Metaheurísticas Populacionais: Algoritmos Genéticos e Enxame de Partículas

Guilherme de Alencar Barreto

gbarreto@ufc.br

Departamento de Engenharia de Teleinformática (DETI)
Engenharias de Computação, Telecomunicações e Teleinformática
Universidade Federal do Ceará – UFC
www.researchgate.net/profile/Guilherme_Barreto2/

Conteúdo dos Slides

1. População de Indivíduos/Partículas: Visão Matricial
2. Enxame de Partículas: Atualização da Posição/Velocidade
3. Algoritmos Genéticos: Seleção/Recombinação/Mutação
4. Exemplos Computacionais

Representação de um indivíduo/cromossomo/partícula

- Nas metaheurísticas populacionais, o i-ésimo indivíduo, cromossomo (GA) ou partícula (PSO) é representado com um vetor-linha de dimensão p: $\mathbf{x}_i \in \mathbb{R}^p$.

$$\mathbf{x}_i = [x_{i1} \;\; x_{i,2} \;\; \cdots \;\; x_{ij} \;\; \cdots \;\; x_{ip}], \tag{1}$$

em que o elemento x_{ij} é chamado de gene no jargão de algoritmos genéticos.

Representação de um indivíduo/cromossomo/partícula

- Assim, uma população de N indivíduos na iteração ou geração t é representada como uma matriz $\mathbf{P}(t)$, de dimensão $N \times p$.

$$\mathbf{P}(t) = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_i \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,j} & \cdots & x_{1,p} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{i,1} & x_{i,2} & \cdots & x_{i,j} & \cdots & x_{i,p} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,j} & \cdots & x_{N,p} \end{bmatrix} \quad (2)$$

- Cada linha dessa matriz corresponde a uma solução-candidata para o problema de interesse.
- Portanto, a cada geração/iteração, uma metaheurística populacional avalia N soluções candidatas.

Formalização Matemática do Problema

- A cada iteração t uma nova população é formada por meio de operações matemáticas sobre as linhas da matriz $\mathbf{P}(t)$.
- Pode-se entender a população a ser avaliada na próxima iteração, $\mathbf{P}(t+1)$, como resultante da aplicação sucessiva de transformações não-lineares às linhas da população $\mathbf{P}(t)$.
- Formalmente, temos a seguinte representação:

$$T\left(\mathbf{P}(t)\right) \rightarrow \mathbf{P}(t+1). \tag{3}$$

Formalização Matemática do Problema

- Em algoritmos genéticos, a transformação (ou operador) T é resultado da aplicação sequencial de três outros operadores, a saber:
  1. Operador **seleção** (S).
  2. Operador **recombinação** (R).
  3. Operador **mutação** (M).
- Assim, que podemos escrever a transformação T como

$$M\left(R\left(S\left(\mathbf{P}(t)\right)\right)\right) \rightarrow \mathbf{P}(t+1). \tag{4}$$

Formalização Matemática do Problema

- Em otimização por enxame de partículas, a transformação (ou operador) T é resultado da aplicação sequencial de dois outros operadores, a saber:
  1. Operador **atualização de velocidade** (V).
  2. Operador **atualização de posição** (X).
- Assim, podemos escrever a transformação T como

$$X\left(V\left(\mathbf{P}(t)\right)\right) \rightarrow \mathbf{P}(t+1). \tag{5}$$

Parte I

Otimização por Enxame de Partículas

Introdução

Metaheurística para otimização global inspirada no comportamento social de animais que se organizam e se movimentam em grupos, tais como cardume de peixes e revoada de pássaros.

(a) Revoada

(b) Cardume

Introdução

- Proposta por Kennedy e Eberhart em 1995, no seguinte artigo:

  J. Kennedy & R. C. Eberhart (1995). "Particle swarm optimization". In: Proceedings of the IEEE International Conference on Neural Networks (ICNN'1995), pp. 1942–1948. doi:10.1109/ICNN.1995.488968.

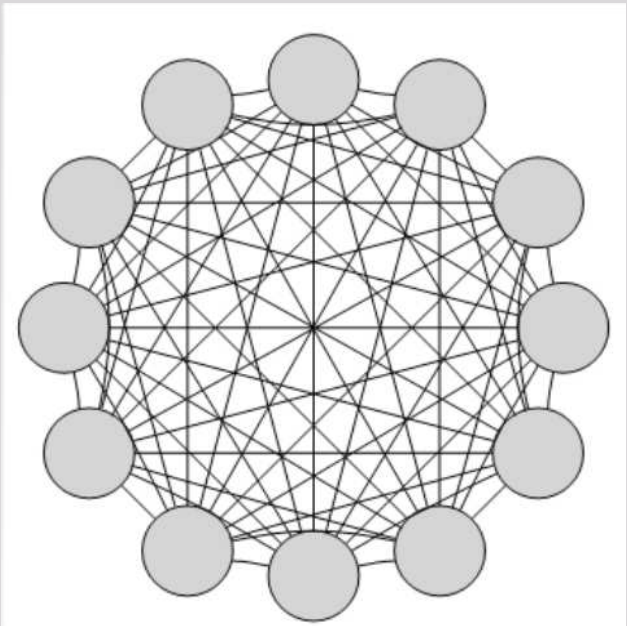
- É um metaheurística populacional, estocástica, livre de gradiente, e de inspiração biológica.
- Soluções são representadas por entidades matemáticas abstratas, denominadas de partículas, com posição e velocidades específicas e que formam um enxame em um espaço de possíveis soluções.

Introdução

- O poder exploratório do algoritmo PSO provém da troca ou compartilhamento de informação entre as partículas e da percepção que cada partícula tem do seu desempenho.
- A informação compartilhada entre partículas confere um elemento *social* ao algoritmo e permite a exploração do espaço de buscas por distâncias maiores.
- A informação de cada partícula sobre o seu próprio desempenho confere um elemento de auto-percepção (i.e. cognitivo) ao algoritmo e, portanto, de exploração local do espaço de busca.
- O embate de forças exploratórias de naturezas distintas, global e local, é chamado de dilema **Exploração** $\times$ **Explotação**.

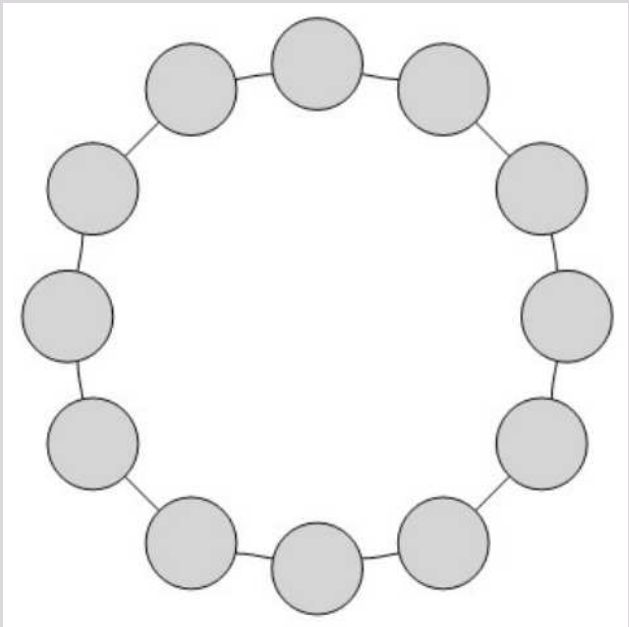
Topologia Global: Cada partícula conectada a todas as outras

- Isto implica que a informação sobre o desempenho de cada partícula está disponível a todas as outras.
- Confere um caráter instantâneo à propagação de informação e, por isso, não é biologicamente plausível.



Topologia Local: Cada partícula conectada à sua vizinhança apenas

- Implica que a informações sobre cada partícula não estão disponíveis de imediato a todas as outras.
- Confere um caráter não-instantâneo à propagação de informação, sendo mais biologicamente plausível.



- No algoritmo PSO, cada partícula $\mathbf{x}_i(t) = [x_{i,1}(t) \quad \cdots \quad x_{i,j}(t) \quad \cdots \quad x_{i,p}(t)]$ deve manter um registro de sua melhor posição $\mathbf{x}_i^{best} = [x_{i,1}^{best} \quad \cdots \quad x_{i,j}^{best} \quad \cdots \quad x_{i,p}^{best}]$ até a iteração atual t.
- Entende-se por *busca local* aquela realizada em torno de $\mathbf{x}_i^{best}$.
- Logo, uma variação de posição na direção de $\mathbf{x}_i^{best}$ produz uma componente de velocidade na iteração t dada por

$$\mathbf{v}_i^{cog}(t) = \mathbf{x}_i^{best} - \mathbf{x}_i(t), \tag{6}$$

em que o sobrescrito cog remete à idéia de um elemento 'cognitivo' influenciando a velocidade da i-ésima partícula.

- Pode-se entender $\mathbf{v}_i^{cog}(t)$ como uma estimativa rude do gradiente na direção de $\mathbf{x}_i^{best}$.

Formalismo Matemático

- No algoritmo PSO global, também é necessário manter um registro da melhor posição $\mathbf{x}_g^{best}$ dentro do enxame até a iteração atual t.
- Assim, *busca global* é aquela realizada na direção de $\mathbf{x}_g^{best}$.
- Logo, uma variação de posição na direção de $\mathbf{x}_g^{best}$ produz uma componente de velocidade na iteração t dada por

$$\mathbf{v}_i^{soc}(t) = \mathbf{x}_g^{best} - \mathbf{x}_i(t), \tag{7}$$

em que o sobrescrito soc remete à idéia de um elemento 'social' influenciando a velocidade da i-ésima partícula.

- Pode-se entender $\mathbf{v}_i^{soc}(t)$ como uma estimativa rude do gradiente na direção de $\mathbf{x}_g^{best}$.

- As componentes de velocidade $\mathbf{v}_i^{cog}(t)$ e $\mathbf{v}_i^{soc}(t)$ são linearmente combinadas em uma equação recursiva de atualização da velocidade da i-ésima partícula:

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(t) \circ \mathbf{v}_i^{cog}(t) + c_2\mathbf{r}_2(t) \circ \mathbf{v}_i^{soc}(t), \quad (8)$$

$$= w\mathbf{v}_i(t) + c_1\mathbf{r}_1(t) \circ (\mathbf{x}_i^{best} - \mathbf{x}_i(t)) + c_2\mathbf{r}_2(t) \circ (\mathbf{x}_g^{best} - \mathbf{x}_i(t)), \quad (9)$$

em que $\circ$ denota o produto de Hardamad. As constantes $c_1 > 0$ e $c_2 > 0$ são chamados de coeficientes de aceleração (uma escolha comum: $c_1 = c_2 = 2,05$). A constante $w \in [0,4 - 0,9]$ é chamado de fator de inércia.

- Os vetores $\mathbf{r}_1(t)$ e $\mathbf{r}_2(t)$ têm dimensão $p \times 1$ e seus elementos são números aleatórios uniformes entre 0 e 1: $r_{1,j}(t) \sim U(0,1)$, $r_{2,j}(t) \sim U(0,1)$, $j = 1, \ldots, p$.

- Uma vez atualizada a velocidade da i-ésima, sua posição é atualizada por meio da seguinte equação recursiva:

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \alpha \mathbf{v}_i(t+1), \tag{10}$$

em que $0 \leq \alpha \leq 1$ é a taxa de aprendizagem cujo o papel é controlar o quanto a atualização na velocidade das partículas influenciará em suas posições. É comum fazer $\alpha = 1$.

- No Octave/Matlab, os vetores aleatórios $\mathbf{r}_1$ e $\mathbf{r}_2$ são facilmente geradas pelos seguintes comandos:
  - `>> r1=rand(`p`,1);`
  - `>> r2=rand(`p`,1);`

- Os vetores aleatórios $\mathbf{r}_1(t)$ e $\mathbf{r}_2(t)$ conferem estocasticidade ao movimento das partículas ao multiplicar os vetores $\mathbf{v}_i^{cog}$ e $\mathbf{v}_i^{soc}$.
- Considere, por exemplo, o 2o. termo à esquerda da igualdade, da Eq. (8):

$$c_1\mathbf{r}_1(t)\circ\mathbf{v}_i^{cog}(t) = \begin{bmatrix} c_1 \cdot r_{11}(t) \cdot v_{i,1}^{cog}(t) \\ c_1 \cdot r_{12}(t) \cdot v_{i,2}^{cog}(t) \\ \vdots \\ c_1 \cdot r_{1j}(t) \cdot v_{i,j}^{cog}(t) \\ \vdots \\ c_1 \cdot r_{1p}(t) \cdot v_{i,p}^{cog}(t) \end{bmatrix} = \begin{bmatrix} c_1 \cdot r_{11}(t) \cdot (x_{i,1}^{best} - x_{i,1}(t)) \\ c_1 \cdot r_{12}(t) \cdot (x_{i,2}^{best} - x_{i,2}(t)) \\ \vdots \\ c_1 \cdot r_{1j}(t) \cdot (x_{i,j}^{best} - x_{i,j}(t)) \\ \vdots \\ c_1 \cdot r_{1p}(t) \cdot (x_{i,p}^{best} - x_{i,p}(t)) \end{bmatrix} \quad (11)$$

- Sem os números aleatórios $r_{1j}(t) \sim U(0,1)$, $j = 1, \ldots, p$, a contribuição das componentes $v_{i,j}^{cog}(t) = x_{i,j}^{best} - x_{i,j}(t)$ ao vetor $\mathbf{v}_i^{cog}$ seria determinística. Ou seja, saberíamos exatamente para onde a partícula se moveria na próxima iteração.

- Com a introdução dos números aleatórios $r_{1j}(t)$, a contribuição das componentes $v_{i,j}^{cog}(t) = x_{i,j}^{best} - x_{i,j}(t)$ ao vetor $\mathbf{v}_i^{cog}$ é estocástica. Ou seja, não sabemos exatamente para onde a partícula se mover na próxima iteração, embora saibamos que o movimento ocorre na vizinhança de $\mathbf{x}_i^{best}$.

- O mesmo raciocínio se aplica à parte ao termo $c_2\mathbf{r}_2(t) \circ \mathbf{v}_i^{soc}(t)$ da equação de ajuste da velocidade.

- É possível ainda escrever a forma matricial das equações de ajuste da velocidade e posição das partículas do enxame.
- Assim, atualizamos uma matriz de velocidade por meio da seguinte expressão:

$$\mathbf{V}(t+1) = w\mathbf{V}(t) + c_1\mathbf{R}_1(t) \circ \mathbf{V}^{cog}(t) + c_2\mathbf{R}_2(t) \circ \mathbf{V}^{soc}(t), \quad (12)$$

em que agora usamos matrizes de números aleatórios $\mathbf{R}_1(t)$ e $\mathbf{R}_1(t)$. O operador $\circ$ denota o produto de Hadamard.

- A matriz de posições das partículas é atualizada por meio da seguinte equação recursiva:

$$\mathbf{X}(t+1) = \mathbf{X}(t) + \alpha\mathbf{V}(t+1), \quad (13)$$

em que α é a taxa de aprendizagem.

- A matriz $\mathbf{V}^{cog}(t) \in \mathbb{R}^{N \times p}$ da Eq. 12 é definida como

$$\mathbf{V}^{cog}(t) = \begin{bmatrix} \mathbf{v}_1^{cog}(t) \\ \vdots \\ \mathbf{v}_i^{cog}(t) \\ \vdots \\ \mathbf{v}_N^{cog}(t) \end{bmatrix}_{N \times p} = \begin{bmatrix} \mathbf{x}_1^{best} - \mathbf{x}_1(t) \\ \vdots \\ \mathbf{x}_i^{best} - \mathbf{x}_i(t) \\ \vdots \\ \mathbf{x}_N^{best} - \mathbf{x}_N(t) \end{bmatrix}_{N \times p} \quad (14)$$

$$= \begin{bmatrix} \mathbf{x}_1^{best} \\ \vdots \\ \mathbf{x}_i^{best} \\ \vdots \\ \mathbf{x}_N^{best} \end{bmatrix}_{N \times p} - \begin{bmatrix} \mathbf{x}_1(t) \\ \vdots \\ \mathbf{x}_i(t) \\ \vdots \\ \mathbf{x}_N(t) \end{bmatrix}_{N \times p} = \mathbf{X}^{best} - \mathbf{X}(t), \quad (15)$$

em que $\mathbf{X}^{best}$ é uma matriz $N \times p$ contendo as melhores posições até o momento para as N partículas do enxame.

- E a matriz e $\mathbf{V}^{soc}(t) \in \mathbb{R}^{N \times p}$ da Eq. 12 é dada por

$$\mathbf{V}^{soc}(t) = \begin{bmatrix} \mathbf{v}_1^{soc}(t) \\ \vdots \\ \mathbf{v}_i^{soc}(t) \\ \vdots \\ \mathbf{v}_N^{soc}(t) \end{bmatrix}_{N \times p} = \begin{bmatrix} \mathbf{x}_g^{best} - \mathbf{x}_1(t) \\ \vdots \\ \mathbf{x}_g^{best} - \mathbf{x}_i(t) \\ \vdots \\ \mathbf{x}_g^{best} - \mathbf{x}_N(t) \end{bmatrix}_{N \times p} \quad (16)$$

$$= \begin{bmatrix} \mathbf{x}_g^{best} \\ \vdots \\ \mathbf{x}_g^{best} \\ \vdots \\ \mathbf{x}_g^{best} \end{bmatrix}_{N \times p} - \begin{bmatrix} \mathbf{x}_1(t) \\ \vdots \\ \mathbf{x}_i(t) \\ \vdots \\ \mathbf{x}_N(t) \end{bmatrix}_{N \times p} = \mathbf{X}_g^{best} - \mathbf{X}(t), \quad (17)$$

em que $\mathbf{X}_g^{best} = \mathbf{1}_N \mathbf{x}_g^{best}$ é uma matriz $N \times p$ contendo o vetor $\mathbf{x}_g^{best}$ (melhor posição de todo o enxame até o momento) repetido N vezes, e $\mathbf{1}_N$ é um vetor de 1's de dimensão $N \times 1$.

- Uma das vantagens da notação matricial consiste em entender o algoritmo PSO global como um operador matemático sobre uma matriz; no caso, a matriz que define a população de partículas do enxame.
- Há também ganhos de velocidade consideráveis ao se implementar o algoritmo PSO global na forma matricial, principalmente em ambientes de programação como Octave e Matlab.
- No Octave/Matlab, tais matrizes são facilmente geradas pelos seguintes comandos:
  - `>> R1=rand(`N, p`));`
  - `>> R2=rand(`N, p`));`

Passo 1: Inicialização da População

1. Inicializar aleatoriamente posição e velocidade das partículas:

$$\mathbf{x}_i(0) \sim U(\mathbf{x}^l, \mathbf{x}^u), \quad i = 1, \ldots, N. \tag{18}$$

$$\mathbf{v}_i(0) \sim U(-|\mathbf{x}^u - \mathbf{x}^l|, |\mathbf{x}^u - \mathbf{x}^l|), \quad i = 1, \ldots, N. \tag{19}$$

   em que $\mathbf{x}^l$ e $\mathbf{x}^u$ são, respectivamente, os limites inferiores e superiores das variáveis do problema.

2. Fazer $\mathbf{x}_i^{best} = \mathbf{x}_i(0)$ e calcular $f(\mathbf{x}_i^{best})$, $i = 1, \ldots, N$.

3. Encontrar $\mathbf{x}_g^{best}$:

$$\mathbf{x}_g^{best} = \arg\min_{\forall i} \left\{ f(\mathbf{x}_i^{best}) \right\}. \tag{20}$$

4. Fazer $f_g^{best} = f(\mathbf{x}_g^{best})$.

Passo 2: Atualização da População

Até a convergência acontecer (ou atingir número máximo de iterações):

2.1 - Atualizar a velocidade das N partículas:

$$\mathbf{V}(t+1) = w\mathbf{V}(t) + c_1\mathbf{R}_1(t) \circ \mathbf{V}^{cog}(t) + c_2\mathbf{R}_2(t) \circ \mathbf{V}^{soc}(t), \qquad (21)$$

2.2 - Atualizar a posição das N partículas:

$$\mathbf{X}(t+1) = \mathbf{X}(t) + \alpha\mathbf{V}(t+1), \qquad (22)$$

Passo 3: Avaliação das Novas Posições ds Partículas

Para as N partículas do enxame:

3.1 - Avaliar a nova posição da i-ésima partícula: $f(\mathbf{x}_i(t+1))$.

3.2 - Atualizar sua melhor posição da partícula até o momento:

$$\textbf{SE } f(\mathbf{x}_i(t+1)) < f(\mathbf{x}_i^{best}),$$

$$\textbf{ENTÃO } \mathbf{x}_i^{best} = \mathbf{x}_i(t+1)$$

$$f(\mathbf{x}_i^{best}) = f(\mathbf{x}_i(t+1)).$$

3.3 - Buscar melhor posição dentro do enxame: $\mathbf{x}_g^{best} = \arg\min_{\forall i} \left\{ f(\mathbf{x}_i^{best}) \right\}$.

3.4 - Fazer $f_g^{best} = f(\mathbf{x}_g^{best})$.

3.5 - Se convergência = TRUE, então terminar execução. Caso contrário, fazer $t = t+1$ e ir para o Passo 2.

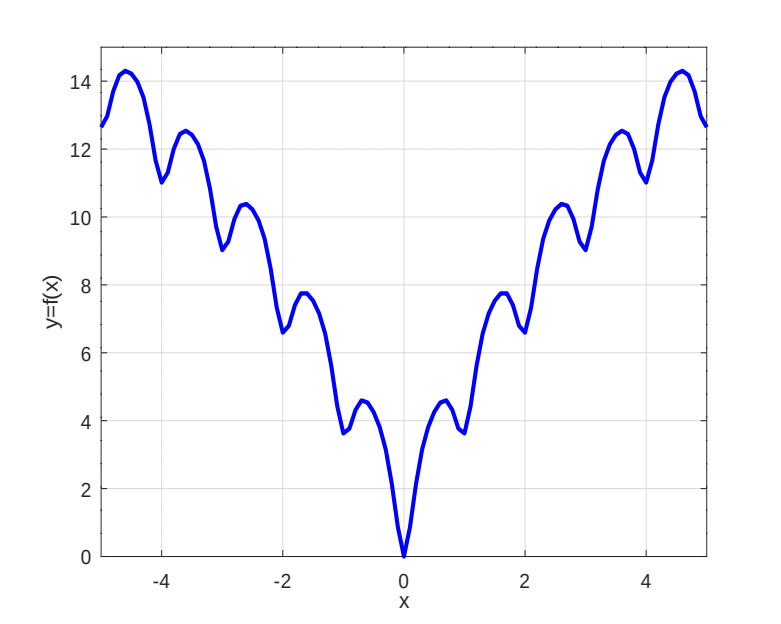
OBS: A convergência do algoritmo pode ser monitorada através do valor da função objetivo para a melhor solução global $f(\mathbf{x}_g^{best})$.

Exemplo 1

Encontrar o mínimo da função de Ackley 1-D[a]:

$$f(x) = -20e^{-0.2|x|} - e^{\cos(2\pi x)} + 20 + e, \tag{23}$$

para $x \in [-5, 5]$. A solução ótima é $f(0) = 0$.



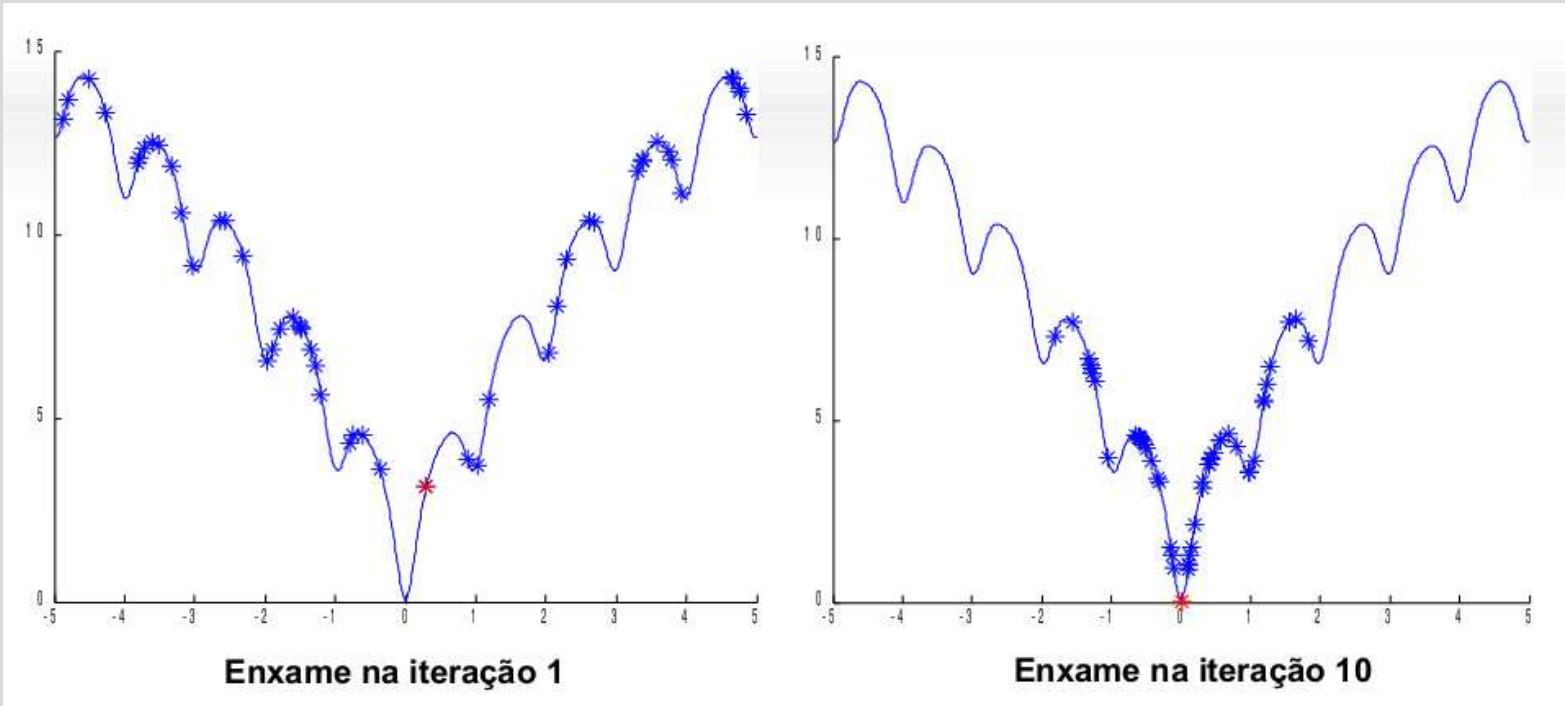
[a] Conferir em `http://www.sfu.ca/~ssurjano/ackley.html`

Exemplo 1

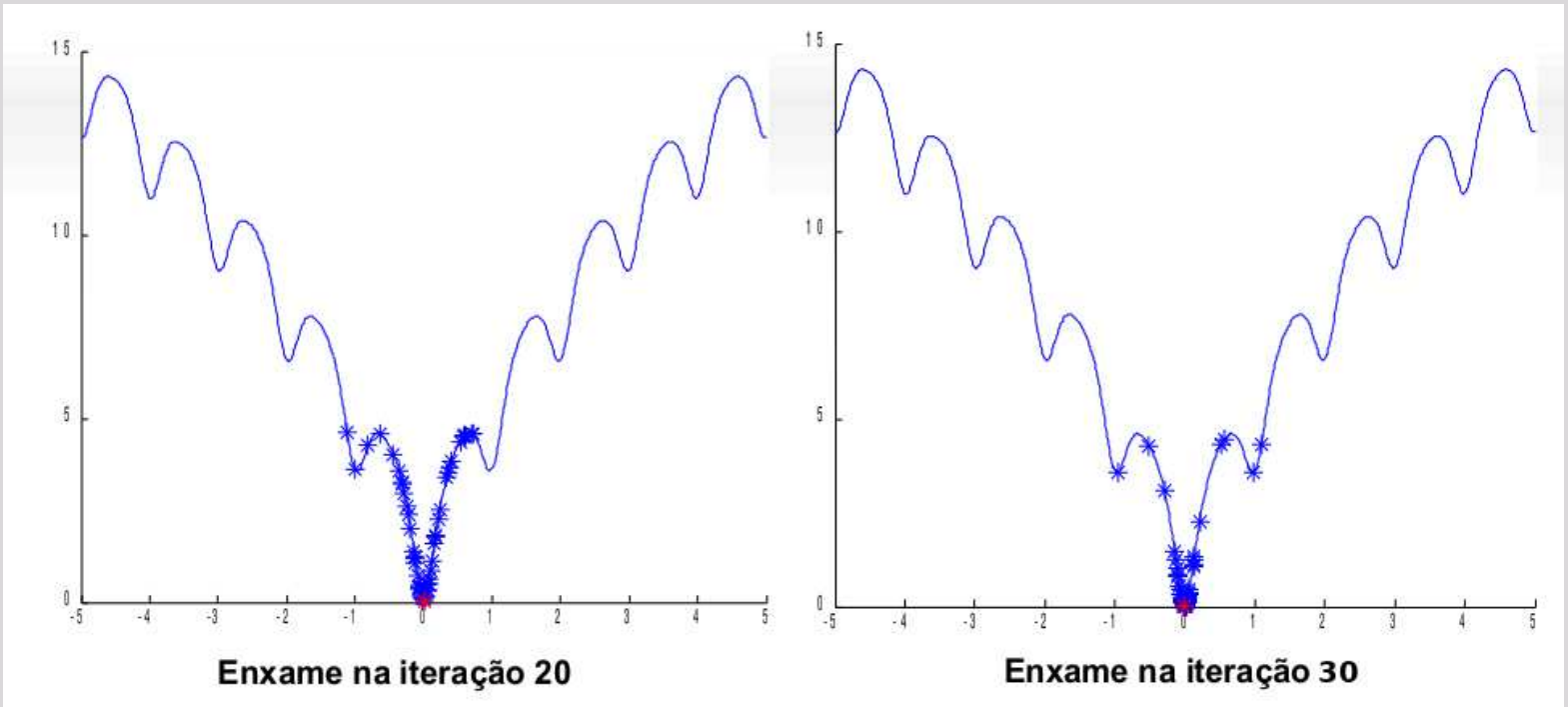
- Solução encontrada após 30 iterações: $f(-0.00013380) = 0.00053616$.

Evolução da função objetivo com um enxame de 50 partículas.

Exemplo 1



Exemplo 1

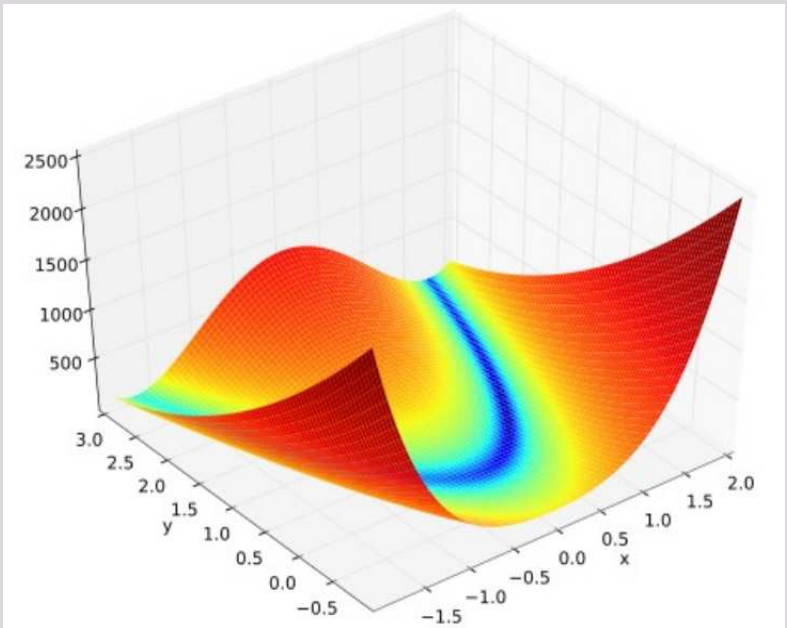


Exemplo 2

Encontrar o mínimo da função de Rosenbrock 2-D [a]:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2, \tag{24}$$

para $x, y \in [-5, 5]$. A solução ótima é $f(1, 1) = 0$.



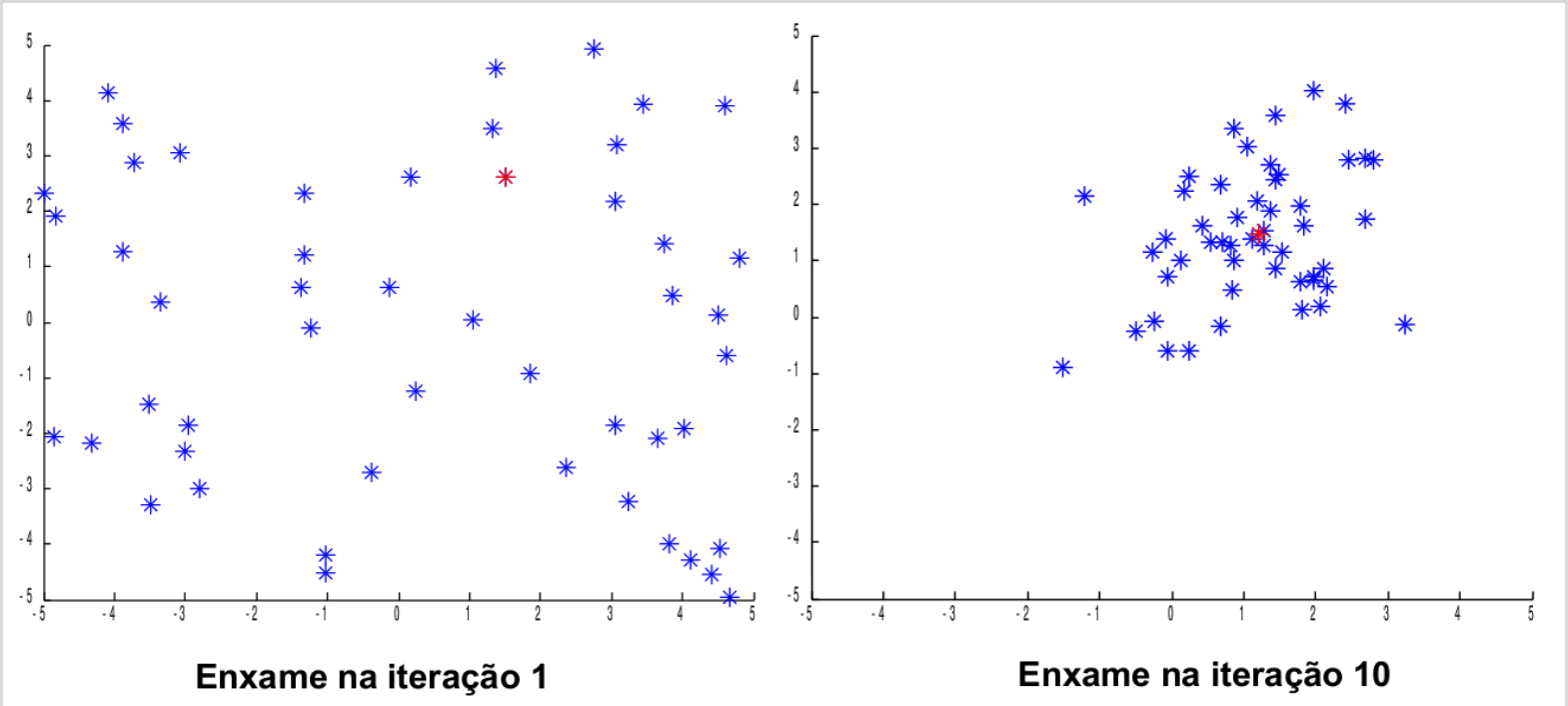
[a] Conferir em `http://www.sfu.ca/~ssurjano/rosen.html`

Exemplo 2

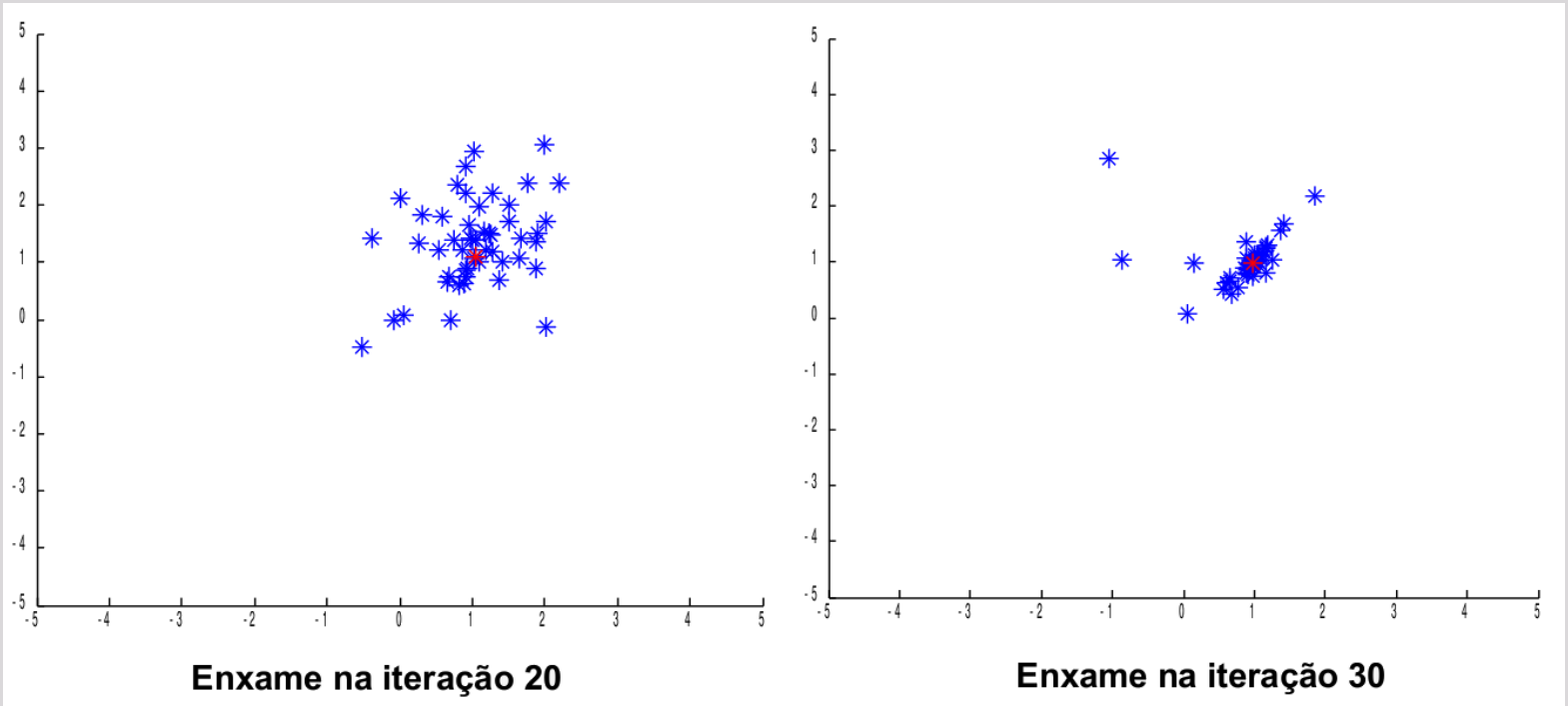
- Solução encontrada após 50 iterações: $f(1.0000, 1.0001) = 0.000003$.

Evolução da função objetivo com um enxame de 50 partículas.

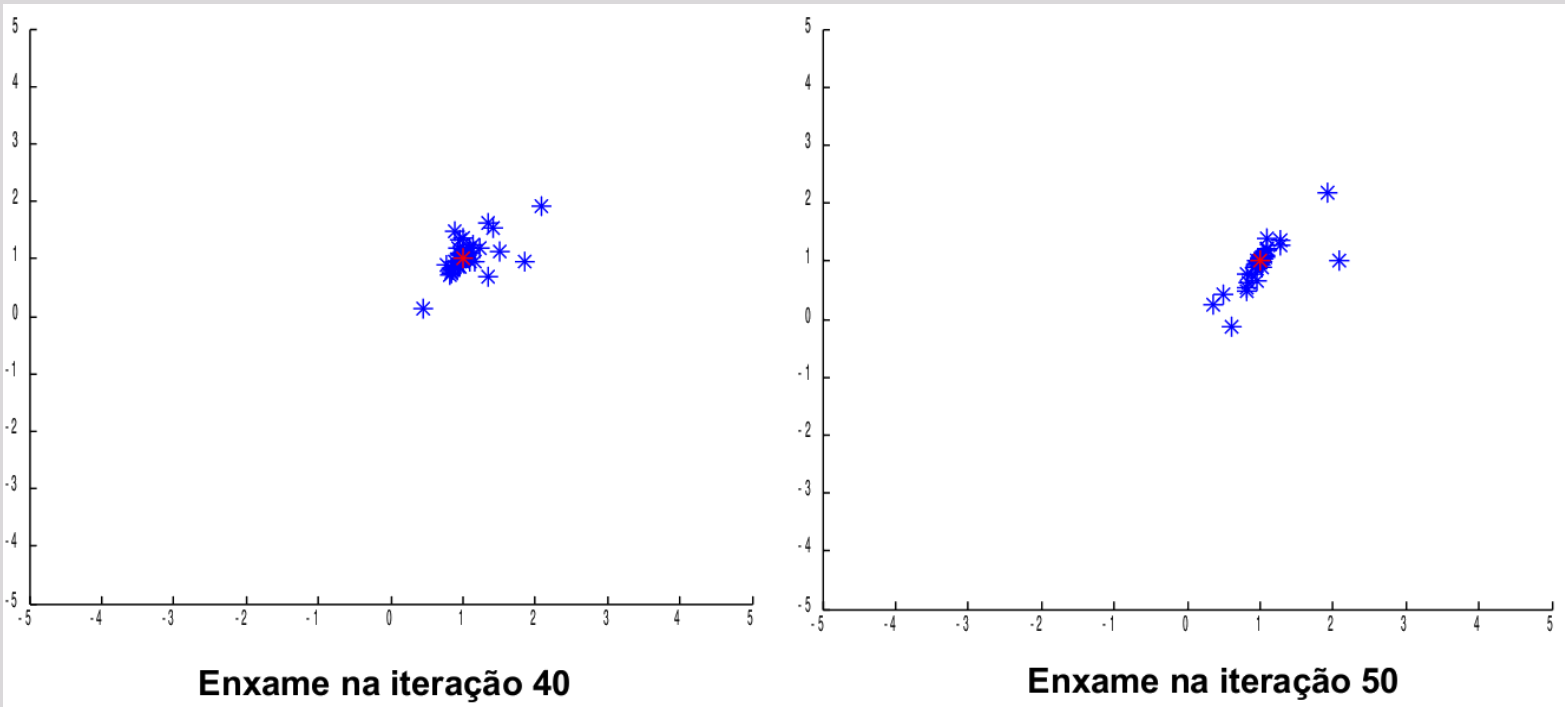
Exemplo 2



Exemplo 2



Exemplo 2



Referências Adicionais

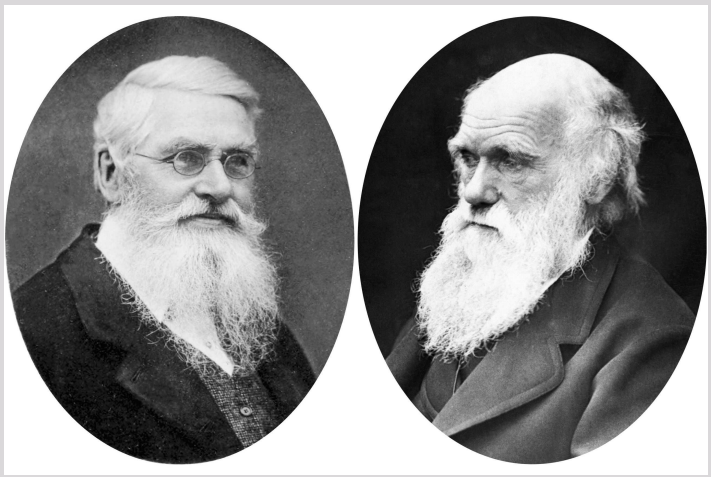
1. D. Bratton and J. Kennedy, "Defining a Standard for Particle Swarm Optimization", 2007 IEEE Swarm Intelligence Symposium, Honolulu, HI, 2007, pp. 120-127, doi: 10.1109/SIS.2007.368035.
2. Eberhart R.C., Shi Y. (1998) Comparison between genetic algorithms and particle swarm optimization. In: Porto V.W., Saravanan N., Waagen D., Eiben A.E. (eds) Evolutionary Programming VII. EP 1998. Lecture Notes in Computer Science, vol 1447. Springer, Berlin, Heidelberg. https://doi.org/10.1007/BFb0040812
3. Mattos, C.L.C., Barreto, G.A. & Cavalcanti, F.R.P. An improved hybrid particle swarm optimization algorithm applied to economic modeling of radio resource allocation. Electron Commer Res 14, 51–70 (2014). https://doi.org/10.1007/s10660-013-9128-x

Parte II

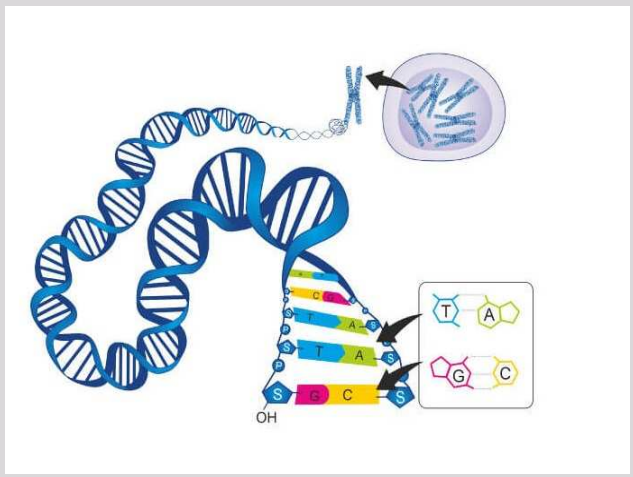
Algoritmos Genéticos

Introdução

Metaheurística para otimização global inspirada na teoria da evolução pela seleção natural de Charles Darwin e Alfred Russel Wallace, com a incorporação de conceitos da genética moderna, tais como genes, cromossomos, seleção, recombinação e mutação.



(c) Wallace & Darwin



(d) Molécula de DNA

Introdução

- Teoria e várias versões de GA (binários e contínuos) estão descritas em detalhes no livro abaixo:

  Andries Engelbrecht (2007). Computational Intelligence: An Introduction. *John Wiley & Sons, 2a. edição.*

- Assim como o algoritmo PSO, GA é um metaheurística populacional, estocástica, livre de gradiente, e de inspiração biológica (metaheurística bioinspirada).
- Soluções são representadas por entidades matemáticas abstratas, denominadas de cromossomos.

Representação de um cromossomo/indivíduo

- Em um GA, o i-ésimo cromossomo ou indivíduo da população é representado com um vetor-linha de dimensão p: $\mathbf{x}_i \in \mathbb{R}^p$.

$$\mathbf{x}_i = [x_{i1} \;\; x_{i,2} \;\; \cdots \;\; x_{ij} \;\; \cdots \;\; x_{ip}], \tag{25}$$

em que o elemento $x_{ij} \in \mathbb{R}$ define o j-ésimo gene do i-ésimo cromossomo.

Representação de um indivíduo/cromossomo

- Assim, uma população de N indivíduos na geração t é representada como uma matriz $\mathbf{P}(t)$, de dimensão $N \times p$.

$$\mathbf{X}(t) = \begin{bmatrix} \mathbf{x}_1(t) \\ \vdots \\ \mathbf{x}_i(t) \\ \vdots \\ \mathbf{x}_N(t) \end{bmatrix} = \begin{bmatrix} x_{1,1}(t) & x_{1,2}(t) & \cdots & x_{1,j}(t) & \cdots & x_{1,p}(t) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{i,1}(t) & x_{i,2}(t) & \cdots & x_{i,j}(t) & \cdots & x_{i,p}(t) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1}(t) & x_{N,2}(t) & \cdots & x_{N,j}(t) & \cdots & x_{N,p}(t) \end{bmatrix} \tag{26}$$

- Cada linha dessa matriz corresponde a uma solução-candidata para o problema de interesse.
- A cada geração, N soluções candidatas são avaliadas por seu valor da função-objetivo: $f_i(t) = f(\mathbf{x}_i(t))$.

Formalização Matemática do Problema

- A cada iteração t uma nova população é formada por meio de operações matemáticas sobre as linhas da matriz $\mathbf{X}(t)$.
- Pode-se entender a população a ser avaliada na próxima iteração, $\mathbf{X}(t+1)$, como resultante da aplicação sucessiva de transformações não-lineares às linhas da população $\mathbf{X}(t)$.
- Formalmente, temos a seguinte representação:

$$T\left(\mathbf{X}(t)\right) \rightarrow \mathbf{X}(t+1). \tag{27}$$

Formalização Matemática do Problema

- Em GAs, o operador T é resultado da aplicação sequencial de três outros operadores:
  1. Operador **seleção** (S).
  2. Operador **recombinação** (R).
  3. Operador **mutação** (M).

$$\text{Ou seja: } M\left(R\left(S\left(\mathbf{X}(t)\right)\right)\right) \rightarrow \mathbf{X}(t+1). \tag{28}$$

- É comum o uso de um quarto operador, chamado de **elitismo**, antes da aplicação do operador seleção.
- Este operador seleciona (deterministicamente) um pequeno número de indivíduos com melhor avaliação para passarem automaticamente à próxima geração.

Operador Seleção: Método do Torneio

- Descreveremos aqui o método do torneio com reposição (*tournament selection with replacement*).
- Para formar um par de cromossomos para recombinação, selecionam-se *aleatoriamente* 2 pares de indivíduos da população atual $\mathbf{X}(t)$.
- Seja $(\mathbf{x}_{r_1}, \mathbf{x}_{r_2})$, o primeiro par selecionado. Deste par, escolher o indivíduo mais apto:

$$\text{SE } f(\mathbf{x}_{r_1}) < f(\mathbf{x}_{r_2}), \text{ENTÃO } \mathbf{x}_{r_1} \text{ é escolhido.} \tag{29}$$

- Seja $(\mathbf{x}_{r_3}, \mathbf{x}_{r_4})$ o segundo par selecionado. Escolher o cromossomo mais apto:

$$\text{SE } f(\mathbf{x}_{r_3}) < f(\mathbf{x}_{r_4}), \text{ENTÃO } \mathbf{x}_{r_3} \text{ é escolhido.} \tag{30}$$

Operador Seleção: Método do Torneio

- O par $(\mathbf{x}_{r_1}, \mathbf{x}_{r_3})$ é copiado para a matriz de selecionados, $\mathbf{S}(t)$, para possível cruzamento.
- O processo de seleção de pares por torneio é repetido até que o total de linhas de $\mathbf{S}(t)$ seja igual N, assumindo que N é par.
- Em seguida, as linhas de $\mathbf{S}(t)$, tomadas de 2 em 2, serão processados pelo operador recombinação (*crossover*).

Operador Recombinação

- Os pares selecionados tem probabilidade p_c de gerar prole. Esta probabilidade, que deve ser alta $(0,85 < p_c < 1)$, é chamada probabilidade de cruzamento.
- Cada par de indivíduos em $\mathbf{S}(t)$ que passar por recombinação de seus genes dará origem a uma nova prole (*offspring*).
- A operação de recombinação pode ser dividida em 3 categorias, a depender do número de pais utilizados:
  - **Assexuada**: Quando a prole é gerada a partir de um só indivíduo.
  - **Sexuada**: Quando dois indivíduos geram prole de um ou mais filhos.
  - **Múltipla**: Quando mais de dois indivíduos geram prole com um ou mais filhos.

Operador Recombinação 1: Método de Wright[a]

[a] A. Wright. Genetic Algorithms for Real Parameter Optimization. In G. J. E. Rawlins, editor, Foundations of Genetic Algorithms, pages 205–220, San Mateo, C.A., 1991. Morgan Kaufmann.

- Seja $(\mathbf{s}_1(t), \mathbf{s}_2(t))$ o par de cromossomos de $\mathbf{S}(t)$ em processamento.

  SE $U(0,1) \leq p_c$, **ENTÃO** escolher os dois mais aptos da prole.

$$\textbf{Indivíduo 1: } \tilde{\mathbf{x}}_1(t) = \mathbf{s}_1(t) + \mathbf{s}_2(t), \tag{31}$$

$$\textbf{Indivíduo 2: } \tilde{\mathbf{x}}_2(t) = 1.5\mathbf{s}_1(t) - 0.5\mathbf{s}_2(t), \tag{32}$$

$$\textbf{Indivíduo 3: } \tilde{\mathbf{x}}_3(t) = -0.5\mathbf{s}_1(t) + 1.5\mathbf{s}_2(t). \tag{33}$$

  SENÃO os dois indivíduos da prole são cópias exatas de seus pais.

- Por mais aptos, entende-se os cromossomos da prole com melhores valores segundo sua função-aptidão (*fitness function*).
- Para problema for de minimização (maximização), buscam-se os menores (maiores) valores da função-aptidão.

Operador Recombinação 2: Simulated Binary Crossover (SBX) [a]

[a] K. Deb & R.B. Agrawal. Simulated Binary Crossover for Continuous Space. Complex Syst., 9:115-148, 1995.

- Para cada par $(\mathbf{s}_1(t), \mathbf{s}_2(t))$ de indivíduos de $\mathbf{S}(t)$, produzir uma prole de outros dois indivíduos por meio das seguintes operações:

$$\textbf{Indivíduo 1: } \tilde{x}_{1j}(t) = 0.5[(1+\gamma_j)s_{1j}(t) + (1-\gamma_j)s_{2j}(t)], \quad (34)$$

$$\textbf{Indivíduo 2: } \tilde{x}_{2j}(t) = 0.5[(1-\gamma_j)s_{1j}(t) + (1+\gamma_j)s_{2j}(t)], \quad (35)$$

em que

$$\gamma_j = \begin{cases} (2r_j)^{\frac{1}{\eta+1}}, & \text{se } r_j \leq 0.5 \\ \left[\frac{1}{2(1-r_j)}\right]^{\frac{1}{\eta+1}}, & \text{C.C.} \end{cases} \quad (36)$$

tal que $r_j \sim U(0,1)$, e $\eta > 0$ é o índice da distribuição. Os autores sugerem $\eta = 1$.

Operador Recombinação 2: Simulated Binary Crossover (SBX) [a]

[a] K. Deb & R.B. Agrawal. Simulated Binary Crossover for Continuous Space. Complex Syst., 9:115-148, 1995.

- O operador SBX gera prole simetricamente em torno dos pais, prevenindo viés na direção de um dos pais.
- O viés aqui significa os filhos serem mais parecidos com um dos pais.
- Para valores altos de η, há uma probabilidade maior de a prole ser gerado próximo aos pais. Ou seja, prole mais semelhante aos pais.
- Para valores pequenos de η, prole será mais distante dos pais. Ou seja, prole menos semelhantes aos pais.

Operador Recombinação 2: Versão vetorial do SBX

- Para cada par $(\mathbf{s}_1(t), \mathbf{s}_2(t))$ de indivíduos de $\mathbf{S}(t)$, produzir uma prole de outros dois indivíduos por meio das seguintes operações:

$$\textbf{Indivíduo 1: } \tilde{\mathbf{x}}_1(t) = 0.5[(\mathbf{1}_p + \boldsymbol{\gamma}) \circ \mathbf{s}_1(t) + (\mathbf{1}_p - \boldsymbol{\gamma}) \circ \mathbf{s}_2(t)], \tag{37}$$

$$\textbf{Indivíduo 2: } \tilde{\mathbf{x}}_2(t) = 0.5[(\mathbf{1}_p - \boldsymbol{\gamma}) \circ \mathbf{s}_1(t) + (\mathbf{1}_p + \boldsymbol{\gamma}) \circ \mathbf{s}_2(t)], \tag{38}$$

em que $\mathbf{1}_p$ é um vetor de 1's de dimensão $1 \times p$, e $\boldsymbol{\gamma}$ é o vetor aleatório de mesma dimensão cujo as componentes são definidas como na Eq. (36).

Operador Recombinação

- Caso o valor do j-ésimo gene do i-ésimo indivíduo, $x_{ij}(t)$, esteja fora dos seus limites, aplicar as seguintes regras.

  Regra 1 - Se ultrapassar limite superior:

  $$\text{SE } x_{ij}(t) > x_{ij}^u, \text{ ENTÃO } x_{ij}(t) = x_{ij}^u.$$

  Regra 2 - Se ultrapassar limite inferior:

  $$\text{SE } x_{ij}(t) < x_{ij}^l, \text{ ENTÃO } x_{ij}(t) = x_{ij}^l.$$

Explicando o Operador Recombinação

- A operação de recombinação tem o objetivo de conferir certa variabilidade genética em função da troca de material genético entre os indivíduos.
- Ao mesmo tempo, mantém uma espécie de *memória genética* da população.
- A intensidade da troca é controlada pela probabilidade de recombinação.
- Se p_c é muito baixa, então haverá menor troca de informação genética devido à recombinação e, consequentemente, maior inércia (lentidão) a mudanças genéticas na população.
- Por este motivo, p_c deve ser mantida em um valor alto.

Operador Mutação

- Para cada indivíduo $\tilde{\mathbf{x}}_i(t)$ da prole gerada, aplicar a seguinte regra:

  SE $U(0,1) \leq p_m$, **ENTÃO**

$$\tilde{\mathbf{x}}'_i(t) = \tilde{\mathbf{x}}_i(t) + \alpha(\mathbf{x}^u - \mathbf{x}^l) \circ \mathbf{n}, \quad i = 1, 2. \tag{39}$$

  em que

  - p_m é a probabilidade de mutação.
  - $0 < \alpha \ll 1$ é passo (incremento) de mutação.
  - O símbolo $\circ$ denota o produto de Hadamard.
  - $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)$ é um vetor aleatório p-dimensional amostrado de uma densidade gaussiana multivariada de vetor médio nulo e de matriz de covariância identidade.

Explicando o Operador Mutação

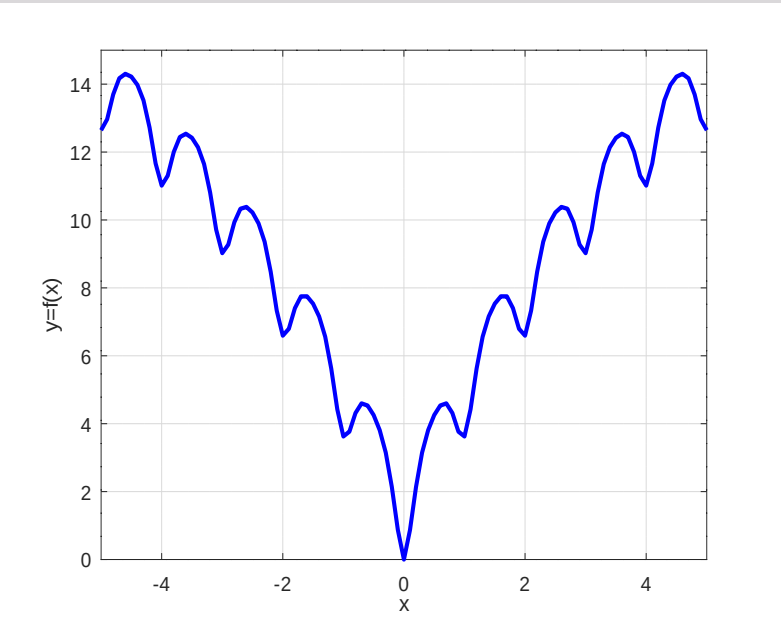
- A operação de mutação tem o objetivo de conferir certa inovação genética aos indivíduos da prole.
- A intensidade da inovação é controlada pela probabilidade de mutação.
- Se p_m é muito alta, haverá excesso de inovação, o que pode destruir a história (i.e. memória) genética da população.
- Neste caso, o GA se assemelha a uma busca global com N soluções independentes por geração.
- Por este motivo, p_m deve ser mantida em um valor bem baixo (e.g. $p_m < 5\%$).

Exemplo 1

Encontrar o mínimo da função de Ackley 1-D[a]:

$$f(x) = -20e^{-0.2|x|} - e^{\cos(2\pi x)} + 20 + e, \tag{40}$$

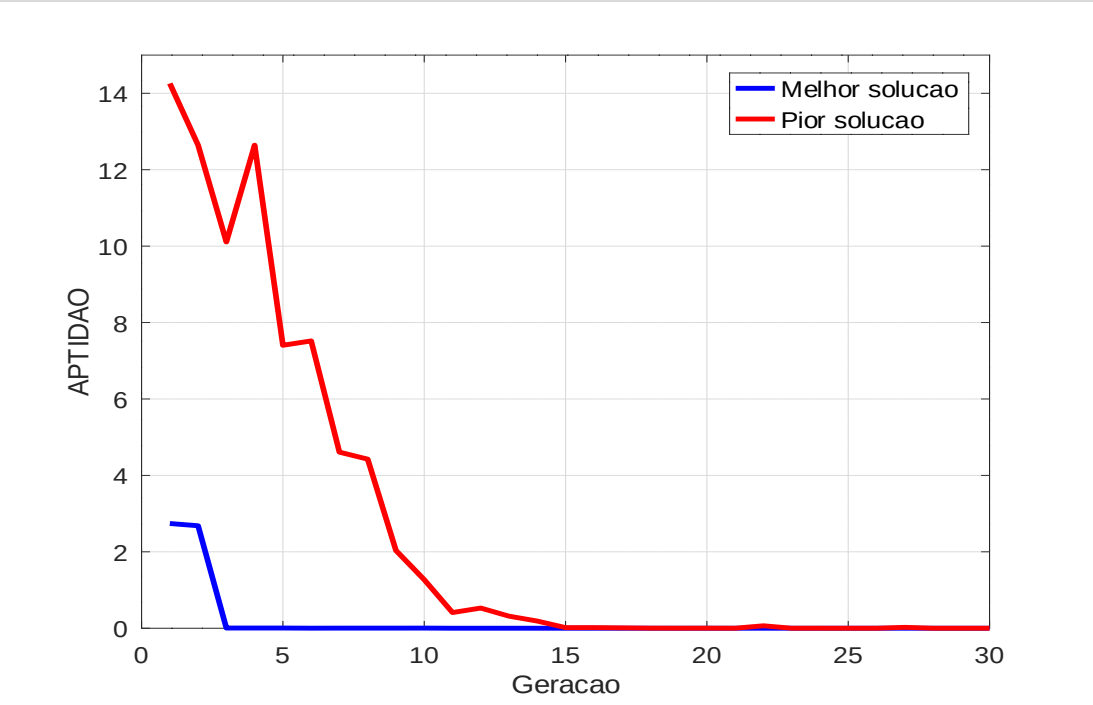
para $x \in [-5, 5]$. A solução ótima é $f(0) = 0$.



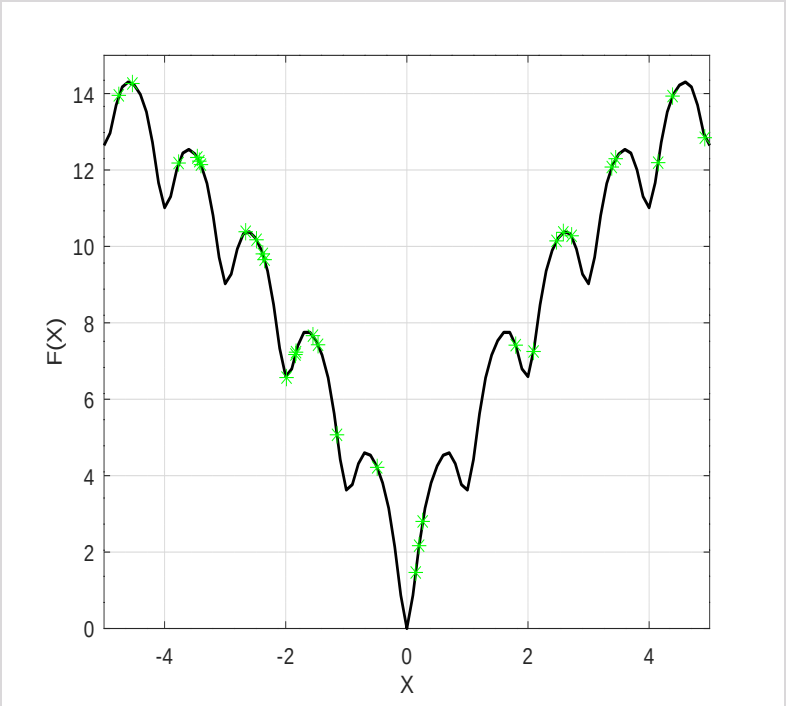
[a] Conferir em `http://www.sfu.ca/~ssurjano/ackley.html`

Exemplo 1

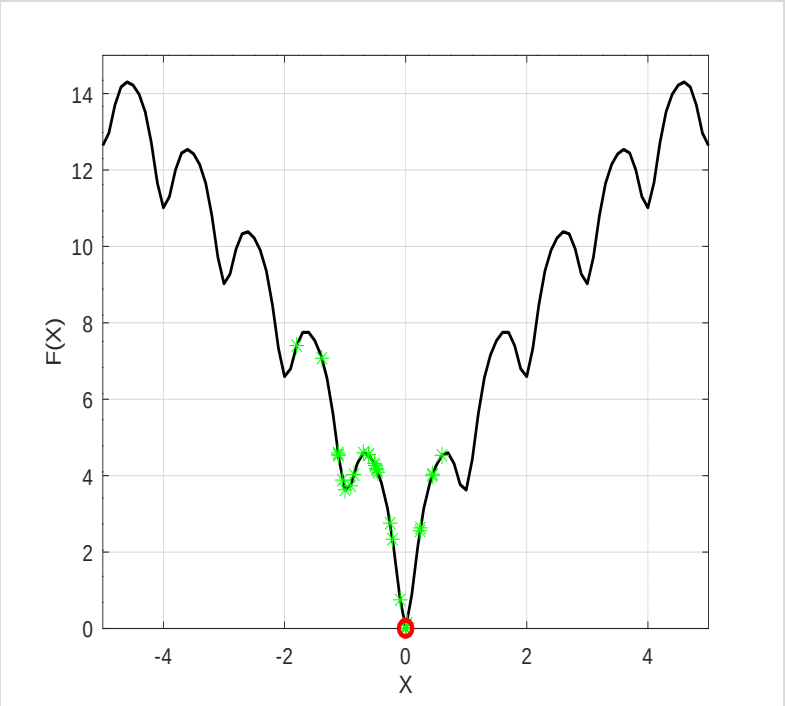
- Solução encontrada após 30 iterações: $f(0.0000011211) = 0.0000044846$.



Exemplo 1

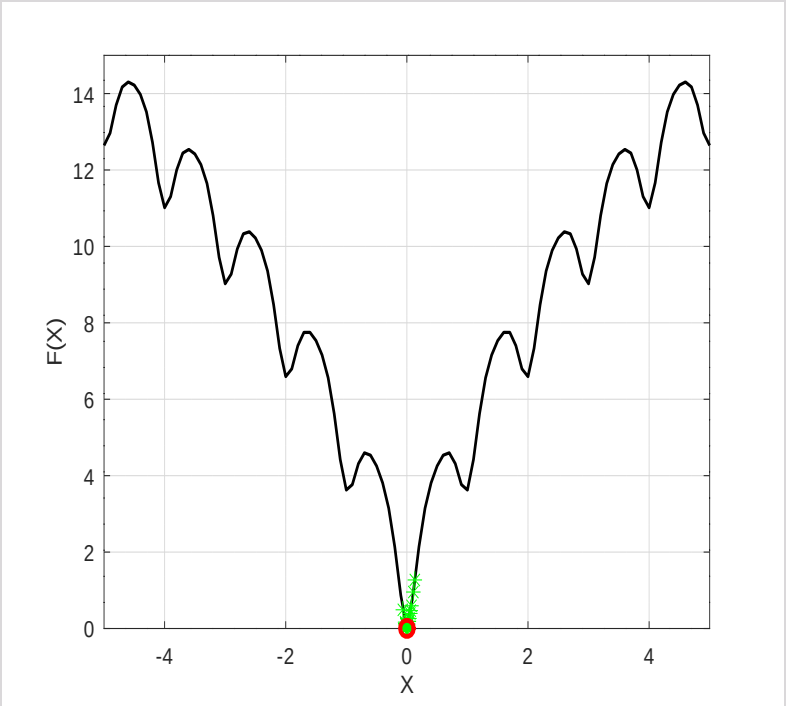


Geração 1

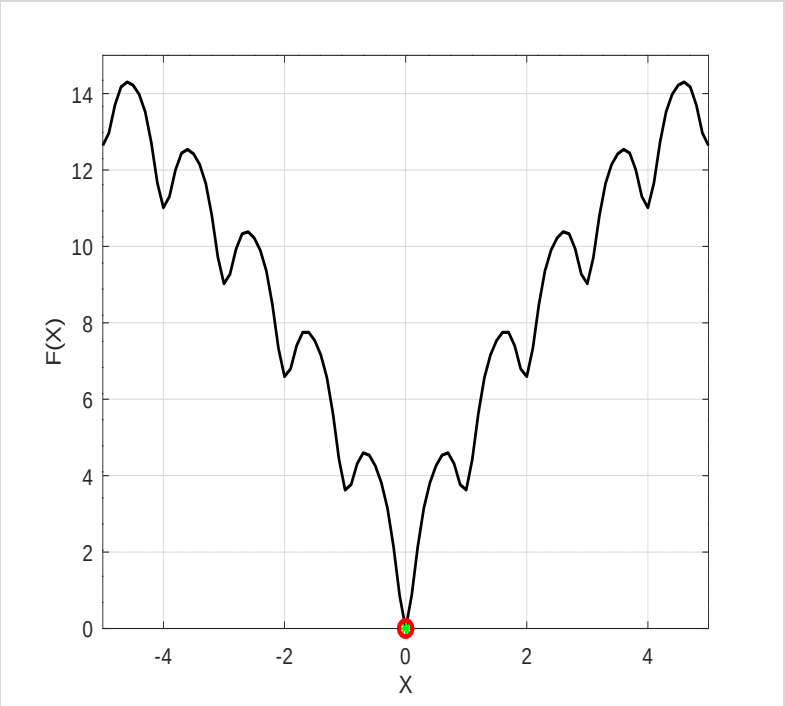


Geração 5

Exemplo 1



Geração 10



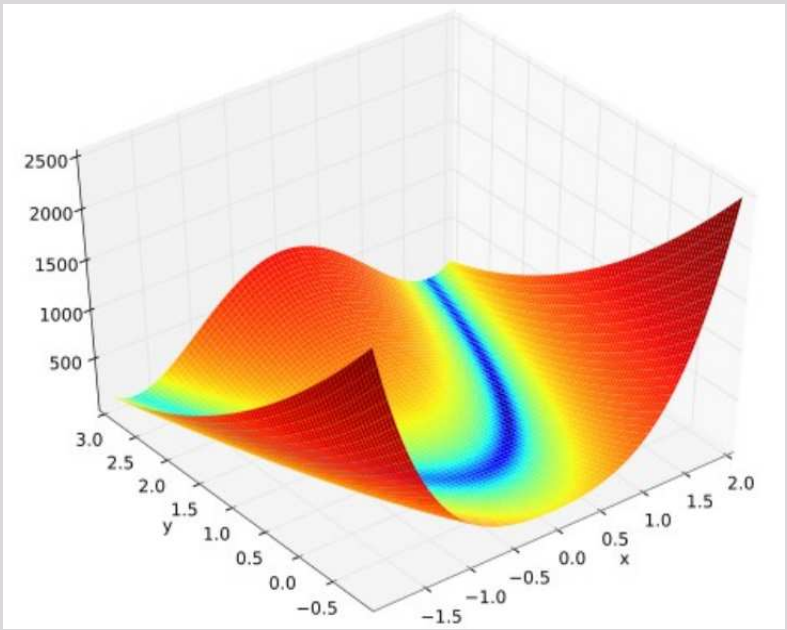
Geração 20

Exemplo 2

Encontrar o mínimo da função de Rosenbrock 2-D [a]:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2, \tag{41}$$

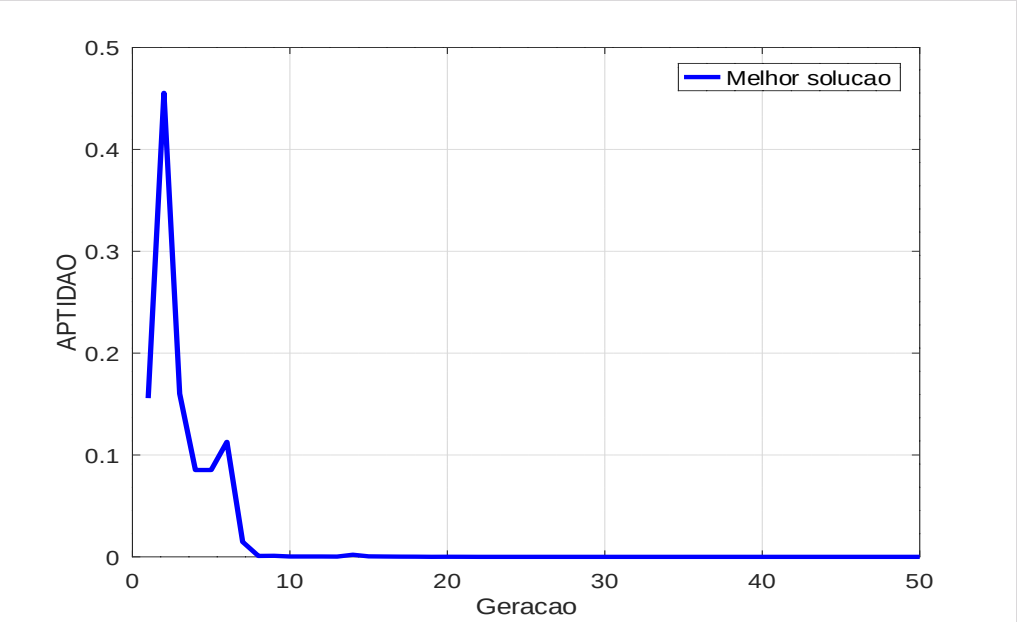
para $x, y \in [-5, 5]$. A solução ótima é $f(1, 1) = 0$.



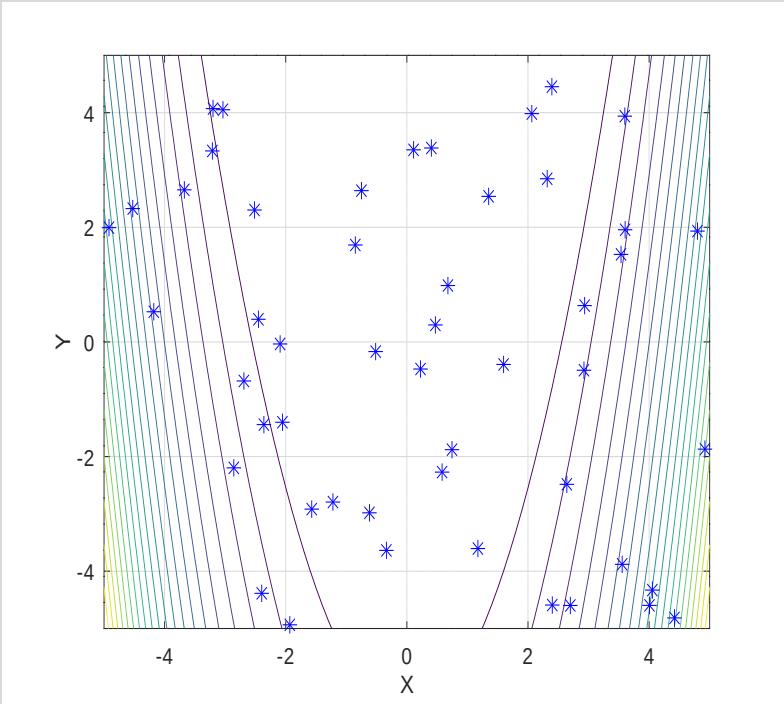
[a] Conferir em `http://www.sfu.ca/~ssurjano/rosen.html`

Exemplo 2

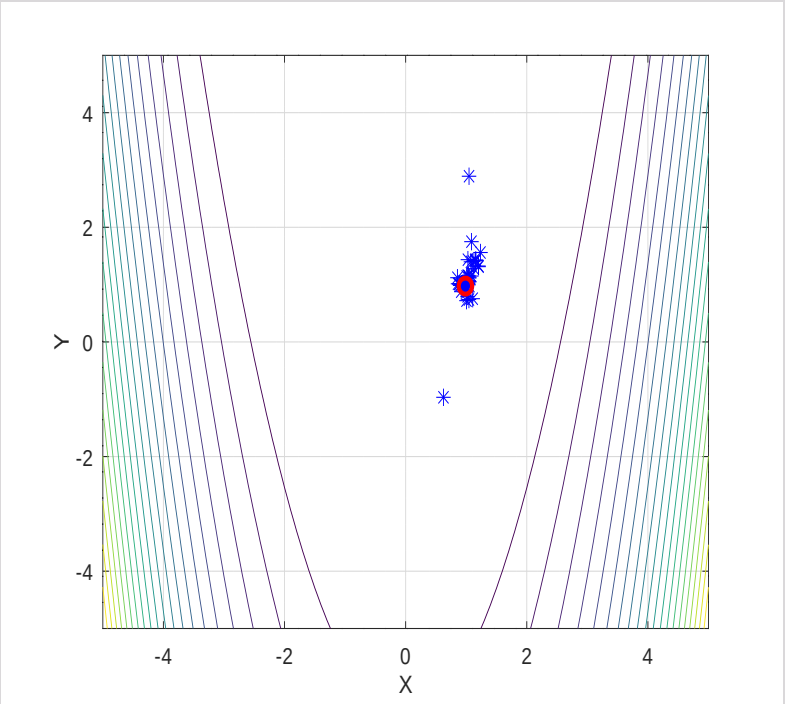
- Solução encontrada após 50 gerações: $f(1.0014, 1.0029) = 0.000002$.



Exemplo 2

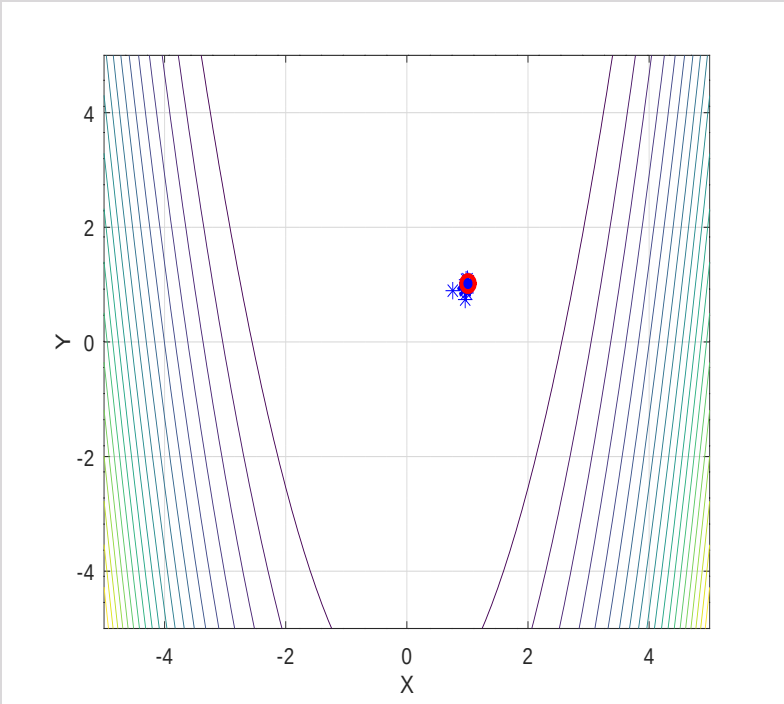


Geração 1

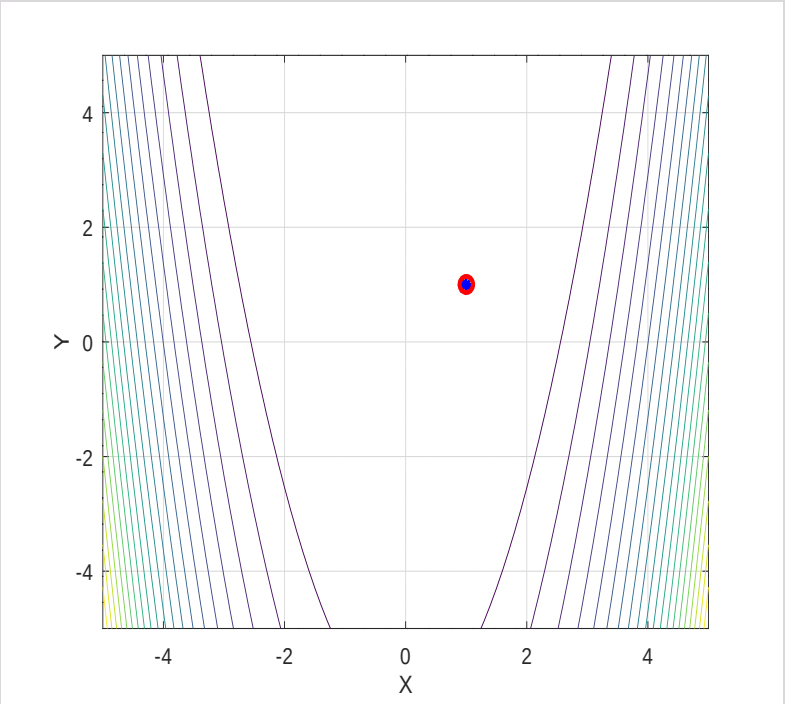


Geração 20

Exemplo 2



Geração 30



Geração 50

Referências Adicionais

1. Goldberg, David (1989). Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley Professional. ISBN 978-0201157673.
2. Fogel, David (2006). Evolutionary Computation: Toward a New Philosophy of Machine Intelligence (3rd ed.). Piscataway, NJ: IEEE Press. ISBN 978-0471669517.
3. Phelipe W. Oliveira, Guilherme A. Barreto & George A. P. Thé (2020). "A General Framework for Optimal Tuning of PID-like Controllers for Minimum Jerk Robotic Trajectories", *Journal of Intelligent & Robotic Systems*, volume 99, pages 467–486.