Fabricantes de Estruturas para realizar a verificação estrutural das estruturas metálicas ou para a redução do seu custo final. Cabe destacar que a otimização do projeto das estruturas das linhas de transmissão, segundo experiências anteriores, resulta em economias do peso da ordem de 10 a 20%, sendo que o material utilizado em estruturas autoportantes de 750KV é da ordem de 20 toneladas/Km.

Como função objetivo o sistema OPTIMIZE-ELTE considera o custo total dos materiais empregados na fabricação da estrutura. As restrições de projeto são divididas em dois grupos: as de origem mecânica e as geométricas. As de origem mecânica tem relação com as tensões admissíveis dos materiais que compõem os elementos estruturais da estrutura, máximos deslocamentos nodais permitidos e o valor mínimo da freqüência natural da estrutura. As restrições geométricas que poderão ser consideradas podem ser divididas em duas classes: as relacionadas às simetrias da estrutura e as relacionadas à posição dos nós da estrutura. As restrições de simetria são consideradas definindo sistemas de coordenadas simétricos uns em relação a outros e definindo variáveis de posição comuns a nós simétricos. Esta estratégia para a consideração das simetrias da estrutura tem a grande vantagem da redução do número de variáveis do problema de otimização e a não introdução das restrições explícitas, o que permite a solução mais eficiente, sobretudo nos casos das estruturas com grande número de nós. Por outra parte, barras com igual seção, por exemplo aquelas em posições simétricas, são consideradas atribuindo-lhes a mesma variável de seção, o que permite uma redução adicional do número de variáveis do problema de otimização. Outras restrições geométricas relacionadas com a posição de um nó em particular podem ser consideradas pelo sistema assim como também condições de colinearidade entre três ou mais nós da estrutura.

OPTIMIZE-ELTE utiliza o "solver" para Otimização Não Linear FAIPA_F90, desenvolvido pela equipe do Laboratório Interdisciplinar de Otimização em Engenharia da COPPE em cooperação com o Centro de P&D da RENAULT em Paris, e registrado no INPI. Este se baseia no FAIPA, *Feasible Arc Interior Point Algorithm*, método numérico para Otimização Não Linear desenvolvido pelo responsável do projeto e bastante utilizado no meio acadêmico e por importantes industrias no país e no exterior.

Soluções ótimas com variáveis discretas podem ser procuradas pelo sistema mediante a realização de uma busca heurística a partir do mínimo contínuo, de modo a permitir a utilização dos perfis normalizados disponíveis no mercado.

A interface simples do sistema OPTIMIZE-ELTE a través de arquivos de entrada-saída de texto formatado permitirá, no futuro, a incorporação das facilidades mais modernas de Projeto Assistido por Computador para constituir-se numa ferramenta "amigável" e eficiente, permitindo a efetiva utilização pelo meio produtivo. Pré e Pós - processadores gráficos poderão ser utilizados reduzindo o custo de elaboração do projeto. Estes processadores porão a disposição do Engenheiro projetista toda a informação necessária originada em normas técnicas e critérios mecânicos de projeto, gerando automaticamente os dados de entrada mais rotineiros. Também poderão ser definidas coleções "customizadas" de estruturas utilizadas em linhas de transmissão, reduzindo de esta forma ao máximo o trabalho prévio a execução do sistema OPTIMIZE-ELTE para análise e otimização da estrutura escolhida.

O sistema OPTIMIZE-ELTE é programado na linguagem FORTRAN e utiliza rotinas para álgebra linear computacional altamente portáveis, o que possibilita uma extensão deste projeto e sua integração futura num Sistema Multidisciplinar de Otimização que incorpore outras disciplinas necessárias para o Projeto de Linhas de Transmissão. Em particular, a Otimização da Trajetória da Linha de Transmissão, a Localização Ótima das Torres e a Otimização Geométrica e Eletromagnética de Cabos e Isoladores.

Entre os pontos inovadores do sistema desenvolvido, destaca-se:

i) A utilização de um novo algoritmo heurístico para otimização não linear com variáveis inteiras.

ii) Utilização de uma nova técnica para otimização com restrições nas freqüências naturais da estrutura.

iii) Integração em problemas de porte real da otimização dimensional e geométrica das estruturas treliçadas.

Este documento conta com seis seções e quatro apêndices. A Seção 2 descreve o modelo de otimização utilizado, função objetivo e restrições do projeto. Esta seção descreve brevemente o algoritmo FAIPA para otimização não linear e a heurística desenvolvida para otimização não linear com variáveis discretas.

A Seção 3 descreve detalhadamente o sistema OPTIMIZE-ELTE para análise e otimização de estruturas treliçadas. A Seção 4 apresenta alguns exemplos de utilização do sistema. Esta seção, além de ilustrar as funcionalidades do sistema, pode ser utilizada como base para a elaboração de um texto complementário para o treinamento do engenheiro usuário.

Outras contribuições e publicações científicas realizadas no marco do projeto OPTIMIZE-ELTE não necessariamente envolvidas na versão final do programa são apresentadas na Seção 5. Finalmente, a Seção 6 fornece uma lista das publicações referenciadas neste texto.

O Apêndice A corresponde ao manual de usuário do sistema OPTIMIZE-ELTE. O Apêndice B corresponde ao manual de usuário do Módulo de análise estrutural, sistema utilizado pelo OPTIMIZE-ELTE para análise estrutural e de sensibilidades. O mesmo foi desenvolvido para utilização do sistema OPTIMIZE-ELTE, porém, concebido independentemente de forma de permitir a sua utilização por separado.

Os Apêndices C e D descrevem, mais detalhadamente o Algoritmo FAIPA e a heurística desenvolvida para problemas não lineares de variáveis discretas. O Apêndice E descreve a metodologia para inclusão de aproximações no FAIPA.

# 2. METODOLOGIA

A pesquisa se apóia em técnicas gerais e robustas como o Método dos Elementos Finitos, Algoritmos de Pontos Interiores para problemas de programação matemática. Neste capítulo descreve-se um conjunto de ferramentas utilizadas ao longo do projeto e implementadas no software OPTIMIZE-ELTE.

O capítulo está organizado da seguinte forma, na Seção 2.1 se apresenta o problema de otimização nas variáveis continuas do projeto de torres de linhas de transmissão. No final desta seção se mostra a formulação matemática. Na Seção 2.2 se explica o procedimento computacional a traves do Algoritmo de Pontos Interiores por Arcos Viáveis, FAIPA e de uma heurística desenvolvida para obter uma solução discreta.

## 2.1. Modelo de Otimização

Considera-se uma estrutura otimizada aquela que submetida a um conjunto de carregamentos (peso próprio, ação do vento, etc), possua a menor quantidade de material, sem que certas restrições mecânicas (deslocamentos/tensões) ou geométricas (posição/orientação dos elementos estruturais) sejam violadas.

O modelo de otimização empregado se apresenta da seguinte forma: na Seção 2.1.1 se define o modelo estrutural utilizado para o cálculo das respostas mecânicas. Na Seção 2.1.2. se especificam as variáveis de projeto. Na Seção 2.1.3. se define a função custo. Na

Seção 2.1.4. se definem as funções de restrição e finalmente, na Seção 2.1.5., se formula o problema de programação matemático.

### 2.1.1. Modelo estrutural

As torres de linha de transmissão são modeladas por treliças espaciais baseadas no Método dos Elementos Finitos [1]. O modelo consiste em um conjunto de $n_{elem}$ barras ou elementos $\{b_1,...,b_i,...,b_{n_{elem}}\}$ articuladas num conjunto de vértices (ou nós). Cada barra $b_i$ tem um comprimento $l_i$, uma seção transversal de área $x_i$ e está associado a um tipo de material $m_i$ (Figura 1).

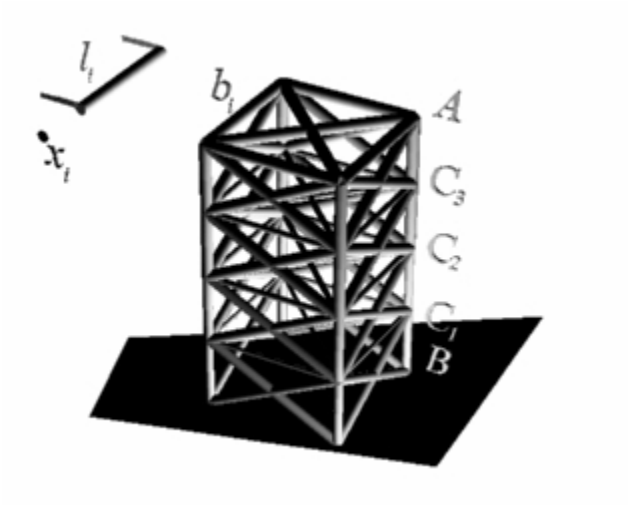



Figura 1: Treliça espacial.

O conjunto de apoios e carregamentos aplicados na estrutura é modelado com um conjunto de deslocamentos prescritos e forças nodais equivalentes. As respostas mecânicas (deslocamento, tensões, etc.) são obtidas da resolução de sistemas lineares.

### 2.1.2. Variáveis de projeto

As variáveis de projeto consideradas são as áreas das seções transversais das barras e as posições dos vértices da estrutura.

As variáveis de projeto estão organizadas no vetor coluna $x \in R^n$ da seguinte forma:

$$x = \begin{bmatrix} x_A \\ x_P \\ x_C \end{bmatrix}.$$

onde $n = n_A + n_P + n_C$ é o número de variáveis de projeto, $n_A$, $n_P$ e $n_C$ são o número de variáveis de área das seções transversais, posição e colinearidade respectivamente. Os vetores coluna $x_A \in R^{n_A}$, $x_P \in R^{n_P}$ e $x_C \in R^{n_C}$ são as variáveis de seção, posição e colinearidade respectivamente. Em geral $n_A$ será da ordem do número de barras, $n_P$ da ordem do número de graus de liberdade $n_{dof}$ e $n_C$ igual ao número de restrições de colinearidade.

### 2.1.3. Função custo (ou função objetivo)

A função custo é dada por:

$$f(x) = \sum_{i=1}^{n_A} c_i \rho_i l_i x_i$$

sendo $\rho_i$ e $c_i$ o peso específico (kg/m$^3$) e o custo unitário (\$/kg) do material $m_i$ respectivamente.

### 2.1.4. Funções de restrição

São considerados os seguintes tipos de restrições:

i) mecânicas

ii) geométricas

#### 2.1.4.1. Restrições mecânicas

O vetor de deslocamentos da estrutura $u \in R^{n_{dof}}$ para o estado de carregamento $p \in R^{n_{dof}}$ é obtido através da resolução do sistema linear:

$$K(x)u = p$$

onde $K(x) \in R^{n_{dof} \times n_{dof}}$ é a matriz de rigidez estrutural.

Em resume, as restrições mecânicas são agrupadas em uma única função vetorial *g* dada por:

$$g(x) = \begin{bmatrix} \dfrac{u_{D_1}(x) - u_{\max}}{u_{\max}} \\ \dfrac{u_{D_2}(x) - u^{\min}}{u^{\min}} \\ \dfrac{\sigma_{S_1}(x) - \sigma_{trac}^{adm}}{\sigma_{trac}^{adm}} \\ \dfrac{\sigma_{comp}^{adm} - \sigma_{S_2}(x)}{\sigma_{comp}^{adm}} \\ \dfrac{\omega^{carregamento} - \omega_{\min}^{natural}}{\omega^{carregamento}} \end{bmatrix}$$

onde as diversas componentes do vetor $g(x)$ são descritas nas seções seguintes.

#### 2.1.4.1.1. Restrição de deslocamento

Definem-se sub-conjuntos de graus de liberdade $D_1, D_2 \subseteq \{1,...,n_{dof}\}$ nos quais se restringe o deslocamento a certos valores máximos ou mínimos para qualquer estado de carregamento. Sejam $u_i^{\min} < 0$ e $u_j^{\max} > 0$ o mínimo e máximo valor de deslocamento admissível do grau de liberdade $i \in D_1$ e $j \in D_2$ respectivamente. As restrições de deslocamento se definem da seguinte forma:

$$\frac{u_i - u_i^{\max}}{u_i^{\max}} \leq 0 \text{ para todo } i \in D_1$$

$$\frac{u_j - u_j^{\min}}{u_j^{\min}} \leq 0 \text{ para todo } j \in D_2$$

ou vetorialmente:

$$\frac{u_{D_1} - u^{\max}}{u^{\max}} \leq 0, \ \frac{u_{D_2} - u^{\min}}{u^{\min}} \leq 0$$

#### 2.1.4.1.2. Restrição de tensão

Definem-se um sub-conjunto de elementos (barras) $S_1, S_2 \subseteq \{1,...,n_{elem}\}$ nos quais se restringe o valor da tensão por tração e compressão para qualquer estado de carregamento. Seja uma barra *i* constituída de um material $m_i$ e sejam $\sigma_{trac_i}^{adm} > 0$ e $\sigma_{comp_i}^{adm} > 0$ (N/m$^2$) os valores admissíveis de tração e compressão definidos para o material $m_i$ respectivamente. A restrição de tensão por tração para o elemento *i* se define da seguinte forma:

$$\frac{\sigma_i - \sigma_{trac_i}^{adm}}{\sigma_{trac_i}^{adm}} \leq 0 \text{ para todo } i \in S_1$$

E a restrição de tensão por compressão para o elemento *j* se define da seguinte forma:

$$\frac{\sigma_{comp_j}^{adm} - \sigma_j}{\sigma_{comp_j}^{adm}} \leq 0 \text{ para todo } j \in S_2$$

onde $\sigma_k$ é a tensão do elemento *k* de material $m_k$. Vetorialmente as condições acima podem escrever-se como:

$$\frac{\sigma_{S_1}(x) - \sigma_{trac}^{adm}}{\sigma_{trac}^{adm}} \leq 0$$
$$\frac{\sigma_{comp}^{adm} - \sigma_{S_2}(x)}{\sigma_{comp}^{adm}} \leq 0$$

#### 2.1.4.1.3. Restrição na freqüência natural da estrutura

A equação de movimento do sistema é:

$$M\ddot{u} + Ku = 0$$

onde *M* e *K* são as matrizes globais de massa e rigidez do modelo em elementos finitos, e *u* é o vetor de deslocamentos nodais.

Supondo soluções do tipo $u = ve^{i\omega t}$, chega-se ao seguinte problema de autovetor-autovalor:

$$Kv = \lambda Mv ,$$

onde os autovalores $\lambda$ são o quadrado das freqüências naturais do sistema ($\lambda = \omega^2$) e os autovetores associados (*v*) representam os modos de vibração da estrutura.

A solução de problema de autovetor/autovalor é feita por um algoritmo iterativo, obtendo-se assim as freqüências naturais $\omega$ da estrutura.

O desejável numa estrutura sob carregamentos dinâmicos é que todas suas freqüências naturais sejam bem maiores do que a máxima freqüência possível dos carregamentos. Para garantir isto, precisa-se impor apenas que a menor das freqüências naturais da estrutura seja maior do que um valor previamente conhecido que considere a maior freqüência possível dos carregamentos multiplicada por um coeficiente de segurança. Em geral, as maiores freqüências naturais da estrutura correspondem a modos

de vibração que são rapidamente dissipados, e, portanto, as menores são as mais críticas em se tratando de fenômenos como a ressonância. Com isto, impede-se que a estrutura falhe sob carregamentos dinâmicos.

Assim, a restrição das freqüências naturais fica:

$$\omega_{\min}^{natural} \geq \omega^{carregamento},$$

onde $\omega_{\min}^{natural} = \min\{\omega_1^{natural},...,\omega_{n_{dof}}^{natural}\}$.

Escrita de maneira padrão e normalizada a restrição fica:

$$\frac{\omega^{carregamento} - \omega_{\min}^{natural}}{\omega^{carregamento}} \leq 0.$$

A função $\omega_{\min}^{natural}$ não é diferenciável, por isto, no algoritmo FAIPA é utilizado o gradiente de uma aproximação diferenciável da função $\omega_{\min}^{natural}$. Para isto, no processo de otimização usa-se uma metodologia com modelos aproximados, como explicado em detalhe na Seção 5.1 e no Apêndice E.

#### 2.1.4.2. Restrições geométricas

##### 2.1.4.2.1. Restrições nas áreas das seções transversais das barras

Sejam $x_{A_i}^{\min}$ e $x_{A_i}^{\max}$ mínimo e máximo valor da variável de área *i* respectivamente. Definem-se as seguintes restrições de mínimo e máximo para as variáveis de área:

$$x_A^{\min} \leq x_A \leq x_A^{\max}$$

##### 2.1.4.2.2. Restrições nas variáveis de posição

Sejam $x_{P_i}^{\min}$ e $x_{P_i}^{\max}$ mínimo e máximo valor da variável de posição *i* respectivamente. Definem-se as seguintes restrições de mínimo e máximo para as variáveis de posição:

$$x_P^{\min} \leq x_P \leq x_P^{\max}$$

##### 2.1.4.2.3. Restrições de colinearidade

Estas restrições geométricas forçam o alinhamento de vértices que se encontram entre um par de vértices. Na estrutura da Figura 1, por exemplo, pode-se estabelecer que os vértices $C_1$, $C_2$ e $C_3$ estejam na reta definida pelos vértices *A* e *B*. Neste caso se

estabelecem três restrições de colinearidade. Assim uma restrição de colinearidade por exemplo para que $C_1$ esteja na reta definida pelos vértices *A* e *B* se estabelece com seguinte equação vetorial:

$$h_1(x) = (1-\alpha_1)A + \alpha_1 B - C_1 = 0$$

Onde $\alpha_1 \in R$ é uma variável de projeto de colinearidade (define a posição do ponto $C_1$ em relação aos pontos *A* e *B*). O conjunto de restrições de colinearidade é agrupado na seguinte equação vetorial:

$$h(x) = 0$$

Onde *h* é uma função vetorial não linear (dado que os vértices dependem das variáveis de projeto de posição).

### 2.1.5. Formulação matemática

O problema de otimização se escreve da seguinte maneira: achar o vetor *x* de variáveis de projeto que minimiza a função custo $f(x)$ sujeito as restrições mecânicas e geométricas definidas na Seção 2.1.4. Matematicamente:

$$\begin{cases} \min\limits_{x \in R^n} f(x) \\ sujeito\ a: \quad g(x) \le 0 \\ \qquad\qquad\quad h(x) = 0 \\ \qquad\quad x_{\min} \le x_A, x_P \le x_{\max} \end{cases}$$

Onde *f* é a função custo descrita na Seção 2.1.3., as funções vetoriais *g* e *h* foram definidas na Seção 2.4.1. respectivamente. As restrições laterais para as variáveis de projeto são definidos como:

$$x_{\min} = \begin{bmatrix} x_A^{\min} \\ x_P^{\min} \end{bmatrix}, x_{\max} = \begin{bmatrix} x_A^{\max} \\ x_P^{\max} \end{bmatrix}$$

sendo $x_A^{\min}$ e $x_A^{\max}$ definidos na Seção 2.1.4.2.1., $x_P^{\min}$ e $x_P^{\max}$ definidos na Seção 2.1.4.2.2.

## 2.2. Procedimento computacional

A otimização dimensional e geométrica das torres é realizada nas seguintes etapas consecutivas:

i) Otimização Contínua: Otimização não linear continua tomando como variáveis de projeto as seções das barras e as coordenadas nodais.

ii) Otimização Discreta: Fixa-se a geometria e realiza-se uma busca heurística de uma estrutura ótima com dimensiones discretas das seções transversais das barras.

iii) Otimização Contínua: Fixam-se as seções transversais das barras e realiza-se uma otimização da geometria tomando como variáveis de projeto as coordenadas nodais.

As etapas (i) e (iii) são consideradas clássicas na literatura de otimização estrutural. Para resolve-las utiliza-se o algoritmo FAIPA. A etapa (ii) utiliza uma busca heurística entorno da solução ótima encontrada na etapa (i). Nas seções seguintes se descreve o algoritmo FAIPA e a heurística utilizada. Estas técnicas são explicadas mais em detalhe nos apêndices deste documento.

### 2.2.1. FAIPA: Otimização em variáveis continuas

O FAIPA, do inglês "Feasible Arc Interior Point Algorithm", proposto por Herskovits [2] é um algoritmo de Pontos Interiores para problemas de otimização não-lineares. Ele faz iterações nas variáveis de projeto (variáveis primais) e nos multiplicadores de Lagrange (variáveis duais) até satisfazer as condições necessárias de otimalidade de primeira ordem de Karush-Kuhn-Tucker. Isto quer dizer que este algoritmo obtém em forma geral um mínimo local do problema matemático descrito na Seção 2.1.5.

O FAIPA parte de um ponto inicial interior à região viável, isto é, a região definida pelo conjunto de pontos que satisfazem as restrições de igualdade e desigualdade. Logo ele gera uma seqüência de pontos dentro desta região que converge a um mínimo local do problema de otimização. Dado o ponto da iteração atual, o FAIPA realiza uma busca linear inexata ao longo de um arco. Para calcular o arco, o FAIPA resolve três sistemas lineares com a mesma matriz de coeficientes. A busca linear define o próximo ponto da seqüência, também interior à região viável e que possui um valor menor para uma função potencial convenientemente definida.

### 2.2.2. Heurística para a otimização em variáveis discretas

Uma solução discreta consiste em uma escolha das variáveis de projeto dentro de um espaço discreto, onde existe um conjunto finito de valores para as variáveis de projeto. No caso do projeto das torres é de interesse na prática a busca da melhor escolha de seções transversais das barras dentro de um conjunto discreto de barras dado pelo fabricante.

A heurística utilizada neste projeto permite achar uma solução discreta do problema de otimização. Parte-se de uma solução discreta viável, um ponto do espaço discreto que

verifica as restrições do problema descrito na Seção 2.1.5. Em cada iteração se calculam os pontos discretos adjacentes, viáveis e de descida. Utiliza-se uma estratégia GREEDY (localmente ótima) [3] para decidir a qual ponto avançar. O próximo ponto discreto é sorteado entre aqueles de maior declive da função custo. Caso não existam pontos discretos para onde avançar o procedimento decide se continuar a busca a partir de um ponto adjacente aumentando o valor da função custo. Esta heurística finaliza quando há ciclagem ou quando atinge um número máximo de iterações sem decréscimo da função objetivo.

# 3. DESCRIÇÃO DO SISTEMA OPTIMIZE-ELTE

Neste capítulo apresentam-se as funcionalidades e organização do programa OPTIMIZE-ELTE. Este sistema é uma implementação da metodologia descrita na Seção 2.

O sistema está organizado em três módulos:

i) Análise (*analysis*)

ii) Otimização (*optimizer*)

iii) Entrada-Saída (*io*)

Cada módulo fornece um conjunto de sub-rotinas, passiveis de serem invocadas por outros módulos ou programas. O módulo de Análise prove sub-rotinas para obter o custo da torre, deslocamentos nodais e tensões das barras assim como a análise de sensibilidade (cálculo de derivadas). O módulo de Otimização utiliza uma implementação do FAIPA na linguagem Fortran 90 e implementa a heurística para busca de soluções discretas. Por último, o módulo de Entrada-Saída se ocupa da interação do programa-usuário. Esquematicamente, o sistema pode ver-se de acordo ao seguinte diagrama:

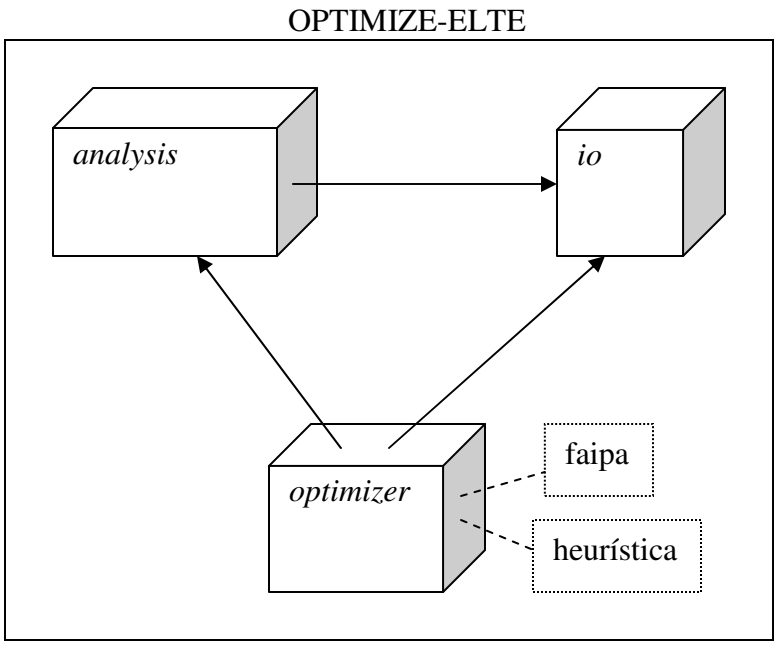



Figura 2: Diagrama de módulos do sistema OPTIMIZE-ELTE.
As flechas representam a relação de uso entre os módulos.

### 3.1. Funcionalidades

OPTIMIZE-ELTE permite minimizar o custo total da torre associado ao material utilizado nas barras da estrutura. A estrutura de mínimo custo verifica restrições de origem mecânica como deslocamento, tensão e freqüência natural. Também verifica restrições geométricas de colinearidade, valores mínimos e máximos para as variáveis de seção e posição.

OPTIMIZE-ELTE utiliza o módulo de Análise e, portanto, o usuário pode solicitar ao sistema a realização de uma análise linear estática ou análise de freqüências naturais da estrutura. O módulo de Análise também oferece um conjunto de sub-rotinas Fortran 90 para o cálculo da sensibilidade das respostas mecânicas frente a perturbações nas variáveis de projeto.

### 3.2. Módulo de Análise Estrutural e de Sensibilidade

Este módulo implementa um conjunto de sub-rotinas escritas na linguagem Fortran 90 destinadas à análise estrutural e cálculo das derivadas da função custos e derivadas das funções de restrição. O módulo foi desenhado especialmente para o sistema OPTIMIZE-ELTE, mas também se pode usar de forma independente como se sugere no apêndice B.

### 3.3. Módulo de Otimização

Este módulo contém uma implementação Fortran 90 do FAIPA denominada FAIPA-F90. A mesma foi desenvolvida no laboratório OptimizE da Universidade Federal do Rio de Janeiro. Para utilizar o código FAIPA-F90 é necessário fornecer o valor da função objetivo das funções de restrição e suas derivadas. Também se inclui neste módulo uma sub-rotina que implementa a Heurística que busca uma solução discreta nas variáveis de seção transversal das barras.

### 3.4. Módulo de Entrada/Saída

A interface de usuário da presente versão do programa OPTIMIZE-ELTE é através de arquivos. O usuário deve especificar em um arquivo todos os parâmetros do problema de otimização. Opcionalmente pode modificar o arquivo de configuração do FAIPA. O módulo de Entrada/Saída realiza a leitura do arquivo de parâmetros do problema de otimização e de configuração do FAIPA. Durante e após a execução do sistema, este módulo escreve em arquivos os resultados solicitados pelo usuário. As informações guardadas nos arquivos de saída dependerão das opções de execução e parâmetros de configuração do FAIPA.

### 3.5. Tipos de execução

OPTIMIZE-ELTE possui quatro tipos de execuções, as mesmas são:

i) Análise Estrutural

O primeiro tipo de execução realiza uma análise estrutural. Não se realiza nenhuma mudança das variáveis de projeto nesta execução. Com esta opção o usuário pode solicitar diversas respostas mecânicas da estrutura, entre elas:

(a) Deslocamentos nodais

(b) Tensões nas barras

(c) Reações nos nós de apoio

(d) Freqüência natural

Esta informação é impressa no arquivo de saída.

ii) Otimização Contínua

Neste tipo de execução o programa encontra um mínimo local do problema de otimização através do FAIPA onde as variáveis de projeto podem assumir valores contínuos. As variáveis de projeto podem ser seções transversais das barras e posições nodais.

iii) Otimização Contínua-Discreta

O terceiro tipo de execução se realiza em duas fases.

(a) Fase Contínua: Na primeira fase otimiza-se de forma idêntica ao tipo de execução (ii).

(b) Fase Discreta: Na segunda fase acha-se uma solução discreta viável "próxima" da solução da fase (i). A partir desta solução discreta executa-se um procedimento heurístico para obter uma solução discreta para as variáveis de seção transversal das barras. Neste processo, as posições nodais não são alteradas, o único que muda são as áreas das seções transversais.

iv) Otimização Contínua-Discreta-Contínua

O quarto tipo de execução se realiza em três fases.

(a) – (b) Fases Contínua-Discreta: Na primeira e segunda fase o sistema procede de forma idêntica ao tipo de execução (iii).

(c) Fase Contínua: Na terceira fase considera um problema de otimização onde o ponto inicial é o resultado da fase (b) e as variáveis de projeto são as posições nodais unicamente. O programa fixa as variáveis de seção transversal e encontra um mínimo local através do FAIPA. As variáveis de projeto podem assumir valores contínuos.

OPTIMIZE_ELTE imprime no arquivo de saída às iterações do FAIPA (Fases Contínuas (a) e (c)) e da heurística da busca discreta (Fase Discreta (b)). Opcionalmente, o usuário pode solicitar uma análise estrutural da estrutura achada e imprimir os resultados no arquivo de saída.

# 4. Exemplos de utilização do sistema OPTIMIZE-ELTE

Apresentam-se dois exemplos de otimização de torres com OPTIMIZE-ELTE. No primeiro exemplo otimiza-se uma torre pequena de 42 barras (Figura 3). Este exemplo permitirá ilustrar as principais funcionalidades do sistema OPTIMIZE-ELTE. O segundo exemplo otimiza-se uma torre tipo Delta de 499, popularmente usada em Linhas de Transmissão de Energia Elétrica (Figura 4).

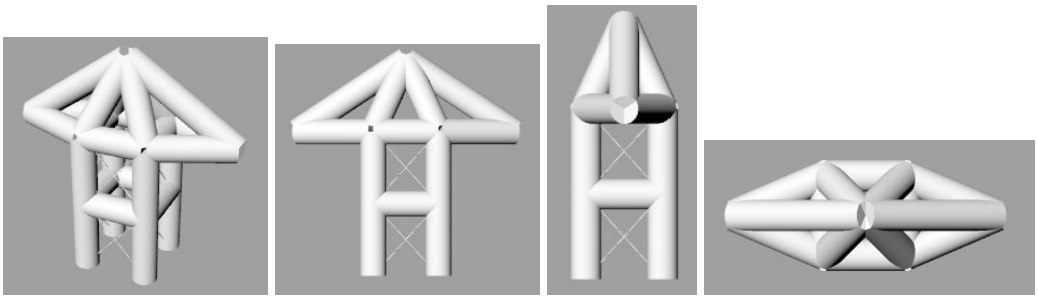


Figura 3: Diferentes pontos de vista da estrutura do Exemplo 1.

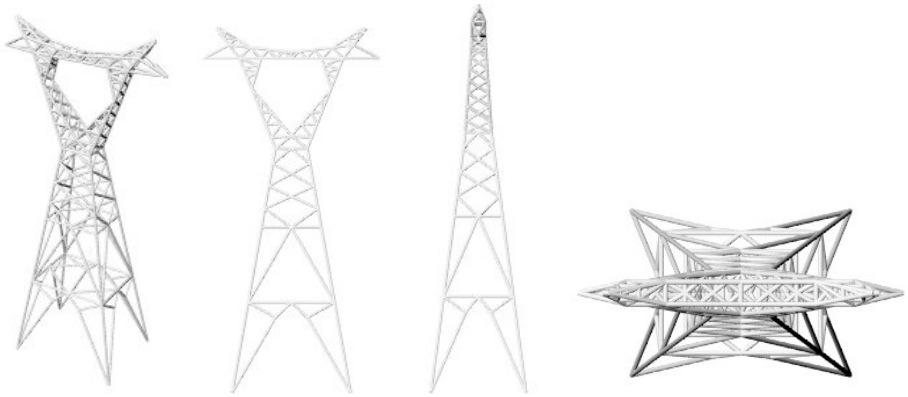


Figura 4: Diferentes pontos de vista da estrutura do Exemplo 2.

### 4.1. Exemplo 1

#### 4.1.1. Dados da estrutura

Nesta seção se apresentam os dados do arquivo de configuração do problema de otimização. Este arquivo pode ser consultado no Apêndice A.

As variáveis de seção se mostram na Figura 5. As variáveis de projeto vão de 1 a 6 e as restantes são fixas (seções 7 e 8).

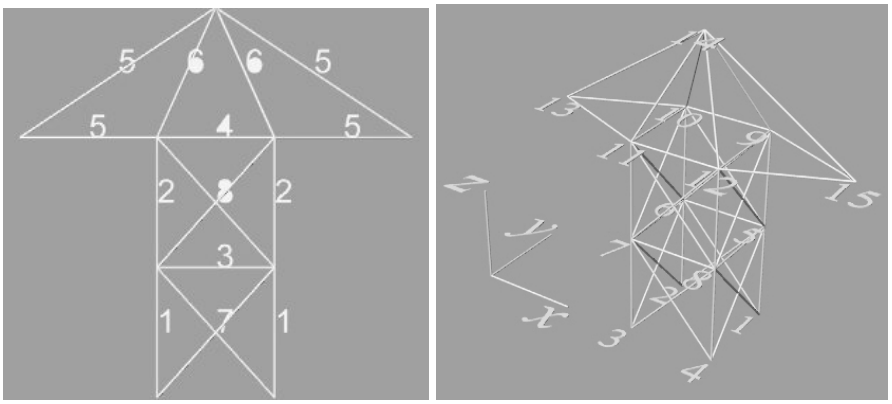


Figura 5: Variáveis de projeto de seção transversal (esquerda) e numeração dos nós (direita).

As variáveis de seção possuem um valor inicial de $1x10^{-1}m^2$. As seções fixas 7 e 8 permanecem num valor constante de $1x10^{-4}m^2$. As variáveis de seções estão limitadas inferiormente por uma área mínima de $1x10^{-4}m^2$.

As variáveis de posição se descrevem na seguinte tabela:

| Número Variável de Posição | Número de Nó | Grau de liberdade |
|---|---|---|
| 1 | 1,2,3,4 | *x* |
| 2 | 1,2,3,4 | *y* |
| 3 | 5,6,7,8 | *x* |
| 4 | 5,6,7,8 | *y* |
| 5 | 9,10,11,12 | *x* |
| 6 | 9,10,11,12 | *y* |
| 7 | 14 | *z* |
| 8 | 13,15 | *z* |

Na Figura 5 se mostra a numeração dos nós. Assim, por exemplo, a variável de posição 1 representa a posição no eixo *x* dos nós 1 a 4. Consideram-se fixas as posições verticais dos nós 1 a 12.

As propriedades do material utilizado são as seguintes:

| | |
|---|---|
| Módulo de Young | $200x10^6$ N/m$^2$ |
| Densidade | $7.8x10^3$ kg/m$^3$ |
| Tensão de escoamento | $250x10^6$ N/m$^2$ |

No presente problema são consideradas restrições de tensão em todas as barras da estrutura. A tensão máxima admissível é de 250 MPa.

Consideram-se três estados de carregamento. O primeiro consiste em forças verticais nos vértices 13 e 15. A segunda é um carregamento assimétrico no vértice 13. E o terceiro uma distribuição constante de carregamentos no eixo *x* aplicados nos vértices 1, 4, 5, 8, 9, 12, 15 e 14 (Figura 6).

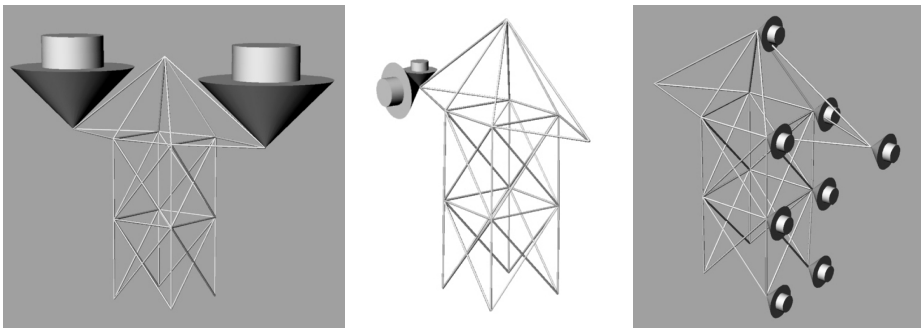


Figura 6: Estados de carregamento 1, 2 e 3.

Na seguinte tabela se detalham os três estados de carregamento.

| Estado de carregamento | Aplicada no vértice | Intensidade (N) | | |
|---|---|---|---|---|
| | | Fx | Fy | Fz |
| 1 | 13, 15 | 0 | 0 | -200 |
| 2 | 13 | 0 | 160 | -100 |
| 3 | 1, 4, 5, 8, 9, 12, 15, 14 | -10 | 0 | 0 |

Consideram-se também restrições geométricas de colinearidade nas barras verticais. Por último se considera uma restrição de deslocamento nos vértices 13 e 15 de no máximo 8 cm nos eixos *x* e *y* e 1 mm no eixo *z*.

### 4.1.2. Otimização Contínua

Para obter a solução ótima da torre, deve-se colocar "1" na opção de execução do arquivo de configuração do problema, ou seja:

```
% Opções de execução
% 0=Análise,1=Contínua,2=Contínua-Discreta,3=Contínua-Discreta-Contínua
1
```

Com a execução do programa OPTIMIZE_ELTE se obteve o seguinte resultado (Figura 7):

| Variáveis de Área | Valor ($m^2$) |
|---|---|
| 1 | 0.2293514225D-02 |
| 2 | 0.2274079570D-02 |
| 3 | 0.1000001785D-03 |
| 4 | 0.5961777301D-03 |
| 5 | 0.2158403466D-02 |
| 6 | 0.4073376491D-03 |

| Variáveis de Posição | Valor (m) |
|---|---|
| 1 | 0.8948765809D+00 |
| 2 | 0.5479345114D+00 |
| 3 | 0.9474382905D+00 |
| 4 | 0.4739672560D+00 |
| 5 | 0.9999999998D+00 |
| 6 | 0.4000000002D+00 |
| 7 | 0.3500000000D+01 |
| 8 | 0.3142602587D+01 |

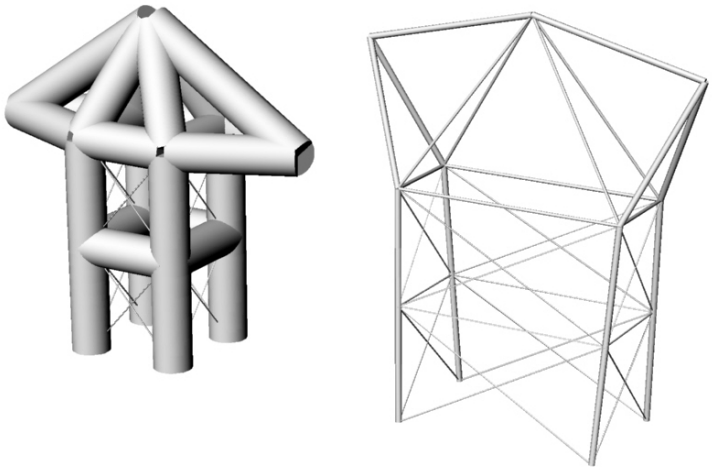


Figura 7: Estrutura inicial (esquerda), estrutura ótima (direita).

### 4.1.3. Otimização Discreta de Área

Adotam-se os seguintes valores discretos para as áreas das seções transversais ($cm^2$):

1, 10, 20, 40

Para obter a solução discreta da torre, deve-se colocar "2" na opção de execução do arquivo de configuração do problema, ou seja:

```
% Opções de execução
% 0=Análise,1=Contínua,2=Contínua-Discreta,3=Contínua-Discreta-Contínua
2
```

O resultado foi o seguinte (Figura 8):

| Variáveis de Área | Valor ($cm^2$) |
|---|---|
| 1 | 40 |
| 2 | 40 |
| 3 | 1 |
| 4 | 1 |
| 5 | 40 |
| 6 | 10 |

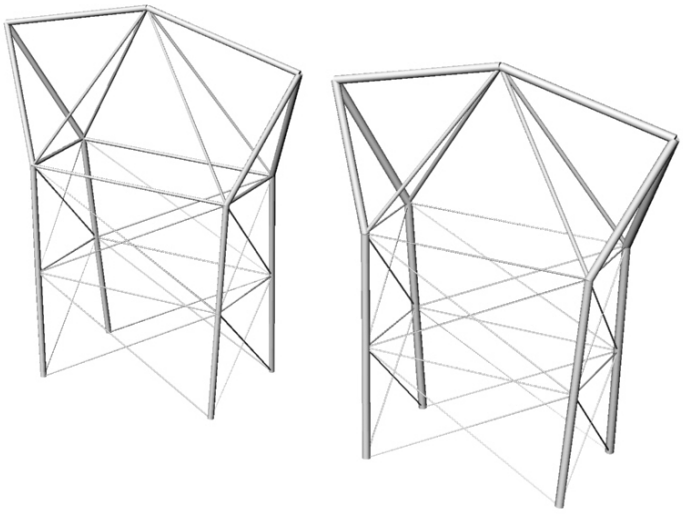


Figura 8: Estrutura ótima (esquerda), estrutura discreta (direita).

### 4.2. Exemplo 2

#### 4.2.1. Dados da estrutura

Neste exemplo se consideram 8 variáveis de seção. Na Figura 5 cada cor representa um grupo de barras com a mesma variável de seção.

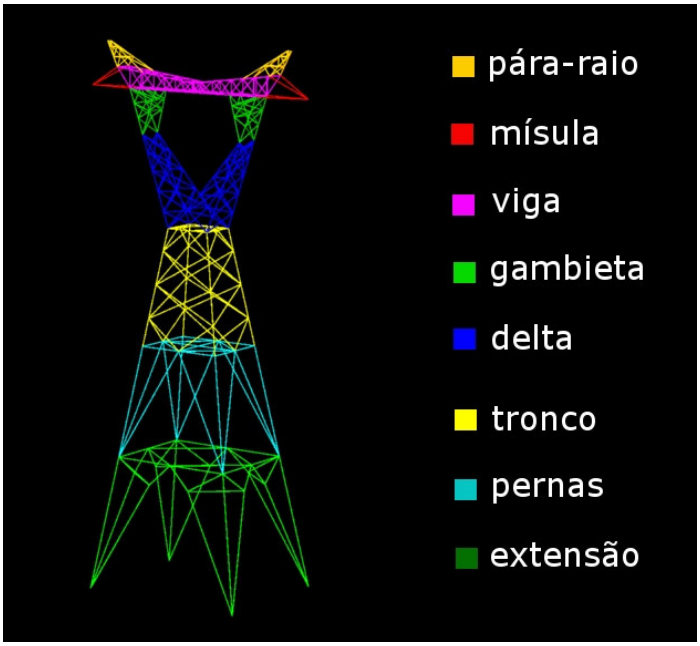



Figura 9: Variáveis de projeto.

As variáveis de seção possuem um valor inicial de $90x10^{-4}m^2$. As variáveis de seções estão limitadas inferiormente por uma área mínima de $1.12x10^{-4}m^2$.

As propriedades do material utilizado são as seguintes:

| | |
|---|---|
| Módulo de Young | $200x10^6$ N/m$^2$ |
| Densidade | $7.8x10^3$ kg/m$^3$ |
| Tensão de escoamento | $250x10^6$ N/m$^2$ |

No presente problema são consideradas restrições de tensão em todas as barras da estrutura. A tensão máxima admissível é de 250 MPa.

Considera-se o carregamento devido aos cabos condutores e o cabo pára-raios. (Figura 6).

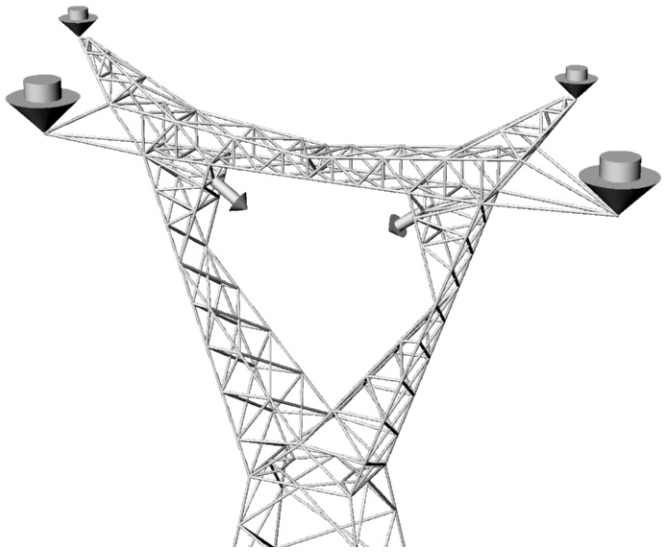


Figura 10: Carregamento.

Na seguinte tabela se detalham os três estados de carregamento.

| Carregamento | Aplicada no vértice | Intensidade (N) | | |
|---|---|---|---|---|
| | | Fx | Fy | Fz |
| Cabos pára-raios | 160, 170 | 0 | 0 | -9370 |
| Condutor central | 93, 104 | -33598 | 0 | -23526 |
| Condutores laterais | 158,159 | 0 | 0 | -71510 |

### 4.2.2. Otimização Contínua

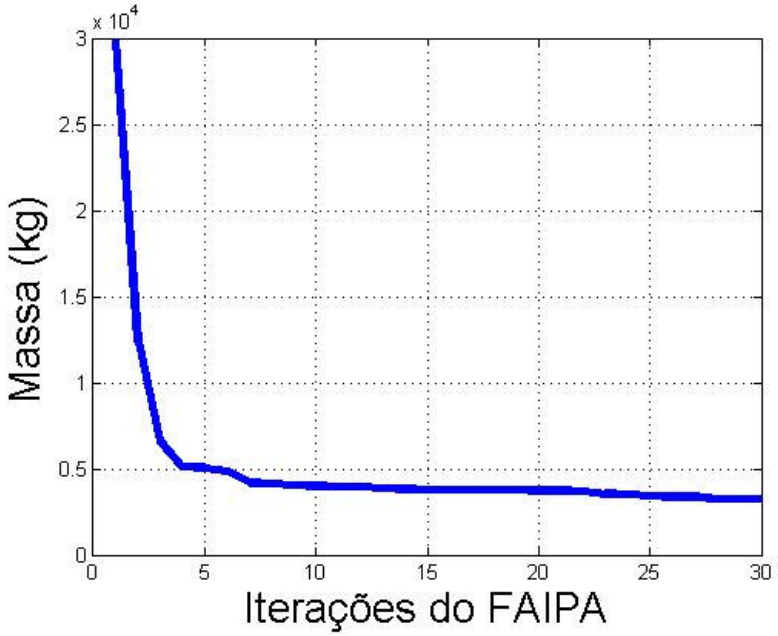
Com a execução do programa OPTIMIZE_ELTE se obteve o seguinte resultado:

| Variáveis de Área | Valor (m$^2$) |
|---|---|
| extensão | 0.2276190045D-03 |
| pernas | 0.2263222623D-03 |

| | |
|---|---|
| tronco | 0.2138136773D-03 |
| delta | 0.2352507596D-03 |
| gambieta | 0.3222496297D-03 |
| viga | 0.5548112693D-03 |
| mísula | 0.3134790115D-03 |
| pára-raio | 0.1808864182D-03 |

### 4.2.3. Redução do custo (massa)

No presente exemplo, o custo é equivalente a massa total da estrutura. A continuação se mostra a redução da massa por iteração do FAIPA.




Observa-se que a estrutura obtida na iteração número 30 contém menos de 10% da massa inicial.

# 5. CONTRIBUIÇÕES E PUBLICAÇÕES CIENTÍFICAS

### 5.1. Otimização baseada em aproximações

No âmbito do projeto OPTIMIZE-ELTE, aplicou-se o esquema de modelos aproximados para obter o gradiente da restrição de mínima freqüência natural, pois esta é uma função não diferençável e o esquema de aproximações proposto é útil para o tratamento desse tipo de funções e mais adequado do que simplesmente aplicar outro esquema tipo diferenças finitas, pois diferenças finitas pressupõe continuidade nas funções.

No apêndice E encontra-se uma pequena revisão da utilização de modelos aproximados, em especial se detalham os modelos mais usados que são basicamente polinômios de baixa ordem (*superfícies de resposta*) ou expansões em séries de Taylor.

A metodologia proposta considera superfícies de resposta que são ajustes polinomiais construídos a partir de dados obtidos por *projeto de experimentos*. Este modelo aproximado poderia substituir completamente à função original para fins de otimização. No entanto como o propósito desta pesquisa requer de um algoritmo de otimização que forneça soluções viáveis em cada iteração, então, para garantir a viabilidade o único jeito é usando sempre a função original na busca linear do ótimo e usando a aproximação somente para fins do cálculo das derivadas.

Nossa aproximação por polinômios de baixa ordem é boa para uma vizinhança pequena, por isso é necessário o controle do tamanho de dita vizinhança, o que é feito seguindo um critério que envolve a conhecida atualização da metodologia da região de confiança (*trust region*) e também outro critério baseado na magnitude do vetor que representa o máximo avanço esperado em cada iteração.

Em cada iteração atualiza-se a região de amostragem como mencionado, e constrói-se uma nova aproximação para fins de obter os gradientes, tanto da função objetivo quanto das restrições. O processo se repete até cumprir o critério de convergência do FAIPA.

O fluxograma da Figura 11 ilustra o esquema de otimização baseada em aproximações com o FAIPA.

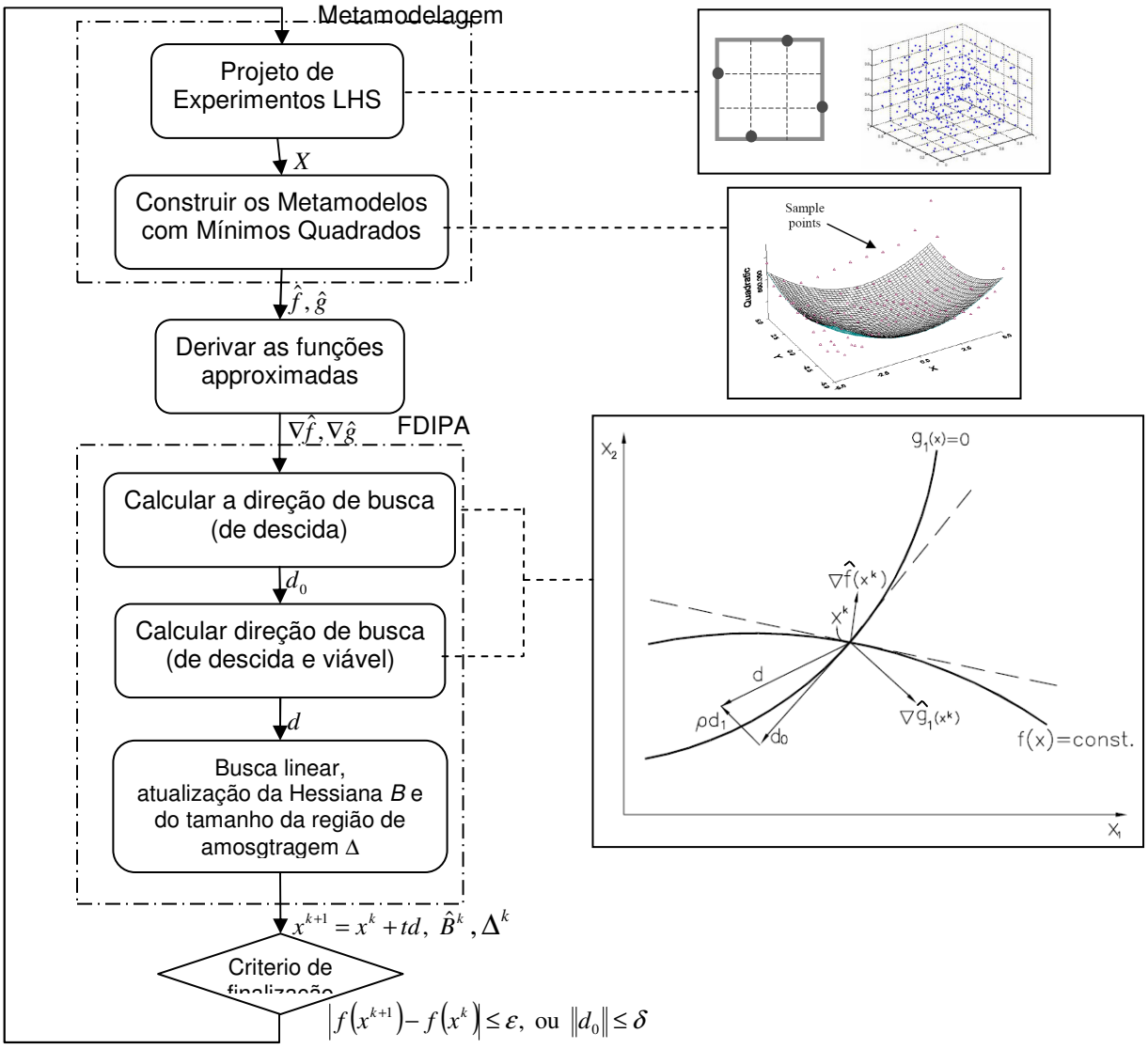



Figura 11: Fluxograma do algoritmo *SB-FAIPA* (*Surrogate Based - FAIPA*).

## 5.2. Algoritmo para otimização topológica

Durante o desenvolvimento deste projeto se teve especial interesse no problema de otimização dimensional [4]. Consideram-se estruturas lineares elásticas, discretizadas por elementos finitos. As variáveis de projeto são as espessuras ou áreas destes elementos. Não se admite espessuras iguais a zero (“furo”) ou maiores que um valor especificado (espessura máxima). A finalidade é obter a melhor distribuição de espessuras da estrutura que minimize seu volume, considerando, no estado de equilíbrio, que a tensão de Von Mises em cada elemento não supere um valor pré-estabelecido.

Em geral, problemas de otimização dimensional podem ser resolvidos utilizando técnicas de programação matemática. Neste trabalho optou-se pela utilização do FAIPA ("*Feasible Arc Interior Point Algorithm*").

Em otimização topológica se enfatiza no problema obter "furos" na estrutura. Isto é, em possibilitar que a variável "espessura" vá para zero. As técnicas de otimização dimensional fornecem informações de onde retirar material, podemos utilizá-la para definir a posição de um orifício.

No projeto OPTIMIZE-ELTE não se empregam técnicas de otimização topológica devido a que o projetista de torres não está, em geral, interessado em eliminar uma barra da torre. Portanto se empregaram técnicas de otimização dimensional unicamente. Porém, durante o processo de pesquisa se descobriu uma técnica de otimização que permite resolver problemas com grande número de variáveis de projeto e restrições mecânicas. A mesma se descreve a continuação.

### 5.2.1. Técnica de otimização de grande porte

O método de memória limitada foi concebido para resolver problemas de otimização de grande porte. Esta técnica, baseada no método Quase-Newton, permite aproximar a matriz quase-Newton da Hessiana do problema (ou a matriz quase-Newton da inversa da Hessiana), através de um conjunto de pares de vetores. Por meio deste conjunto de pares é possível representar tais matrizes sem necessidade de alocar seu espaço. Além disso, o produto de matriz por vetor pode realizar-se eficientemente através desta metodologia [5][6].

#### 5.2.1.1. Metodologia proposta

As características desta metodologia são as seguintes:

i) Usa-se o FDIPA, ("*Feasible Direction Interior Point Algorithm*").

ii) Resolvem-se os sistemas lineares duais do FDIPA.

iii) Usa-se a técnica de memória limitada para representar as inversas das matrizes quase-Newton e dos sistemas lineares do FDIPA.

iv) Os sistemas duais se resolvem pelo Método do Gradiente Conjugado pré-condicionado.

v) Não se calcula a matriz de gradientes das restrições $\nabla g$. Basta computar o produto do gradiente por vetor.

#### 5.2.1.2. Benefícios da nova técnica

i) Redução tempo computacional:

(a) Com a utilização de apropriados pre-condicionadores da matriz do sistema do FDIPA é possível reduzir o número iterações do processo de otimização.

(b) No caso de otimização estrutural: Dado que não se calcula o gradiente das restrições, mas sim o produto da mesma por vetores, se obtém uma economia em termos de número de resoluções de sistemas estruturais.

ii) Redução no uso da memória: Devido a que não se precisa do cálculo do gradiente das restrições, não é necessário alocar espaço de memória para tal informação.

# 6. REFERENCIAS


[1] HUGHES, T.J.R., The Finite Element Method, Englewood Cliffs, New Jersey, Prentice-Hall, 1987.

[2] HERSKOVITS, J., SANTOS, G., Feasible Arc Interior Point Algorithms for Nonlinear Optimization. *Fourth World Congress on Computational Mechanics*, in CD-ROM, Buenos Aires, 1998.

[3] AHO, A.V., ULLMAN J.D., HOPCROFT J.E., Data Structures and Algorithms, Addison-Wesley Series in Computer Science and Information, 1983.

[4] BENDSOE, M.P., SIGMUND, O., Topology Optimization, Theory, Methods and Applications, Berlin, Springer, 2003.

[5] GOULART, E., Matrizes Quase – Newton Esparsas para Problemas de Otimização Não Linear de Grande Porte, Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2005.

[6] DUBEUX, V.J.C., Técnicas de Programação Não Linear para Otimização de Grande Porte. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2005.

# APÊNDICE A

# MANUAL DO USUÁRIO OPTIMIZE-ELTE

## 1. INTRODUÇÃO

OPTIMIZE-ELTE é um software para otimização de torres de transmissão de energia elétrica. Este programa automatiza o processo de dimensionamento das barras, localização espacial de nós e barras para a obtenção da estrutura viável de menor custo. OPTIMIZE-ELTE foi desenhado para trabalhar com grande número de variáveis de projeto podendo-se considerar estruturas com grande numero de barras, nós e estados de carregamento.

### 1.1. Organização do manual

Este manual descreve as configurações necessárias prévias à execução do sistema OPTIMIZE-ELTE. O usuário deve definir do problema e tipo de otimização, assim como que resultados devem ser extraídos após a execução do sistema.

O manual está organizado da seguinte forma, na Seção 2 se descreve o arquivo de configuração do problema de otimização, as opções de execução e dados de saída. Na Seção 3 descreve-se um exemplo simples do arquivo de configuração. Por último, na Seção 4 se explica a forma de executar o programa.

## 2. ARQUIVO DE CONFIGURAÇÃO DO PROBLEMA DE OTIMIZAÇÃO

O arquivo de configuração do problema de otimização consiste em um arquivo de texto plano. Este arquivo está divido nas seguintes partes:

1. Descrição da torre
2. Variáveis de projeto
3. Sistemas de simetria e coordenadas nodais
4. Materiais e barras
5. Apoios e carregamento
6. Restrições de colinearidade e de deslocamento
7. Opções de execução
8. Opções de saída

A continuação detalha-se as partes supracitadas.

### 2.1. Descrição da torre

A descrição da torre consiste em uma linha de texto com qualquer comentário sobre o projeto. A linha deve começar com o símbolo '%' e não pode superar 80 caracteres alfanuméricos (0-9, a-z, A-Z). Exemplo:

```
% ESTRUTURA AUTOPORTANTE TIPO CMSP
```

## 2.2. Variáveis de projeto

Nesta parte do arquivo se definem as variáveis de projeto. Devem-se definir os seguintes números e matrizes:

a. NumVarSec, número de variáveis de seção transversal das barras
b. NumVarPos, número de variáveis de posição
c. NumGrDisc, número de grupos de seções transversais discretas
d. NumSecFix, número de seções fixas
e. NumPosFix ,número de posições fixas

1. MatVarSec, matriz das variáveis de seção
2. MatVarPos, matriz das variáveis de posição
3. MatGrDisc, matriz dos grupos de seções transversais discretas
4. MatValDisc, matriz dos valores discretos para um grupo especifico
5. MatSecFix, matriz das seções fixas
6. MatPosFix, matriz das posições fixas

O formato se apresenta à continuação:

```
% Número de variáveis de seção transversal das barras
NumVarSec

% Numero de variáveis de posição
NumVarPos

% Número de grupos de seções transversais discretas
NumGrDisc

% Número de seções fixas
NumSecFix

% Numero de posições fixas
NumPosFix

% MatVarSec, matriz das variáveis de seção
% Num       Area        Tmin  Tmax  Amin  Amax  GrD
(num_a      x_area      tmin  tmax  amin  amax  num_gr)

% MatVarPos, matriz das variáveis de posição
```

```
% Num        Valor Tmin  Tmax  Pmin  Pmax
(num_p       x_pos tmin  tmax  pmin  pmax)

% MatGrDisc, matriz dos grupos de seções transversais discretas
% NumGr      NumVal
(num_gr      NumValDisc)

(%MatValDisc, matriz dos valores discretos para um grupo especifico
  % Num      Val
  (num_grd   x_area_d))

% MatSecFix, matriz das seções fixas
% Num        Area
(num_a       a)

% MatPosFix, matriz das posições fixas
% Num        Valor
(num_p       p)
```

Onde (c1 c2 ... cn) representa uma matriz de n colunas e cada linha contém os elementos c1, c2, ..., cn. A seguir se detalha cada matriz e o significado de cada coluna. Toda linha que começa com o caractere '%' é ignorada pelo sistema.

### 2.2.1. MatVarSec, matriz de variáveis de seção

Dimensão:

NumVarSec x 7.

Coluna num_a:

Número Inteiro entre 1 e NumVarSec. Representa o identificador da variável de projeto.

Coluna x_area:

Número Real positivo. Representa o valor inicial da variável de área da seção transversal.

Coluna tmin:

0 ou 1. Se tmin=1, o programa considera a restrição de caixa mínima amin. Caso contrário, desconsidera amin.

Coluna tmax:

0 ou 1. Se tmax=1, o programa considera a restrição de caixa máxima amax. Caso contrário, desconsidera amax.

Coluna amin:

Número Real positivo. Representa a área mínima da variável de seção. Esta restrição estará ativa se tmin=1, caso contrário, não se terá em conta o valor de amin.

Coluna amax:

Número Real positivo. Representa a área máxima da variável de seção. Esta restrição estará ativa se tmax=1, caso contrário, não se terá em conta o valor de amax.

Coluna num_gr:

Número Inteiro entre 1 e NumGrDisc. Representa o identificador do grupo de variável discreta à qual a variável de projeto pertence.

### 2.2.2. MatVarPos, matriz das variáveis de posição

Dimensão:

NumVarPos x 6.

Coluna num_p:

Número Inteiro entre 1 e NumVarPos. Representa o identificador da variável de projeto.

Coluna x_pos:

Número Real positivo. Representa o valor inicial da variável de posição.

Coluna tmin:

0 ou 1. Se tmin=1, o programa considera a restrição de caixa mínima pmin. Caso contrário, desconsidera pmin.

Coluna tmax:

0 ou 1. Se tmax=1, o programa considera a restrição de caixa máxima pmax. Caso contrário, desconsidera pmax.

Coluna pmin:

Número Real positivo. Representa a posição mínima da variável de seção. Esta restrição estará ativa se tmin=1, caso contrário, não se terá em conta o valor de pmin.

Coluna pmax:

Número Real positivo. Representa a posição máxima da variável de seção. Esta restrição estará ativa se tmax=1, caso contrário, não se terá em conta o valor de pmax.

### 2.2.3. MatGrDisc, matriz dos grupos de seções transversais discretas

Dimensão:

NumGrDisc x 2.

Coluna num_gr:

Número Inteiro entre 1 e NumGrDisc. Representa o identificador do grupo de variáveis discretas.

Coluna NumValDisc:

Número Inteiro positivo. Representa o número de valores discretos do grupo.

### 2.2.4. MatValDisc, matriz dos valores discretos para um grupo especifico

Dimensão:

A dimensão dependerá do grupo de valores discretos, numValDisc x 2.

Coluna num_grd:

Número Inteiro entre 1 e NumValDisc. Representa o identificador do grupo de valores discretos.

Coluna x_area_d:

Número Real positivo. Representa o valor discreto da área da seção transversal.

### 2.2.5. MatSecFix, matriz das seções fixas

Dimensão:
NumSecFix x 2.
Coluna num_a:
Número Inteiro entre NumVarSec+1 e NumVarSec+NumSecFix. Representa o identificador de uma seção transversal fixa.
Coluna a:
Número Real positivo. Representa o valor da área da seção transversal fixa.

### 2.2.6. MatPosFix, matriz das posições fixas

Dimensão:
NumPosFix x 2.
Coluna num_p:
Número Inteiro entre NumVarPos+1 e NumVarPos+NumPosFix. Representa o identificador de uma posição fixa.
Coluna p:
Número Real positivo. Representa o valor da posição fixa.

## 2.3. Sistemas de simetria e coordenadas nodais

Nesta seção do arquivo se definem as coordenadas nodais da torre. Devem-se definir os seguintes números e matrizes:

a. NumSisCoord, número de sistemas de coordenadas.
b. NumNo, número de nós.

1. MatSim, matriz de sistemas de coordenadas
2. MatNo, matriz de coordenadas nodais

O formato desta parte do arquivo se mostra à continuação:

```
% Numero de sistemas de coordenadas
NumSisCoord

% MatSim, matriz de sistemas de coordenadas
% Num       X     Y     Z
(num_s      i     i     i)

% Numero de nós
NumNo

% MatNo, matriz de coordenadas nodais
```

```
% Num        Cx     Cy     Cz     Sistema
(num_n       num_p  num_p  num_p  num_s)
```

A seguir se detalha cada matriz e o significado de cada coluna.

### 2.3.1. MatSim, matriz de sistemas de coordenadas

Esta matriz contém os fatores multiplicativos para as coordenadas nos três eixos espaciais {x,y,z}.

Dimensão:
NumSisCoord x 4.
Coluna num_s:
Número Inteiro entre 1 e NumSisCoord. Identificador do sistema de coordenada.
Coluna i:
-1 ou 1. Fator multiplicativo.

### 2.3.2. MatNo, matriz de coordenadas nodais

Esta matriz define as coordenadas nodais da estrutura.

Dimensão:
NumNo x 5.
Coluna num_n:
Número Inteiro entre 1 e NumNo. Identificador do nó.
Coluna num_p:
Número Inteiro entre 1 e NumVarPos+NumPosFix. Identificador de posição. Pode ser o identificador da variável de posição ou identificador da posição fixa.
Coluna num_s:
Número Inteiro entre 1 e NumSisCoord. Sistema de fatores de coordenada do nó.

A coordenada nodal se calcula multiplicando o fator do sistema de coordenada pelo valor da posição correspondente.

## 2.4. Materiais e barras

Nesta seção do arquivo se definem os parâmetros dos materiais e barras da torre. Devem-se definir os seguintes números e matrizes:

a. NumMat, número de materiais.
b. NumElem, número de barras.

1. MatMat, matriz de parâmetros dos materiais
2. MatElem, matriz de barras

O formato no arquivo se descreve à continuação:

```
% Numero de materiais
NumMat

% MatMat, matriz de parâmetros dos materiais
% Num       Y     P     D     C     Tac   Tat   tac   tat
(num_m      y     p     d     c     tc    tt    tacom tatrac)

% Numero de barras
NumElem

% MatElem, matriz de barras
% Num       N1          N2          Seção       Material
(num_b      num_n       num_n       num_a       num_m)
```

A seguir se detalha cada matriz e o significado de cada coluna.

### 2.4.1. MatMat, matriz de parâmetros dos materiais

Dimensão:

NumMat x 9.

Coluna num_m

Número Inteiro entre 1 e NumMat. Identificador do material.

Coluna y:

Número Real positivo. Módulo de Young do material.

Coluna p:

Número Real positivo. Módulo de Poisson do material.

Coluna d:

Número Real positivo. Densidade do material.

Coluna c:

Número Real positivo. Custo do material por unidade de massa.

Coluna tc:

0 ou 1. Se tc=1, ativa-se a restrição de tensão admissível por compressão. Caso contrario, a restrição fica desativada.

Coluna tt:

0 ou 1. Se tt=1, ativa-se a restrição de tensão admissível por tração. Caso contrario, a restrição fica desativada.

Coluna tacom:

Número Real positivo. Tensão admissível por compressão.

Coluna tatrac:

Número Real positivo. Tensão admissível por tração.

### 2.4.2. MatElem, matriz de barras

Dimensão:

NumElem x 5.

Coluna num_b

Número Inteiro entre 1 e NumElem. Identificador da barra ou elemento estrutural.

Coluna num_n:

Número Inteiro entre 1 e NumNo. Identificador do nó.

Coluna num_a:

Número Inteiro entre 1 e NumVarSec+NumSecFix. Pode ser o identificador da variável de seção ou identificador da seção fixa.

Coluna num_m:

Número Inteiro entre 1 e NumMat. Identificador do material.

Toda barra terá uma restrição de tensão por tração ou compressão se o material da barra tem tt=1 ou tc=1, respectivamente.

### 2.5. Apoios e carregamento

Nesta parte se definem as condições de contorno e carregamento do modelo de elementos finitos. Devem-se definir os seguintes números e matrizes:

a. NumEstCarga, número de estados de carregamento
b. NumCondCont, número de condições de contorno
c. NumCondNo, número de condições de contorno em nós

1. MatCondCont, matriz de condições de contorno
2. MatCondNo, matriz de condições de contorno em nós

O formato se descreve à continuação:

```
% Numero de estados de carregamento
NumEstCarga

% Numero de condições de contorno
NumCondCont

% MatCondCont, matriz de condições de contorno
% Num       Tx    Ty    Tz    Vx    Vy    Vz
(num_bc     tx    ty    tz    vx    vy    vz)

% Numero de condições em nós
NumNodeCond

% MatCondNo, matriz de condições de contorno em nós
% Num       No          Estado_de_carregamento  Condição_de_contorno
(num_cn     num_n       num_lc                  num_bc)
```

A seguir se detalha cada matriz e o significado de cada coluna.

### 2.5.1. MatCondCont, matriz de condições de contorno

Dimensão:
NumCondCont x 7.
Coluna num_bc
Número inteiro entre 1 e NumCondCont. Identificador da condição de contorno.
Colunas tx, ty, tz:
0 ou 1. Se tx=0, a condição é do tipo deslocamento. Caso contrário, ac condição é de força.
Coluna vx, vy, vz:
Número real. Deslocamento ou força prescrita.

### 2.5.2. MatCondNo, matriz de condições de contorno em nós

Dimensão:
NumCondCont x 4.
Coluna num_cn
Número inteiro entre 1 e NumCondNo. Identificador da condição de contorno no nó.
Colunas num_n:
Número inteiro entre 1 e NumNo. Identificador do nó.
Coluna num_lc:
Número inteiro entre 1 e NumEstCarga. Identificador de estado de carregamento.
Coluna num_bc:
Número inteiro entre 1 e NumCondCont. Identificador da condição de contorno.

## 2.6. Restrições de colinearidade e de deslocamento

Nesta parte se definem as restrições de colinearidade entre nós e restrições de deslocamento. Devem-se definir os seguintes números e matrizes:

a. NumRestrCol, número de restrições de colinearidade
b. NumRestrDesp, número de restrições de deslocamento

1. MatRestrCol, matriz de restrições de colinearidade
2. MatRestrDesp, matriz de restrições de deslocamento

O formato se descreve à continuação:

```
% Numero de restrições de colinearidade
NumRestrCol

% MatRestrCol, matriz de restrições de colinearidade
% Num       No_central  No_1        No_2
(num_col    num_n       num_n       num_n)
```

```
% Numero de restrições de deslocamento
NumRestrDesp

% MatRestrDesp, matriz de restrições de deslocamento
% Num       No          Grau_de_liberdade Tmin  Tmax  Dmin  Dmax
(num_dc     num_n       gdl               tmin  tmax  dmin  dmax)
```

A seguir se detalha cada matriz e o significado de cada coluna.

### 2.6.1. MatRestrCol, matriz de restrições de colinearidade

Dimensão:

NumRestrCol x 4.

Coluna num_col

Número inteiro entre 1 e NumRestrCol. Identificador da restrição de colinearidade.

Colunas num_n:

Número inteiro entre 1 e NumNo. Identificador do nó. A segunda coluna especifica o nó interno, a terceira e quarta coluna especificam os nós extremos.

### 2.6.2. MatRestrDesp, matriz de restrições de deslocamento

Dimensão:

NumRestrDesp x 7.

Coluna num_dc

Número Inteiro entre 1 e NumRestrDesp. Identificador de restrição de deslocamento.

Coluna num_n:

Número inteiro entre 1 e NumNo. Identificador do nó.

Coluna gdl:

1, 2 ou 3. Grau de liberdade.

Coluna tmin:

0 ou 1. Se tmin=1, o programa considera a restrição de caixa mínima dmin. Caso contrário, desconsidera dmin.

Coluna tmax:

0 ou 1. Se tmax=1, o programa considera a restrição de caixa máxima dmax. Caso contrário, desconsidera dmax.

Coluna dmin:

Número Real positivo. Deslocamento mínimo. Esta restrição estará ativa se tmin=1, caso contrário, não se terá em conta o valor dmin.

Coluna dmax:

Número Real positivo. Deslocamento máximo. Esta restrição estará ativa se tmax=1, caso contrário, não se terá em conta o valor dmax.

## 2.7. Opções de execução

Nesta porção do arquivo se especifica o arquivo de configuração do FAIPA, o tipo de execução do programa, o arquivo de saída de resultados, e que tipo de resultados será colocado no arquivo de saída.

Devem-se definir os seguintes parâmetros:

1. fdata, arquivo de parâmetros do FAIPA
2. num, opções do tipo de execução
3. saída, arquivo de resultados da execução
4. bin, tipo de resultados

O formato é o seguinte:

```
% Arquivo de configuração do FAIPA
fdata

% Opções de execução
% 0=Análise,1=Contínua,2=Contínua-Discreta,3=Contínua-Discreta-Contínua
num

% Arquivo de resultados
saida

% Variáveis de Projeto
binvp

% Deslocamentos nodais
bindn

% Tensão nas barras
bintb

% Reações
binr

% Freqüência Natural
binfn
```

### 2.7.1. fdata, arquivo de parâmetros do FAIPA

Este parâmetro define o nome do arquivo com os parâmetros do algoritmo FAIPA. Recomenda-se não mudar este parâmetro dado que o arquivo de configuração do FAIPA foi escolhido convenientemente para este tipo de aplicação. O parâmetro fdata deve ser uma string de até 80 caracteres alfanuméricos (0-9, a-z, A-Z).

### 2.7.2. num, opções do tipo de execução

Os tipos de execuções do programa são os seguintes:

0. Análise estrutural:
   Realizam-se operações para obtenção das respostas mecânicas da estrutura. Não se realiza nenhum tipo de processo de otimização.
1. Otimzação Contínua
   Otimiza-se a torre a traves do FAIPA. Nesta execução não se considera o grupo de valores discretos para as seções transversais das barras. A solução obtida, portanto não é discreta nas variáveis de seções das barras.
2. Otimização Contínua-Discreta
   Otimiza-se a torre como na opção 1 (Contínua) obtendo-se uma solução ótima. A partir desta solução ótima e através de uma heurística, se obtém uma solução discreta nas variáveis de seção transversal das barras. No processo heurístico não se modifica a geometria da torre, isto é, não se modifica as variáveis de posição.
3. Otimização Contínua-Discreta-Contínua
   Otimiza-se a torre como na opção 2 (Contínua-Discreta) obtendo-se uma solução discreta nas variáveis de seção transversal das barras. A partir desta solução e através do FAIPA, se obtém uma solução ótima para as variáveis de posição. Neste processo não se modificam as áreas das seções das barras. Por tanto se obtém uma solução discreta para as variáveis de seção transversal e continua para as variáveis de posição.

O parâmetro num deve ser o número 0, 1, 2 ou 3. Caso num seja 0, realiza-se uma execução do tipo Análise Estrutural. Caso num seja 1, realiza-se uma execução do tipo Otimização Contínua. Caso num seja 2, realiza-se uma execução do tipo Otimização Contínua-Discreta. Caso num seja 3, realiza-se uma execução do tipo Otimização Contínua-Discreta-Contínua.

### 2.7.3. saida, arquivo de resultados da execução

Este parâmetro define o nome do arquivo para guardar os resultados da execução do sistema. O parâmetro saida deve ser uma string de até 80 caracteres alfanuméricos (0-9, a-z, A-Z).

### 2.7.4. bin, tipo de resultados

Estes parâmetros definem quais resultados da execução serão guardados no arquivo de saída. Caso o parâmetro seja igual a 1, significa que tal informação será impressa no arquivo de saída, caso contrário, não será impressa no arquivo de saída.

Os dados de saída podem ser os seguintes:

binvp: variáveis de projeto
bindn: deslocamentos nodais
bintb: tensão das barras
binr: reações

binfn: freqüência natural

# 3. EXEMPLO DE ARQUIVO DE CONFIGURAÇÃO DO PROBLEMA DE OTIMIZAÇÃO

```
% torre 42 B

% Número de variáveis de seção transversal das barras
6

% Numero de variáveis de posição
8

% Número de grupos de seções transversais discretas
1

% Numero de seções fixas
2

% Numero de posições fixas
4

% MatVarSec, matriz das variáveis de seção  (1e-4 metro quadrado = 1 cm
quadrado)
% Num         Area     Tmin     Tmax      Amin     Amax      GrD
 1             1e-1     1        1        1e-4     2e-1      1
 2             1e-1     1        1        1e-4     2e-1      1
 3             1e-1     1        1        1e-4     2e-1      1
 4             1e-1     1        1        1e-4     2e-1      1
 5             1e-1     1        1        1e-4     2e-1      1
 6             1e-1     1        1        1e-4     2e-1      1

% MatVarPos, matriz das variáveis de posição
% Num            Valor   Tmin     Tmax  Boxmin  Boxmax ( metros )
 1               0.45     1        1       0.4   1.0
 2               0.45     1        1       0.4   1.0
 3               0.45     1        1       0.4   1.0
 4               0.45     1        1       0.4   1.0
 5               0.45     1        1       0.4   1.0
 6               0.45     1        1       0.4   1.0
 7               3.0      1        1       2.5   3.5
 8               2.0      1        1       1.5   3.5

% MatGrDisc, matriz dos grupos de seções transversais discretas
%      NumGr     NumVal
        1          4
```

```
% Valores discretos do grupo 1
%    Num          Val
    1             1e-4
    2            10e-4
    3            20e-4
    4            40e-4

% MatSecFix, matriz das seções fixas
% Num       Area (1e-4 metro quadrado = 1 cm quadrado)
 7          1e-4
 8          1e-4

% MatPosFix, matriz das posições fixas
% Num            Valor ( metro )
 9               0.0
 10              1.0
 11              2.0
 12              1.5

% Numero de sistemas de coordenadas
 4

% Sistemas de coodenadas
% Num           X       Y       Z
 1              1       1       1
 2             -1       1       1
 3             -1      -1       1
 4              1      -1       1

% Numero de nos
 15

% Nos
% Num          Cx      Cy      Cz      Sistema
 1              1       2       9       1
 2              1       2       9       2
 3              1       2       9       3
 4              1       2       9       4
 5              3       4      10       1
 6              3       4      10       2
 7              3       4      10       3
 8              3       4      10       4
 9              5       6      11       1
10              5       6      11       2
11              5       6      11       3
12              5       6      11       4
13             12       9       8       2
```

```
14              9       9       7       1
15             12       9       8       1

% Numero de materiais
1

% Material -> AÇO ASTM A36 (E = 200e6 N/m2, Densidade = 7.8e3 kg/m3,
Tensão de escoamento = 250e6 N/m2)
% Num  Young  Poisson Dens  custo  Ttac  Ttat  tens_comp tens_trac
 1     200e6   0.3    7.8e3   1.0    1     1   250e6         250e6

% Numero de barras
 42

% Barras
% Num N1      N2      Secao   Material
 1     1       5       1       1
 2     2       6       1       1
 3     3       7       1       1
 4     4       8       1       1
 5     5       9       2       1
 6     6       10      2       1
 7     7       11      2       1
 8     8       12      2       1
 9     5       6       3       1
10     6       7       3       1
11     7       8       3       1
12     8       5       3       1
13     9       10      4       1
14     10      11      4       1
15     11      12      4       1
16     12      9       4       1
17     4       5       7       1
18     1       8       7       1
19     1       6       7       1
20     2       5       7       1
21     2       7       7       1
22     3       6       7       1
23     3       8       7       1
24     4       7       7       1
25     8       9       8       1
26     5       12      8       1
27     5       10      8       1
28     6       9       8       1
29     6       11      8       1
30     7       10      8       1
31     7       12      8       1
32     8       11      8       1
```

```
33      11      13      5       1
34      10      13      5       1
35      13      14      5       1
36      12      15      5       1
37      9       15      5       1
38      14      15      5       1
39      11      14      6       1
40      10      14      6       1
41      12      14      6       1
42      9       14      6       1

% Numero de estados de carregamento
 3

% Numero de condicoes de contorno
 4

% Condicoes de contorno
% Num   Tx  Ty   Tz    Vx      Vy      Vz ( 200 Newton = 20Kg.10m/s2 )
 1     0     0    0     0.0     0.0     0.0
 2     1     1    1     0.0     0.0  -200.0
 3     1     1    1     0.0   160.0  -100.0
 4     1     1    1   -10.0     0.0     0.0
% Numero de condicoes en nos
15

% Condicoes de contorno en nos
% Num   No   Estado_de_carregamento  Condicao_de_contorno
 1     1       1                       1
 2     2       1                       1
 3     3       1                       1
 4     4       1                       1
 5    13       1                       2
 6    15       1                       2
 7    13       2                       3
 8     1       3                       4
 9     4       3                       4
 10    5       3                       4
 11    8       3                       4
 12    9       3                       4
 13    12      3                       4
 14    15      3                       4
 15    14      3                       4

% Numero de restricoes de colinearidade
1

% Restricoes de colinearidade
```

```
% Num   No_central      No1     No2
  1     5                 1     9

% Numero de restricoes de deslocamento
 3

% Restricoes de deslocamento
% Num   No      Grau_de_liberdade       Tmin    Tmax    Rmin    Rmax
 1    13             1                  1       1     80e-3    80e-3
 2    13             2                  1       1     80e-3    80e-3
 3    13             3                  1       1     1e-3      1e-3

% Arquivo de configuração do FAIPA
fdata.txt

% Opções de execução
% 0=Análise, 1= Contínua, 2= Contínua-Discreta, 3= Contínua-Discreta-
Contínua
2

% Arquivo de resultados
torre42B.out.txt

% Variáveis de Projeto
1

% Deslocamentos nodais
0

% Tensão nas barras
0

% Reações
0

% Freqüência Natural
0
```

# 4. EXECUÇÃO DO PROGRAMA OPTIMIZE-ELTE

Para executar o programa deve-se rodar o código binário em uma consola do computador:

>optimize_elte arquivo.txt

O argumento ‘arquivo.txt’ é o arquivo de configuração do problema de otimização. Este arquivo deve ser criado pelo usuário. O código binário do sistema deve

ser compilado para sistema operacional do usuário. Para a execução do programa devem existir os seguintes arquivos no diretório de execução:

a. fdata.txt

Arquivo de configuração do algoritmo FAIPA. Este arquivo deve ser fornecido pelo laboratório OptimizE da UFRJ.

b. arquivo.txt

Arquivo de configuração do problema de otimização. Este arquivo deve ser fornecido pelo usuário.

A execução do programa gera um arquivo de texto com os resultados da execução.

# APÊNDICE B

# MÓDULO DE ANÁLISE ESTRUTURAL – MANUAL DO PROGRAMADOR

## 1. INTRODUÇÃO

O Módulo de análise estrutural é uma coleção de rotinas escritas na linguagem Fortran90 que permitem a análise estrutural e análise de sensibilidade de estruturas treliçadas. Este módulo foi especialmente concebido para a sua utilização pelo software OPTIMIZE-ELTE de otimização de torres de transmissão de energia elétrica. Com este propósito, o módulo permite realizar a análise de qualquer estrutura das consideradas pelo software OPTIMIZE-ELTE e é especialmente indicado para estruturas de grande tamanho, isto é, estruturas com grande número de barras, nós e estados de carregamento.

### 1.1. Conteúdo deste documento

Este documento descreve o Módulo de análise estrutural, para seu uso pelos programadores do software OPTIMIZE-ELTE. Este documento proporciona toda a informação necessária para a utilização das rotinas públicas do módulo.

### 1.2. Distribuição do Módulo de análise estrutural

O Módulo de análise estrutural é parte do software OPTIMIZE-ELTE e não é distribuído por separado. Na distribuição do software OPTIMIZE-ELTE, o Módulo de análise estrutural corresponde ao arquivo:

```
Analysis.f90
```

### 1.3. Características importantes

O Módulo de análise estrutural permite realizar a análise linear estática e análise de freqüências naturais das estruturas treliçadas consideradas pelo software OPTIMIZE-ELTE. Para uma determinada estrutura, o Módulo de análise estrutural permite obter as seguintes medidas de performance:

i) Custo da estrutura.

ii) Deslocamentos dos nós da estrutura para cada estado de carregamento.

iii) Tensões nas barras para cada estado de carregamento.

iv) Mínima Freqüência natural da estrutura.

v) Função que define as restrições de colinearidade.

vi) Sensibilidade de cada uma das funções de performance mencionadas nos itens (i)-(v) em relação a qualquer das variáveis geométricas (de posição de nós ou área de seção transversal) utilizadas pelo software OPTIMIZE-ELTE para a definição do problema de otimização estrutural.

A implementação realizada das rotinas para cálculo das funções de performance mencionadas nos itens (i)-(vi) foi feita considerando maximizar a capacidade de resolução de problemas de grande porte. Para isso foram seguidas as seguintes diretrizes:

vii) Minimização da quantidade de memória necessária para armazenamento das variáveis auxiliares ou intermediárias.

viii) Minimização do custo computacional requerido para o cálculo das funções de performance indicadas e de suas derivadas assim como das variáveis auxiliares utilizadas.

Para calcular as funções de performance indicadas nos itens (i)-(vi) de forma de satisfazer os critérios (vii)-(viii) é utilizado o Método dos Elementos Finitos (MEF). A implementação desenvolvida do MEF é específica para a categoria de problemas de otimização que serão resolvidos utilizando o software OPTIMIZE-ELTE. Esta característica do software implementado teve por objetivo aproveitar as particularidades do problema de forma de maximizar o desempenho do software OPTIMIZE-ELTE.

As características importantes da implementação desenvolvida do MEF que permitem satisfazer os requerimentos (vii)-(viii) são:

ix) Armazenamento esparso da matriz de rigidez da estrutura.

x) Utilização de um software específico para solução dos sistemas lineares simétricos e esparsos do MEF. Para isso foi utilizada a rotina MA27 [1] pertencente à "Harwell Subroutine Library" [2], programada na linguagem Fortran77 e apropriada para problemas simétricos e esparsos de grande porte.

xi) Utilização de um software específico para a solução dos problemas de valores próprios da análise de freqüências naturais. Para isso é utilizado o software ARPACK [3], uma coleção de rotinas de Fortran77 desenhada para solução de diversos problemas de valores próprios de grande porte.

### 1.4. Rotinas fornecidas pelo módulo

As rotinas fornecidas pelo Módulo de análise estrutural são classificadas nos seguintes grupos:

i) Rotinas de inicialização e informação básica

```
analysis_init
analysis_getNumGrLib
analysis_getNumGrFix
analysis_getGrLib
analysis_result
```

ii) Rotinas relativas à função custo:

```
analysis_getCost
analysis_getDCost
```

iii) Rotinas relativas às restrições geométricas de colinearidade

```
analysis_getCol
analysis_getDCol
```

iv) Rotinas relativas à análise linear estática:

```
analysis_getUS
analysis_getDUDS
```

v) Rotinas relativas à análise de freqüências naturais:

```
analysis_getMaxEig
```

### 1.5. Dependência de outras rotinas e códigos

O Módulo de análise estrutural depende das rotinas:

i) Módulo de Entrada-Saída do software OPTIMIZE-ELTE.

ii) Rotina MA27 [1] da "Harwell Subroutine Library". Esta rotina pode ser obtida gratuitamente para fins acadêmicos ou sob uma licencia comercial no sítio de Internet:

http://www.cse.scitech.ac.uk/nag/hsl/

iii) Conjunto de rotinas ARPACK. Este conjunto de rotinas pode ser obtido gratuitamente por ftp anônimo do servidor:

ftp.caam.rice.edu

ou no sítio de internet:

http://www.caam.rice.edu/software/ARPACK

### 1.6. Limitações conhecidas

Entre qualquer par de nós da estrutura pode existir, como máximo, uma única barra (consultar documentação do Módulo de Entrada-Saída do software OPTIMIZE-ELTE).

A dimensão da variável MatK do módulo é definida pela rotina analysis_init igual ao dobro do tamanho máximo necessário para armazenar a matriz de rigidez da estrutura. Este valor pode ser menor do necessário para armazenar a decomposição da matriz. Em caso de apresentar problemas corrigir as seguintes linhas da rotina analysis_init:

```
TAM = 2*(6*NumNodos+9*NumElementos)
```

Este valor usualmente funciona bem. Caso contrário redefina a variável TAM assinando-lhe um valor maior como, por exemplo:

```
TAM = 4*(6*NumNodos+9*NumElementos)
```

Consultar a documentação da rotina MA27 para escolher um valor adequado para o problema particular que está sendo resolvido.

# 2. USO DO MÓDULO DE ANÁLISE ESTRUTURAL

Para utilizar o Módulo de análise estrutural, este deve ser declarado previamente no módulo ou rotina no qual se deseja utilizá-lo mediante a sentença:

```
use analysis
```

A sentença anterior deve estar localizada após a sentença PROGRAM, FUNCTION, SUBROUTINE, MODULE, ou BLOCK DATA, dependendo do tipo de rotina que utiliza o Módulo de análise estrutural.

### 2.1. Inicialização e informação básica

As seguintes são as rotinas de inicialização e informação básica:

```
analysis_init
analysis_getNumGrLib
analysis_getNumGrFix
analysis_getGrLib
analysis_result
```

#### 2.1.1. Rotina analisis_init

A rotina `analisis_init` é uma rotina de inicialização. A mesma deve ser invocada antes que qualquer outra rotina do módulo de análise. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_init

      Sentenças executáveis

end subroutine analysis_init
```

Para invocar a rotina `analisis_init,` a sintaxe é a seguinte:

```
call analisis_init
```

### 2.1.2. Rotina analysis_getNumGrLib

A rotina analysis_getNumGrLib fornece o número de graus de liberdade da estrutura. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getNumGrLib(NGL)

      integer, intent(out) :: NGL

      Sentenças executáveis

end subroutine analysis_getNumGrLib
```

O valor da variável de saída NGL corresponde ao número de graus de liberdade da estrutura. Para invocar a rotina analysis_getNumGrLib, a sintaxe é a seguinte:

```
call analysis_getNumGrLib(NGL)
```

### 2.1.3. Rotina analysis_getNumGrFix

A rotina analysis_getNumGrFix fornece o número de restrições do deslocamento da estrutura, ou seja, número de apoios. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getNumGrFix(NGF)

      integer, intent(out) :: NGF

      Sentenças executáveis

end subroutine analysis_getNumGrFix
```

O valor da variável de saída NGF corresponde ao número apoios da estrutura. Para invocar a rotina analysis_getNumGrFix, a sintaxe é a seguinte:

```
subroutine analysis_getNumGrFix(NGF)

      integer, intent(out) :: NGF

      Sentenças executáveis

end subroutine analysis_getNumGrFix
```

call analysis_getNumGrFix(NGF)

### 2.1.4. Rotina analysis_getGrLib

A rotina `analysis_getGrLib` fornece o número de restrições do deslocamento da estrutura, ou seja, número de apoios. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getGrLib(GrG,Nod,GrL)

      integer, intent(out) :: GrG
      integer, intent(in)  :: Nod
      integer, intent(in)  :: GrL

      Sentenças executáveis

end subroutine analysis_getGrLib
```

O valor da variável de saída `GrG` é a numeração que o Módulo de análise estrutural assina ao grau de liberdade correspondente ao nó `Nod` na direção `GrL`. `GrL` deve ser 1, 2 ou 3, respectivamente, para o primeiro, segundo e terceiro grau de liberdade do nó. Para invocar a rotina `analysis_getGrLib`, a sintaxe é a seguinte:

```
call analysis_getGrLib(GrG,Nod,GrL)
```

### 2.1.5. Rotina analysis_result

A rotina `analysis_result` realiza uma análise elástica linear e de freqüências naturais e passa os resultados ao Módulo de Entrada-Saída. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_result(VecX)

      real*8, intent(in) :: VecX(NumVar)
```

```
      Sentenças executáveis

end subroutine analysis_result
```

A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. A variável `NumVar` é privada do Módulo de Análise estrutural e o seu valor é atribuído previamente pela rotina `analysis_init`. Para invocar a rotina `analysis_result`, a sintaxe é a seguinte:

```
call analysis_result
```

### 2.2. Rotinas relativas à função custo

As seguintes são as rotinas relativas à função custo:

```
analysis_getCost
analysis_getDCost
```

#### 2.2.1. Rotina analysis_getCost

A rotina `analysis_getCost` calcula o custo da estrutura considerada em função das variáveis geométricas que a definem. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getCost(f,VecX)

      real*8, intent(out) :: f
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getCost
```

O valor da variável de saída `f` e o custo da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. A variável `NumVar` é privada do Módulo de Análise estrutural e o seu valor é atribuído previamente pela rotina `analysis_init`. Para invocar a rotina `analysis_getCost`, a sintaxe é a seguinte:

```
call analysis_getCost(f,VecX)
```

#### 2.2.2. Rotina analysis_getDCost

A rotina `analysis_getDCost` calcula as derivadas parciais da função custo da estrutura com relação às variáveis geométricas que a definem. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getDCost(Df,VecX)

      real*8, intent(out) :: Df(NumVar)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getDCost
```

O valor da variável de saída `Df` e o vetor de derivadas da função custo da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. A variável `NumVar` é privada do Módulo de Análise estrutural e o seu valor é atribuído previamente pela rotina `analysis_init`. Para invocar a rotina `analysis_getDCost`, a sintaxe é a seguinte:

```
call analysis_getDCost(Df,VecX)
```

### 2.3. Rotinas relativas às restrições geométricas de colinearidade

As seguintes são as rotinas relativas às restrições geométricas de colinearidade:

```
analysis_getCol
analysis_getDCol
```

#### 2.3.1. Rotina analysis_getCol

A rotina `analysis_getCol` calcula a função que define as restrições de colinearidade da estrutura com relação às variáveis geométricas que a definem. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getCol(h,VecX)

      real*8, intent(out) :: h(NumResCol*NGN)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getCol
```

O valor da variável de saída h e a função das restrições de colinearidade da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. As variáveis `NumVar` e `NumResCol` são privadas do Módulo de Análise estrutural e o seus valores são atribuídos previamente pela rotina `analysis_init`. A variável `NGN` é uma constante privada do módulo. Para invocar a rotina `analysis_getCol`, a sintaxe é a seguinte:

```
call analysis_getCol(h,VecX)
```

### 2.3.2. Rotina analysis_getDCol

A rotina `analysis_getDCol` calcula a as derivadas parciais da função que define as restrições de colinearidade da estrutura com relação às variáveis geométricas que a definem. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getDCol(Dh,VecX)

      real*8, intent(out) :: Dh(NumResCol*NGN,NumVar)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getDCol
```

A variável de saída Dh é a matriz de derivadas parciais da função das restrições de colinearidade da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. As variáveis `NumVar` e `NumResCol` são privadas do Módulo de Análise estrutural e os seus valores são atribuídos previamente pela rotina `analysis_init`. A variável `NGN` é uma constante privada do módulo. Para invocar a rotina `analysis_getDCol`, a sintaxe é a seguinte:

```
call analysis_getDCol(Dh,VecX)
```

## 2.4. Rotinas relativas à análise linear estática

As seguintes são as rotinas relativas à análise linear estática:

```
analysis_getUS
analysis_getDUDS
```

### 2.4.1. Rotina analysis_getUS

A rotina `analysis_getUS` calcula os deslocamentos dos nós e as tensões das barras da estrutura. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getUS(VecU,VecS,VecX)

      real*8, intent(out) :: VecU(NumGrLib*NumEstCarga)
      real*8, intent(out) :: VecS(NumElementos*NumEstCarga)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getUS
```

Os valores das variáveis de saída `VecU` e `VecS` correspondem, respectivamente, aos deslocamentos dos nós e às tensões das barras da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. As variáveis `NumVar`, `NumGrLib`, `NumElementos` e `NumEstCarga` são privadas do Módulo de Análise estrutural e os seus valores são atribuídos previamente pela rotina `analysis_init`. Para invocar a rotina `analysis_getUS`, a sintaxe é a seguinte:

```
call analysis_getUS(VecU,VecS,VecX)
```

### 2.4.2. Rotina analysis_getDUDS

A rotina `analysis_getUS` calcula as derivadas parciais dos deslocamentos dos nós e as tensões das barras da estrutura com relação às variáveis geométricas que a definem. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getDUDS(MatDU,MatDS,VecX)

      real*8, intent(out) :: MatDU(NumGrLib*NumEstCarga,NumVar)
      real*8, intent(out) :: MatDS(NumElementos*NumEstCarga,NumVar)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getDUDS
```

Os valores das variáveis de saída `VecDU` e `VecDS` correspondem, respectivamente, as derivadas dos deslocamentos dos nós e às derivadas das tensões das barras da estrutura. A variável de entrada `VecX` contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. As variáveis `NumVar`, `NumGrLib`, `NumElementos` e `NumEstCarga` são privadas do Módulo de Análise estrutural e os seus valores são atribuídos previamente pela rotina

analysis_init. Para invocar a rotina analysis_getDUDS, a sintaxe é a seguinte:

```
call analysis_getDUDS(VecDU,VecDS,VecX)
```

### 2.5. Rotinas relativas à análise de freqüências naturais:

As seguintes são as rotinas relativas à análise de freqüências naturais:

```
analysis_getMaxEig
```

#### 2.5.1. Rotina analysis_getMaxEig

A rotina analysis_getMaxEig calcula as menores NumEig freqüências naturais da estrutura. No Módulo de análise estrutural é definida na seguinte forma:

```
subroutine analysis_getMaxEig(NumEig,MaxEig,VecX)

      integer, intent(in) :: NumEig
      real*8, intent(out) :: MaxEig(NumEig)
      real*8, intent(in)  :: VecX(NumVar)

      Sentenças executáveis

end subroutine analysis_getUS
```

Onde NumEig é o número de freqüências Os valores das variáveis de saída VecU e VecS correspondem, respectivamente, aos deslocamentos dos nós e às tensões das barras da estrutura. A variável de entrada VecX contém os valores atuais das variáveis do problema de otimização estrutural tal como definida pelo software Optimize_ELTE. A variável NumVar é privada do Módulo de Análise estrutural e o seu valor é atribuído previamente pela rotina analysis_init. Para invocar a rotina analysis_getUS, a sintaxe é a seguinte:

```
call analysis_getUS(VecU,VecS,VecX)
```

# 3. REFERÊNCIAS


[1] Duff, I. S.; Reid, J. K. The multifrontal solution of indefinite sparse symmetric linear equations. ACM Trans. Math. Software 9 (1983), no. 3, 302--325

[2] http://www.cse.scitech.ac.uk/nag/hsl/

[3] http://www.caam.rice.edu/software/ARPACK

# APÊNDICE C

# DESCRIÇÃO DO FAIPA

Considere o problema de otimização não-linear com restrições:

$$\min_x f(x) \tag{1}$$

$$\text{Sujeito a } g(x) \le 0 \tag{2}$$

$$\text{e } h(x) = 0 \tag{3}$$

onde $x \in R^n$ é o vetor de variáveis de projeto, $g : R^n \to R^m$ é a função que define as restrições de desigualdade e $h : R^n \to R^p$ é a função que define as restrições de igualdade do problema de otimização.

Em um mínimo local do problema de otimização (1)-(3) são satisfeitas as condições necessárias de otimalidade de primeira ordem de Karush-Kuhn-Tucker (KKT), ou seja, existem vetores $\lambda \in R^m$ e $\mu \in R^p$ que satisfazem:

$$\nabla f(x) + \sum_{i=1}^{m} \nabla g_i(x)\lambda_i + \sum_{i=1}^{p} \nabla h_i(x)\mu_i = 0 \tag{4}$$

$$g_i(x)\lambda_i = 0, \quad 1 \le i \le m \tag{5}$$

$$h(x) = 0 \tag{6}$$

$$g(x) \le 0 \tag{7}$$

$$\lambda \ge 0 \tag{8}$$

O algoritmo FDIPA [1][2] é um método iterativo para achar um ponto que satisfaça as condições de KKT, Eq. (4)-(8). Este algoritmo requer um ponto inicial $x_0$ pertencente ao interior da região $\Delta$ definida como:

$$\Delta = \left\{ x \in R^n \mid g(x) \ge 0, h(x) \ge 0 \right\} \tag{9}$$

A partir do ponto inicial o algoritmo FDIPA gera uma seqüência $\{x_k\}_{k \in N}$ totalmente contida no interior de $\Delta$ e que converge a um ponto de KKT do Problema (1)-(3). A seqüência gerada pelo algoritmo reduz em cada iteração a função potencial:

$$\phi_c(x) = f(x) + \sum_{i=1}^{p} c_i \left| h_i(x) \right| \tag{10}$$

onde o vetor $c \in R^p$ é definido pelo algoritmo. Pode ser mostrado que se as componentes $c_i$ do vetor $c$ são suficientemente grandes, então a função $\phi_c$ é uma função de penalidade exata para o Problema (1)-(3), ver [3]. Em outras palavras, existe um vetor $c$

tal que os mínimos da função $\phi_c$ sujeitos as restrições de desigualdade são soluções do Problema (1)-(3).

A iteração do algoritmo FAIPA é dada pela regra $x_{k+1} = x_k + t_k\, d_x + t_k^2\, \tilde{d}_x$ onde $x_k$ é o ponto da iteração atual, $t_k$ é o passo da iteração $k$ definido por um procedimento chamado busca linear. A definição da direção de busca $d_x$ do algoritmo FAIPA é inspirada na aplicação do método de Newton ao sistema não linear de equações dado pelas Eqs. (4)-(6). A iteração de Newton para este sistema de equações é $x_{k+1} = x_k + d_x$, onde $d_x$ é obtida resolvendo o sistema linear:

$$\begin{pmatrix} H_k & \nabla g_k & \nabla h_k \\ \Lambda_k \nabla g_k^T & G_k & 0 \\ \nabla h_k^T & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x \\ d_\lambda \\ d_\mu \end{pmatrix} = \begin{pmatrix} -\nabla_x L_k \\ -G_k \lambda_k \\ -h_k \end{pmatrix} \tag{11}$$

onde o subíndice "$k$" implica avaliação das funções no ponto $(x_k, \lambda_k, \mu_k)$, $\Lambda_k = \text{diag}(\lambda_k)$, $G_k = \text{diag}(g_k)$, $L$ e $H$ são, respectivamente, a função Lagrangiana e a matriz Hessiana definidas como:

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m} g_i(x)\lambda_i + \sum_{i=1}^{p} h_i(x)\mu_i \tag{12}$$

$$H(x, \lambda, \mu) = \nabla^2 f(x) + \sum_{i=1}^{m} \nabla^2 g_i(x)\lambda_i + \sum_{i=1}^{p} \nabla^2 h_i(x)\mu_i \tag{13}$$

Definindo as variáveis auxiliares $\overline{\lambda} = \lambda_k + d_\lambda$ e $\overline{\lambda} = \lambda_k + d_\lambda$ o sistema (11) pode ser reescrito como:

$$\begin{pmatrix} H_k & \nabla g_k & \nabla h_k \\ \Lambda \nabla g_k^T & G_k & 0 \\ \nabla h_k^T & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x \\ \overline{\lambda} \\ \overline{\mu} \end{pmatrix} = \begin{pmatrix} -\nabla f_k \\ 0 \\ -h_k \end{pmatrix} \tag{14}$$

É mostrado na referência [3] que a direção de Newton pode não ser uniformemente viável para a região $\Delta$ nem de descida para a função potencial $\phi_c$. Por esta razão a direção de busca do algoritmo FAIPA é diferente da dada pela Eq. (14). O pseudo-código do algoritmo FAIPA é o seguinte:

**Algoritmo FAIPA**

Parâmetros: $\xi \in (0,1)$, $\eta \in (0,1)$, $\varphi > 0$, e $\nu \in (0,1)$.

Dados: $x_0 \in \Delta^0$, $\lambda_0 \in R^m$ positivo, $B_0 \in R^{n \times n}$ positiva definida, $\omega^I \in R^m$ positivo, $\omega^E \in R^p$ positivo e $c_0 \in R^m$ positivo.

**Passo 1:** Teste de convergência

**Passo 2:** Cálculo da direção de busca $d_x$

**2.1** Ache $(d_x^\alpha, \lambda^\alpha, \mu^\alpha)$ e $(d_x^\beta, \lambda^\beta, \mu^\beta)$ solução do sistema:

$$\begin{pmatrix} B_k & \nabla g_k & \nabla h_k \\ \Lambda \nabla g_k^T & G_k & 0 \\ \nabla h_k^T & 0 & 0 \end{pmatrix} \begin{pmatrix} d_x^\alpha & d_x^\beta \\ \lambda^\alpha & \lambda^\beta \\ \mu^\alpha & \mu^\beta \end{pmatrix} = \begin{pmatrix} -\nabla f_k & 0 \\ 0 & -\Lambda_k \omega^I \\ -h_k & -\omega^E \end{pmatrix} \tag{15}$$

**2.2** Se $d_x^\alpha = 0$ pare.

**2.3** Se $(c_k)_i < -1.2\mu_i^\alpha$ faça $(c_k)_i < -2\mu_i^\alpha$, para todo $i \in \{1,..., p\}$.

**2.4** Se $\nabla \phi_{c_k} d_x^\beta > 0$ faça:

$$\rho = \min\left( \varphi \left\| d_x^\alpha \right\|^2, (\xi - 1)\nabla \phi_{c_k} d_x^\alpha / \nabla \phi_{c_k} d_x^\beta \right)$$

Se não faça:

$$\rho = \varphi \left\| d_x^\alpha \right\|^2$$

**2.5** Calcule a direção de busca $d_x$ e o vetor $\overline{\lambda}$:

$$d_x = d_x^\alpha + \rho d_x^\beta, \quad \overline{\lambda} = \lambda^\alpha + \rho \lambda^\beta$$

**Passo 3:** Cálculo da direção para busca em arco $\tilde{d}_x$

**3.1** Calcule:

$$\omega_i^I = g_i(x_k + d_x) - g_i(x_k) - \nabla g_i^T d_x, \quad i \in \{1,..., m\}$$
$$\omega_i^E = h_i(x_k + d_x) - h_i(x_k) - \nabla h_i^T d_x, \quad i \in \{1,..., p\}$$

**3.2** Ache $(\tilde{d}_x, \tilde{\lambda}, \tilde{\mu})$ solução do sistema:

$$\begin{pmatrix} B_k & \nabla g_k & \nabla h_k \\ \Lambda \nabla g_k^T & G_k & 0 \\ \nabla h_k^T & 0 & 0 \end{pmatrix} \begin{pmatrix} \tilde{d}_x \\ \tilde{\lambda} \\ \tilde{\mu} \end{pmatrix} = \begin{pmatrix} 0 \\ -\Lambda_k \tilde{\omega}^I \\ -\tilde{\omega}^E \end{pmatrix} \tag{16}$$

**Passo 4:** Busca linear

Calcule $t_k$ como o primeiro número da seqüência $\{1, \nu, \nu^2, \nu^2, ...\}$ que satisfaz:

$$\phi_{c_k}(x_k + t_k d_x + t_k^2 \tilde{d}_x) \le \phi_{c_k}(x_k) + t_k \eta \nabla \phi_{c_k} d_x$$
$$h(x_k + t_k d_x + t_k^2 \tilde{d}_x) \le 0$$
$$g_i(x_k + t_k d_x + t_k^2 \tilde{d}_x) \le 0 \qquad \text{Se } \overline{\lambda}_i \ge 0 \text{ ou}$$
$$g_i(x_k + t_k d_x + t_k^2 \tilde{d}_x) \le g_i(x_k) \quad \text{caso contrário}$$

**Passo 5:** Atualização

**5.1** Faça:

$$x_{k+1} = x_k + t_k d_x + t_k^2 \tilde{d}_x$$

$$c_{k+1} = c_k$$

e defina novos valores: $\omega^I > 0$, $\omega^E > 0$, $\lambda_{k+1} > 0$ e $B_{k+1}$ positiva definida.

**5.2** Retorne ao Passo 1.

**Nota:** O algoritmo FDIPA tem convergência global a pontos que satisfazem as condições de KKT se algumas condições de regularidade forem satisfeitas pelas funções que definem o problema de otimização, ver [3]. A matriz $B_k$ é definida por um processo de atualização quasi-Newton BFGS. A busca linear do Passo 4 é busca de Armijo. Outras buscas lineares como as de Wolfe ou a de Goldstein podem ser utilizadas, ver [1][3].

# 1. REFERÊNCIAS


[1] HERSKOVITS, J., A View on Nonlinear Optimization, in Advances in Structural Optimization, Edited by J. Herskovits, Kluwer Academic Publishers, Dordrecht, Holland, p. 71-117, 1995.

[2] HERSKOVITS, J., Santos, G., Feasible Arc Interior Point Algorithms for Nonlinear Optimization. Fourth World Congress on Computational Mechanics, in CD-ROM, Buenos Aires, 1998.

[3] HERSKOVITS, J., A Feasible Directions Interior Point Technique for Nonlinear Optimization. JOTA – Journal of Optimization Theory and Applications, Vol. 99, n. 1, p. 121-146, 1998.

# APENDICE D

# DESCRIÇÃO DA HEURÍSTICA PARA VARIÁVEIS DISCRETAS

A heurística desenvolvida neste projeto minimiza uma função considerando um espaço de soluções discreto e limitado. A solução obtida por esta heurística encontra um mínimo local do seguinte problema:

$$\begin{aligned} &\min f(x) \\ &\textit{sujeito a}: \; x \in \Omega \end{aligned} \tag{1}$$

onde o conjunto viável $\Omega = \{x \in D \,|\, g(x) \leq 0\}$ é um conjunto finito de pontos do espaço $R^n$. O conjunto $D$ é o produto cartesiano de conjuntos discretos em $R$, ou seja, $D = D_1 \times ... \times D_n$, $D_i \subset R$, $\# D_i < \infty$, $i \in \{1,...,n\}$, sendo $n$ o número de variáveis. A função $g$ é uma função vetorial de $m$ restrições do problema, $g : R^n \to R^m$, sendo $m$ o número de restrições. A função $f$ é a função custo e vem dada por $f : R^n \to R$.

Define-se conjunto de pontos de descida como:

$$d(x) = \{y \in D \,|\, f(y) \leq f(x)\} \tag{2}$$

e conjunto de pontos adjacentes a $x$ como:

$$a(x) = \{y \in D \,|\, \delta(x, y) = 1\} \tag{3}$$

onde $\delta : D \times D \to \mathbb{Z}$ é uma distancia entre dois pontos do conjunto $D$. A definição de distancia é a seguinte:

$$\delta(x, y) = \sum_{i=1}^{n} B(x(i) \neq y(i)) \tag{4}$$

Onde $B(b)$ é uma função que retorna 1 se a condição booleana $b$ for verdadeira e 0 caso contrario.

A continuação pode-se expor a seguinte Condição de Otimalidade: O ponto $x$ é um mínimo local do problema (1) se e somente se:

$$x \in \Omega \text{ e } a(x) \cap d(x) \cap \Omega = \varnothing \tag{5}$$

Em palavras isto quer dizer que os pontos adjacentes a $x$ que são adjacentes e de descida, não estão na região viável.

A continuação apresenta-se uma heurística que busca uma solução do problema (1). A presente técnica não usa as derivadas da função custo $f$ nem das derivadas da função de restrições. A heurística requer de um ponto inicial dentro da região discreta $\Omega$. Em cada iteração avança para um ponto adjacente, de descida dentro da região viável. Ou seja, neste algoritmo se optou por uma estratégia *greedy* [1]. O processo de otimização termina quando não houver mais pontos adjacentes de descida dentro da região viável, de acordo com as condições de otimalidade (5).

**Heurística**

Dados. Valores discretos $D$, ponto inicial $x \in \Omega$.

**Passo 1.** Calcular os adjacentes $A := a(x)$. Inicializar $C := \varnothing$.

**Passo 2.** Para cada adjacente $y \in A$, fazer:

Se $f(y) \le f(x)$ e $y \in \Omega$, fazer $C := C \cup \{y\}$.

**Passo 3.** Se $C = \varnothing$, terminar, caso contrario continuar.

**Passo 4.** Sortear um ponto $y \in C$.

**Passo 5.** Designar novo ponto $x := y$ e voltar ao Passo 1.

A presente heurística foi utilizada na busca de uma solução discreta num entorno de uma solução continua. Por solução continua se entende a solução do problema (1) onde as variáveis de projeto podem tomar valores no conjunto dos números reais (ou seja, o conjunto $D = R$x...x$R = R^n$). Para a otimização continua se utilizou um algoritmo de pontos interiores chamado FAIPA. Este algoritmo obtém um mínimo local $x^*$ que verifica as condições de Karush-Kuhn-Tucker do problema nas variáveis continuas.

O custo computacional da Heurística esta dado no Passo 2 onde se calcula a função custo e se verifica a viabilidade do ponto adjacente $y$. Observe-se que segundo nossa definição de adjacente, a cardinalidade do conjunto $A$ é $2.n$ sendo $n$ o número de variáveis. A verificação de viabilidade $y \in \Omega$ implica a computação da desigualdade $g(y) \le 0$ e se esta desigualdade for custosa, a heurística torna-se inviável para problemas com grande numero de variáveis. Uma solução a este problema consiste em considerar uma aproximação linear $g_{LIN}$ da função $g$ no ponto $x^*$ como a continuação:

$$g_{LIN}(x) = g(x^*) + (x - x^*)^t \nabla g(x^*) \tag{6}$$

Finalmente, no Passo 2. substitui-se $y \in \Omega$ por $y \in \Omega_{LIN}$ onde está dado por:

$$\Omega_{LIN} = \{x \in D \mid g_{LIN}(x) \le 0\} \tag{7}$$

# 2. REFERÊNCIAS

[1] AHO, A.V., ULLMAN J.D., HOPCROFT J.E., Data Structures and Algorithms, Addison-Wesley Series in Computer Science and Information, 1983.

# APÊNDICE E

# OTIMIZAÇÃO BASEADA EM APROXIMAÇÕES

A otimização baseada em aproximações tem-se estabelecido como uma estratégia efetiva para o projeto de modelos computacionalmente custosos. Seu uso é conveniente em muitas situações e em alguns casos torna-se indispensável. Esse é o caso de modelos numéricos construídos com dados experimentais, modelos em que a análise de sensibilidade é computacionalmente custosa, modelos com derivadas descontínuas ou sem derivadas disponíveis e problemas de otimização complexos.

Os métodos de otimização que usam modelos aproximados foram originados em 1970´s [1] e são muito populares dentro da comunidade da engenharia. Existem inúmeras revisões destes métodos [2][3]. No entanto, muitos destes métodos têm sido inerentemente heurísticos, carecendo do rigor matemático necessário para ter um desempenho previsível. Em particular, eles funcionam bem em alguns problemas, mas não convergem à solução do modelo original em outros.

Nas aplicações da engenharia os modelos usados para aproximação podem ser classificados em dois tipos básicos: modelos "físicos" ou também chamados "fenomenológicos" e modelos "funcionais". Os modelos *fenomenológicos* consistem em soluções numéricas das equações governantes dos sistemas físicos incorporando assim conhecimento do sistema sob estudo, enquanto os modelos *funcionais* são aproximações funcionais das soluções numéricas sem recorrer ao conhecimento do sistema físico.

A distinção entre esses dois tipos de modelos não é absoluta. Há alguns modelos que podem ser definidos como pertencendo aos dois tipos. Por exemplo, os modelos em séries de Taylor são modelos funcionais já que a diferenciação das equações governantes é um processo puramente matemático que pode ser feito sem conhecimento algum do sistema. Pode-se dizer que este também é um modelo físico já que as derivadas das equações governantes descrevem o comportamento do mesmo sistema que as equações governantes descrevem.

Adicionalmente, os modelos funcionais podem se subdividir em "interpolantes" e "não interpolantes", segundo eles consigam, ou não, reproduzir exatamente os dados iniciais fornecidos para a construção do modelo.

Exemplos de modelos interpolantes são os obtidos ao resolver diretamente um sistema linear que representa os parâmetros de polinômios de Lagrange ou a combinação de parâmetros de uma tendência polinomial junto com os parâmetros de uma função de densidade de probabilidade, como é feito na Metodologia Kriging. Exemplos de modelos não interpolantes são os obtidos usando um ajuste por mínimos quadrados para obter uma tendência do tipo polinomial, como é feito na Metodologia das Superfícies de Resposta (*RSM, Response Surface Methodology*).

A Figura 1 ilustra a classificação mencionada.

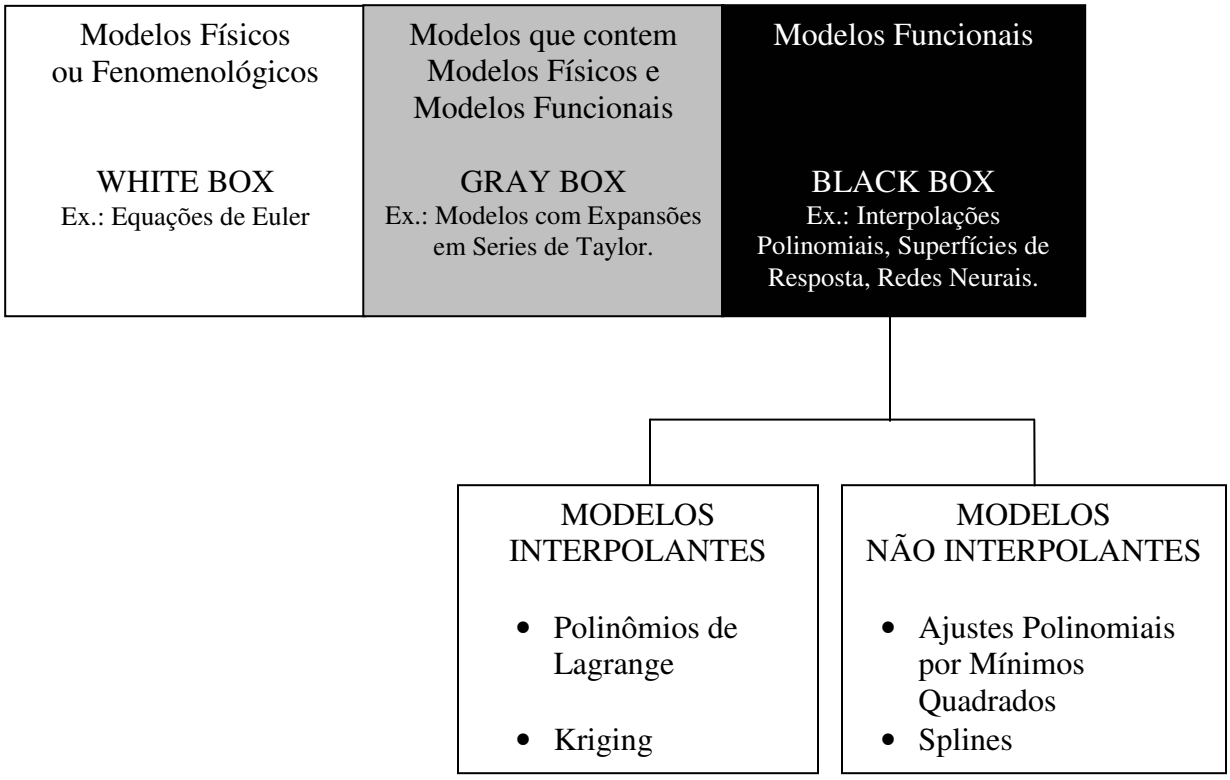



Figura 1: Classificação básica dos principais modelos aproximados usados em otimização.

A seguir, ilustram-se os conceitos relacionados aos principais modelos aproximados usados em otimização.

# 1. APROXIMAÇÕES EM SERIE DE TAYLOR

Uma abordagem usual em otimização em engenharia usando aproximações é mediante o uso de aproximações formais para criar um modelo simplificado do problema original complexo. Desta maneira a fase de análise não é custosa e o número de analises usadas durante a otimização não é crítica.

A metodologia mais comum consiste em criar algumas funções de aproximação para a função objetivo e as restrições. Para depois efetuar a otimização usando diretamente essas aproximações. Por ultimo atualizam-se as aproximações usando uma análise acurada do ótimo proposto e o processo continua até atingir a convergência ao ótimo exato.

A aproximação mais comum é o uso de expansões em séries de Taylor

$$f(\mathbf{x}) = f(\mathbf{x}^0) + \nabla \mathbf{f}(\mathbf{x}^0)^T \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T \left[ H(\mathbf{x}^0) \right] \delta\mathbf{x} + \ldots \tag{1}$$

onde $f(\mathbf{x})$ pode ser a função objetivo ou qualquer das funções de restrição, $\nabla f(\mathbf{x}^0)$ é o gradiente, e $\mathbf{H}(\mathbf{x}^0)$ é a matriz Hessiana. O vetor de perturbação $\delta\mathbf{x}$ é

$$\delta\mathbf{x} = \mathbf{x} - \mathbf{x}^0 \tag{2}$$

As aproximações baseadas em expansões em séries de Taylor são em geral boas num alcance local do espaço de projeto.

Se usarmos somente os primeiros dois termos do lado direito da Eq. (1), teremos uma aproximação linear do problema. Se resolvermos ele, seguido de um re-analise e resolvemos de novo repetidamente, teremos o método de Programação Linear Sucessiva (*SLP, Sequential Linear Programming*). Adicionando o terceiro termo teremos uma expansão de segunda ordem equivalente ao método de Newton para funções sem restrições. No entanto, o desejável é usar essas aproximações tanto para a função objetivo quanto para as restrições. Assumindo que a matriz Hessiana ***H*** é fornecida para a função objetivo e que também se dispõe dos gradientes $\nabla\mathbf{f}$ para cada restrição, isto vira um problema de Programação Quadrática Sucessiva. Dito problema pode ser resolvido usando métodos formais de programação quadrática ou por algum dos diversos métodos existentes para otimização com restrições.

O novo aqui é que o processo completo de otimização é realizado usando a Eq. (1), em vez de atualizar a informação do gradiente em cada iteração durante a otimização. Quando atualizamos nossa aproximação, pode não ser necessário a atualização de todos os termos. Se atualizamos somente o termo constante, isto requer uma avaliação da função. O termo linear requer $n$ avaliações de função adicionais em caso de usar gradientes obtidos por diferencias finitas. Finalmente, o computo da matriz simétrica Hessiana requer $n(n+1)/2$ avaliações de função adicionais. Então, dependendo da facilidade com que é obtida a informação poderíamos querer realizar vários ciclos de aproximação atualizando somente o termo constante. Depois atualizar os termos lineares e repetir, atualizando a matriz Hessiana só quando não se obtenha nenhum progresso adicional.

Seguindo esta linha de razoamento, pode ser desejável começar só com uma pequena quantidade de informação e criar a melhor aproximação possível. Depois otimizar usando esta aproximação, com limites móveis razoáveis nas variáveis de projeto. Este resultado é analisado com maior acurácia e os valores correspondentes da função usados para melhorar a aproximação. O conceito importante aqui é que não é necessario usar pequenos passos de diferencias finitas para aproximar $\nabla\mathbf{f}(\mathbf{x})$ e $\mathbf{H}(\mathbf{x})$, e em lugar disso, pode-se usar qualquer informação que dispor no momento.

Para esclarecer isto, expandindo a Eq. (1) usando somente os termos linear e quadrático obtemos

$$\begin{aligned}\Delta f &= \Delta f_1 \delta x_1 + \Delta f_2 \delta x_2 + \ldots + \Delta f_n \delta x_n \\ &+ \frac{1}{2}\left( H_{11}\delta x_1^2 + H_{22}\delta x_2^2 + \ldots + H_{nn}\delta x_n^2 \right) \\ &+ H_{12}\delta x_1 \delta x_2 + H_{13}\delta x_1 \delta x_3 + \ldots + H_{1n}\delta x_1 \delta x_n \\ &+ H_{23}\delta x_2 \delta x_3 + \ldots + H_{n-1,n}\delta x_{n-1} \delta x_n\end{aligned} \tag{3}$$

onde

$$\Delta f = f(\mathbf{x}) - f(\mathbf{x}^0) \equiv f - f^0 \tag{4}$$

$$\nabla f_i = \nabla f_i(\mathbf{x}^0) \qquad H_{ij} = H_{ij}(\mathbf{x}^0) \tag{5}$$

Agora assumindo que um projeto "nominal" tem sido analisado para dar $f^0$. Também, outros projetos $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^k$ têm sido analisados para dar $f^1, f^2, \ldots, f^k$.

Seja

$$\delta \mathbf{x}^i = \mathbf{x}^i - \mathbf{x}^0 \qquad i = 1, \ldots, k \tag{6}$$

e

$$\Delta f^i = f^i - f^0 \qquad i = 1, \ldots, k \tag{7}$$

Agora podem ser escritas $k$ equações na forma da Eq. (3). As incógnitas são $\nabla f_1, \nabla f_2, \ldots, \nabla f_n$ e $H_{11}, H_{12}, \ldots, H_{nn}$ para um total de $l$ = n+n(n+1)/2 incógnitas. Lembrando que para o projeto nominal se requer de uma analise, a quantidade total requerida de avaliações de projetos é $l$+1. Assim, se $k \geq l+1$, as incógnitas podem ser determinadas.

A Equação (3) é linear nas incógnitas. Escrevendo esta equação para cada um dos $k$ projetos leva a equações que podem ser resolvidas em forma direta. Se $k \geq l$, pode ser usado um ajuste por mínimos quadrados ponderados no qual aqueles projetos com a menor magnitude $\left|\delta \mathbf{x}^i\right|$ (projetos mais perto do nominal) recebem maior ponderação. Se menos de $l$ projetos vão ser usados, poucos coeficientes são calculados. Em geral, é desejável (mas não obrigatório) que sejam especificados $k \geq n+1$ projetos iniciais para prover ao menos uma aproximação inicial linear para cada função. Também, na medida em que $k \leq l$, os projetos têm que ser linearmente independentes para evitar uma matriz singular na determinação dos coeficientes. Se todos os projetos estão muito perto de $\mathbf{x}^0$, a Eq. (3) fornece a forma de diferencias finitas de uma expansão em series de Taylor de

segunda ordem. Se os projetos estão distribuídos de uma maneira mais randômica através do espaço de projeto, estaríamos no caso de uma aproximação polinomial quadrática.

É importante reconhecer que a aproximação quadrática mencionada aqui não é a única opção possível. Qualquer ajuste por curvas pode ser usada para aproximar as funções de maneira razoável. Também, não é necessário que as funções estejam definidas numérica ou analiticamente. Poder-se-iam usar dados experimentais, usando otimização com aproximações para identificar uma proposta de projeto melhorada para testar experimentalmente. Desta maneira, a otimização pode ser usada para reduzir dramaticamente o número de pontos de teste num estúdio experimental.

### 1.1. Aproximações Conservativas

As dificuldades em aplicar técnicas de aproximação a respostas gerais foi superada com o uso de "aproximações conservativas".

Este conceito foi proposto por Starnes e Haftka [4] e depois foi refinado por Fleury e Braibant [5]. A idéia básica é criar uma aproximação conservativa para a resposta em questão.

Primeiro considera-se uma aproximação direta (linear) de uma restrição. Isto é

$$g_d(\mathbf{x}) \approx g(\mathbf{x}^0) + \sum_{i=1}^{N} \frac{\partial g(\mathbf{x}^0)}{\partial x_i}(x_i - x_i^0) \tag{8}$$

Alternativamente, pode se usar uma aproximação recíproca, ou seja, a aproximação da restrição vai estar expressada em termos do recíproco da variável de projeto. Isto é

$$g_r(\mathbf{x}) \approx g(\mathbf{x}^0) - \sum_{i=1}^{N} \frac{\partial g(\mathbf{x}^0)}{\partial x_i}\left(\frac{1}{x_i} - \frac{1}{x_i^0}\right)(x_i^0)^2 \tag{9}$$

Nenhuma destas duas aproximações pode ser melhor em geral. Também, em certos casos a aproximação recíproca pode não ser válida, já que a variável de projeto pode se aproximar ou cruzar o zero. Na ausência de melhor informação, pode se desejar que a aproximação seja conservativa relativa a uma aproximação linear, assim

$$\begin{aligned} &\text{Se} \quad x_i \frac{\partial g(\mathbf{x}^0)}{\partial x_i} \geq 0 \qquad \text{Usar Expansão Direta} \\ &\text{Se} \quad x_i \frac{\partial g(\mathbf{x}^0)}{\partial x_i} < 0 \quad \text{Usar Expansão Recíproca} \end{aligned} \tag{10}$$

Isto é considerado como a melhor aproximação na ausência de melhor informação. No entanto, tem que ser lembrado que isto somente é conservativa em relação a uma aproximação linear. Esta não pode ser conservativa em relação à função real.

# 2. APROXIMAÇÕES BASEADAS EM META-MODELOS OU MODELOS SUBSTITUTOS

A maioria dos projetos em engenharia precisa aplicar-se vários análises, e com isto, muitas execuções de códigos computacionais complexos, fornecendo um vetor de variáveis de projeto (entradas) *x* e calculando um vetor de respostas (saídas) *y*. Isto é ilustrado na Figura 2.

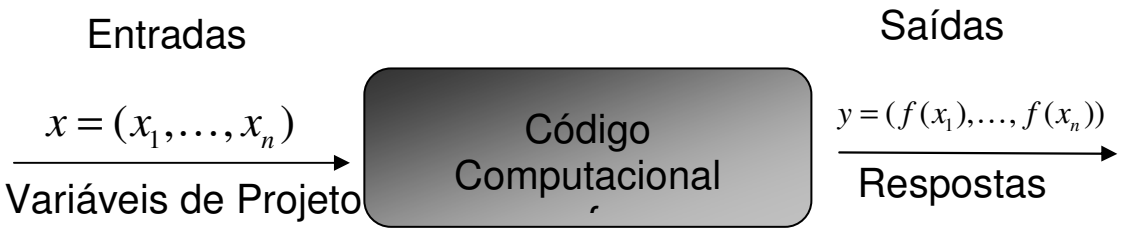



Figura 2: Representação esquemática das Superfícies de Resposta.

O custo de executar muitos códigos de análise na atualidade ainda é alto. Por exemplo, em aerodinâmica e mecânica dos fluidos.

O alto custo computacional dos analises complexos limitam ou proíbem seu uso em Otimização Multidisciplinar (*MDO, Multidisciplinary Design Optimization*) e em geral em otimização em engenharia.

Na prática procura-se estabelecer uma relação funcional entre *x* e *y* usando técnicas estadísticas, construindo aproximações dos códigos computacionais custosos. Tais aproximações são mais "econômicas" de avaliar e permitem uma maior compreensão na relação entre *x* e *y*. Assim, se a relação funcional do código de análise computacional real é

$$y = f(x)$$

Então, o modelo de aproximação pode se expressar como

$$\hat{y} = \hat{f}(x) \quad \text{e assim} \quad y = \hat{y} + \varepsilon \text{, onde } \varepsilon = \varepsilon_{\text{desvio}} + \varepsilon_{\text{aleatório}}$$

O erro $\varepsilon$ contempla tanto o erro de aproximação (*desvio*) $\varepsilon_{\text{bias}}$ quanto o erro de medição (*aleatório*) $\varepsilon_{\text{random}}$.

Já que o modelo de aproximação atua como um substituto (*surrogate*) para o código original, este é chamado com freqüência "modelo substituto", "modelo aproximado", ou "metamodelo" (isto é um "modelo de um modelo").

A estratégia mais comum para trabalhar com metamodelos consiste em aplicar projeto de experimentos [6] (*DOE, Design Of Experiments*) para identificar um conjunto eficiente de valores de parâmetros de entrada $(x_1, x_2, \ldots, x_n)$ e depois usar regressão para criar um ajuste polinomial do código de análise (também chamado modelo de alta fidelidade).

A construção de metamodelos ou modelos substitutos é conhecida como "*metamodelagem*". Como se ilustra na Figura 3, a metamodelagem pode ser vista como um problema inverso não linear para o qual o objetivo é determinar uma função continua ( $\hat{f}(x)$, $\hat{f} : R^n \rightarrow R$ ) a partir de uma quantidade limitada de dados disponíveis (**f**). Os dados disponíveis **f** são determinísticos por natureza e podem representar avaliações exatas da função *f* ou observações com ruído, e em geral, não podem levar a informação suficiente para identificar de modo único a $\hat{f}$ (múltiplos modelos substitutos podem ser consistentes com os dados disponíveis).

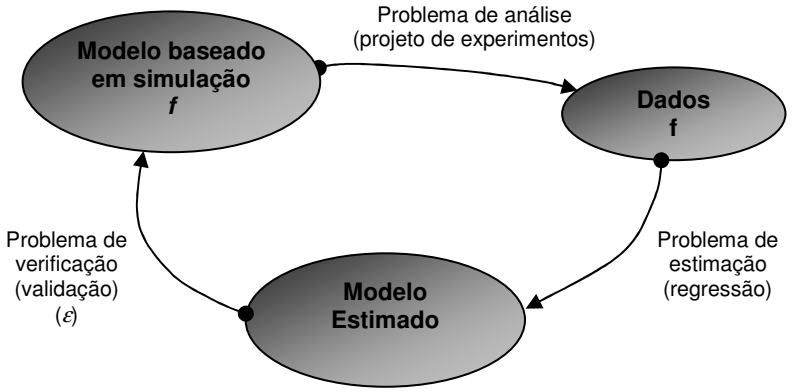



Figura 3: Anatomia da Metamodelagem.

Uma estratégia típica de metamodelagem abrange os seguintes passos:

i) escolher um projeto de experimentos para gerar os dados,

ii) escolher um modelo para representar os dados, e

iii) ajustar o modelo aos dados observados.

Existem varias opções para cada um destes passos, como se ilustra na Figura 4, e nesta seção explicar-se-á brevemente a técnica usada na presente pesquisa.

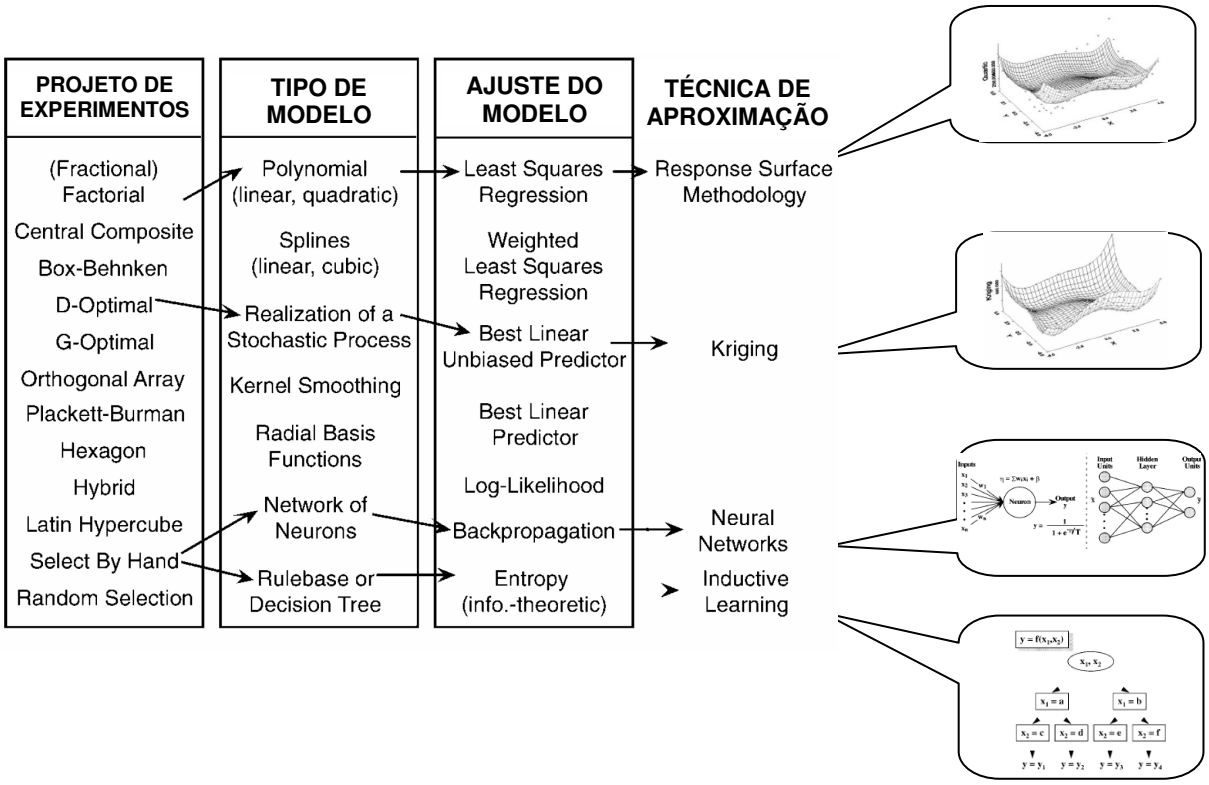



Figura 4: Técnicas para metamodelagem.

## 2.1. Projeto de Experimentos

Para a primeira fase do procedimento de otimização precisa-se de um plano inicial de experimentos. Isto ajuda a realizar uma exploração global simplificada do espaço de projeto e também ajuda a construir o modelo aproximado.

A seleção de pontos no espaço de projeto onde a resposta tem que ser avaliada é freqüentemente chamada projeto de experimentos. Assim, um projeto de experimentos representa a seqüência de experimentos (analises) a ser realizada, expressada em termos de *fatores* (variáveis de projeto) fixos em *níveis* especificados (valores predefinidos). Um projeto de experimentos é comumente representado pela matriz *X* onde as linhas denotam execuções de experimentos, e as colunas denotam as configurações de um dado fator. Assim, *X* pode ser representada como

$n_e$-essimo experimento (11)

**configurações do $n_e$-essimo fator**

Existem muitas maneiras de criar um projeto de experimentos. Algumas delas podem ser adotadas da teoria clássica de projeto de experimentos [7]: projeto fatorial reduzido, projeto composto central, arranjo ortogonal e projeto ótimo (A-, D-, G-, etc...).

### 2.1.1. Projeto fatorial completo

Antes de criar um projeto de experimentos, define-se o alcance permitido para cada uma das *n* variáveis mediante limite inferior e superior. Depois se discretiza o alcance em níveis igualmente espaçados. Para conseguir estabilidade numérica e para facilitar a notação escala-se o alcance de cada variável para cobrir [-1,1].

A região confinada pelos limites inferior e superior nas variáveis é denominada espaço de projeto, e seus vértices determinam um cubo *n*-dimensional. Assim, um projeto $2^n$ fatorial completo é criado se cada uma das variáveis é especificada somente nos limites inferior e superior (dois níveis). Similarmente um projeto $3^n$ fatorial completo é criado ao especificar os limites inferior, intermédio e superior (três níveis: [-1,0,1]) para cada uma das *n* variáveis. Na Figura 5 se ilustra um projeto $3^3$ fatorial completo.

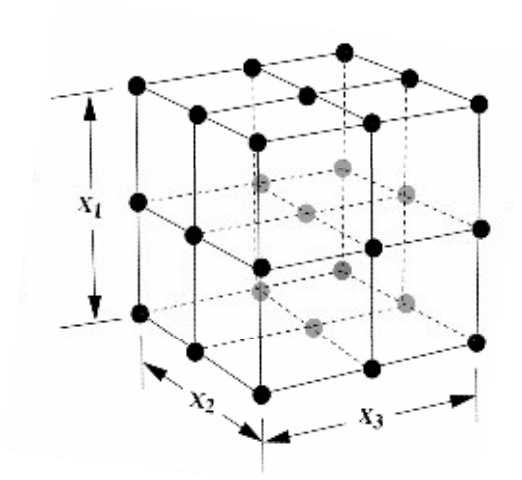



Figura 5: Projeto fatorial completo para três variáveis com três níveis ($3^3$).

Os projetos fatoriais completos viram impraticáveis na medida em que aumenta o número de variáveis de projeto ($n > 10$). Por exemplo para $n = 29$ usando o projeto $2^n$ fatorial completo precisaríamos de $2^{29} \approx 5.4*10^8$ avaliações (analises). Por isto, o projeto fatorial completo é usado tipicamente para dez ou menos variáveis.

### 2.1.2. Projeto aleatório

Um projeto aleatório consiste num número específico de pontos de projeto, distribuídos de maneira aleatória através do espaço de projeto. Isto significa que todo ponto de projeto no espaço de projeto tem a mesma probabilidade de ser escolhido como o próximo ponto aleatório.

Em geral evita-se o uso de projetos aleatórios pois eles não são construídos a partir de critérios estadísticos.

### 2.1.3. Projeto de hiper-cubo de Latin

O projeto de hiper-cubo de Latin (*LH, Latin Hypercube*) é uma estratégia de amostragem estratificada, o que garante que todas as porções de uma dada região sejam amostradas. Uma amostra de tamanho $n_e$ pode-se construir ao dividir o alcance de cada variável de entrada em $n_e$ estratos de igual probabilidade marginal e pegar uma amostra de cada estrato. Denotamos esta amostra por $x_i^{(p)}$, $i = 1,\ldots,n$; $p = 1,\ldots,n_e$. A amostra é feita de componentes de cada uma das ($x_i$), combinadas aleatoriamente. A Figura 6 ilustra um projeto *LHS* para obter 5 experimentos ($n_e$ = 5) a partir de duas variáveis de projeto ($n$ = 2).

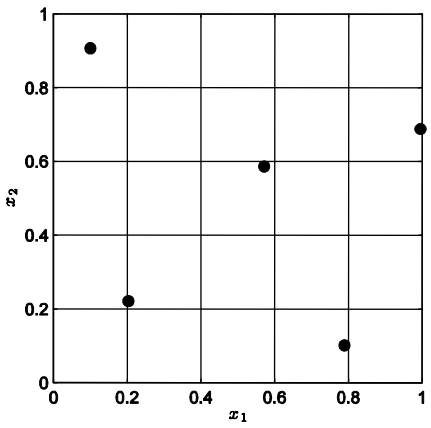



Figura 6: Projeto *LHS* com n = 2, $n_e$ = 5 para $x = (x_1, x_2)$ uniformemente distribuído num quadrado unitário.

### 2.1.4. Projeto *D*-ótimo

Os projetos *D*-ótimos são selecionados usando um critério de eficiência conhecido como *D*-eficiência, que classifica quantitativamente os diferentes pontos selecionados para possibilitar comparações e a identificação do "melhor" dos projetos. O uso de critérios baseados na variância, de maneira similar à *D*-eficiência, procura selecionar um conjunto de pontos que fazem que o modelo de superfície de resposta seja insensível ao ruído nos dados usados para a construção deles. Isto vem da noção de que um bom conjunto de pontos pode atingir certas propriedades na matriz de informação *X'X*, que ingresa nas equações da regressão por mínimos quadrados.

Assim, um projeto de experimentos *D*-ótimo é por definição aquele que maximiza o determinante $|X'X|$. Adicionalmente, a matriz *X'X* precisa ser não singular. O determinante $|X'X|$ é inversamente proporcional ao quadrado do (hyper)volume da região de confiança nos coeficientes da regressão, com isto o projeto *D*-ótimo conduz a

coeficientes de regressão com mínima região de confiança e com isto a um alto grau de certeza. Uma característica importante do projeto D-ótimo é que o número de avaliações a serem feitas não está restrito a, por exemplo, potências de 2 ou 3 como no caso de projetos fatoriais.

## 2.2. Superfícies de Resposta

Um modelo de aproximação por superfícies de resposta consiste numa função, $\hat{\mathbf{y}}(\mathbf{X},\boldsymbol{\beta})$, que contém um vetor $\boldsymbol{\beta}$ de $n_{cr}$ coeficientes de regressão, a ser determinado partindo de $n_e$ ($n_e > n_{cr}$) observações (analises). O excedente do número de experimentos para o número de coeficientes de regressão implica um modelo de superfície de resposta com erros entre o valor verdadeiro e o aproximado da resposta em cada ponto de amostragem (ver Eq. (12)).

$$\mathbf{y} = \hat{\mathbf{y}} + \mathbf{e} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e} \tag{12}$$

onde $\mathbf{y}$ é o vetor de valores verdadeiros da função de resposta, $\hat{\mathbf{y}}$ é o vetor correspondente de respostas aproximadas, $\mathbf{e}$ é o vetor de erros de aproximação, $\boldsymbol{\beta}$ é o vetor de coeficientes de regressão, e $\boldsymbol{X}$ é a Matriz do Modelo, que depende do polinômio usado em particular.

Os erros na superfície de resposta representam fontes de variabilidade diferentes aos que estão inerentemente conteúdos no modelo. Eles podem ser subdivididos em: a) ruído numérico aleatório, e b) erros de tendência no modelo como resultado de um modelo insuficiente.

Se assume que os erros são independentes e identicamente distribuídos segundo a distribuição normal com media zero e variância, $\sigma^2$. Atualmente, não existe garantia de que isto seja certo mas é uma suposição muito prática que possibilita o tratamento estadístico das observações.

Na Eq. (12), tem sido escrito um modelo de regressão linear como o produto entre a matriz do modelo $\mathbf{X}$ e o vetor de coeficientes de regressão $\boldsymbol{\beta}$. Cada linha na matriz do modelo $\mathbf{X}$ contém valores das funções de forma avaliadas nos pontos amostrados.

Considerando um modelo quadrático de superfície de resposta a Eq. (12) pode ser escrita como:

$$\begin{aligned} \hat{y} = \hat{f}(x) &= b_0 + \sum_{j=1}^{n} b_j x_j + \sum_{j=1}^{n} b_{jj} x_j^2 + \sum\sum_{i<j} b_{ij} x_i x_j \\ &\equiv f_0 + \tilde{f}^T x + \frac{1}{2} x^T \hat{S} x \end{aligned} \tag{13}$$

onde $\hat{S}$ é a Hessiana exata de $\hat{f}$, e representa a Hessiana aproximada de *f*. Para este caso o número de coeficientes de regressão é de $n_{cr} = \frac{1}{2}(n+1)(n+2)$.

Considerando um modelo linear a Eq. (12) pode ser escrita como:

$$\hat{y} = \hat{f}(x) = b_0 + \sum_{j=1}^{n} b_j x_j \equiv f_0 + \tilde{f}^T x \tag{14}$$

Para este caso o número de coeficientes de regressão é de $n_{cr} = n+1$.

A obtenção do vetor de coeficientes de regressão **b** por mínimos quadrados requer resolver o seguinte problema:

$$\underset{\mathbf{b}}{\text{minimizar}} \ \left\| \mathbf{Xb} - \mathbf{y} \right\|_2 \tag{15}$$

A solução do problema (15) é dada por

$$\mathbf{b} = \left( \mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{Xy} \tag{16}$$

A técnica original da aproximação por superfícies de resposta consiste em obter a função de aproximação $\hat{f}(x)$, onde **b** é a solução de (16), e representa o vetor de coeficientes do polinômio de aproximação. No entanto, neste trabalho precisamos impor um requerimento de consistência que consiste em que o valor da função aproximada coincida com o valor da função exata no projeto atual $\mathbf{x}^k$, isto é

$$\hat{f}(\mathbf{x}) = f(\mathbf{x}) \tag{17}$$

Para atingir o requerimento de consistência da Eq. (17) precisamos fazer uma pequena mudança no sistema de referência das variáveis de projeto, é preciso fazer que o novo origem das variáveis de projeto seja o projeto atual $\mathbf{x}^k$. Isto é chamado "shifting". Para isto definimos a nova variável $x_{sh} = x - x^k$. Com isto resolvemos para $b_{sh}$ a partir de (16) substituindo **X** por $\mathbf{X}_{sh}$, **y** por $\mathbf{y}^k$. Isto é: $\mathbf{X} \leftarrow \mathbf{X}_{sh}$, $\mathbf{y} \leftarrow \mathbf{y} - \mathbf{y}^k$.

Com isto o polinômio de aproximação que pasa por $\mathbf{x}^k$ é dado por:

$$\hat{f}(\mathbf{x}) = \mathbf{X}^T \mathbf{b}_{sh} + f(\mathbf{x}^k) - \left( \mathbf{x}^k \right)^T \mathbf{b}_{sh} \tag{18}$$

# 3. REFERÊNCIAS

[1] SCHMIT JR L. A. AND MIURA H. Approximation concepts for efficient structural synthesis. CR 2552, NASA, March 1976.

[2] BARTHELEMY J. F. M. AND HAFTKA R. T. “Function Approximations”, volume 150 of Progress in Astronautics and Aeronautics, chapter 4. AIAA, Washington, D.C., 1993.

[3] SOBIESZCZANSKI-SOBIESKI J. AND HAFTKA R. T. “multidisciplinary aerospace design optimization: Survey of recent developments”. Structural Optimization, v. 14, n. 1, PP. 1-23, 1997.

[4] STARNES, J. H. JR. AND R. T. HAFTKA: Preliminary Design of Composite Wings for Buckling, Stress and Displacement Constraints, Journal of Aircraft, Vol. 16, Aug. 1979, PP. 564-570.

[5] FLEURY, C. AND V. BRAIBANT: Structural Optimization: A New Dual Method using Mixed Variables, Int. J. of Numerical Methods in Engineering, Vol. 23, No. 3, 1986, PP. 409-429.

[6] MONTGOMERY D. C. Design and Analysis of Experiments. John Wiley, New York, 1997.

[7] SACKS J., WELCH W.J., MITCHELL T.J., WYNN H.P. “Design and analysis of computer experiments”. Statistical Science, Vol. 4, PP 409-435, 1989.