

Trabalho prático 05

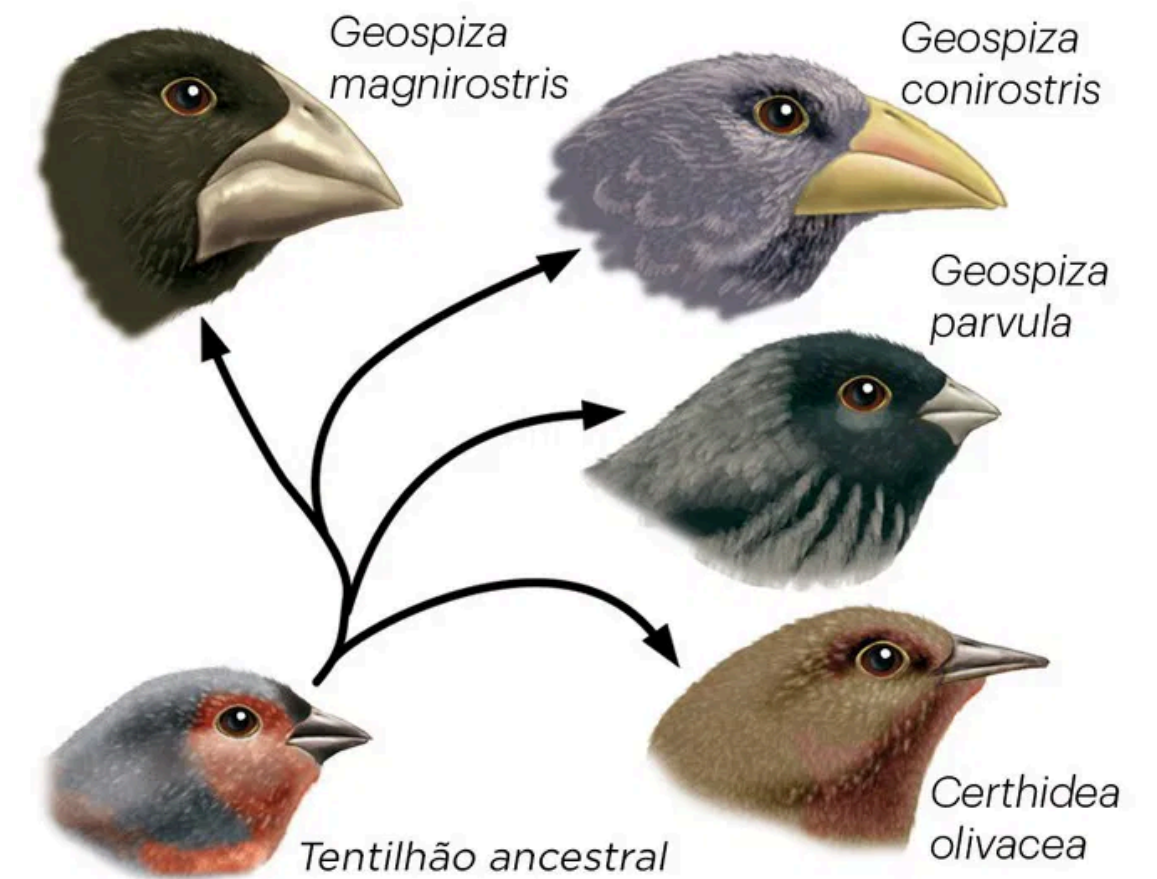
ALGORITMOS GENÉTICOS

Maria Eduarda Ferreira da Silva
Vitória Christie Amaral Santos



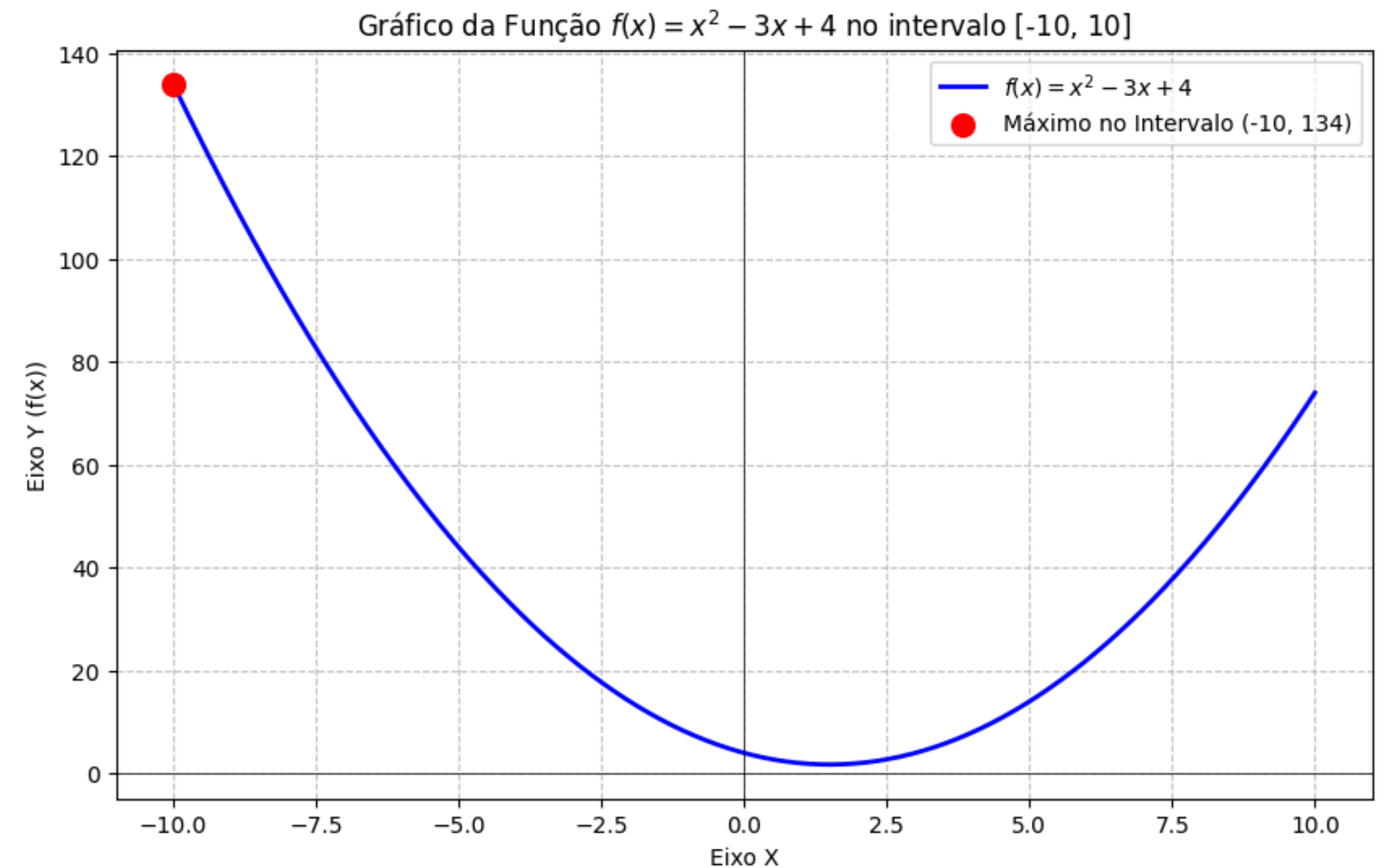
O CONCEITO DE AG

- O que é? Técnica de busca probabilística.
- **Inspiração:** Teoria da Evolução (Darwin).
- **Diferencial:**
 - Não usa derivadas.
 - Trabalha com uma **população** de soluções simultâneas.
 - Utiliza operadores de cruzamento e mutação.



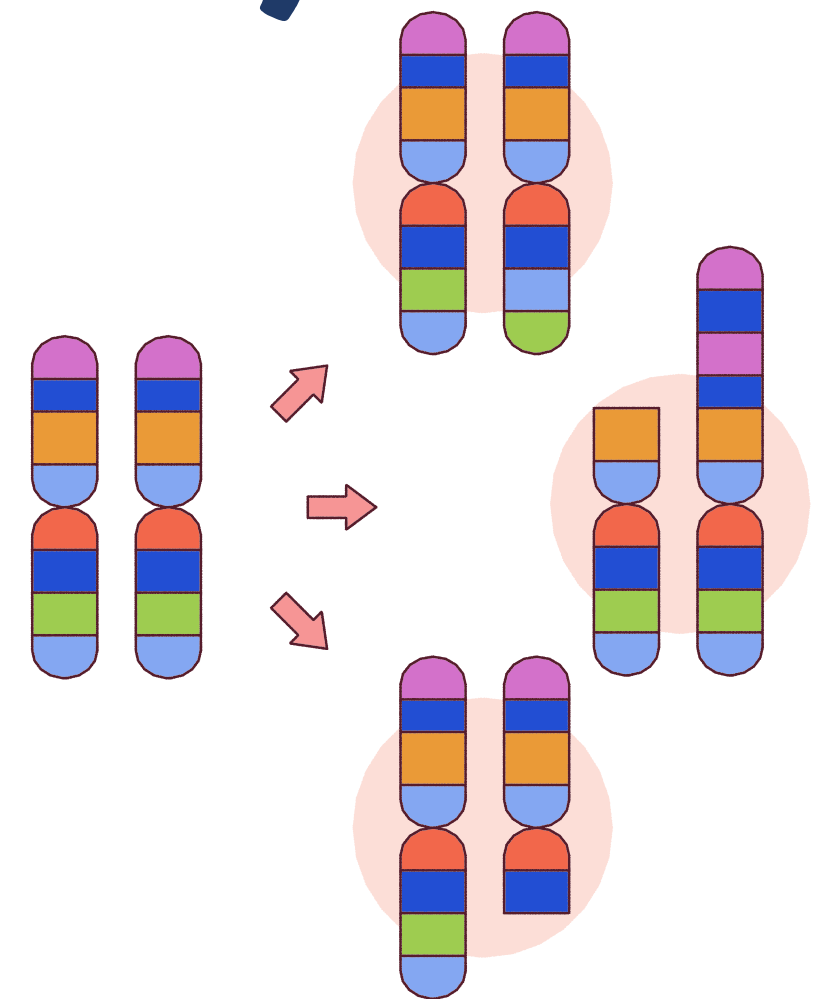
O PROBLEMA MATEMÁTICO

- Função Objetivo: $f(x) = x^2 - 3x + 4$
- Intervalo de Busca: $x \in [-10, +10]$
- **Objetivo:** Encontrar o valor MÁXIMO.
- Análise do Alvo:
- Se $x = 10 \rightarrow f(10) = 74$
- Se $x = -10 \rightarrow f(-10) = 134$
- **Meta:** O algoritmo deve encontrar $x = -10$



REPRESENTAÇÃO (GENÓTIPO VS. FENÓTIPO)

- Codificação: Binária (Vetor de Bits).
- Cromossomo: 10 bits (Genótipo).
- Espaço de Busca: $2^{10} = 1024$ combinações.
- Mapeamento:
- 0000000000 -> -10 (Mínimo)
- 1111111111 -> +10 (Máximo)



DECODIFICAÇÃO

- Desafio: Converter Bits -> Número Real.
- Fórmula de Normalização:

$$Real = Min + \frac{Decimal}{maxDecimal} \times (Max - Min)$$

- Onde:
 - Min = -10
 - Max = +10
 - MaxDecimal = 1023 ($2^{10} - 1$)

FUNÇÃO DE APTIDÃO (FITNESS)

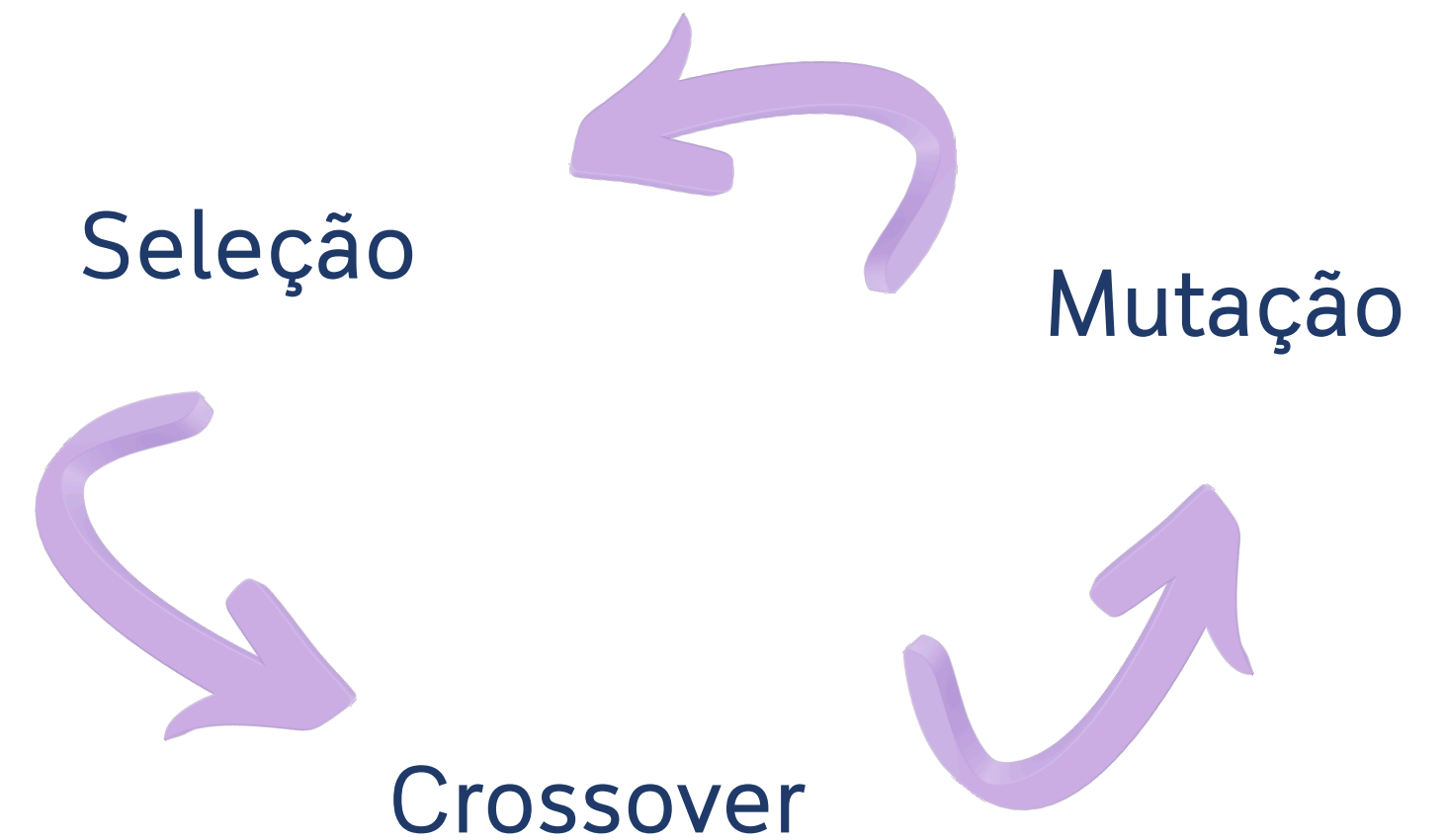
- Fitness (Aptidão): Medida de qualidade da solução.
- Neste problema: $\text{Fitness} = f(x)$.
- Critério:
- Maior valor = Mais chance de reproduzir.
- Menor valor = Provável descarte.



O CICLO EVOLUTIVO (OPERADORES)

Ciclo de Gerações:

- Seleção: Quem vai ser pai?
- Crossover: Mistura genética (70%).
- Mutação: Diversidade (1%).
- Modelo Generacional: População antiga é 100% substituída.
- Processo repetido por 20 gerações.



SELEÇÃO E CROSSOVER

1. Seleção por Torneio:

- Escolhe 2 aleatórios → vence o melhor.
- Vantagem: Pressão seletiva controlada.

```
def selecao_torneio(populacao):  
    """  
    Seleciona um indivíduo por torneio.  
    Escolhe 'k' indivíduos aleatórios e retorna o melhor.  
    """  
    competidores = random.sample(populacao, TORNEIO_SIZE)  
    melhor = max(competidores, key=calcular_fitness)  
    return melhor
```

2. Crossover (1 ponto):

- Taxa: 70%.
- Corte e troca de “caudas” dos vetores.
- Objetivo: Combinar boas características.

```
def crossover(pai1, pai2):  
    """  
    Aplica crossover de um ponto com base na TAXA_CROSSOVER.  
    """  
    if random.random() < TAXA_CROSSOVER:  
        ponto_corte = random.randint(1, BITS_POR_INDIVIDUO - 1)  
        filho1 = pai1[:ponto_corte] + pai2[ponto_corte:]  
        filho2 = pai2[:ponto_corte] + pai1[ponto_corte:]  
        return filho1, filho2  
    else:  
        return pai1[:], pai2[:] # Retorna cópias se não houver crossover
```

MUTAÇÃO

3. Mutação (Bit-flip):

- Taxa: 1% por bit.
- Ação: Inverte o bit ($0 \rightarrow 1$ ou $1 \rightarrow 0$).
- Evita estagnação (máximos locais).
- Reintroduz material genético perdido.

```
def mutacao(individuo):  
    """  
    Aplica mutação bit-flip com base na TAXA_MUTACAO.  
    """  
    novo_ind = []  
    for bit in individuo:  
        if random.random() < TAXA_MUTACAO:  
            novo_ind.append(1 if bit == 0 else 0) # Inverte o bit  
        else:  
            novo_ind.append(bit)  
    return novo_ind
```

```
# 1. População Inicial
populacao = criar_populacao(TAMANHO_POPULACAO)
historico_fitness_melhor = []
historico_fitness_media = [] # NOVA LISTA

print(f"--- Iniciando AG com {TAMANHO_POPULACAO} indivíduos por {NUM_GERACOES} gerações ---\n")

for geracao in range(NUM_GERACOES):
    nova_populacao = []

    # Avaliação de TODOS os indivíduos
    valores_fitness = [calcular_fitness(ind) for ind in populacao]

    # Estatísticas da Geração
    melhor_fitness = max(valores_fitness)
    media_fitness = sum(valores_fitness) / len(populacao) # CÁLCULO DA MÉDIA

    # Identifica o melhor indivíduo para mostrar o X
    indice_melhor = valores_fitness.index(melhor_fitness)
    melhor_ind = populacao[indice_melhor]
    melhor_x = binario_para_real(melhor_ind)

    # Salva no histórico para o gráfico
    historico_fitness_melhor.append(melhor_fitness)
    historico_fitness_media.append(media_fitness)
```

```

# Print mais completo
print(f"Geração {geracao+1}: Melhor X = {melhor_x:.4f} | Melhor Fit = {melhor_fitness:.4f} | Média Fit = {media_fitness:.4f}")

# Loop para criar nova geração (Reprodução)
while len(nova_populacao) < TAMANHO_POPULACAO:
    pai1 = selecao_torneio(populacao)
    pai2 = selecao_torneio(populacao)

    f1, f2 = crossover(pai1, pai2)

    f1 = mutacao(f1)
    f2 = mutacao(f2)

    nova_populacao.append(f1)
    if len(nova_populacao) < TAMANHO_POPULACAO:
        nova_populacao.append(f2)

populacao = nova_populacao

# --- RESULTADO FINAL ---
valores_fitness = [calcular_fitness(ind) for ind in populacao]
melhor_geral_fit = max(valores_fitness)
melhor_geral_ind = populacao[valores_fitness.index(melhor_geral_fit)]
melhor_geral_x = binario_para_real(melhor_geral_ind)

print("\n" + "="*40)
print("RESULTADO FINAL")
print(f"Melhor X encontrado: {melhor_geral_x:.5f}")
print(f"Valor Máximo da Função: {melhor_geral_fit:.5f}")
print("="*40)

```

COMPARAÇÃO DE CENÁRIOS

Cenário 1: População inicial (4 indivíduos)

- Resultado: Instável.
- Problema: Baixa diversidade genética.
- Depende da sorte inicial.

```
--- Iniciando AG com 4 indivíduos por 5 gerações ---

Geração 1: Melhor X = 9.5308 | Melhor Fit = 66.2436 | Média Fit = 34.9417
Geração 2: Melhor X = 9.5308 | Melhor Fit = 66.2436 | Média Fit = 60.9302
Geração 3: Melhor X = 9.5308 | Melhor Fit = 66.2436 | Média Fit = 65.9304
Geração 4: Melhor X = 9.5308 | Melhor Fit = 66.2436 | Média Fit = 66.2436
Geração 5: Melhor X = 9.5308 | Melhor Fit = 66.2436 | Média Fit = 57.7609

=====
RESULTADO FINAL
Melhor X encontrado: 9.53079
Valor Máximo da Função: 66.24362
=====
```

COMPARAÇÃO DE CENÁRIOS

Cenário 2: População ajustada (30 indivíduos)

- Resultado: Consistente.
- Maior cobertura do espaço de busca.
- Convergência atingida plenamente.

```
--- Iniciando AG com 30 indivíduos por 5 gerações ---
```

```
Geração 1: Melhor X = -8.5533 | Melhor Fit = 102.8183 | Média Fit = 31.7771  
Geração 2: Melhor X = -8.5533 | Melhor Fit = 102.8183 | Média Fit = 45.9678  
Geração 3: Melhor X = -9.1789 | Melhor Fit = 115.7886 | Média Fit = 69.7452  
Geração 4: Melhor X = -9.5894 | Melhor Fit = 124.7257 | Média Fit = 95.1094  
Geração 5: Melhor X = -9.7849 | Melhor Fit = 129.1000 | Média Fit = 105.2392
```

```
=====
```

```
RESULTADO FINAL
```

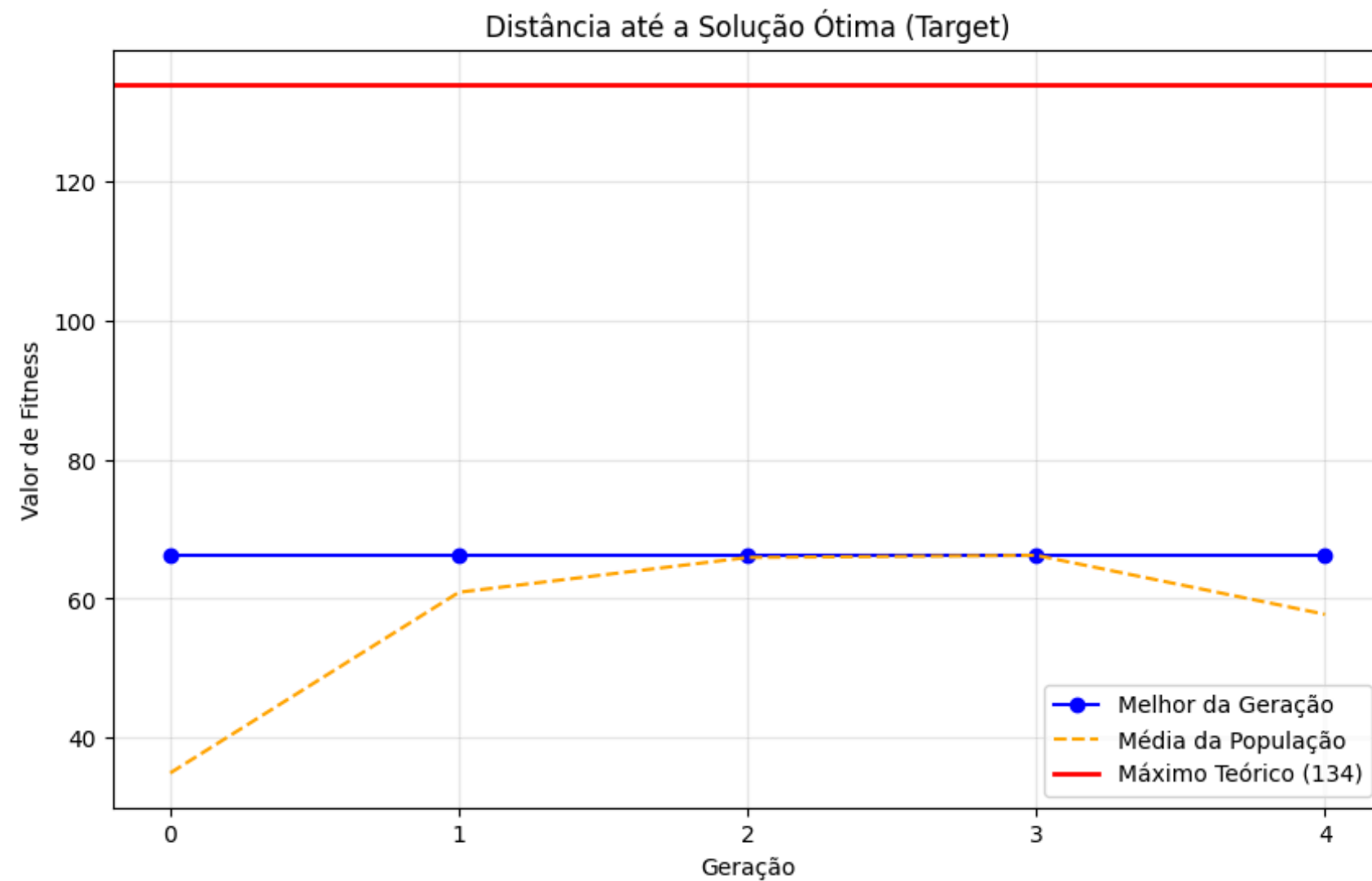
```
Melhor X encontrado: -9.80450
```

```
Valor Máximo da Função: 129.54164
```

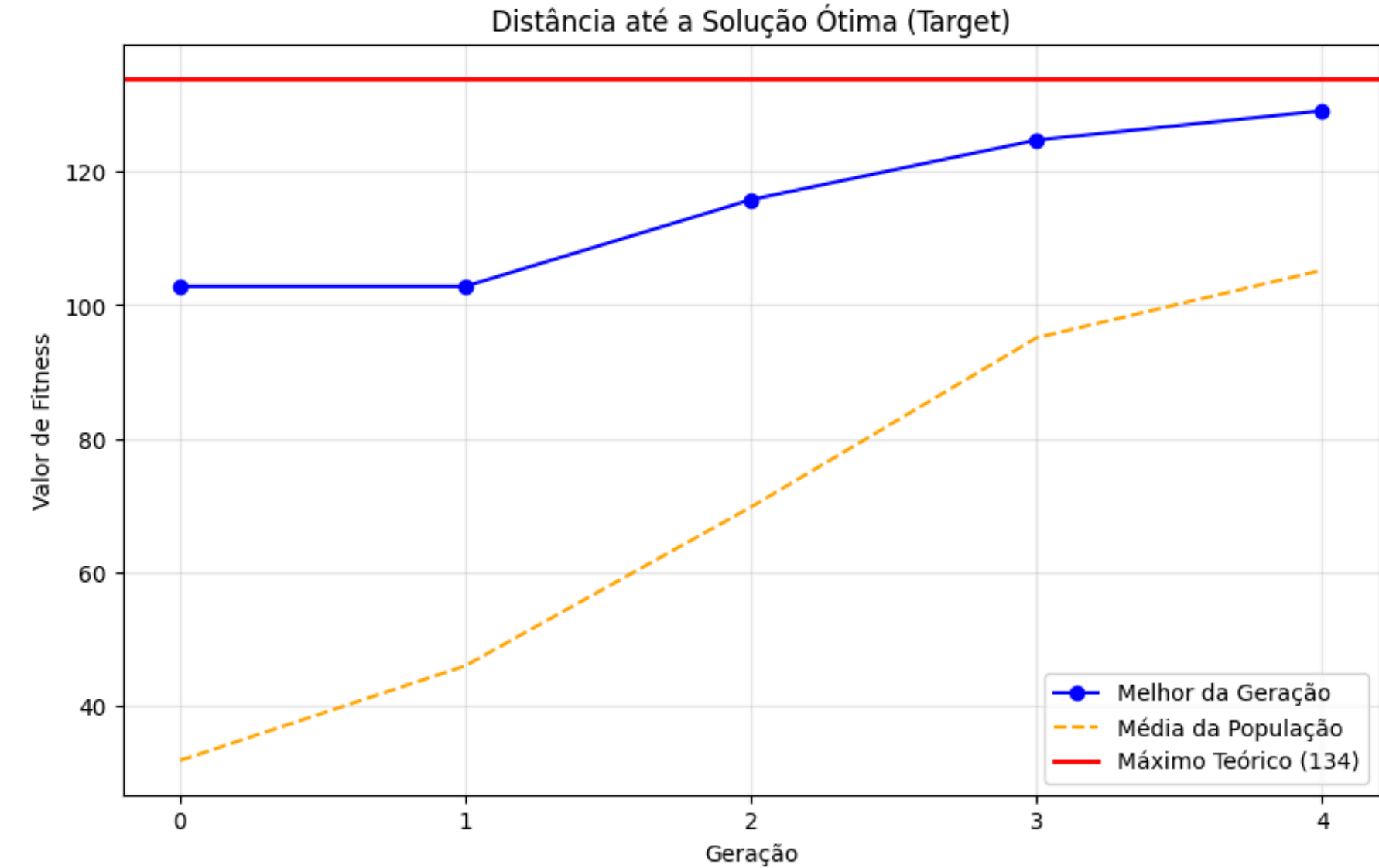
```
=====
```

ANÁLISE VISUAL

4 indivíduos



30 indivíduos



ANÁLISE CRÍTICA (O FENÔMENO DA PERDA)

- Máximo encontrado: $x \approx -10$
- Fitness: 129
- Oscilação no topo
- Causa: ausência de elitismo
- Crossover/Mutação podem piorar o campeão

CONCLUSÃO

- AG maximizou a função sem derivadas.
- Representação binária funcionou no intervalo real.
- Aumentar população ($4 \rightarrow 30$) foi crucial.



OBRIGADA!

LINK DA APRESENTAÇÃO

[HTTPS://YOUTU.BE/4AYEYYDCZBA](https://youtu.be/4AYEYYDCZBA)