Notas em Matemática Aplicada 6

Editado por

**Eliana X.L. de Andrade**
Universidade Estadual Paulista-UNESP
São José do Rio Preto, SP, Brasil

**Rubens Sampaio**
Pontifícia Universidade Católica do Rio de Janeiro
Rio de janeiro, RJ, Brasil

**Geraldo N. Silva**
Universidade Estadual Paulista-UNESP
São José do Rio Preto, SP, Brasil

**Sociedade Brasileira de Matemática Aplicada e Computacional**

# Notas em Matemática Aplicada

1. Restauração de Imagens com Aplicações em Biologia e Engenharia
   Geraldo Cidade, Antônio Silva Neto e Nilson Costa Roberty
2. Fundamentos, Potencialidades e Aplicações de Algoritmos Evolutivos
   Leandro dos Santos Coelho
3. Métodos Matemáticos e Métodos Numéricos em Águas Subterrâneas
   Edson Wendlander
4. Métodos Numéricos para Equações Diferenciais Parciais
   Maria Cristina de Castro Cunha e Maria Amélia Novais Schleicher
5. Modelagem em Biomatematica
   Joyce da Silva Bevilacqua, Marat Rafikov e Cláudia de Lello Courtouke Guedes
6. **Métodos de Otimização Randômica: algoritmos genéticos e "simulated annealing"**
   **Sezimária F. Pereira Saramago**

# MÉTODOS DE OTIMIZAÇÃO RANDÔMICA: ALGORITMOS GENÉTICOS E SIMULATED ANNEALING

Sezimária F. Pereira Saramago
Faculdade de Matemática Universidade Federal de Uberlândia
Uberlândia - MG
saramago@ufu.br

**Sociedade Brasileira de Matemática Aplicada e Computacional**

São Carlos - SP, Brasil
2003

Coordenação Editorial: Geraldo Nunes Silva

Coordenação Editorial da Série: Geraldo Nunes Silva

Editora: SBMAC

Impresso na Gráfica: Soraya

Capa: Matheus Botossi Trindade

Patrocínio: SBMAC



**Catalogação elaborada pela Biblioteca do IBILCE/UNESP.**

Saramago, Sezimária F. Pereira
Métodos de Otimização Randômica: algoritmos genéticos e "simulated annealing- São Carlos, SP : SBMAC, 2003
x, 37 p. - (Notas em Matemática Aplicada; 6)

ISBN

1. Otimização 2. Algorítimos Genéticos 3. Matemática aplicada. 4. Computação científica. I. Saramago, Sezimária F. Pereira II. Título. III. Série

AMS 90C59/65K05
CDU 518.734

# PREFÁCIO DA SÉRIE

A Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC, desde a realização do primeiro CNMAC - Congressso Nacional de Matemática Aplicada e Computacional, publica monografias de cursos que são ministrados durante o Evento. A atual diretoria decidiu fazer a indexação bibliográfica dessas monografias através do ISBN para efeito de catalogação para preservação das mesmas para a memória dos CNMAC.

A coleção recebeu o título de "Notas em Matemática Aplicada" e consistirá das monografias dos cursos ministrados nos CNMAC. O livro correspondente a cada minicurso deve ser preparado em até 100 páginas para servir como texto introdutório, de modo que é aconselhável que contenha uma boa revisão bibliografica e exercícios de verificação de aprendizagem. A clareza do texto é um dos fatores mais importantes.

A coleção incluirá, gradativamente, os textos dos Encontros Regionais de Matemática Aplicada e Computacional, os ERMACs e de outros eventos patrocinados pela SBMAC.

Além disso, é objetivo desta coleção publicar volumes com coletâneas de *preprints* de trabalhos apresentados em reuniões científicas patrocinadas pela SBMAC.

Esta primeira coleção, composta das monografias dos minicursos do XXVI CNMAC, foi especialmente preparada em comemoração aos 25 anos da SBMAC.

E. X. L. de Andrade
R Sampaio
G N Silva

# Agradecimentos

Às pessoas que contribuíram de diferentes formas para a realização deste trabalho, lembrando dos alunos Eduardo Duarte Faria e Lidiane Sartini de Oliveira que desenvolveram pesquisas em Simulated Annealing e Algoritmos Genéticos.
Um agradecimento especial ao professor Antônio Carlos Nogueira que auxiliou na formatação da apostila.

# Conteúdo

# Capítulo 1

# Introdução

Otimizar é melhorar o que já existe, projetar o novo com mais eficiência e menor custo. A otimização visa determinar a melhor configuração de projeto sem ter que testar todas as possibilidades envolvidas.
Problemas de otimização são caracterizados por situações em que se deseja maximizar ou minimizar uma função numérica de várias variáveis, num contexto em que podem existir restrições. Tanto as funções acima mencionadas como as restrições dependem dos valores assumidos pelas variáveis de projeto ao longo do procedimento de otimização.

Pode-se aplicar otimização em várias áreas, como por exemplo no projeto de sistemas ou componentes, planejamento e análise de operações, problemas de otimização de estruturas, otimização de forma, controle de sistemas dinâmicos.

A otimização tem como vantagens diminuir o tempo dedicado ao projeto, possibilitar o tratamento simultâneo de uma grande quantidade de variáveis e restrições de difícil visualização gráfica e/ou tabular, possibilitar a obtenção de algo melhor, obtenção de soluções não tradicionais, menor custo.

Como limitações tem-se o aumento do tempo computacional quando aumenta-se o número de variáveis de projeto, pode-se surgir funções descontínuas que apresentam lenta convergência, funções com presença de muitos mínimos locais onde o mínimo global raramente é obtido.

As técnicas clássicas de otimização são conhecidas à bem mais de um século, sendo utilizadas na física e na geometria, servindo-se de ferramentas associadas às equações diferenciais ao Cálculo Variacional. A sofisticação dos recursos computacionais, desenvolvidos nos últimos anos, tem motivado um grande avanço nas técnicas de otimização. Aliado ao fato de que os problemas tornam-se cada vez mais complexos.

Técnicas clássicas de otimização são confiáveis e possuem aplicações nos mais diferentes campos de engenharia e de outras ciências. Porém, estas técnicas podem apresentar algumas dificuldades numéricas e problemas de robustez relacionados com: a falta de continuidade das funções a serem otimizadas ou de suas restrições, funções não convexas, multimodalidade, existência de ruídos nas funções,

necessidade de se trabalhar com valores discretos para as variáveis, existência de mínimos ou máximos locais, etc. Assim, os estudos de métodos heurísticos, com busca randômica controlada por critérios probabilísticos, reaparecem como uma forte tendência nos últimos anos, principalmente devido ao avanço dos recursos computacionais, pois um fator limitante destes métodos é a necessidade de um número elevado de avaliações da função objetivo [17].
Os algoritmos genéticos são mecanismos de busca baseados nos processos de seleção natural da luta pela vida e da genética de populações. Trata-se de um método pseudo-aleatório, portanto pode-se dizer que é um método, um procedimento de exploração inteligente, no espaço de parâmetros codificados [2].
O surgimento dos algoritmos genéticos deu-se por volta de 1950 quando vários biólogos usavam técnicas computacionais para a simulação de sistemas biológicos. Entre 1960 e 1970, na Universidade de Michigan, sob a direção de John Holland [8], iniciou-se o estudo de algoritmos genéticos como os conhecidos atualmente. David Goldberg [5] apresentou a solução de complexos problemas de engenharia usando algoritmos genéticos, o que ajudou o método a se tornar popular entre os pesquisadores.
Simulated Annealing, que pertence à classe de métodos que tentam simular os processos usados pela natureza para resolver problemas difíceis. Neste caso, a analogia é com o crescimento de um cristal simples de um metal fundido, que corresponde a encontrar o mínimo local da energia interna do metal, que é uma função da disposição dos átomos [4].

## 1.1 Problema Geral de Otimização

O problema geral de otimização consiste em minimizar uma função objetivo, sujeita, ou não, a restrições de igualdade, desigualdade e restrições laterais.

A função objetivo e as funções de restrições podem ser funções lineares ou não lineares em relação às variáveis de projeto, implícitas ou explícitas, calculadas por técnicas analíticas ou numéricas.

Seja o problema geral de otimização dado por:

Minimizar

$$F(X),\ X = [X_1, X_2, \ldots, X_n]^T,\ X \in \mathbb{R}^n \tag{1.1}$$

sujeito a

$$\begin{array}{ll} g_j \geq 0, & j = 1, 2, \cdots, J \\ h_k = 0, & k = 1, 2, \cdots, K \\ X_i^{(L)} \leq X \leq X_i^{(U)}, & i = 1, 2, \cdots, n \end{array} \tag{1.2}$$

onde, $F(X)$ representa a função objetivo, $g_j$ e $h_k$ as restrições de desigualdade e de

igualdade, $X_i^{(L)}$ e $X_i^{(U)}$ as restrições laterais. Todas estas funções assumem valores em $\mathbb{R}^n$ e são, na maioria dos casos, não-lineares.

## 1.2 Classificação dos Métodos

Os métodos para a solução de problemas de otimização se dividem em dois grupos, os métodos baseados no cálculo (Deterministic Optimization) e os métodos randômicos ou aleatórios (Random Strategies).

Quanto a presença de limitantes ao problema, tem-se a otimização irrestrita e a otimização restrita. Na otimização restrita tem-se os métodos indiretos (Métodos Seqüenciais e outros) e os métodos diretos (Programação Linear e outros).

Quanto ao número de variáveis, os métodos determinísticos não-lineares, se classificam da seguinte forma:

- para funções de uma única variável utiliza-se métodos de busca unidimensional como o método de busca uniforme, busca de Fibonacci, método da Seção Áurea, Aproximação Polinomial, Newton-Raphson e Bisseção;
- Para funções de várias variáveis utiliza-se métodos de primeira ordem que são baseados no cálculo do gradiente, métodos de segunda ordem que são baseados no cálculo da matriz Hessiana e métodos Quasi-Newton que utilizam uma matriz pseudo-Hessiana.

No grupo dos métodos aleatórios encontra-se os métodos de ordem zero (métodos tradicionais), Algoritmos Genéticos, Simulated Annealing, Redes Neurais e outros.

## 1.3 Revisão sobre Técnicas Seqüenciais

A maioria dos algoritmos seqüenciais de otimização requer um conjunto inicial de variáveis de projeto $X^0$ [18]. A partir daí, o projeto é atualizado iterativamente:

$$X^q = X^{q-1} + \alpha^* S^q, \tag{1.3}$$

onde $q$ representa o número da iteração, $S$ o vetor direção de busca no espaço de projeto, $\alpha^*$ é o escalar que define o passo que se deseja dar na direção de $S$.

Os algoritmos de otimização não-linear, baseados no cálculo, necessitam da determinação da direção de busca $S$ e do parâmetro escalar $\alpha^*$.

### 1.3.1 Existência e unicidade da solução ótima

Raramente pode-se garantir a existência e unicidade de um projeto ótimo, isto ocorre devido a existência de várias soluções, mal condicionamento numérico ou lenta convergência.

A estratégia prática utilizada é inicializar o processo de otimização à partir de diferentes configurações de $X^0$, caso se encontre o mesmo valor para $F_{min}$, tem-se alguma garantia de mínimo global.

Considerando problemas sem restrição, para que *F(x)* seja mínimo uma condição necessária, mas não suficiente, é que $\nabla F(x)=0$.

Para *F(x)*, função de uma variável, no ponto de mínimo a $2^a$ derivada deve ser positiva. Para o caso de uma função de várias variáveis, a matriz Hessiana $H$ deve ser positiva definida, o que implica dizer que todos os autovalores de $H$ devem ser positivos.

### 1.3.2 Estratégia geral de otimização

Os algoritmos de otimização são baseados na solução iterativa:

$$X^{q+1} = X^q + \alpha^* S^q, \tag{1.4}$$

ou seja,

- Determinar a direção de busca $S$;
- Executar a busca unidimensional e obter $\alpha_q^*$;
- Determinar quando o processo converge.

### 1.3.3 Critérios de convergência

Os critérios mais utilizados para testar a convergência de um programa de otimização podem se resumidos como [1]:

- Estabelecer o número máximo de iterações;
- Alteração absoluta da função objetivo:

$$\left|F(X^{q+1}) - F(X^q)\right| < \xi_a \tag{1.5}$$

- Alteração relativa da função objetivo:

$$\frac{\left|F(X^{q+1}) - F(X^q)\right|}{max\left[F(X^q) * 10^{-8}\right]} < \xi_r \tag{1.6}$$

- Obediência a condição Kuhn-Tucker:

$$|\nabla F(X^q)| < \xi_K \tag{1.7}$$

## 1.4 Métodos de Primeira Ordem (Método da Descida Máxima)

A direção de busca é dada pelo gradiente $\nabla F(X)$. Usando $\nabla F(X)$ limita-se a direção de busca, evitando a busca em todo espaço de projeto [15]. O gradiente deve ser recalculado a cada nova direção, conforme representado na Figura 1.1. Sua

importância é permitir estabelecer o ponto inicial para métodos mais sofisticados. Toma-se como direção de busca aquela oposta ao gradiente:

$$S^q = -\nabla F(X^q) \text{ onde } X^{q+1} = X^q + \alpha_q^* S^q. \tag{1.8}$$

A cada passo, determina-se $\alpha^*$ que ao mínimo nesta direção, busca unidimensional. Nestes métodos, a convergência é boa no início, mas muito lenta ao se aproximar do mínimo.

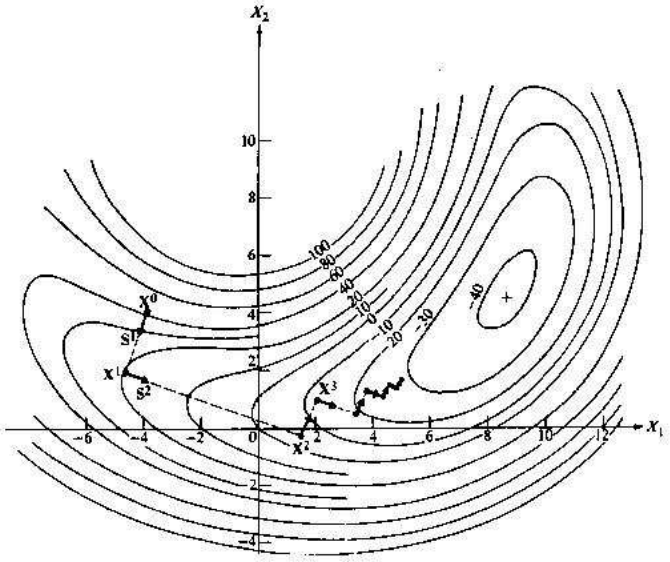



Figura 1.1: Representação do Método da Descida Máxima

# Capítulo 2

# Métodos de Ordem Zero

## 2.1 Introdução

São métodos simples, de fácil implementação, confiáveis e capazes de trabalhar com valores discretos. Requerem apenas o cálculo de $F(X)$, não dependem do gradiente e da continuidade da função. Necessitam de um grande número de avaliações da função objetivo, o que aumenta o custo computacional.

A idéia básica é selecionar um grande número de vetores de projeto $X$ e calcular $F(X)$ correspondente a cada um. O vetor correspondente ao menor valor de $F(X)$ será adotado como o valor ótimo $X^*$.

O vetor $X$ é selecionado de forma randômica no espaço de projeto. Para limitar a busca, utiliza-se as restrições laterais. Um número randômico $r$ é gerado, $r \in [0, 1]$ e as variáveis de projeto da q-ésima iteração atualizadas:

$$X_i^q = X_i^l + r(X_i^u - X_i^l). \tag{2.1}$$

O processo do método de Ordem Zero está apresentado no fluxograma da Figura 2.1. Neste caso, o critério de parada adotado é o número máximo de iterações. Porém, outros critérios podem ser incorporados ao programa.

## 2.2 Exemplo ilustrativo

Considere o problema de minimização de uma função escrita por:

$$\begin{array}{rcl} g(x,y) & = & xsen(4x) + 1,1ysen(2y) \\ \text{restrições laterais} & : & 8 < x < 10, \\ & & 8 < y < 10. \end{array} \tag{2.2}$$

A Figura 2.2 ilustra o gráfico da função $g(x, y)$ e suas curvas de nível, respectivamente.
Acompanhando, por exemplo, uma evolução do método de Ordem Zero aplicado

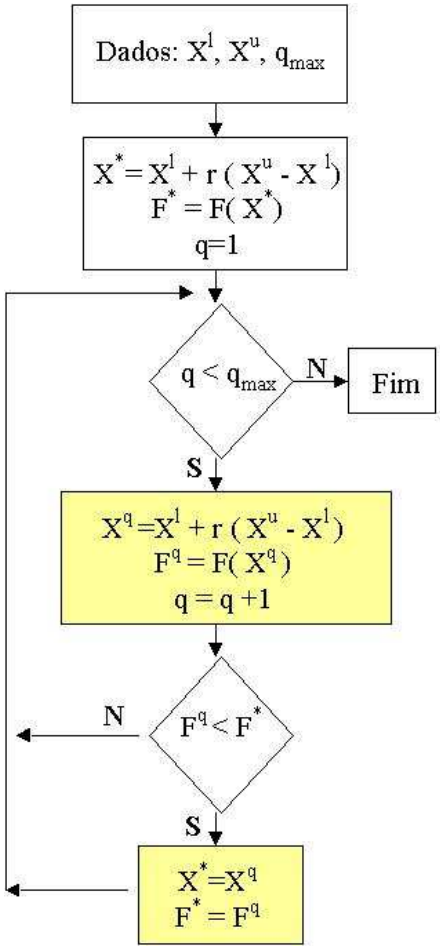
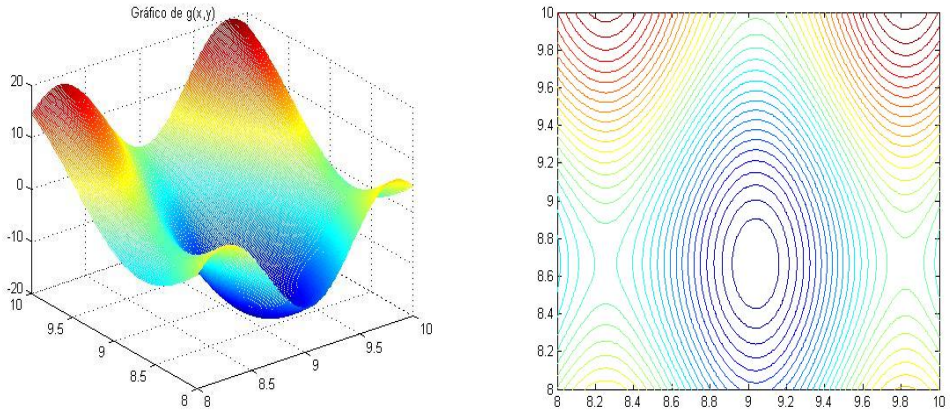



Figura 2.1: Esquema do Método de Ordem Zero

ao problema 2.2, para um máximo de 100 iterações, os melhores resultados podem ser acompanhados na Tabela 2.1, sendo que o valor ótimo foi encontrado na 58° iteração.




(a) (b)

Figura 2.2: (a) Gráfico da função da Equação 2.2; (b) Curvas de nível desta função

O ponto de mínimo obtido foi $F^* = -18.2155$, e as variáveis de projeto correspondentes ao ponto de mínimo $X^* = [9.0111; 8.7895]$.

| Iteração | X(1) | X(2) | F(X) |
|---|---|---|---|
| 3 | 8.1776 | 8.3915 | -0.2517 |
| 5 | 9.3963 | 8.2914 | -8.0677 |
| 8 | 9.1431 | 8.3260 | -15.674 |
| 26 | 8.9579 | 8.5686 | -17.898 |
| 58.000 | 9.0111 | 8.7895 | -18.216 |

Tabela 2.1: Evolução do Método de Ordem Zero aplicado à Equação 2.2

Os pontos randômicos criados pelo algoritmo podem ser visualizados na Figura 2.3, nota-se que o ponto mínimo obtido ainda pode ser melhorado.

## 2.3 Exercícios

I) Desenvolver um programa computacional para o Método de Ordem Zero.

II) Usando o programa obter o ponto de mínimo das funções abaixo:

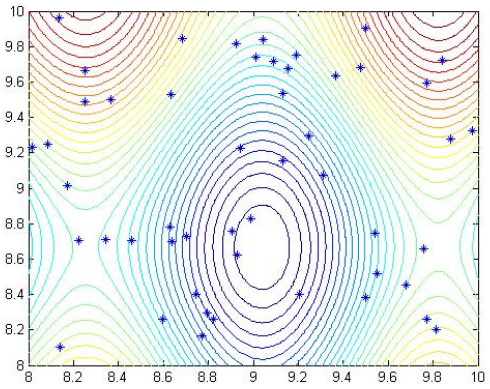



Figura 2.3: Evolução do Método de Ordem Zero aplicado à Equação 2.2

(a) $F(x, y) = (x + 2)^2 + (y - 1)^2$, $x \in [10; 10]$ e $y \in [-5; 5]$, qmax $= 100$;

(b) Mesmo exemplo, qmax$= 500$; qmax$= 1000$;

(c) Repetir 0a e 0b, com nova região viável: $x, y \in [-3; 3]$.

III) Obter o mínimo das seguintes funções :

(a) $F(X) = X_1^4 + X_2^4 + 32X_1 - 4X_2 + 52$;

(b) $F(X) = (X_1 + 10X_2)^2 + (X_3 - X_4)^2 + (X_2 - 2X_3)^4 + 10(X_1 - X_4)^4$.

# Capítulo 3

# Algorítmos Genéticos

Os algorítmos genéticos usam um vocabulário emprestado da genética natural. Fala-se sobre indivíduos (genótipos) de uma população. Estes indivíduos também são chamados de cromossomos [13].
Cromossomos são compostos de unidades ou elementos, cada elemento equivale a um gene, dispostos em uma seqüência linear. A Figura 3.1 exemplifica o exposto acima:

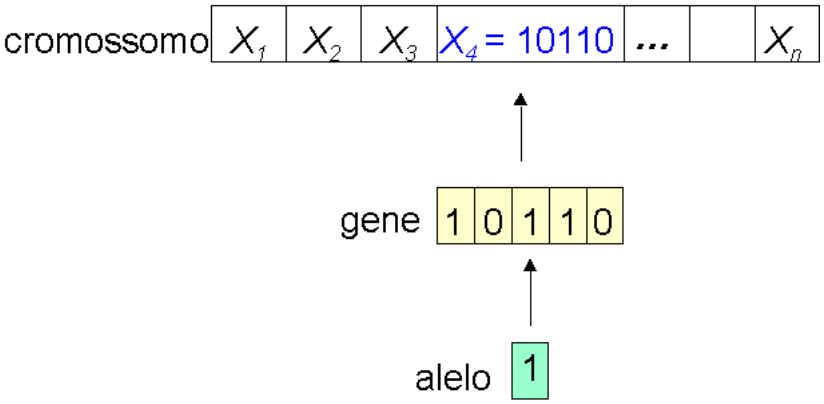



Figura 3.1: Esquematização de um cromossomo

Sendo n o número de variáveis (genes) e a cada gene tem o comprimento m. Assim, uma função de duas variáveis $f(x,\ y)$, n = 2, será representada através de um cromossomo com 2 genes. Seja m = 7, o número de alelos de cada gene. Neste caso, tem-se:

Cromossomo: $\left[ \underbrace{1100001}_{x} \quad \underbrace{0011101}_{y} \right]$

Algorítmos genéticos são algorítmos iterativos, e a cada iteração a população é modificada, usando as melhores características dos elementos da geração anterior e submetendo-as a três tipos básicos de operadores, para produzir melhores resultados. São eles:

**Reprodução:** é um processo no qual cada cadeia é copiada levando em conta os valores da função de adaptação $f$;

**Cruzamento:** é um processo no qual a combinação em partes de cada um de dois cromossomos gera um novo descendente;

**Mutação:** é a modificação aleatória ocasional (de baixa probabilidade) do valor de um alelo da cadeia.

O primeiro passo para a aplicação de algorítmos genéticos a um problema qualquer é encontrar alguma representação cromossômica conveniente, cujo gene represente o espaço de busca do problema, com vetores binários de zeros e um (0,1), os quais são gerados aleatoriamente. O comprimento m do gene depende da precisão requerida para o problema. Temos na Figura 3.2 um fluxograma do algorítmo genético segundo Haupt [7].

Com a finalidade de ilustrar a aplicação dos operadores, vamos considerar o problema de minimização de uma função por

$$g(x, y) = xsen(4x) + 1, 1ysen(2y) \tag{3.1}$$

no intervalo, $8 < x < 10$, $8 < y < 10$, que define a região viável do problema. O gráfico da função e suas curvas de nível já foram apresentados na Figura 2.2.

A maioria dos códigos computacionais para algorítmos genéticos costuma maximizar a função, portanto, a função objetiva em estudo será reescrita como

$$max\, g(x, y) = -[xsen(4x) + 1, 1ysen(2y)]. \tag{3.2}$$

Para este exemplo, será adotada a precisão de duas casas decimais. Como o espaço de busca, ou seja, o domínio da função tem amplitude $I = 10 - 8 = 2$ e precisão de duas casas decimais, este intervalo deve ser dividido em $I * 10^m$ subintervalos iguais, neste caso, $2 \times 10^2 = 200$ pontos. Portanto a seqüência binária (cada gene) deverá ter pelo menos 8 bits, pois

$$128 = 2^7 < 200 < 2^8 = 256.$$

Seja a seguinte população inicial, obtida aleatoriamente:

$C_1$ - 10000101 00100111
$C_2$ - 00001110 00001001
$C_3$ - 10010001 00000001
$C_4$ - 11000101 00101001

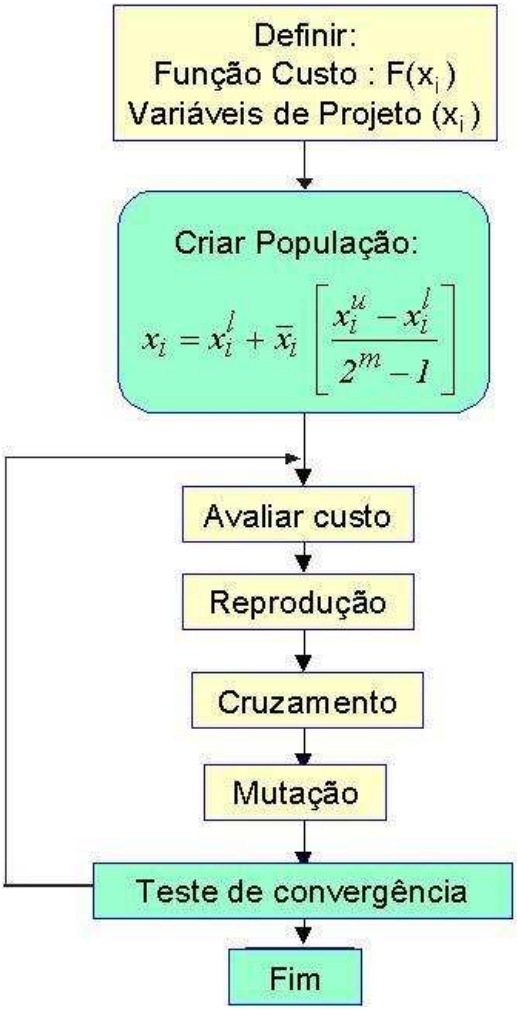



Figura 3.2: Fluxograma do algorítmo genético contínuo

$C_5$ - 01111100 10101100
$C_6$ - 11100010 01001010

Tem-se assim, a população inicial de cromossomos definida, onde cada gene é um vetor binário de m bits, sendo m função da precisão exigida ($10^{-2}$) e da amplitude do intervalo definido pelas restrições laterais (I = 2). A seguir, todos esses indivíduos serão modificados, submetendo-os aos operadores genéticos.

## 3.1 Reprodução

Reprodução é um processo que será atribuído às cadeias que possuem o maior valor objetivo e, portanto uma probabilidade mais elevada de contribuir à geração seguinte, criando pelo menos um descendente. Quanto maior o valor da função objetivo, maiores são as chances do indivíduo sobreviver no ambiente e reproduzir-se passando parte de seu material genético a gerações posteriores [2].

Usando a probabilidade, expressa pela Equação 3.3, tem-se que se o indivíduo for de baixa adequabilidade, tem alta probabilidade de desaparecer da população, caso contrário, os indivíduos terão grandes chances de permanecer na população.

$$P_i = \frac{f(x)}{F(x)}, \text{ sendo } F(x) = \sum f(x) \tag{3.3}$$

Para se calcular o valor da função de adaptação $f$, deve-se converter a seqüência binária (base 2) para base 10, ou seja, deve-se decodificar um cromossomo, conforme Equação 3.4.

$$C = [b_7 b_8 \cdots b_2 b_1 b_0 a_7 a_6 \cdots a_2 a_1 a_0]$$

$$\bar{x} = \sum_{i=0}^{m-1} b_i \times 2^i \text{ e } \bar{y} = \sum_{i=0}^{m-1} a_i \times 2^i \tag{3.4}$$

Feito isso, deve-se calcular o valor de $x$ e $y$ reais, dentro da região viável, através das seguintes equações:

$$y = a_i + decimal\,(010...100)_2 \times \tfrac{b_i - a_i}{2^m - 1}, \tag{3.5}$$

sendo

$a_i$ e $b_i$ - domínio das variáveis $x$ e $y$
$m$ - comprimento total do gene
decimal $(100...010)_2 = \overline{x}$
decimal $(010...100)_2 = \overline{y}$.

Como a população inicial já está definida, o próximo passo será o cálculo da função objetivo (adaptação).

A título de ilustração, será mostrado o cálculo da função objetivo do primeiro cromossomo da população criada para o Exemplo (3.2) em estudo.

Seja $C_1 = 1000010100100111$.

Passando para a base 10, utilizando as Equações 3.5 e 3.6, tem-se:

$$\overline{x} = \sum_{i=0}^{8-1} b_i \times 2^i = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 0 \times 2^4 + 0 \times 2^5 + 0 \times 2^6 + 1 \times 2^7$$

$$\overline{x} = 133,$$

e

$$\overline{y} = \sum_{i=0}^{7} a_i \times 2^i = 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 0 \times 2^4 + 1 \times 2^5 + 0 \times 2^6 + 0 \times 2^7$$

$$\overline{y} = 39.$$

Os valores reais $x$ e $y$,dentro da região viável, são dados por:

$$x = 8 + \frac{133 \times (10 - 8)}{2^8 - 1} \Rightarrow x = 9,04$$

e

$$y = 8 + \frac{39 \times (10 - 8)}{2^8 - 1} \Rightarrow y = 8,31.$$

O valor da função de adaptação é

$$g(x, y) = -[xsen(4x) + 1,1ysen(2y)]$$

$$g(x, y) = -[9,04sen(4 \times 9,04) + 1,1 \times 8,31sen(2 \times 8,31)]$$

$$g(x, y) = +16,26.$$

Obteve-se os resultados mostrados na Tabela 3.1, para cada cromossomo da população inicial.

Como citado anteriormente, a função de adaptação $g(x, y)$ é o árbitro final que decide sobre a vida ou a morte de cada cromossomo. O mecanismo para seleção das melhores cadeias, ou seja, as mais adaptadas, são definidas pelo uso das probabilidades proporcionais, dadas pela Equação 3.3, resultando os seguintes valores:

| Cromossomos | $x$ | $y$ | $g(x,\ y)$ |
|---|---|---|---|
| 1000010100100111 | 9,04 | 8,31 | 16,26 |
| 0000111000001001 | 8,11 | 8,07 | -3,21 |
| 1001000100000001 | 9,14 | 8,01 | 11,01 |
| 1100010100101001 | 9,55 | 8,32 | 2,76 |
| 0111110010101100 | 8,98 | 9,35 | 10,32 |
| 1110001001001010 | 9,77 | 8,58 | - 0,22 |
| g(x,y) | | | 36,92 |

Tabela 3.1: Cromossomos da população inicial

$$p_1 = \frac{16,26}{36,92} = 0,44$$

$$p_2 = \frac{-3,21}{36,92} = -0,09$$

$$p_3 = \frac{11,01}{36,92} = 0,30$$

$$p_4 = \frac{2,76}{36,92} = 0,07$$

$$p_5 = \frac{10,32}{36,92} = 0,28$$

$$p_6 = \frac{-0,22}{36,92} = -0,00$$

sendo $p_1 + p_2 + p_3 + p_4 + p_5 + p_6 = 1{,}00$.

Considerando que as probabilidades acumulativas $q_i$ cada cromossomo são dadas por:

$$q_i = \sum_{j=1}^{i} p_j. \tag{3.6}$$

Obtém-se os seguintes valores acumulativos:

$$q_1 = p_1 = 0,44$$

$$q_2 = p_1 + p_2 = 0,44 - 0,09 = 0,35$$

$$\begin{aligned} q_3 &= p_1 + p_2 + p_3 \\ &= 0,44 - 0,09 + 0,30 = 0,65 \end{aligned}$$

$$\begin{aligned} q_4 &= p_1 + p_2 + p_3 + p_4 \\ &= 0,44 - 0,09 + 0,30 + 0,07 = 0,72 \end{aligned}$$

$$\begin{aligned} q_5 &= p_1 + p_2 + p_3 + p_4 + p_5 \\ &= 0,44 - 0,09 + 0,30 + 0,07 + 0,28 = 1,00 \end{aligned}$$

$$\begin{aligned} q_6 &= p_1 + p_2 + p_3 + p_4 + p_5 + p_6 = \\ &= 0,44 - 0,09 + 0,30 + 0,07 + 0,28 + 0,00 = 1,00 \end{aligned}$$

A seguir deve-se selecionar as cadeias que irão contribuir para a geração seguinte. Esta seleção considera um conjunto de números r, escolhidos randomicamente entre [0,1], em quantidade igual ao número de cadeias.

A análise é feita através das seguintes opções:
Se r < $q_1$, então se seleciona o 1° cromossomo $C_1$.
Se r > $q_1$, então se passa para o subseqüente e faz a análise novamente.
Vale ressaltar que alguns cromossomos poderão ser selecionados mais de uma vez, ou seja, os melhores serão copiados mais vezes, enquanto que outros irão morrer.

Seja o exemplo em estudo. Considere que foram gerados os seguintes números randômicos:

$$\begin{aligned} \mathrm{r}_1 &= 0{,}64 \\ \mathrm{r}_2 &= 0{,}08 \\ \mathrm{r}_3 &= 0{,}47 \\ \mathrm{r}_4 &= 0{,}88 \\ \mathrm{r}_5 &= 0{,}93 \\ \mathrm{r}_6 &= 0{,}70 \end{aligned}$$

A seleção dos cromossomos é dada por:

$$r_1 = 0,64 > q_1 = 0,44 \Rightarrow r_1 = 0,64 > q_2 = 0,35 \Rightarrow r_1 = 0,64 < q_3 = 0,65,$$

e portanto seleciona-se $q_3 \Rightarrow C_3$.

$$r_2 = 0,08 < q_1 = 0,44,$$

e portanto seleciona-se $q_1 \Rightarrow C_1$.

$$r_3 = 0,47 > q_1 = 0,44 \Rightarrow r_3 = 0,47 > q_2 = 0,35 \Rightarrow r_3 = 0,47 < q_3 = 0,65,$$

e portanto seleciona-se $q_3 \Rightarrow C_3$.

$$\begin{aligned} r_4 = 0,88 > q_1 &= 0,44 \Rightarrow r_4 = 0,88 > q_2 = 0,35 \Rightarrow r_4 = 0,88 > q_3 = \\ &= 0,65 \Rightarrow r_4 = 0,88 > q_4 = 0,72 \Rightarrow r_4 = 0,88 < q_5 = 1,00, \end{aligned}$$

e portanto seleciona-se $q_5 \Rightarrow C_5$.

$$\begin{aligned} r_5 = 0,93 > q_1 &= 0,44 \Rightarrow r_5 = 0,93 > q_2 = 0,35 \Rightarrow r_5 = 0,93 > q_3 = \\ &= 0,65 \Rightarrow r_5 = 0,93 > q_4 = 0,72 \Rightarrow r_5 = 0,93 < q_5 = 1,00, \end{aligned}$$

e portanto seleciona-se $q_5 \Rightarrow C_5$.

$$\begin{aligned} r_6 = 0,70 > q_1 &= 0,44 \Rightarrow r_6 = 0,70 > q_2 = 0,35 \Rightarrow r_6 = 0,70 > q_3 = \\ &= 0,65 \Rightarrow r_6 = 0,70 < q_4 = 0,72, \end{aligned}$$

e portanto seleciona-se $q_4 \Rightarrow C_4$.

Depois de selecionados, os cromossomos dão origem a uma nova população representada como:

$C'_1 - 1001000100000001 \Rightarrow$ gerados de $C_3$; $g(x,y) = 11,01$
$C'_2 - 1000010100100111 \Rightarrow$ gerados de $C_1$; $g(x,y) = 16,26$
$C'_3 - 1001000100000001 \Rightarrow$ gerados de $C_3$; $g(x,y) = 11,01$
$C'_4 - 0111110010101100 \Rightarrow$ gerados de $C_5$; $g(x,y) = 10,32$
$C'_5 - 0111110010101100 \Rightarrow$ gerados de $C_5$; $g(x,y) = 10,32$
$C'_4 - 1100010100101001 \Rightarrow$ gerados de $C_4$; $g(x,y) = 2,76$.

## 3.2 Cruzamento

Tem-se várias formas para se obter o cruzamento, neste curso será utilizada a seguinte técnica para se fazer o cruzamento:

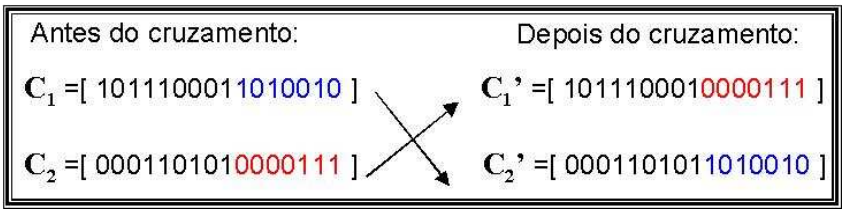



Figura 3.3: Representação do Operador Cruzamento

Seja o ponto k que define a posição de cruzamento na cadeia de bits de cada cromossomo escolhido aleatoriamente. A quantidade de cromossomos a ser submetida ao processo de cruzamento é definida através da probabilidade de cruzamento $p_c$, especificada pelo usuário. Cada cadeia é partida neste ponto k e todas as informações do cromossomo A, a partir do ponto escolhido, são copiadas para o cromossomo B e vice-versa, conforme esquematizada na Figura 3.3.

O processo de escolha de quem será cruzado deve ser feito em pares, sorteando números randômicos ($r_i$). Quando não for possível formar os pares um novo sorteio deverá ser feito até obter os pares necessários para o cruzamento. Por exemplo, se $r_1$ for menor que a probabilidade $p_c$, então o cromossomo $C_1$' será selecionado.

Na maioria das literaturas especializadas, a probabilidade de cruzamento é de $p_c = 25\%$, a qual será adotada neste trabalho.

Após de ter feito isso, temos que gerar um novo número randômico para determinar a posição $k$, onde duas novas cadeias são formadas pela troca de todos os caracteres compreendidos entre as posições $k + 1$ e $m$. Esta posição $k$ é determinada pela seguinte fórmula:

$$k = 1 + rand[(m - 1) - 1]. \tag{3.7}$$

Dando continuidade ao Exemplo (3.2) em estudo, submetem-se as populações ($C_i$') ao cruzamento. Sejam os seguintes números randômicos, $r_i$:

$$\begin{array}{l} r_1 - 0,50 \Rightarrow C'_1 > p_c \\ r_2 - 0,17 \Rightarrow C'_2 < p_c \\ r_3 - 0,40 \Rightarrow C'_3 > p_c \\ r_4 - 0,15 \Rightarrow C'_4 < p_c \\ r_5 - 0,20 \Rightarrow C'_5 < p_c \\ r_6 - 0,23 \Rightarrow C'_6 < p_c \end{array}$$

sendo selecionados para o cruzamento, os cromossomos $C_2$ e $C'_4$; $C'_5$ e $C'_6$.

Agora, é só gerar um número randômico para determinar $k$, a posição de cruzamento usando a Equação (3.7). Seja rand = 0,7; segue-se que:

$$\begin{array}{rcl} k & = & 1 + 0,7[(16 - 1) - 1] = 1 + 0,7(15 - 1) \\ k & = & 1 + 0,7.14 \Rightarrow k = 10,8. \end{array}$$

Como $k$ é um número inteiro, então $k = 11$. Daí,

$C'_2 - 1000010100100111$
$C'_4 - 0111110010101100.$

Trocando os caracteres, tem-se:

$C''_2 - 1000010100101100$
$C''_4 - 0111110010100111$
e

$C'_5 - 0111110010101100$
$C'_6 - 1100010100101001.$

Trocando os caracteres, tem-se:

$C_5''$ – 0111110010101001
$C_6''$ – 1100010100101100.

Assim, após a aplicação do operador cruzamento, a população é dada por:

$C_1'$ – 1001000100000001; $g(x, y) = 11, 01$.
$C_2''$ – 1000010100101100; $g(x, y) = 16, 72$.
$C_3'$ – 1001000100000001; $g(x, y) = 11, 01$.
$C_4''$ – 0111110010100111; $g(x, y) = 11, 02$.
$C_5''$ – 0111110010101001; $g(x, y) = 10, 67$.
$C_6''$ – 1100010100101100; $g(x, y) = 3, 10$.

## 3.3 Mutação

A mutação é uma modificação aleatória do valor de um alelo da cadeia. Caso o alelo escolhido seja zero passa a ser um e vice-versa, conforme esquematizado na Figura 3.4.

Segundo Haupt [7], uma técnica de se fazer a mutação é gerar pares *(a, b)* randômicos onde *a* representa a linha e *b* representa a coluna da mudança do bit. Nesta forma de aplicar o operador mutação exclui-se da seleção o melhor cromossomo. No exemplo em estudo, sejam os pares (1, 10) e (5, 3), logo tem-se (o cromossomo $C_2$' não será objeto de mutação por apresentar o maior valor para a função objetiva):

$$(1, 10) \Rightarrow C_1' \text{ e posição } 10$$
$$1001000100000001 \Rightarrow 1001000101000001$$
$$(5, 3) \Rightarrow C_5'' \text{ e posição } 3$$
$$0111110010101001 \Rightarrow 0101110010101001$$

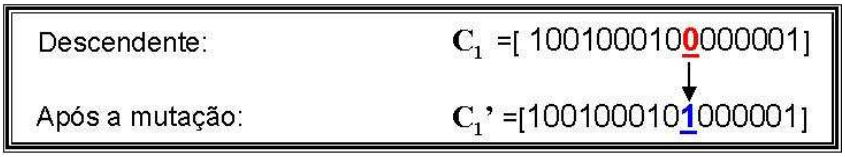



Figura 3.4: Representação do Operador Mutação

Neste curso, será utilizada outra metodologia onde se seleciona randomicamente uma posição em um cromossomo, obedecendo a probabilidade de mutação $p_m$, e muda o valor deste bit.

| Posição | Cromossomo | Probabilidade ($p_m$) |
|---|---|---|
| 13 | $C_1$' | 0,009 |
| 39 | $C_3$' | 0,0025 |
| 83 | $C_6$" | 0,0004 |

Tabela 3.2: Seleção da posição para aplicação do operador mutação

O processo de mutação é controlado por um parâmetro fixo $p_m$, probabilidade de mutação, que é geralmente recomendado como de 1%. Este operador tem um papel importante e necessário, porque a reprodução e o cruzamento podem perder material genético potencialmente útil. O operador de mutação protege os algorítmos genéticos contra perdas irreparáveis. Tomada isoladamente, a mutação se constituiria na exploração aleatória do espaço das cadeias. Utilizada com cuidado, juntamente com os outros dois operadores, protege-se o procedimento da perda prematura de informações importantes [2].

Aplicando o operador mutação ao Exemplo (3.2) em estudo, torna-se necessário gerar 96 ($m$ x $N$) números randômicos r entre [0,1]. Se r for menor que a probabilidade $p_m$ = 0,01 será feita a mutação no bit correspondente. Considere que foram gerados 96 números r entre 0 e 1 e que três tiveram probabilidades menores que $p_m$. Foram os seguintes números r:

$$r_{13} = 0,009 < p_m = 0,01$$
$$r_{39} = 0,0025 < p_m = 0,01$$
$$r_{83} = 0,0004 < p_m = 0,01$$

Considerando a população atual,

$C'_1 - 1001000100000001$
$C''_2 - 1000010100101100$
$C'_3 - 1001000100000001$
$C''_4 - 0111110010100111$
$C''_5 - 0111110010101001$
$C''_6 - 1100010100101100$

torna-se possível selecionar a posição onde deve-se ocorrer a mutação, conforme Tabela 3.2.

Submetendo os bits 13, 39 e 83 ao processo de mutação têm-se:

$C'_1 - 1001000100001001$
$C''_2 - 1000010100101100$
$C'_3 - 1001001100000001$
$C''_4 - 0111110010100111$
$C''_5 - 0111110010101001$

| Cromossomos | $x$ | $y$ | $g(x,\ y)$ |
|---|---|---|---|
| $C_1$'– 1001000100001001 | 9,14 | 8,04 | 11,52 |
| $C_2$" –1000010100101100 | 9,04 | 8,35 | 16,72 |
| $C_3$' – 1001001100000001 | 9,15 | 8,00 | 10,68 |
| $C_4$"– 0111110010100111 | 8,97 | 9,31 | 11,06 |
| $C_5$"– 0111110010101001 | 8,97 | 9,33 | 10,63 |
| $C_6$"– 1110010100101100 | 9,80 | 8,35 | -2,09 |
| $\Sigma g(x,y)$ | | | 58,52 |

Tabela 3.3: Cromossomos da população após a $1^a$ geração

$C_6'' - 1110010100101100$.

Após a aplicação dos três operadores, tem-se encerrado o ciclo da $1^a$ geração. Assim, torna-se interessante observar os valores das funções de adaptação para se ter uma idéia de como está ocorrendo a evolução dos cromossomos da população inicial. Estes dados podem ser acompanhados na Tabela 3.3.

Observando as Tabelas 3.1 e 3.3, nota-se que a população inicial melhorou no sentido de caminhar na direção da maximização da função objetiva, após aplicar os três operadores. Observa-se que o valor de $\sum$ g(x,y) passou de 36,92 para 58,52. Nesta primeira iteração, o ponto ótimo obtido corresponde a: x= 9,04 y= 8,35 e f(x,y)= - 16,72. Obviamente, executando outras iterações espera-se uma adaptação muito melhor da população.

## 3.4 Exemplo Ilustrativo

Neste caso, o problema de obtenção das raízes de equações não-lineares será tratado como um problema equivalente, no qual procura-se o máximo de uma função objetivo. Ou seja, a solução de $f(x) = 0$ será obtida através da função objetivo (ou função de mérito) escrita como:

$$F_m = max - |f(x)|. \tag{3.8}$$

A título de ilustração, considere a equação

$$e^x(x-1)^3 = 0 \tag{3.9}$$

cujo zero pode ser calculado usando a Equação 3.8, ou seja,

$$F_m = -|e^x(x-1)^3|. \tag{3.10}$$

Os respectivos gráficos estão representados nas Figuras 3.5 e 3.6. Pode-se observar que os valores dessa função de mérito estarão sempre situados no intervalo $(-\infty, 0]$.

Para a utilização dos algorítmos genéticos adotou-se a codificação binária, cujo tamanho do cromossomo depende da precisão requerida, para representar os valores reais da variável $x$.

Para o exemplo considerado, $f(x) = e^x\ (x - 1)^3$, seja o domínio da variável $x$ dado pelo intervalo [-1,2], de amplitude 3. Seja a precisão adotada $10^{-3}$. Assim, o intervalo [-1,2] deve ser dividido em, pelo menos, $3 \times 10^3$ subintervalos iguais. Isto significa que é necessário um vetor binário (cromossomo) com 12 bits, pois

$$2048 = 2^{11} < 3000 < 2^{12} = 4096.$$

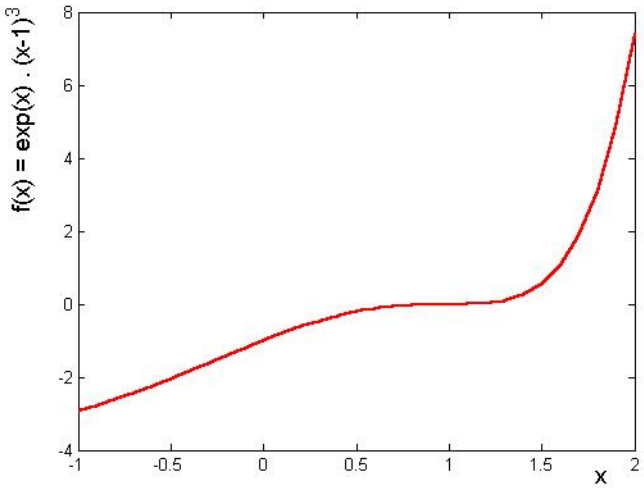



Figura 3.5: Representação Gráfica de $f(x) = e^x\ (x\text{-}1)^3$

Conforme apresentado na secção anterior os cromossomos são gerados aleatoriamente e para o cálculo da função objetivo seguem-se as seguintes etapas:

1. Conversão da cadeia binária < $b_{11}$ $b_{10}$... $b_0$ > da base 2 para a base 10.

$$(\langle b_{11}b_{10}\cdots b_0\rangle)_2 = \sum_{i=0}^{11} b_i \times 2^i = \overline{x};$$

2. Determinação do número real $x$ correspondente:

$$x\ =\ -1,0\ +\ \overline{x}\frac{(2-(-1))}{2^{11}-1} \Rightarrow x\ =\ -1,0\ +\ \overline{x}\frac{3}{2^{11}-1},$$

onde -1,0 é o limite inferior do domínio e 3 é a amplitude do mesmo.

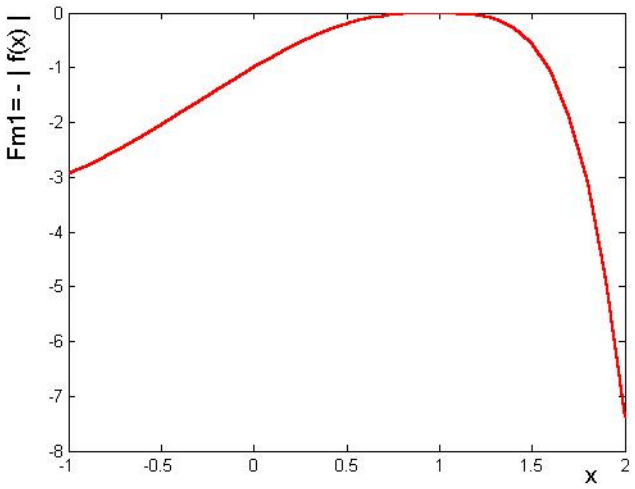



Figura 3.6: Representação Gráfica de $F_m$ = - | $e^x$ (x-1) $^3$ |

Por exemplo, o cromossomo (010100011100) representa o número 0,91; uma vez que

$$\overline{x} = (010100011100)_2 = 1308$$

e

$$x = -1,0 + 1308\frac{3}{2047} = 0,91 \text{ e } f(0,91) = e^{0,91}(0,91-1)^3 = -0,0018111.$$

Obviamente, os cromossomos (000000000000) e (011111111111) representam os extremos do domínio $-1,0$ e $2,0$; respectivamente.

Utilizando o código computacional GAOT, toolbox do Matlab®0, obteve-se o valor máximo do funcional $Fm$, que corresponde à raiz da equação $f(x) = 0$, dado por:

$$\xi = 0,99999999965708.$$

## 3.5 Exercícios

I) Obter a raiz, para as funções e intervalos apresentados na Tab. 3.4, aplicando algorítmos genéticos. (Utilizar o código computacional GAOT, toolbox do Matlab®.)

| Exemplo | $f(x)$ | Intervalo |
|---|---|---|
| a | (x-1)e$^{--nx}$ + x$^{nn}$; n=15 | [0;1] |
| b | (cos(x) – x.e$^{--x}$)^3 | [0;3] |
| c | (x-10) x +23 – x$^{00.4}$ | [0;7] |
| d | x$^{22}$ ( $\frac{1}{3}$ x$^{22}$ + $\sqrt{2}$ sen(x)) - $\frac{\sqrt{3}}{19}$ | [-1; 2] |
| e | 2 x + ln(x) | [0.1;1] |
| f | x$^{33}$ +7 - e$^{xx}$ | [0;5] |
| g | e$^{11-x}$ (x-1)10 -1 | [0;2] |
| h | x$^{22}$ – ln(x) -2 | [1;3] |

Tabela 3.4: Equações para obtenção das raízes

II) Utilizando o código computacional GAOT, toolbox do Matlab®, obter o mínimo das seguintes funções:

Adotar $X_i \in [-10, 10]$.

(a)
$$F(X) = 10\,X_1^4 - 20\,X_1^2\,X_2 + 10\,X_2^2 + X_1^2 - 2\,X_1 + 5;$$

(b)
$$F(X) = 100\,\left(X_2 - X_1^2\right)^2 + (1 - X_1)^2;$$

(c)
$$F(X) = X_1^4 + X_2^4 + 32\,X_1 - 4\,X_2 + 52;$$

(d)
$$F(X) = (X_1 + 10\,X_2)^2 + 5\,(X_3 - X_4)^2 + (X_2 - 2X_3)^4 + 10\,(X_1 - X_4)^4;$$

(e)
$$F(X) = X_1^2 + 2\,X_1\,X_2 + 2X_2^2 + X_3^2 - X_2\,X_3 + X_1 + 3\,X_2 - X_3;$$

(f)
$$F(X) = (X_1 + 2\,X_2 - 7)^2 + (2\,X_1 + X_2 - 5)^2.$$

III) Sob a ação de uma força $F$, o sistema move de $A$ até a posição de equilíbrio $B$. Obtenha esta posição de equilíbrio, que corresponde à mínima energia potencial:

$$\min \quad Ep = W\,y - F\,x,$$

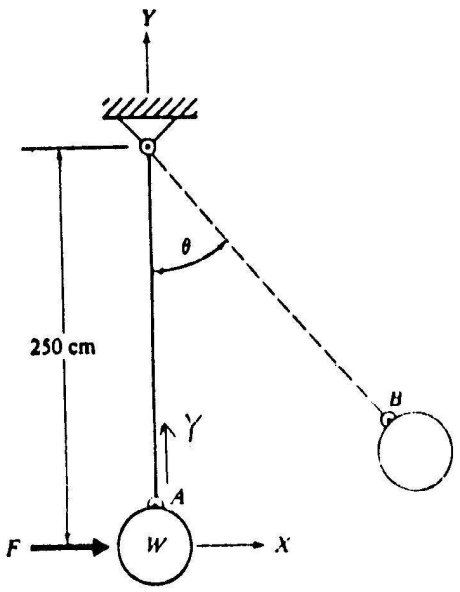
onde

$$x = L\ sin\theta$$
$$y\ =\ L\,(1-\cos\theta) \quad .$$

Dados:

$$W = 500 \text{ N};$$
$$F = 100 \text{ N};$$
$$\text{L} = 2{,}5 \text{ m}.$$

Restrições laterais: $\theta \in [0\ ,\ \pi/2]$.




IV) Determinar a posição de equilíbrio estático de um sistema constituído de 2 molas ($\text{K}_1$ e $\text{K}_2$) solicitado por duas forças constantes ($\text{P}_1$ e $\text{P}_2$), de forma a minimizar sua energia potencial:

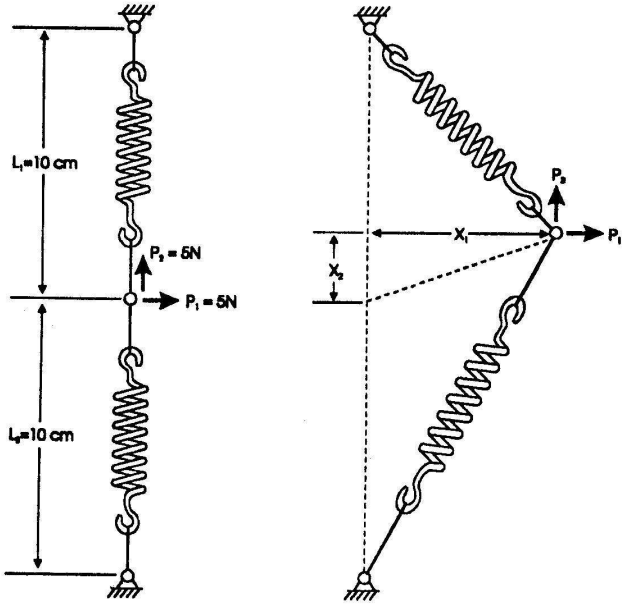
$$\begin{aligned}\min\ Ep \quad = \quad & 0{,}5\,K_1\left[\sqrt{X_1^2 + (l_1 - X_2)^2}\ -\ l_1\right]^2 \\ & +\ 0{,}5\,K_2\left[\sqrt{X_1^2 + (l_2 + X_2)^2}\ -\ l_2\right]^2 - P_1\,X_1 - P_2\,X_2.\end{aligned}$$

Dados:

$P_1 = P_2 = 5$ N;$K_1 = 8$ N/cm; $K_2 = 1$ N/cm;

$l_1 = l_2 = 10$ cm .

Restrições laterais: $X_i \in [0\ ,\ 10]$.



# Capítulo 4

# Simulated Annealing

Simulated Annealing pertence à mesma classe dos métodos das Redes Neurais e Algoritmos Genéticos, no sentido que simulam métodos Naturais. O algoritmo Simulated Annealing permite uma útil conecção entre a mecânica estatística (comportamento de um sistema de vários graus de liberdade em equilíbrio térmico a uma temperatura finita) e a otimização combinatória (encontrar um mínimo de uma dada função dependendo de vários parâmetros). Alguns resultados publicados utilizando Simulated Annealing, em particular devido aos esquemas de resfriamento rápido [11], têm merecido a atenção de físicos e engenheiros.

Este método de otimização faz uma analogia com o processo de recozimento (annealing) da metalurgia. Sabe-se da Metalurgia que, se o metal é resfriado em condições apropriadas, o cristal simples pode ser obtido [9]. No recozimento o metal é aquecido a altas temperaturas, causando um choque violento nos átomos. Se o metal for resfriado de forma brusca, a microestrutura tende a um estado randomicamente instável, porém, se o metal é resfriado de forma suficientemente lenta, o sistema procurará um ponto de equilíbrio caracterizado por uma microestrutura ordenada e estável.

As variáveis de projeto são perturbadas randomicamente e armazena-se o melhor valor da função objetivo a cada perturbação. A temperatura é então reduzida (annealing) e novas tentativas executadas. Este procedimento continua até escaparmos de um mínimo local. Ao final do processo é possível que se obtenha um mínimo global.

Metropolis et al [12], introduziram um método numérico simples que representa o estato equilíbrio de um conjunto de átomos a uma dada temperatura. A analogia com o processo de otimização pode ser feita analisando a Figura 4.1.

Seja $\Delta$E a energia de um sistema de átomos a uma temperatura T. Em cada passo do algoritmo, é dado um deslocamento aleatório a um átomo, o que implica uma nova energia do sistema $\Delta$E. Se esta nova energia $\Delta$E é menor ou igual a zero ($\Delta E \leq 0$), o deslocamento é aceito, caso contrário ($\Delta E > 0$), a probabilidade da configuração ser aceita será dada pela equação

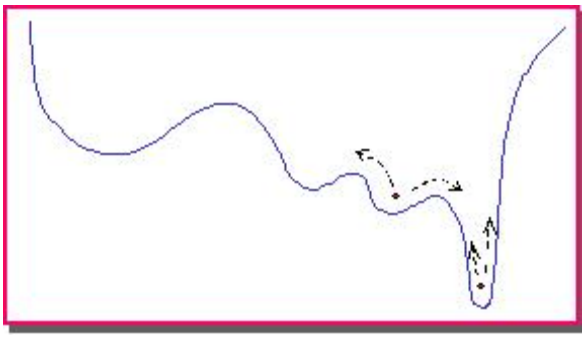


Figura 4.1: Analogia entre o processo de otimização e o recozimento simulado

$$P(\Delta E) \;=\; e^{\left(\frac{-\Delta E}{K_b T}\right)} \tag{4.1}$$

onde $K_b$ é a constante de Boltzmann.

Um número randômico $r$, uniformemente distribuído, deve ser gerado no intervalo [0, 1]. Se $r \leq$ P($\Delta$E) a nova configuração é aceita. Se $r >$ P($\Delta$E) a configuração anterior é utilizada para iniciar um novo passo.

A escolha da função de probabilidade P($\Delta$E), conforme acima descrito, se deve ao fato de que o sistema evolui segundo uma distribuição de Boltzman.

Os parâmetros do algoritmo são: a função custo, que representa a energia do sistema; as variáveis de projeto, que descrevem sua configuração e a temperatura, que é um parâmetro de controle [3].

Se T tiver magnitude muito superior ao desvio padrão da função no intervalo, quase todos os pontos são aceitos. Ao passo que se T for igual a zero, o método se torna uma busca aleatória do mínimo. Assim, adota-se: $T_i$ como o valor do desvio padrão da função objetivo no intervalo estudado e $T_f$ com a ordem de grandeza desejada para o precisão do ponto ótimo.

## 4.1 Implementação do Método

Na otimização via Simulated Annealing considera-se:

- A perturbação randômica das variáveis de projeto;

- A manutenção do melhor valor da função objetivo.

Após algumas tentativas o melhor valor da função é chamado de centro, em torno do qual ocorreram as perturbações na próxima temperatura.

A temperatura é então reduzida (annealing) e novas tentativas executadas. Este procedimento continua até escaparmos de um mínimo local. Ao final do processo é possível que se obtenha um mínimo global.

O algoritmo do método, desenvolvido por Saramago et al [16], pode ser acompanhado através da Figura **??**. Os parâmetros de controle para iniciar o procedimento são:

- A função objetivo $F$(X);
- As variáveis de projeto iniciais (X);
- O número de variáveis de projeto(nvars);
- A temperatura inicial $T_i$;
- A temperatura final $T_f$;
- O número de iterações para cada temperatura (niters);
- O numero de temperaturas (ntemps);
- O número de avaliações da função objetivo (maxcalls);
- O critério de parada.

A configuração inicial das variáveis de decisão é adotada como centro. O valor inicial da função objetivo adotado como o melhor valor (best fucntion value).

No próximo passo, o número randômico $r$ é gerado e as variáveis são modificadas ("shake"):

$$r = r_1 + r_2 - r_3 - r_4 \tag{4.2}$$

e

$$\begin{aligned} T &= T\frac{\sqrt{12}}{12} \\ x &= \text{center } T^* r. \end{aligned} \tag{4.3}$$

Na Equação 4.2 observa-se que quatro valores randômicos são gerados, de forma que a variável $r$ seja adotada como média zero [11]. Assim, uma nova configuração é obtida pela Equação 4.3 e um novo valor da função objetivo pode ser calculado.

O esquema inicia-se com uma temperatura alta, que é reduzida discretamente (usando o fator $r_t$ , $0 < r_t < 1$) até que o sistema "resfrie", conforme as expressões abaixo:

$$\begin{aligned} r_t &= e^{\frac{\ln\left(\frac{T_f}{T_i}\right)}{n_{temp}-1}} \\ T &= T^* r_t. \end{aligned} \tag{4.4}$$

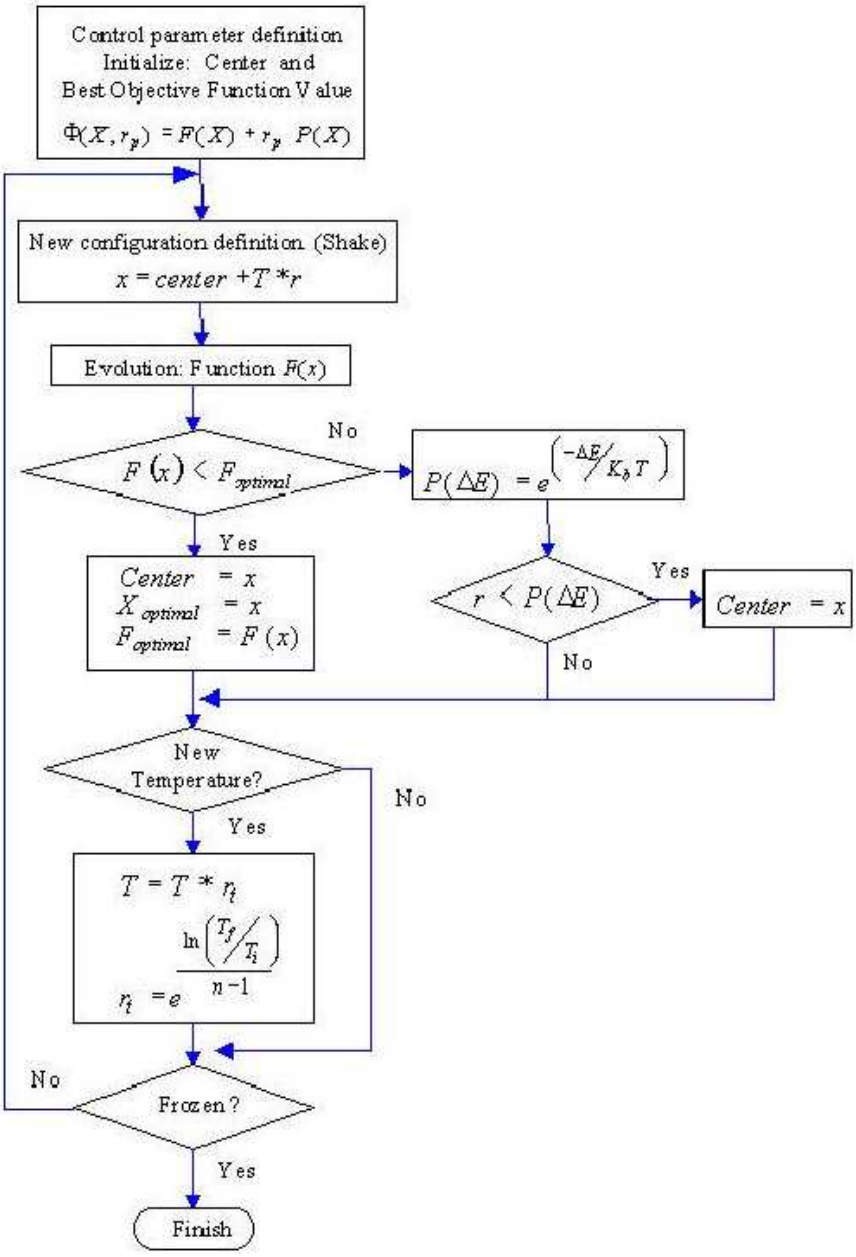



Figura 4.2: Algoritmo do Simulated Annealing

A temperatura é então reduzia e novas tentativas executadas. Este procedimento continua até escaparmos de um mínimo local. Ao final do processo é possível que se obtenha um mínimo global.

Para otimizar problemas restritos, o algoritmo deve ser modificado, conforme apresentado na próxima seção. Utilizando o método das configurações viáveis, para cada nova configuração verifica-se se as restrições são satisfeitas. Encontra-se em estudo a utilização de funções de penalidade para se trabalhar com problemas de otimização restritos.

## 4.2 Exemplo Ilustrativo

Com finalidade de comparar os resultados, vamos considerar o problema de minimização definido na seção anterior:

$$g(x, y) = xsen(4x) + 1,1ysen(2y) \tag{4.5}$$

no intervalo, $8 < x < 10$, $8 < y < 10$, que define a região viável do problema.

Considere os seguintes os dados de entrada para o programa:

nvars= 2
$\mathrm{X}^L$ = [ 8 8]
$\mathrm{X}^U$ = [ 10 10]
ntemps=10
niters = 300
$\mathrm{T}_i$ = 0.5
$\mathrm{T}_f$ = 0.01
maxcalls=10000.

Utilizando o código computacional AS desenvolvido por Saramago et al [16], obteve-se o seguinte ponto ótimo:

$$x = 9,0388,\ y = 8,6682 \ \text{ e } \ g(x, y) = -18,5547.$$

## 4.3 Exercícios

I) Obter a raiz, para as funções e intervalos apresentados na Tabela 3.4, aplicando Simulated Annealing. (Utilizar o código computacional SA)

II) Obter o mínimo das seguintes funções:
Adotar $X_i \in [-10, 10]$.

(a)
$$F(X) = 10\,X_1^4 - 20\,X_1^2\,X_2 + 10\,X_2^2 + X_1^2 - 2\,X_1 + 5;$$

(b)
$$F(X) = 100\,\left(X_2 - X_1^2\right)^2 + (1 - X_1)^2;$$

(c)
$$F(X) = X_1^4 + X_2^4 + 32\,X_1 - 4\,X_2 + 52;$$

(d)
$$\begin{aligned} F(X) \quad = \quad & (X_1 + 10\,X_2)^2 + 5\,(X_3 - X_4)^2 \\ & + (X_2 - 2X_3)^4 + 10\,(X_1 - X_4)^4; \end{aligned}$$

(e)
$$\begin{aligned} F(X) \quad = \quad & X_1^2 + 2\,X_1\,X_2 + 2X_2^2 + X_3^2 \\ & - X_2\,X_3 + X_1 + 3\,X_2 - X_3; \end{aligned}$$

(f)
$$F(X) = (X_1 + 2\,X_2 - 7)^2 + (2\,X_1 + X_2 - 5)^2.$$

III) Sob a ação de uma força $F$, o sistema move de $A$ até a posição de equilíbrio $B$. Obtenha esta posição de equilíbrio, que corresponde à mínima energia potencial:

$$\min \; Ep = W\,y - F\,x$$

onde

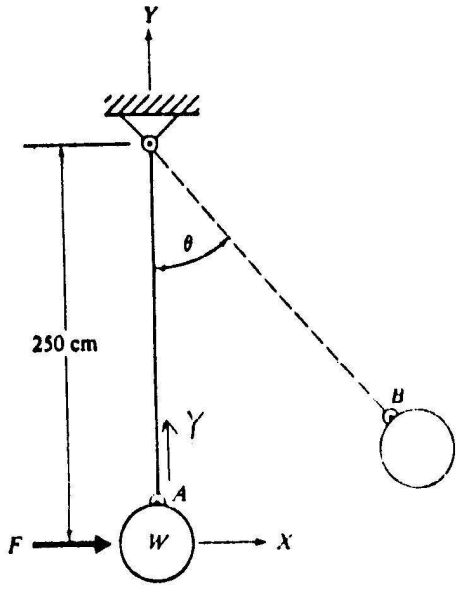
$$\begin{aligned} x &= L\;sin\theta \\ y &= L\,(1 - \cos\theta) \quad . \end{aligned}$$

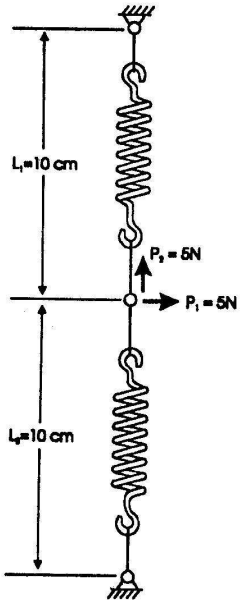
Dados: $W = 500$ N; $F = 100$ N; L = 2,5 m, $\theta \in [0\ ,\ \pi/2]$.

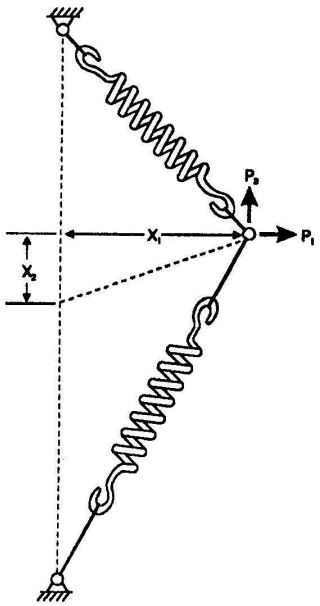
IV) Determinar a posição de equilíbrio estático de um sistema constituído de 2 molas ($K_1$ e $K_2$) solicitado por duas forças constantes ($P_1$ e $P_2$), de forma a minimizar sua energia potencial:

$$\begin{aligned} \min \; Ep \quad = \quad & 0,5\,K_1\left[\sqrt{X_1^2 + (l_1 - X_2)^2} - l_1\right]^2 \\ & + 0,5\,K_2\left[\sqrt{X_1^2 + (l_2 + X_2)^2} - l_2\right]^2 - P_1\,X_1 - P_2\,X_2. \end{aligned}$$

Dados: $P_1 = P_2 = 5$ N; $K_1 = 8$ N/cm ; $= 1$ N/cm ; $l_1 = l_2 = 10$ cm.
Restrições laterais: $X_i \in [0\ ,\ 10]$.


Y
θ
250 cm
B
Y
A
F
W
X


L₁=10 cm
P₂ = 5N
P₁ = 5N
L₂=10 cm


P₂
X₁
P₁
X₂

# Bibliografia


[1] M.S. Bazaraa, H.D. Shetali e C.M. Shetty, "Nonlinear Programming: Theory and Algorithms", John Wiley & Sons, second edition, New York, 1993.

[2] C.G. Braga, "O Uso de Algoritmos Genéticos para Aplicação em Problemas de Otimização de Sistemas Mecânicos", Dissertação de Mestrado, Universidade Federal de Uberlândia, Uberlândia, MG, 1998.

[3] A. Corana, M. Marchesi, C. Martini e S. Ridella, Minimizing multimodal functions of continuous variables with the simulated annealing algorithm, *ACM Transactions on Mathematical Software*, **13,** n. 3, (1987),262-280.

[4] E.D. Faria e S.F.P. Saramago, Constraint optimization problems using simulated annealing, *Ciência & Engenharia*, Brasil, **10**, n. 1, (2001) , 69-75.

[5] D.E. Goldberg, "Genetic Algorithms in Search, Optimization and Machine Learning", New York, Addison- Wesley, 1989.

[6] A. Grace, "Optimization Toolbox- For Use with Matlab", The Math Works Inc., Natick, 1992.

[7] R.L. Haupt e S.E. Haupt, "Pratical Genetic Algorithm", John Wiley G. Sons Inc; New York, pp.25-48, 1998.

[8] J.H. Holland, "Adaptation in Natural and Artificial Systems", Ann Arbor, The University of Michigan Press, United States of America ,1975.

[9] S. Kirkpatrick, C.D. Gelatt Jr. e M.P. Vecchi, Optimization by simulated annealing, *Science*, **220**, n. 4598, (1983), 671-680.

[10] L.C.G. Lopes, E.A.A. Santos e V. Karlec, Determinação de raízes de equações não lineares utilizando algoritmos genéticos, Computacional Methods in Engineering (Cd room), pp. 141.1-141.11, XX CILAMCE, São Paulo,1999.

[11] T. Masters, "Practical Neural Network Recipes In C++", Academic Press, pp. 117-134, 1993.

[12] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth e A.H. Teller, Equations of state calculations by fast computing machines, *Journal of Chemical Physics*, **21**, (1953),1087-1092.

[13] Z. Michalewicz, Z., "Evolutionary Computation for NonLinear Programming Problems", ftp://ftp.uncc.eduftp.uncc.edu, directory coe/evol, 1998,.

[14] G.V. Reklaitis, A. Ravindran, e K.M. Ragsdell, "Engineering Optimization - Methods and Applications", John Wiley and Sons, United States of America, 1983.

[15] S.F.P. Saramago e V. Steffen Jr., Aspectos fundamentais ao usar técnicas de otimização no projeto de sistemas mecânicos, Proc. IV Congr. de Eng. Mecânica- NNE, V.1, pp.421-426, Recife, Brasil, Junho, 1996.

[16] S.F.P. Saramago, S.F.P., E.G. Assis e Steffen Jr., V., 1999, Simulated anneling: some applications in mechanical systems optimization, Computacional Methods in Engineering (Cd room), XX CILAMCE, São Paulo,1999.

[17] H.P. Schwefel e L Taylor, "Evolution and Optimum Seeking", John Wiley & Sons Inc, United States of America, pp. 87-88, 1994.

[18] G. Vanderplaats, "Numerical Optimization Techniques for Engineering Design", McGraw-Hill, United States of America, 70-97, 1984.