“Planeta Água: cultura oceânica para enfrentar as mudanças climáticas no meu território”

Método de Otimização por Enxame de Partículas

Dr. Denis Costa, Professor Titular do IFPA
Bacharelado em Ciência e Tecnologia

Otimizar

O que é Otimização

É o campo da Matemática Aplicada destinado a encontrar a melhor solução (possível) para um problema, dadas certas condições ou restrições.

Em essência, trata-se de um conjunto de técnicas e algoritmos utilizados para **maximizar** ou **minimizar** uma determinada função, conhecida como **função objetivo**, selecionando valores para um conjunto de variáveis de decisão.


Função Objetivo
Restrições Operacionais
Variáveis de Decisão

Os Pilares da Otimização

1. **Variáveis de Decisão:** São as incógnitas do problema.
2. **Função Objetivo:** É a expressão matemática que quantifica o objetivo do problema.
3. **Restrições:** São as limitações ou condições que as variáveis de decisão devem satisfazer. Elas representam as limitações do mundo real.

Classificação dos Métodos de Otimização

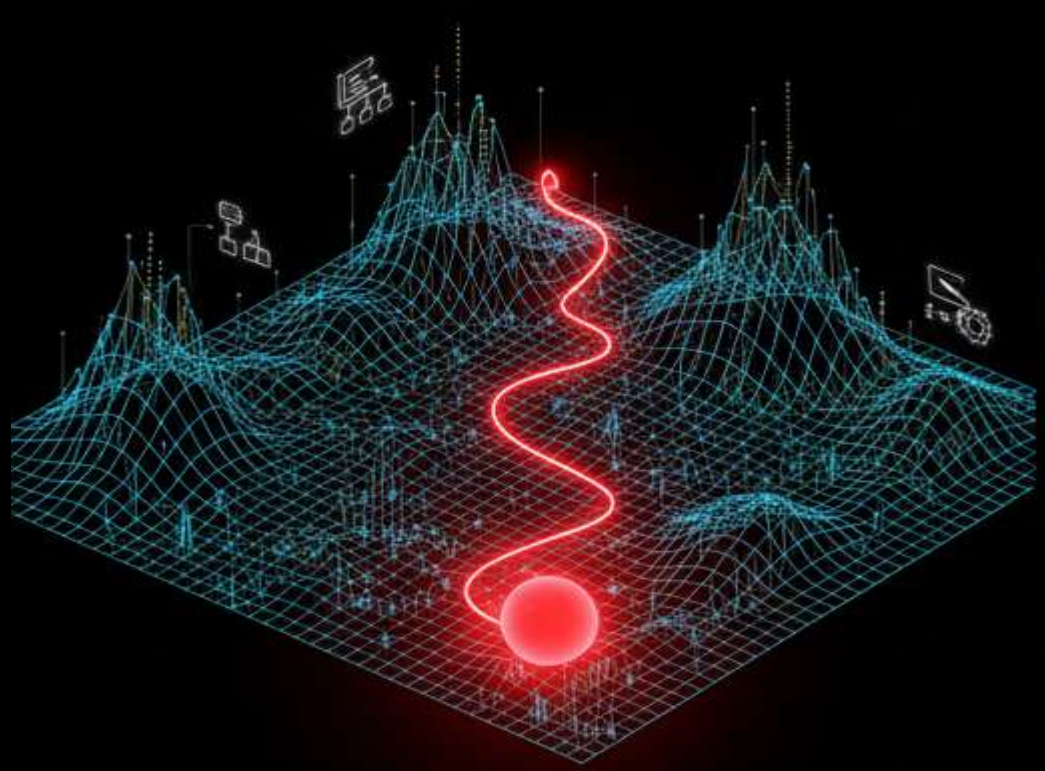
Otimização Irrestrita: Quando não há restrições sobre as variáveis de decisão.

Otimização Restrita: Quando as variáveis de decisão devem obedecer a um conjunto de restrições.

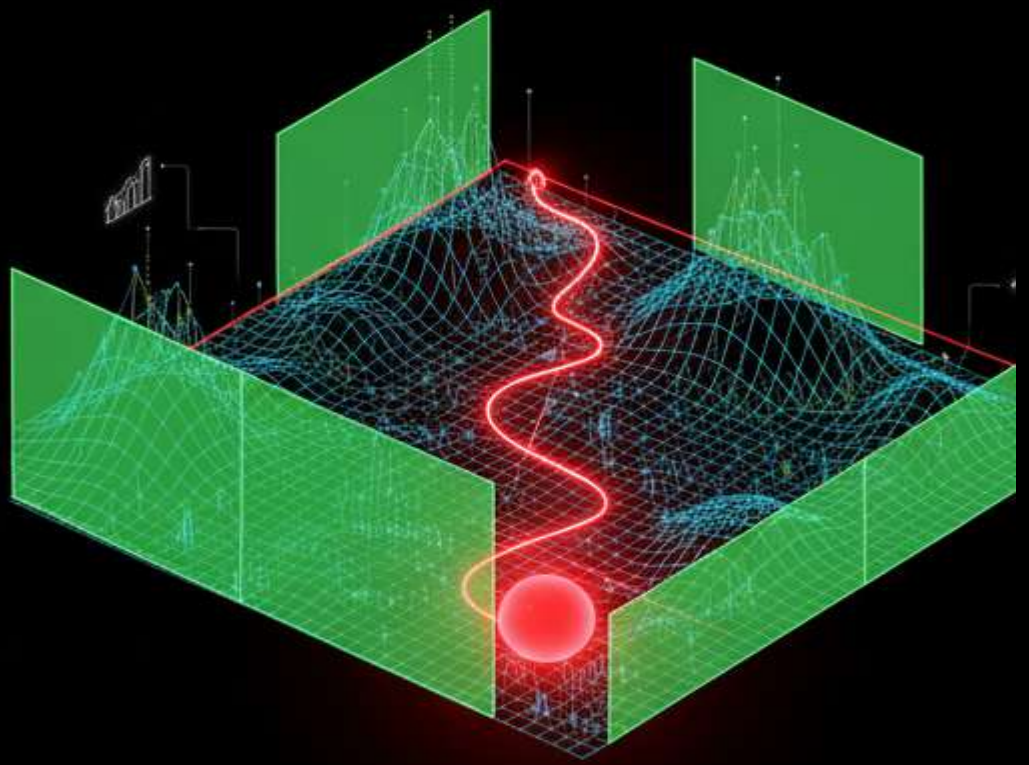
Programação Linear: Função Objetivo e todas as Restrições são equações lineares.

Programação Não-Linear: Função Objetivo ou pelo menos uma das Restrições é uma função não-linear das variáveis de decisão.

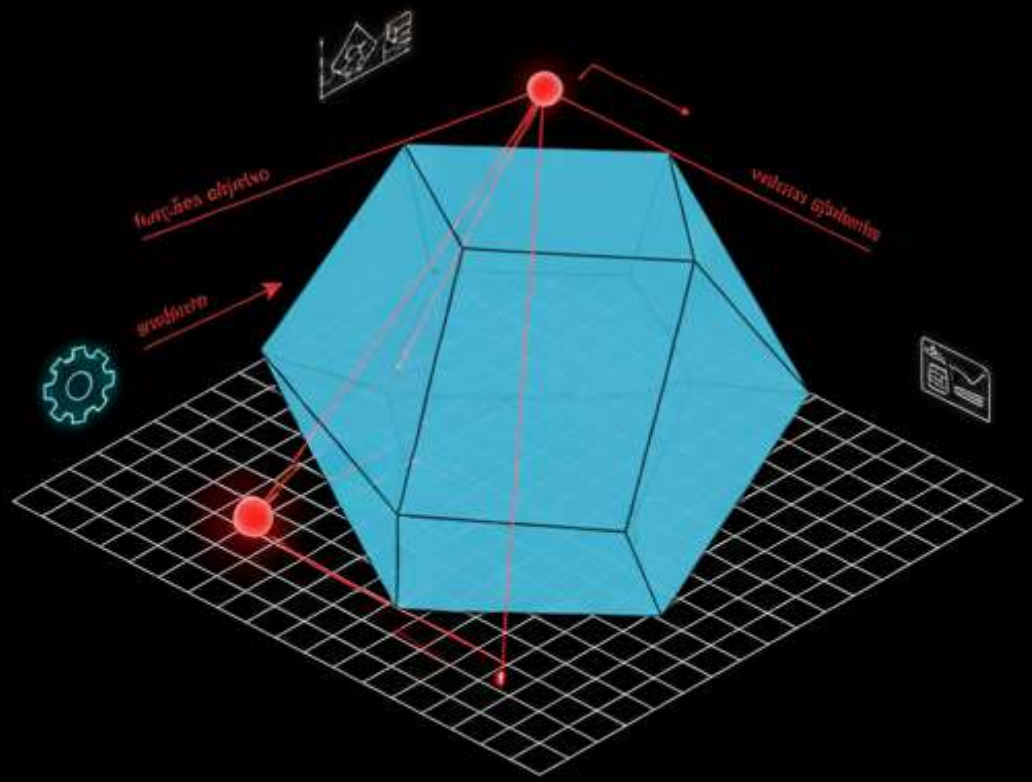
Otimização Irrestrita



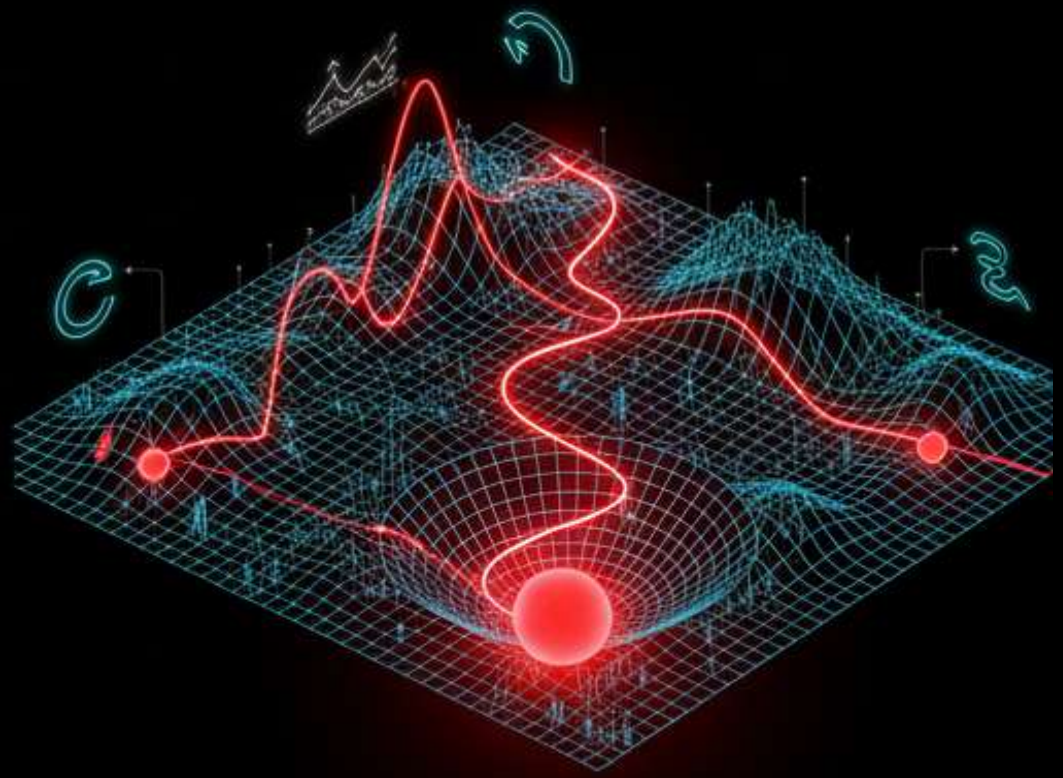
Otimização Restrita



Programação Linear



Programação Não-Linear

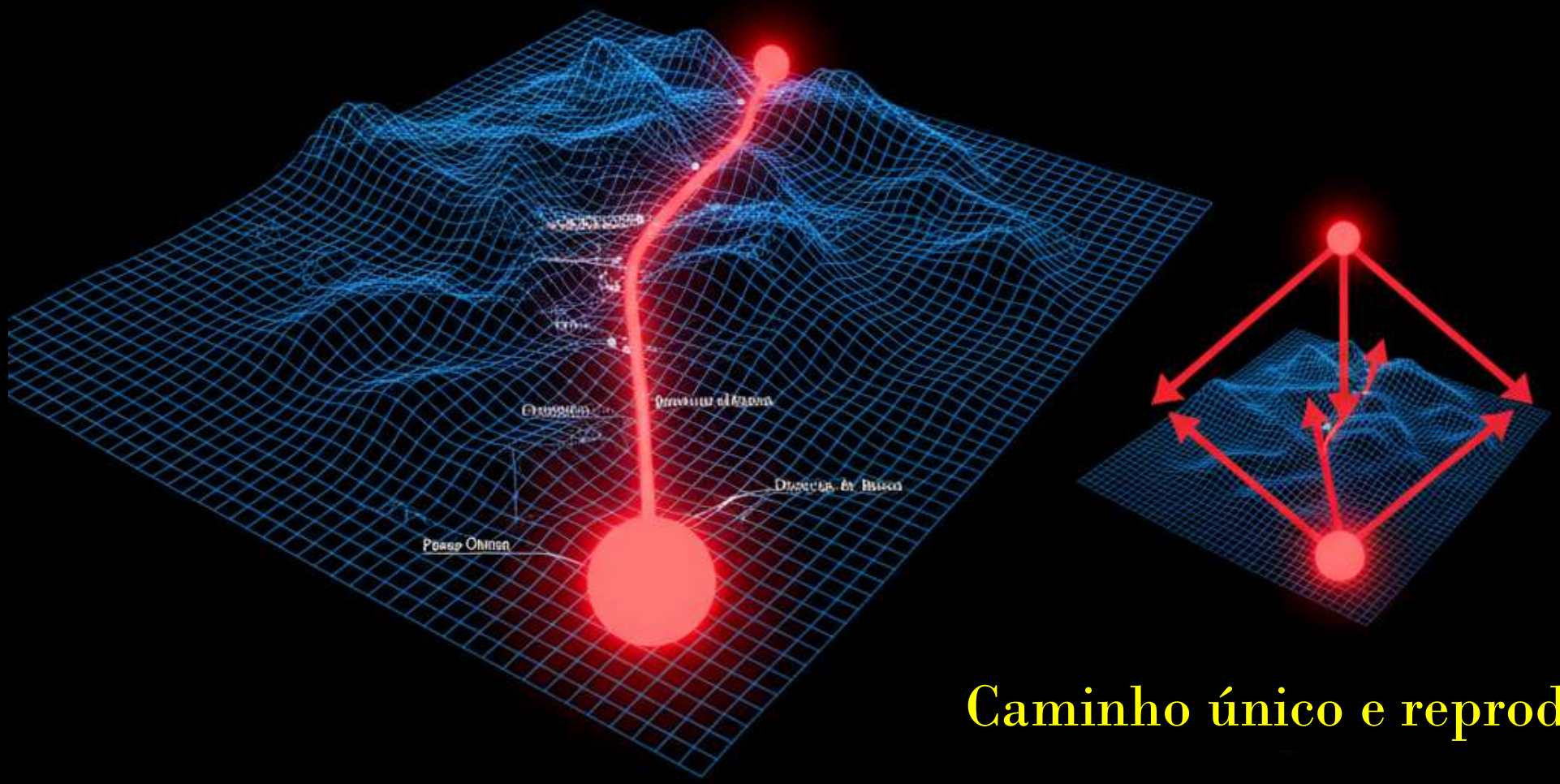


Métodos Determinísticos vs. Estocásticos

Métodos Determinísticos: São algoritmos que, partindo de um mesmo ponto inicial, seguirão sempre o mesmo caminho para encontrar a solução. Eles geralmente se baseiam em informações matemáticas precisas, como Gradiente (Derivadas) da Função Objetivo.

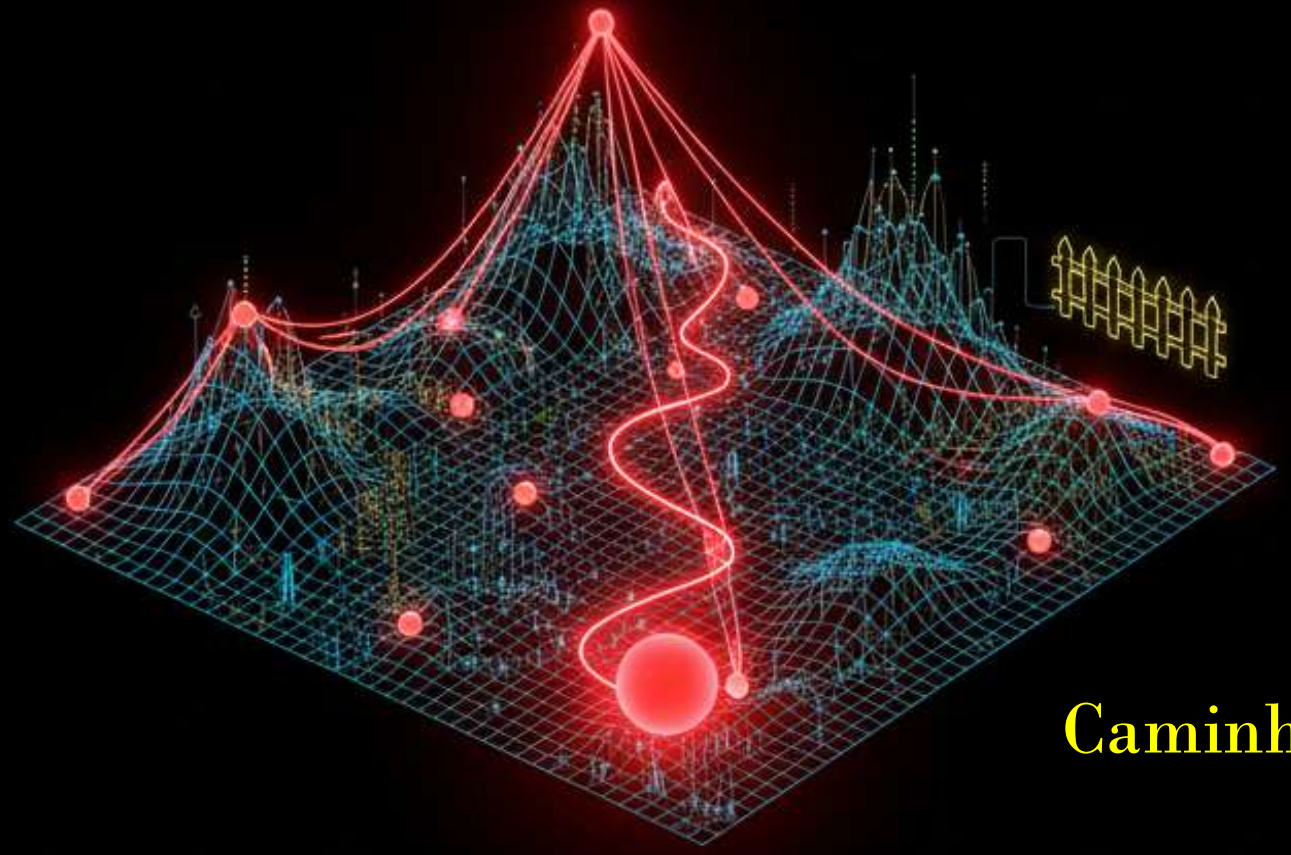
Métodos Estocásticos : Estes métodos incorporam elementos de aleatoriedade em sua lógica de busca. Buscam encontrar soluções de alta qualidade em um tempo computacional razoável. Exemplos :

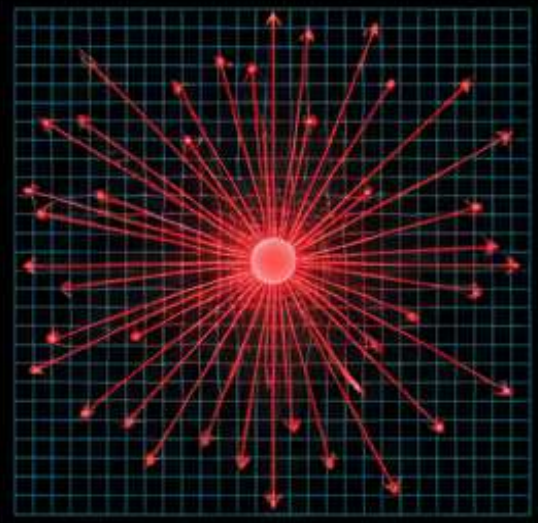
MÉTODO DETERMINÍSTICO



Caminho único e reproduzível

MÉTODO ESTOCÁSTICO





Caminhos aleatórios e exploratórios

- **Algoritmo Genético (*Genetic Algorithm*):** Inspirado na teoria da evolução de Darwin, trabalham com uma "população" de soluções que evoluem ao longo das gerações.

- **Otimização por Colônia de Formigas (*Colony Optimization Algorithm*):** Inspirado no comportamento de formigas em busca de alimento.

- **Algoritmo do Morcego** (***Bat Algorithm***): Inspirado no comportamento de ecolocalização de morcegos.

- **Otimizador Multiverso (*Multi-Verse Optimizer*):** Baseia-se em três conceitos da cosmologia: buraco branco, buraco negro e buraco de minhoca.

- **Otimização por Enxame de Partículas (*Particle Swarm Optimization* ou PSO):** Modelado a partir do comportamento social de pássaros ou peixes.

Otimização por Enxame de Partículas (*Particle Swarm Optimization* - PSO)

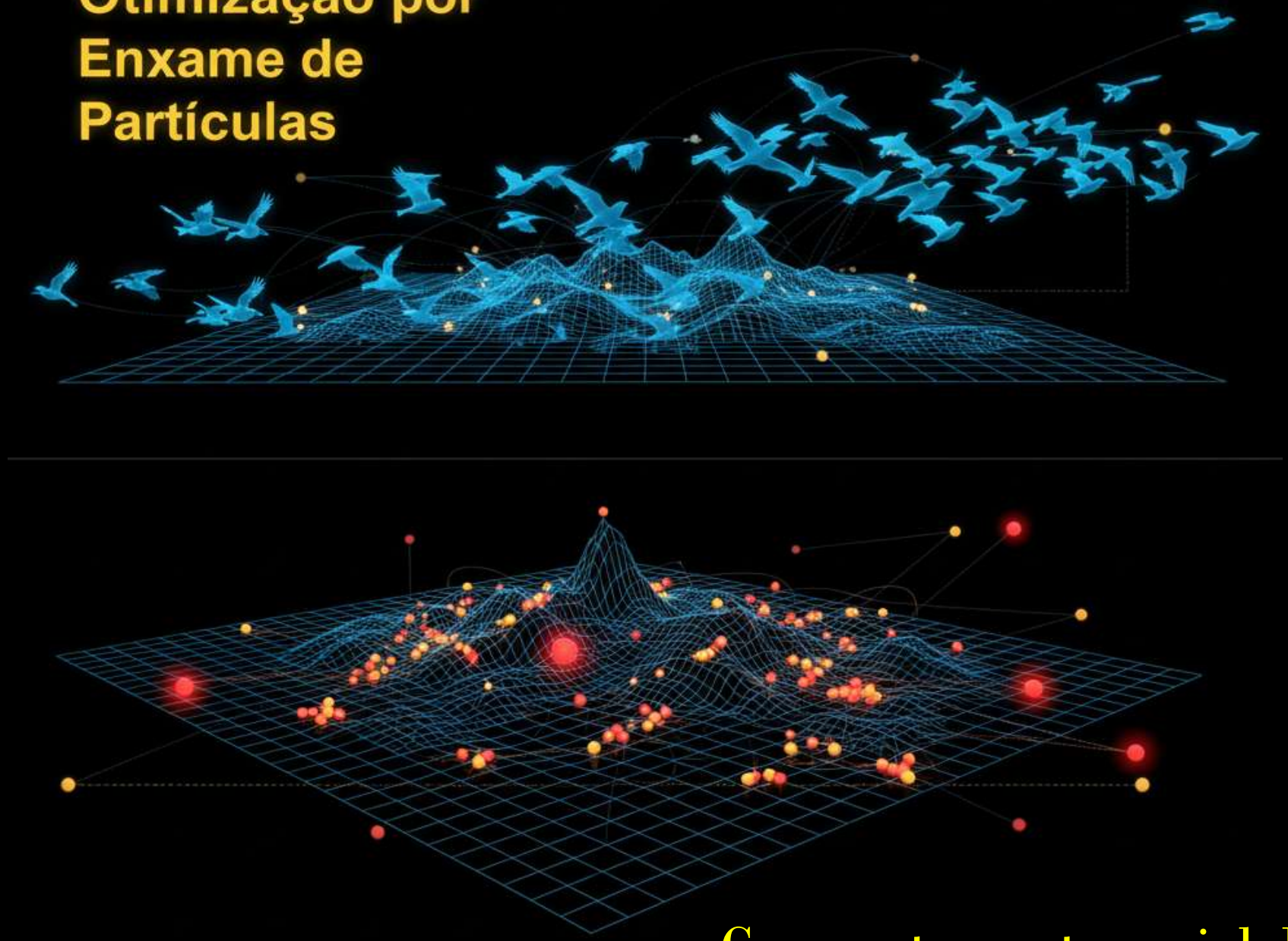
Desenvolvido por **James Kennedy** e **Russell Eberhart**, é uma meta-heurísticas baseada em padrões da natureza (como a representação do movimento de cada individuo dentro de um bando de pássaros ou de um cardume de peixes).

PSO é inspirado pelo comportamento social e cooperativo exibido por várias espécies de forma a realizar as suas experiências no espaço de pesquisa (*search-space*).

James Kennedy (nascido em 5/11/1950): Psicólogo social norte-americano.

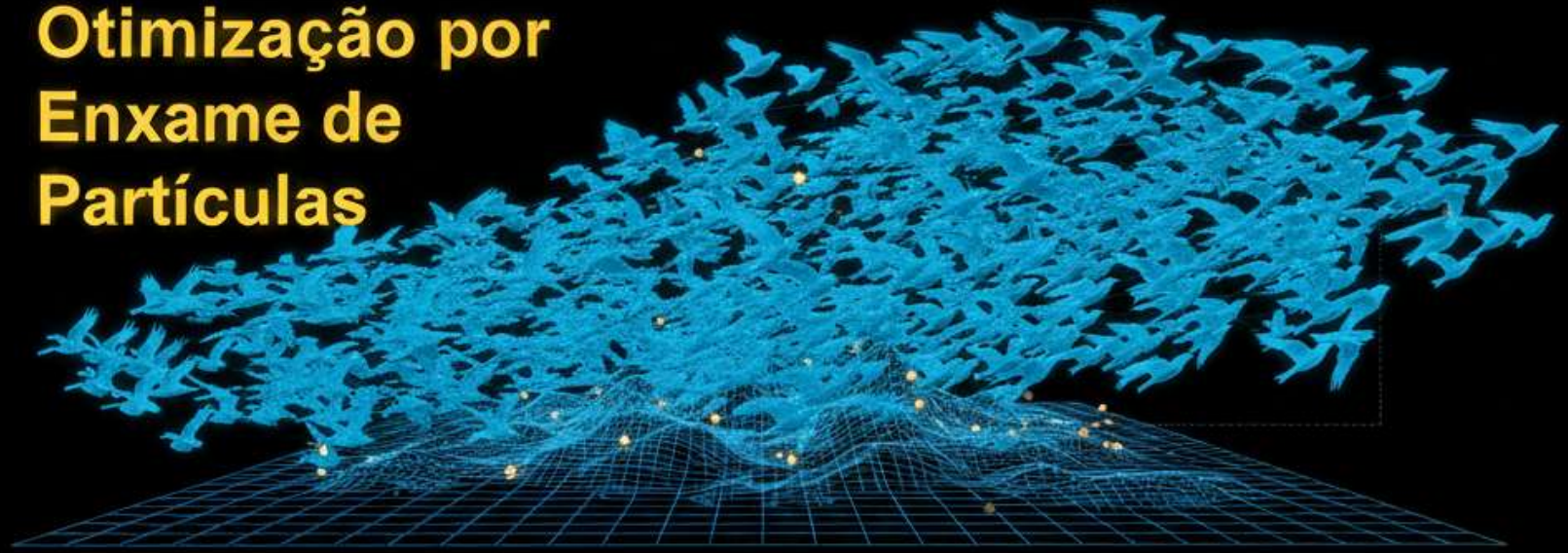
Russell Eberhart (26/01/1923 – 16/01/2011): Engenheiro eletricista norte-americano.

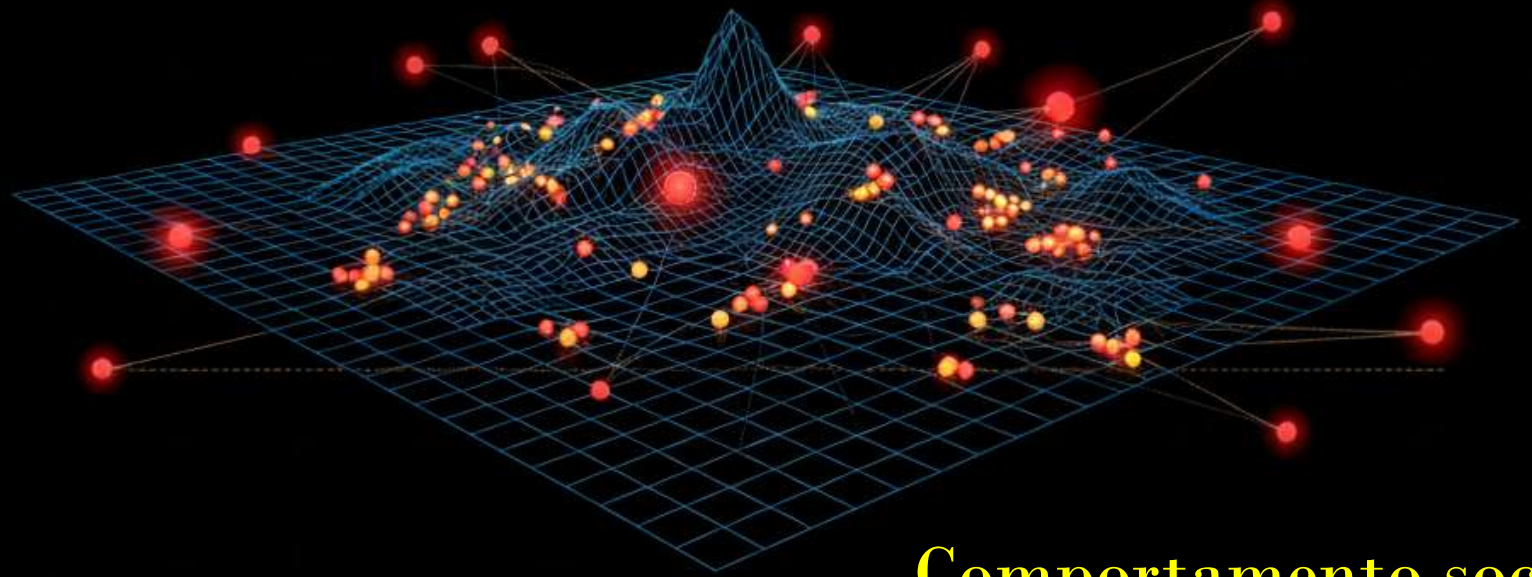
Otimização por Enxame de Partículas



Comportamento social de pássaros

Otimização por Enxame de Partículas





Comportamento social de peixes

Modelo Matemático do PSO

Posição (x_i) → $x_i = x_1, x_2, x_3, \dots, x_n$.

Velocidade (v_i) → $v_i = v_1, v_2, v_3, \dots, v_n$.

Melhor Posição Pessoal ($pbest_i$) → A melhor posição (solução) encontrada pela partícula i, até o momento.

Melhor Posição Global ($gbest$) → A melhor posição encontrada por qualquer partícula em todo o enxame até o momento.

Equação de Atualização da Velocidade:

$$v_i(t+1) = w.v_i(t) + c_i.r_i.\left(pbest_i(t) - x_i(t)\right) + c_2.r_2.(gbest(t) - x_i(t)$$

- $v_i(t+1)$: A nova velocidade da partícula i para a próxima iteração.
- $w \cdot v_i(t)$: **Termo de Inércia.**
  - w (Peso de Inércia): É um coeficiente que controla a tendência da partícula de manter sua direção e velocidade atuais. Este termo representa o *momentum* da partícula.
- $c_1 \cdot r_1 \cdot (pbest_i(t) - x_i(t))$: **Componente Cognitivo (Pessoal).**
  - c_1 (Coeficiente de Aceleração Cognitivo): Pondera a atração da partícula em direção à sua própria melhor posição histórica.
  - r_1: Número aleatório uniformemente distribuído entre [0, 1] para introduzir estocasticidade na busca.
  - $(pbest_i(t) - x_i(t))$: É a "memória" ou "confiança" da partícula em si mesma.

- **$c_2 \cdot r_2 \cdot (gbest(t) - x_i(t))$: Componente Social (Global).**
  - c_2: Coeficiente de Aceleração Social: Pondera a atração da partícula em direção à melhor posição encontrada pelo enxame inteiro (*gbest*).
  - r_2: Um segundo número aleatório uniforme [0, 1].
  - $gbest(t) - x_i(t)$: Direciona a partícula para o melhor ponto conhecido pelo grupo. É a "influência social" ou a "inteligência coletiva".

Equação de Atualização da Posição →

$$x_i(t+1) = x_i(t) + v_i(t+1)$$

$x_i(t+1)$: A nova posição (nova solução candidata) da partícula i.

$x_i(t)$: A posição atual.

$v_i(t+1)$: a nova velocidade calculada.

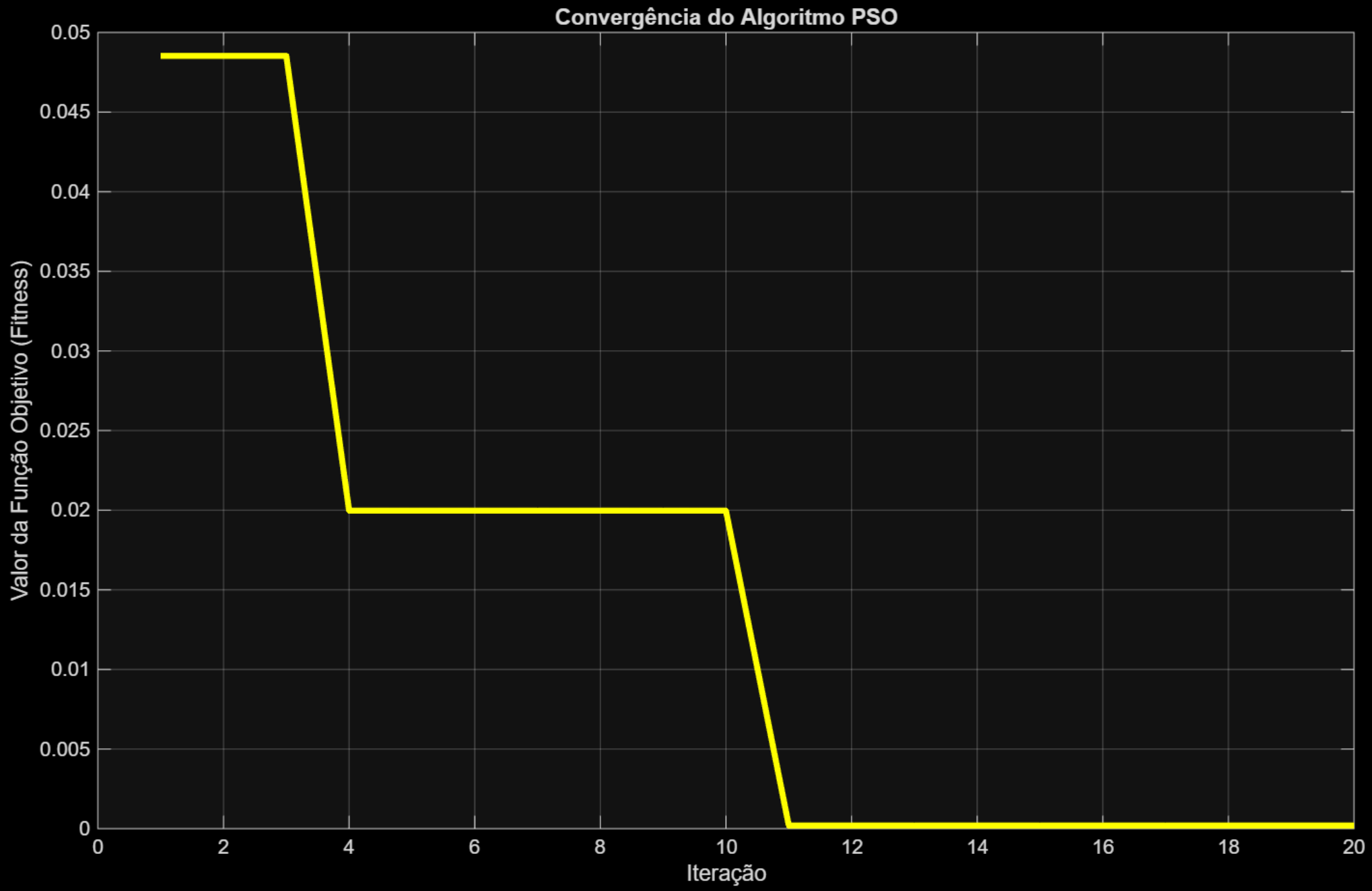
Modelagem Computacional

Função Objetivo: $f(x)=(x-15)^2$.

Intervalo de observação → $x = [0; 10]$.

O objetivo é encontrar o valor de x que minimiza esta função. Sabe-se que a resposta exata é $x = 15$, sendo $f(15) = 0$.

Valor ótimo:
x = 14.986069
f(x) = 1.940718e-04
Mínimo.


Convergência do Algoritmo PSO
Valor da Função Objetivo (Fitness)
Iteração
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
2
4
6
8
10
12
14
16
18
20

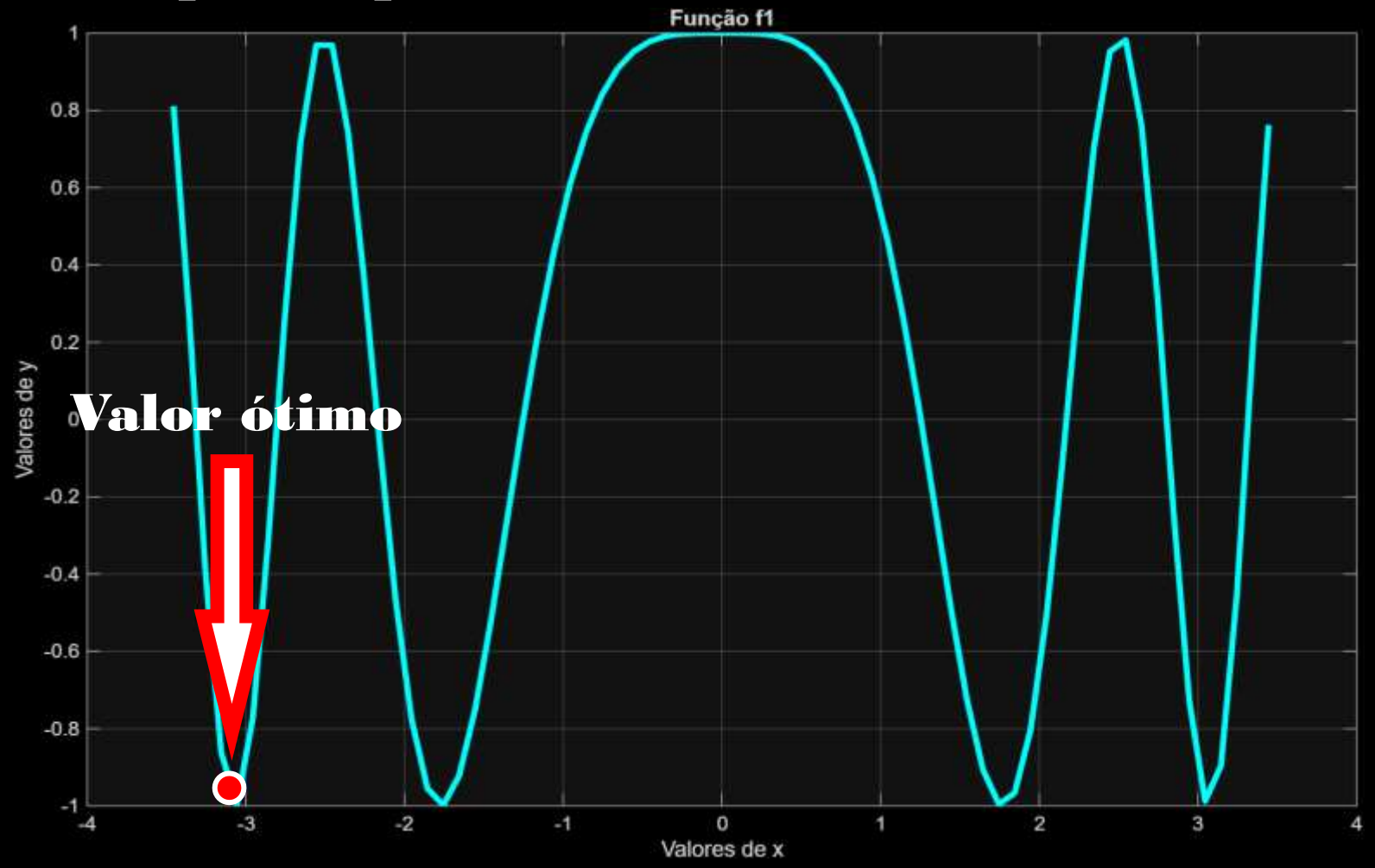
Modelagem Computacional

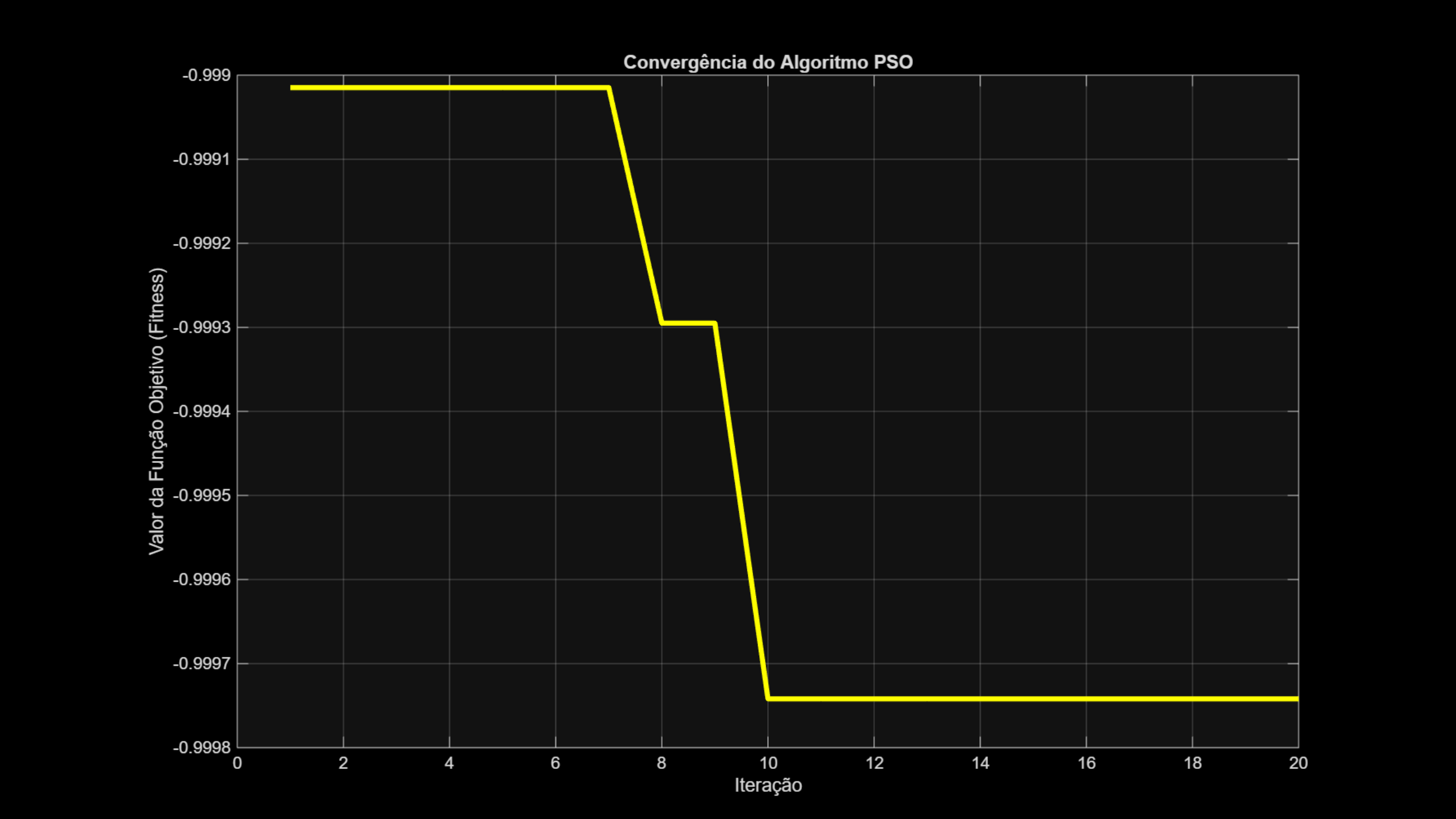
Função Objetivo: $f(x) = cos(x)^2$.

Intervalo de observação → $x = [-\pi; \pi]$.

O objetivo é encontrar o melhor valor para x e para f(x).

Valor ótimo:
x = -3.069590.
f(x) = -9.999971e-01
Mínimo.




Convergência do Algoritmo PSO
Valor da Função Objetivo (Fitness)
-0.999
-0.9991
-0.9992
-0.9993
-0.9994
-0.9995
-0.9996
-0.9997
-0.9998
0
2
4
6
8
10
12
14
16
18
20
Iteração

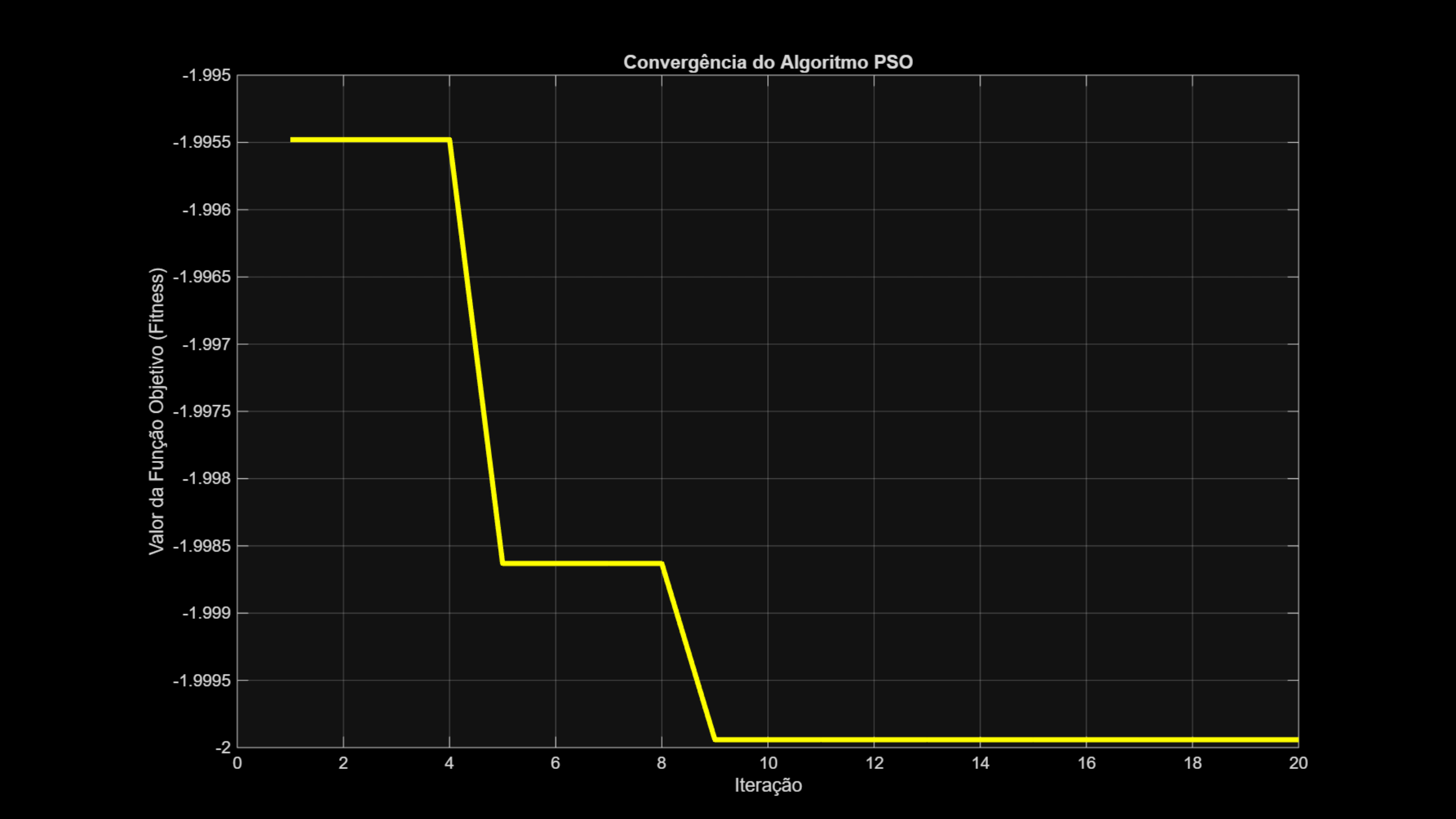
Modelagem Computacional

Função Objetivo: $f(x) = 2*sen(e^x)$.

Intervalo de observação → $x = [-\pi; 2\pi/3]$.

O objetivo é encontrar o melhor valor para x e para f(x).

Valor ótimo:
x = 0.451586.
f(x) = 2.00000e-00
Máximo.


Convergência do Algoritmo PSO
Valor da Função Objetivo (Fitness)
Iteração
-1.995
-1.9955
-1.996
-1.9965
-1.997
-1.9975
-1.998
-1.9985
-1.999
-1.9995
-2
0
2
4
6
8
10
12
14
16
18
20

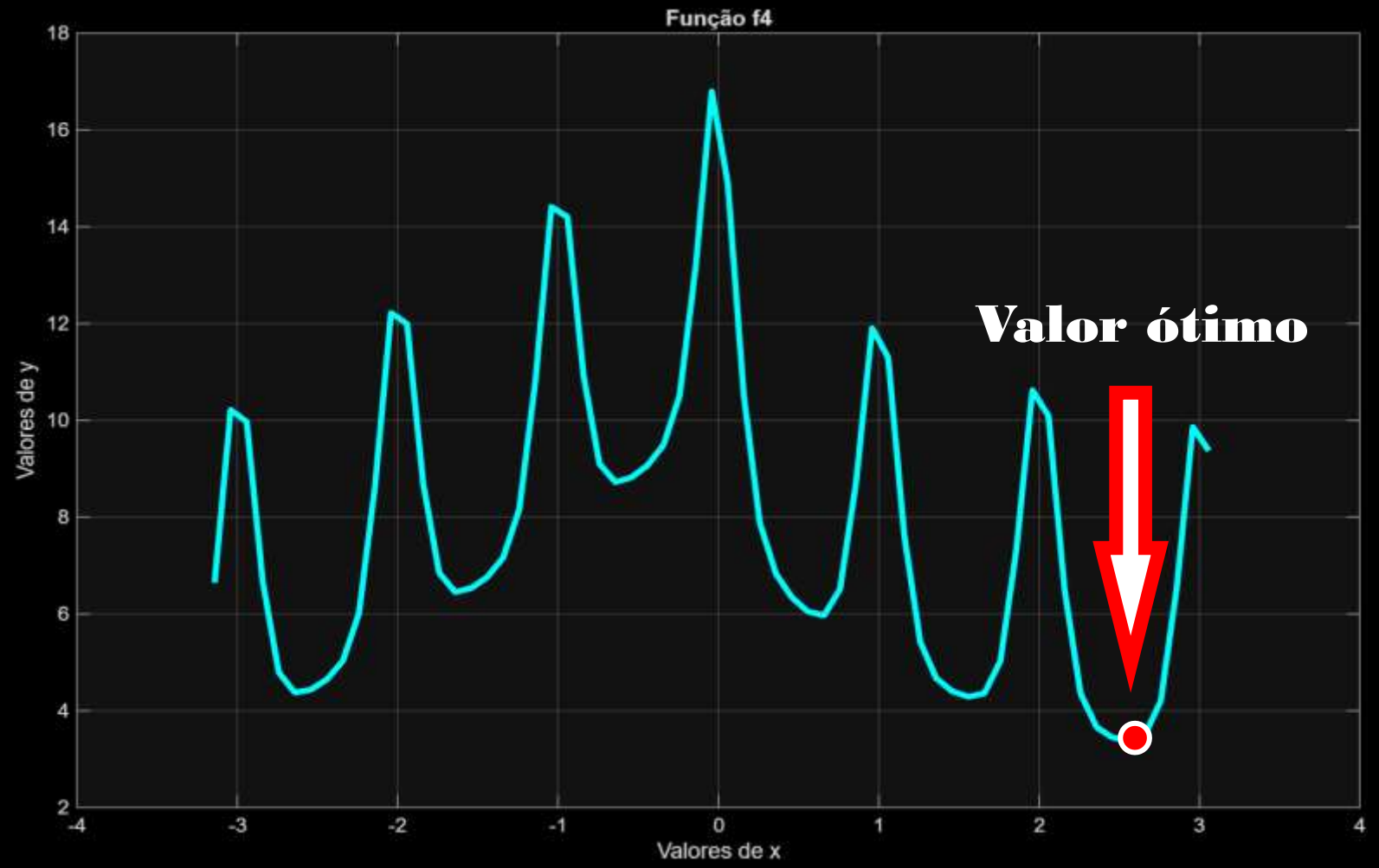
Modelagem Computacional

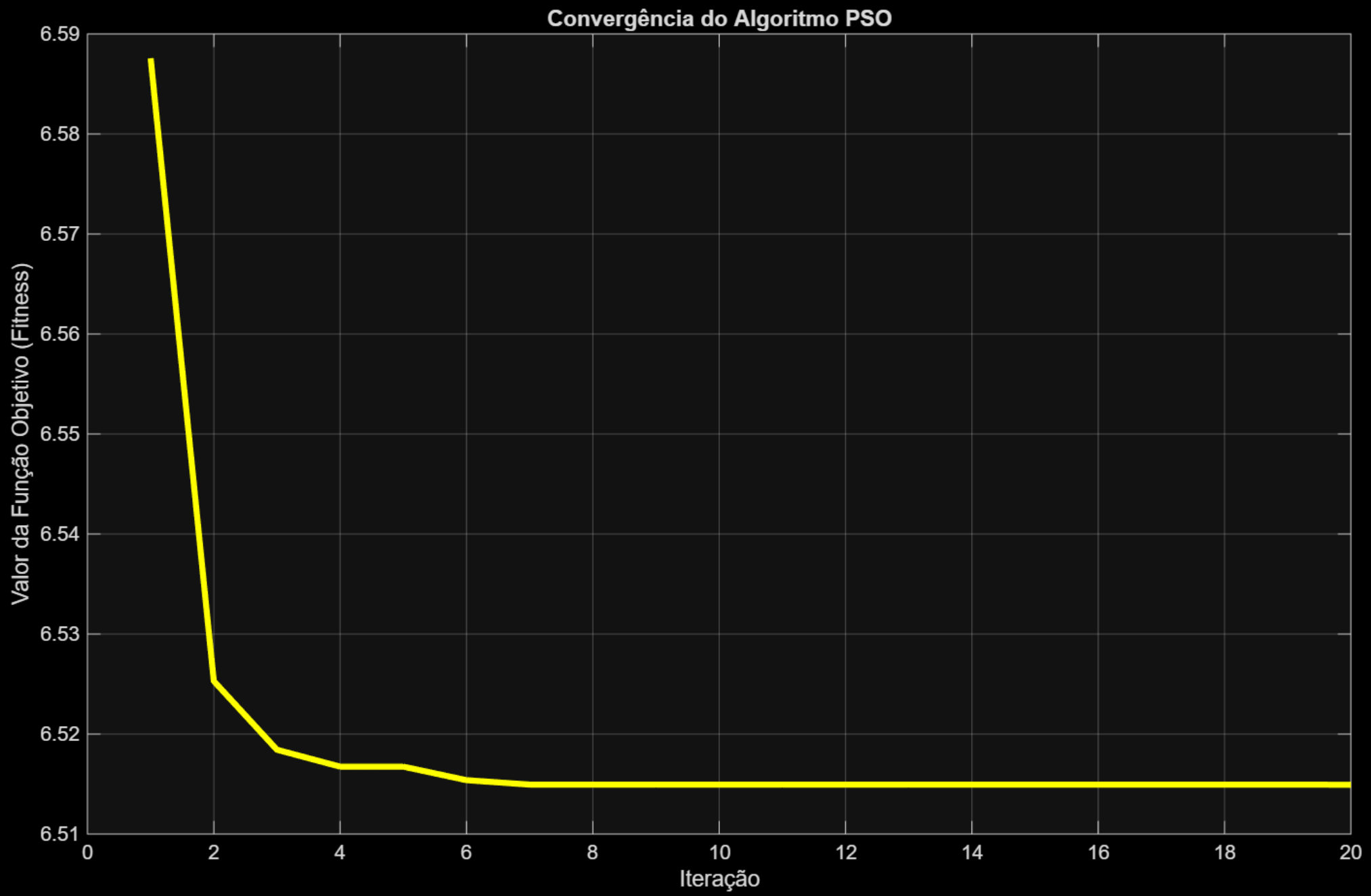
Função Objetivo: $f(x) = 10e^{-0.5\sqrt{2x}} + e^{2*cos(2\pi x)}$

Intervalo de observação → $x = [-\pi; \pi]$.

O objetivo é encontrar o melhor valor para x e para f(x).

Valor ótimo:
x = 2.548294.
f(x) = 4.42338e+00
Máximo.




Convergência do Algoritmo PSO
Valor da Função Objetivo (Fitness)
Iteração
6.59
6.58
6.57
6.56
6.55
6.54
6.53
6.52
6.51
0
2
4
6
8
10
12
14
16
18
20


MATLAB R2025a

PSO:
Código principal

```
% ========================================================================
% 22ª Semana Nacional de Ciência e Tecnologia - SNCT
% Minicurso: ALGORITMO DE OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS - PSO
% Professor: Dr. Denis Costa
% Coordenação: Bacharelado em Ciência e Tecnologia
% ========================================================================
clear all; clc; close all;
disp('=======================================')
disp('  Otimização por Enxame de Partículas')
disp('=======================================')

%% 1. Declarando a Função Objetivo
% Defina aqui a função que você deseja minimizar.
% O comando '.^' é usado para garantir que a função funcione com vetores.
funcao_objetivo = @(x) (x - 15).^2;
%funcao_objetivo = @(x) cos(x.^2);
%funcao_objetivo = @(x) 1*(2*sin(exp(x)));
%funcao_objetivo = @(x) 10*exp(0.5*sqrt(2*x)) +...
%                    exp(2*(cos(2*pi*x)));
%% 2. Parâmetros do PSO
n_particulas = 30;      % Número de partículas no enxame
max_iter = 20;          % Número máximo de iterações

% Coeficientes do PSO
w = 0.8;                % Peso de inércia
c1 = 1.5;               % Coeficiente cognitivo (pessoal)
c2 = 1.5;               % Coeficiente social (global)

% Limites do espaço de busca [limite_inferior, limite_superior]
limite_inferior = -3;
limite_superior = 3;

```

```
%% 3. Inicialização do Enxame
% Inicializa as posições e velocidades das partículas de forma aleatória

% Posição inicial (coluna de vetores)
posicao = limite_inferior + (limite_superior - limite_inferior) *...
          rand(n_particulas, 1);

% Velocidade inicial (pode ser zero ou aleatória)
velocidade = zeros(n_particulas, 1);

% Calcula o fitness (valor da função objetivo) inicial para cada partícula
fitness = funcao_objetivo(posicao);

% Inicializa a melhor posição pessoal (pbest) de cada partícula
pbest_posicao = posicao;
pbest_fitness = fitness;

% Inicializa a melhor posição global (gbest) do enxame
[gbest_fitness, gbest_idx] = min(pbest_fitness);
gbest_posicao = pbest_posicao(gbest_idx);

% Vetor para armazenar o melhor fitness de cada iteração (para o gráfico)
historico_gbest = zeros(max_iter, 1);

%% 4. Loop Principal do PSO
fprintf('Iniciando otimização com PSO...\n');

for t = 1:max_iter

    % Para cada partícula no enxame...
    for i = 1:n_particulas

        % Gera números aleatórios
        r1 = rand();
        r2 = rand();

        % a) Atualiza a velocidade da partícula
        velocidade(i) = w * velocidade(i) ...
                      + c1 * r1 * (pbest_posicao(i) - posicao(i)) ...
                      + c2 * r2 * (gbest_posicao - posicao(i));

        % b) Atualiza a posição da partícula
        posicao(i) = posicao(i) + velocidade(i);

        % c) Garante que a partícula permaneça dentro dos limites de busca
        if posicao(i) > limite_superior
            posicao(i) = limite_superior;
        end
        if posicao(i) < limite_inferior
            posicao(i) = limite_inferior;
        end

        % d) Avalia o fitness da nova posição
        novo_fitness = funcao_objetivo(posicao(i));

        % e) Atualiza a melhor posição pessoal (pbest) se necessário
        if novo_fitness < pbest_fitness(i)
            pbest_fitness(i) = novo_fitness;
            pbest_posicao(i) = posicao(i);
        end

    end % fim do loop das partículas

    % f) Atualiza a melhor posição global (gbest) para a iteração atual
    [melhor_fitness_iter, melhor_idx_iter] = min(pbest_fitness);
    if melhor_fitness_iter < gbest_fitness
        gbest_fitness = melhor_fitness_iter;
        gbest_posicao = pbest_posicao(melhor_idx_iter);
    end

    % Armazena o melhor fitness da iteração para o gráfico de convergência
    historico_gbest(t) = gbest_fitness;

    % Opcional: Exibe o progresso
    fprintf('Iteração %d: Melhor Fitness = %f\n', t, gbest_fitness);

end % fim do loop principal

%% 5. apresentação dos Resultados Encontrados
fprintf('\nOtimização concluída!\n');
fprintf('=====================================\n');
fprintf('Melhor posição encontrada (x): %f\n', gbest_posicao);
fprintf('Valor da função nesse ponto (mínimo): %e\n', gbest_fitness);
fprintf('=====================================\n');

%% 6. Gráfico de Convergência
figure
plot(1:max_iter, historico_gbest, 'y-', 'LineWidth', 3)
title('Convergência do Algoritmo PSO')
xlabel('Nº de iterações')
ylabel('Valor da Função Objetivo (Fitness)')
grid on
```

PSO: Código auxiliar

```
% ========================================================================
% 22ª Semana Nacional de Ciência e Tecnologia - SNCT
% Minicurso: ALGORITMO DE OTIMIZAÇÃO POR ENXAME DE PARTÍCULAS - PSO
% Professor: Dr. Denis Costa
% Coordenação: Bacharelado em Ciência e Tecnologia
% ========================================================================
clear all; clc; close all;
disp('=========================================')
disp('Representação gráfica da Função Objetivo ')
disp('=========================================')
% Gráfico das Funções Objetivos
x1 = [0:0.1:30];
x2 = [-1.1*pi:0.1:1.1*pi];
x3 = [-pi:0.1:2*pi/3];
x4 = [-pi:0.1:pi];
f1 = (x1 - 15).^2;
f2 = cos(x2.^2);
f3 =  2*sin(exp(x3));
f4 = 10*exp(-0.5*sqrt(2*x4)) + exp(2*(cos(2*pi*x4)));
plot(x4,f4,'c-', 'LineWidth', 3)
xlabel('Valores de x')
ylabel('Valores de y')
title('Função f14')
grid on
```

Obrigado!

“Se você está disposto a mover uma pedra, esteja preparado para lidar com a (provável) cobra que está sob ela”.

Niels Bohr.