

Tópicos Especiais de Otimização - Técnicas Inteligentes – 210115-IC

Programa de Pós-Graduação em Engenharia Elétrica / UFJF

Prof. Leonardo Willer de Oliveira/ Prof. Ivo Chaves da Silva Junior

Relatório de Técnicas Inteligentes:

Comparação entre da otimização

via ecolocalização de morcegos e algoritmos genéticos

Ajuste dos coeficientes de uma equação, três dimensões

Doutorando: Thainan Santos Theodoro

Matrícula: 103080112

thainan.theodoro@engenharia.ufjf.br

Juiz de Fora, 15 de maio de 2017

Sumário

[1. Introdução 5](#_Toc482627445)

[2. Problema 5](#_Toc482627446)

[3. Algoritmo BAT –Otimização via ecolocalização de morcegos 6](#_Toc482627447)

[3.1. Passo 1 – Escolhas dos parâmetros 6](#_Toc482627451)

[3.2. Passo 2 – Inicialização aleatória 6](#_Toc482627452)

[3.3. Passo 3 – Avaliação inicial 7](#_Toc482627453)

[3.4. Passo 4 – Melhor morcego 7](#_Toc482627454)

[3.5. Passo 5 – Velocidade e deslocamento 7](#_Toc482627455)

[3.6. Passo 6 e 7 – Busca local 8](#_Toc482627456)

[3.7. Passo 8 – Perturbação em uma dimensão 9](#_Toc482627457)

[3.8. Passo 9 e 10 – Busca Global 9](#_Toc482627458)

[3.9. Passo 11 – Melhor morcego 9](#_Toc482627459)

[4. Testes no BAT Algorithm 9](#_Toc482627460)

[4.1. Variação de λ e α em função do número de iterações 9](#_Toc482627465)

[4.2. Testes propostos 10](#_Toc482627466)

[4.3. Testes para limites [-10,10] 11](#_Toc482627467)

[4.3.1. Variação do número de iterações 11](#_Toc482627468)

[4.3.2. Variação do número de morcegos 12](#_Toc482627469)

[4.3.3. Variação de α 13](#_Toc482627470)

[4.3.4. Variação de λ 14](#_Toc482627471)

[4.4. Testes para limites [3,10] 16](#_Toc482627472)

[4.4.1. Variação do número de iterações 16](#_Toc482627473)

[4.4.2. Variação do número de morcegos 17](#_Toc482627474)

[4.4.3. Variação de α 18](#_Toc482627475)

[4.4.4. Variação de λ 19](#_Toc482627476)

[5. Algoritmos Genéticos 21](#_Toc482627477)

[5.1. Parâmetros a serem alterados 21](#_Toc482627479)

[5.2. Testes iniciais com limites [-10,10] 21](#_Toc482627480)

[5.2.1. Variação de indivíduos 22](#_Toc482627481)

[5.2.2. Variação de gerações 23](#_Toc482627482)

[5.2.3. Variação do elitismo 24](#_Toc482627483)

[5.2.4. Variação do crossover 25](#_Toc482627484)

[5.3. Testes com limites [3, 10] 26](#_Toc482627485)

[5.3.1. Variação de indivíduos 26](#_Toc482627486)

[5.3.2. Variação de gerações 27](#_Toc482627487)

[5.3.3. Variação do elitismo 28](#_Toc482627488)

[5.3.4. Variação do crossover 28](#_Toc482627489)

[5.4. Testes com seleção 30](#_Toc482627490)

[5.5. Testes com cruzamento 31](#_Toc482627491)

[6. Conclusão 32](#_Toc482627492)

[7. Bibliografia 33](#_Toc482627493)

Índice de figuras

[Figura 1 – Fluxograma de funcionamento da otimização via BAT Algorithm. 2](#_Toc482574213)

[Figura 2 – Taxa de emissão em função da iteração. 2](#_Toc482574214)

[Figura 3 – Análise do aumento do número de iterações, limite [-10,10]. 2](#_Toc482574215)

[Figura 4 – Evolução da FOB para os números de iteração selecionados, limite [-10,10]. 2](#_Toc482574216)

[Figura 5 – Análise do aumento do número de morcegos, limite [-10,10]. 2](#_Toc482574217)

[Figura 6 – Evolução da FOB para o número de morcegos selecionados, limite [-10,10]. 2](#_Toc482574218)

[Figura 7 – Análise da variação de α (esquerda), resultados dos testes 9 a 11 (direita), limite [-10,10]. 2](#_Toc482574219)

[Figura 8 – Evolução da FOB para valores de α selecionados, limite [-10,10]. 2](#_Toc482574220)

[Figura 9 – Análise da variação de λ (esquerda), resultados dos testes 12 a 14 (direita), limite [-10,10]. 2](#_Toc482574221)

[Figura 10 – Evolução da FOB para valores de λ selecionados, limite [-10,10]. 2](#_Toc482574222)

[Figura 11 – Curva do problema e curva de ajuste 2](#_Toc482574223)

[Figura 12 – Curva do problema e curva de ajuste 2](#_Toc482574224)

[Figura 13 – Análise do aumento do número de iterações, limite [3,10] 2](#_Toc482574225)

[Figura 14 – Evolução da FOB para os números de iteração selecionados, limite [3,10]. 2](#_Toc482574226)

[Figura 15 – Análise do aumento do número de morcegos, limite [3,10]. 2](#_Toc482574227)

[Figura 16 – Evolução da FOB para o número de morcegos selecionados, limite [3,10]. 2](#_Toc482574228)

[Figura 17 – Análise da variação de α (esquerda), resultados dos testes 9 a 11 (direita), limite [3,10]. 2](#_Toc482574229)

[Figura 18 – Evolução da FOB para valores de α selecionados, limite [3,10]. 2](#_Toc482574230)

[Figura 19 – Análise da variação de λ (esquerda), resultados dos testes 12 a 14 (direita), limite [3,10]. 2](#_Toc482574231)

[Figura 20 – Evolução da FOB para valores de λ selecionados, limite [3,10]. 2](#_Toc482574232)

[Figura 21 – Curva do problema e curva de ajuste 2](#_Toc482574233)

[Figura 22 – Curva do problema e curva de ajuste 2](#_Toc482574234)

[Figura 23 – Variação do número de indivíduos no AG, testes 1 a 4, limite [-10,10]. 2](#_Toc482574235)

[Figura 24 – Evolução da FOB para 10 indivíduos. 2](#_Toc482574236)

[Figura 25 – Evolução da FOB para 100 indivíduos. 2](#_Toc482574237)

[Figura 26 – Variação do número de gerações no AG, testes 5 a 7, limite [-10,10]. 2](#_Toc482574238)

[Figura 27 – Evolução da FOB para 30 gerações. 2](#_Toc482574239)

[Figura 28 – Evolução da FOB para 75 gerações. 2](#_Toc482574240)

[Figura 29 – Variação do elitismo no AG, testes 8 a 10, limite [-10,10]. 2](#_Toc482574241)

[Figura 30 – Evolução da FOB para 0% de elitismo. 2](#_Toc482574242)

[Figura 31 – Evolução da FOB para 10% de elitismo. 2](#_Toc482574243)

[Figura 32 – Variação da porcentagem de cruzamento no AG, testes 11 a 13, limite [-10,10]. 2](#_Toc482574244)

[Figura 33 – Evolução da FOB para 50% crossover. 2](#_Toc482574245)

[Figura 34 – Evolução da FOB para 90% de crossover. 2](#_Toc482574246)

[Figura 35 – Variação do número de indivíduos no AG, testes 1 a 4, limite [3,10]. 2](#_Toc482574247)

[Figura 36 – Evolução da FOB para 30 gerações. 2](#_Toc482574248)

[Figura 37 – Evolução da FOB para 75 gerações. 2](#_Toc482574249)

[Figura 38 – Variação do número de gerações no AG, testes 5 a 7, limite [3,10]. 2](#_Toc482574250)

[Figura 39 – Evolução da FOB para 30 gerações. 2](#_Toc482574251)

[Figura 40 – Evolução da FOB para 75 gerações. 2](#_Toc482574252)

[Figura 41 – Variação do elitismo no AG, testes 8 a 10, limite [3,10]. 2](#_Toc482574253)

[Figura 42 – Evolução da FOB para 0% de elitismo. 2](#_Toc482574254)

[Figura 43 – Evolução da FOB para 10% de elitismo. 2](#_Toc482574255)

[Figura 44 – Variação da porcentagem de cruzamento no AG, testes 11 a 13, limite [3,10]. 2](#_Toc482574256)

[Figura 45 – Evolução da FOB para 50% crossover. 2](#_Toc482574257)

[Figura 46 – Evolução da FOB para 90% de crossover. 2](#_Toc482574258)

[Figura 47 – Variação do tipo de seleção no AG, testes 1 a 5, limite [-10,10]. 2](#_Toc482574259)

[Figura 48 – Variação do tipo de cruzamento no AG, testes 1 a 4, limite [-10,10]. 2](#_Toc482574260)

# Introdução

Este relatório tem a finalidade estabelecer uma comparação de desempenho entre o algoritmo de otimização por eco localização de morcegos (BAT, *Bat* *Algorithm*” [1]) e o Algoritmo Genético (GA, *Genetic Algorithm* [2]) no ajuste dos coeficientes de uma determinada equação. O algoritmo BAT foi desenvolvido inteiramente, enquanto que o AG foi baseado na toolbox do MatLab. Baterias de testes são feitos para comparar o desempenho dos algoritmos na obtenção da resposta quanto ao número de morcegos/população formigas, número de iterações/gerações dentre outros.

# Problema

O objetivo do problema é encontrar os coeficientes que melhor se ajustam na Equação (1) considerando os valores de , e da Tabela 1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | () | Tabela – Dados do problema.   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | -1 | 0 | 1 | 2 | 4 | 5 | 5 | 6 | |  | -2 | -1 | 0 | 1 | 1 | 2 | 3 | 4 | |  | 13 | 11 | 9 | 4 | 11 | 9 | 1 | -1 | |

A estratégia de cálculo da função objetivo, para os dois métodos BAT e AG, foi o cálculo do RMSE (“*Root Mean Square Error*”), dado por [3],

|  |  |
| --- | --- |
|  | () |

Onde é o número de testes, é a saída desejada e é o valor encontrado com a solução BAT/AG.

# Algoritmo BAT –Otimização via ecolocalização de morcegos

O fluxograma de aplicação do algoritmo de otimização via ecolocalização de morcegos é mostrado na Figura 1 [2].



Figura – Fluxograma de funcionamento da otimização via BAT Algorithm [2] adaptado.

Nas secções seguintes serão discutidos cada passo e como foram implementados pelo autor.



## Passo 1 – Escolhas dos parâmetros

A tabela a seguir resume os parâmetros escolhidos inicialmente. Na fase de testes estes parâmetros serão variados a fim de comparação da performance.

Tabela – Parâmetros usados no algoritmo.

|  |  |
| --- | --- |
| Parâmetro | Valor |
| N° de morcegos | 15 |
| α, Taxa de decréscimo da amplitude | 0,0768 |
| λ, taxa de emissão de pulso | 0,9532 |

## 

## Passo 2 – Inicialização aleatória

Para a inicialização das possíveis soluções (morcegos) foi usada a função “unifrnd” que define uma matriz [m,n] aleatória uniforme dentre limites superior e inferior. Conforme código abaixo. A mesma matriz de morcegos iniciais foi usada durantes os testes.

|  |
| --- |
| d = 3; % Dimensão  N = 15; % N° de morcegos  Linf = -10; % Limite inferior  Lsup = 10; % Limite superior  SOLUCOES0 = unifrnd(Linf,Lsup, N, d); |

## Passo 3 – Avaliação inicial

Neste passo é obtida o valor referente à FOB para cada morcego separadamente. Inicialmente são calculados os 8 valores de (cada coluna da Tabela 1) com a equação (1). Posteriormente, calcula-se o erro médio quadrático usando a equação (2). Para isso foi criada a função FOB\_ga conforme mostra o trecho.

|  |
| --- |
| function [ fob ] = FOB\_ga( x )  X = [x(1) x(2) x(3)]; % Problema de 3 dimensões  restricoes = [1 1 1 1 1 1 1 1; % A  -1 0 1 2 4 5 5 6; % B  -2 -1 0 1 1 2 3 4; % C  13 11 9 4 11 9 1 -1];  % Cálculo dos 8 valores yi  y1 = restricoes ([1 2 3],:)'\*X;  % Cálculo do erro  erro = y1 - restricoes(end,:)';  % Cálculo do erro médio quadrático  RMSE = sqrt(erro.^2)/length(erro);  RMSE = mean(fob);  end |

## Passo 4 – Melhor morcego

O melhor morcego, , é aquele que tem o menor RMSE. Optou-se por armazenar o valor da FOB juntamente com as soluções, na última coluna, por exemplo:

Tabela – Armazenamento de morcegos e FOB.

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | FOB |
|  |  |  |  |
|  |  |  |  |
| ... | ... | ... | ... |
|  |  |  |  |

O código a seguir mostra a forma de implementação da seleção do melhor morcego.

|  |
| --- |
| [index, lin] = find(MORCEGOS(:,end) == min(MORCEGOS(:,end)) );  MELHOR = MORCEGOS(index, :); |

## Passo 5 – Velocidade e deslocamento

Os passos 5 ao 12 estão em dois laços, o primeiro diz respeito ao número de iterações que foi o critério de parada escolhido. O segundo diz respeito ao número de morcegos, ou seja, os passos são executados para um morcego de cada vez. Assim, para todos os trechos, subentende-se um laço como o que segue:

|  |
| --- |
| for m = 1:N  ...  ...  end |

Um dos pontos chaves da estratégia do algoritmo é calcular a velocidade e o deslocamento dos morcegos em função do melhor morcego, encontrando o morcego temporário. São empregadas as seguintes equações:

|  |  |
| --- | --- |
|  | () |

Onde é a iteração, e são a velocidade na iteração atual e na passada, é o melhor morcego da iteração, é o morcego em análise, é o morcego em análise. O trecho extraído do código mostra a forma de implementação.

|  |
| --- |
| % Cálculo da velocidade  v(m,:) = v(m,:) + MELHOR(1:d) - MORCEGOS(m,1:d) )\*unifrnd(0,1);  % Atualização deslocamento  TEMP(1:d) = MORCEGOS(m,1:d) + v(m,:); |

## Passo 6 e 7 – Busca local

A busca global tem maior probabilidade de ocorrer no começo do processo e menor no final, já a busca local tem maior probabilidade de ocorrer no fim do processo, portanto essas buscas são complementares. As equações a seguir mostram a forma de obtenção da taxa de emissão de pulso e a alteração da amplitude da onda sonora respectivamente.

|  |  |
| --- | --- |
|  | () |

Onde e são a taxa de emissão de pulso e a amplitude sonora na iteração seguinte, taxa de aumento da emissão de pulso da onda sonora e é a taxa de decréscimo da amplitude sonora. A Figura 2 a seguir mostra a evolução de e em função do número de iterações.



Figura – Taxa de emissão em função da iteração.

Para a busca local é sorteado um número de [0,1], se for maior que é executada a alteração do melhor morcego em uma pequena amplitude em direções aleatórias [-1,1] em função da amplitude média, conforme:

|  |  |
| --- | --- |
|  | () |

O trecho do código a seguir mostra a forma de implementação da busca local:

|  |
| --- |
| if rand < Solucao.r(m)  ep = unifrnd(-1, 1, 1, d);  TEMP(1:d) = MELHOR(1:d) + ep\*mean(A);  end |

## Passo 8 – Perturbação em uma dimensão

Esse passo possibilita que o método busque novas soluções e evite ficar preso em máximos e mínimos locais. Ele consiste de uma perturbação aleatória em uma determinada dimensão. Optou-se por aplicar essa perturbação em 80% das vezes, pois, quando se tem pequena quantidade de morcegos a solução muitas vezes é perdida. O código abaixo ilustra a forma de implementação da perturbação.

|  |
| --- |
| if rand < 0.8  TEMP(1,randsample(1:d,1)) = unifrnd( Linf, Lsup);  end |

## Passo 9 e 10 – Busca Global

A busca global consiste na atualização de e conforme (4), além do valor do morcego atual, conforme:

|  |  |
| --- | --- |
|  | () |

As atualizações ocorrem um número sorteado aleatoriamente for menor que ou a FOB do morcego temporário for melhor que a FOB do morcego atual. O trecho do código a seguir ilustra a operação.

|  |
| --- |
| if (rand < A(m))||(TEMP(1,end)<=MORCEGOS(m,end))  MORCEGOS(m,:) = TEMP; % Atualização do morcego  r(m) = 1-exp(-y\*t); % atualização da taxa de emissao  A(m) = a\*A(m); % atualização da amplitude  end |

## Passo 11 – Melhor morcego

A atualização do melhor morcego da iteração é idêntica ao passo 4.

# Testes no BAT Algorithm



## Variação de λ e α em função do número de iterações

É interessante que a taxa de decréscimo de amplitude e a taxa de emissão de pulso (α e λ) sejam escolhidas em função do número de iterações a fim de preservar a forma característica das curvas, como na Figura 2. Dessa forma, definiu-se um ponto para cada curva em função do número de iterações de acordo com a Tabela 4.

Tabela – Pontos escolhidos para adequação das curvas.

|  |  |  |
| --- | --- | --- |
|  | Amplitude da onda  (Ai) | Taxa de emissão  (ri) |
| Iteração |  |  |
| Valor |  | 0,9 |

Sendo assim, uma forma de implementar α e λ em função das iterações é manipular as Equações (4) a fim de chegar em:

|  |  |
| --- | --- |
|  | () |

O trecho abaixo mostra a implementação.

|  |
| --- |
| iter = 100; % N° de iterações  lambda = -log(1 - 0.9)/(0.3\*iter); % (30%.iter , 0.9)  alpha = 10^((log10(0.1))/(0.5\*iter)); % (50%.iter , 0.1) |

Assim, no decorrer do trabalho α e λ serão definidas em termos de porcentagem por simplificação.

## Testes propostos

Os 14 testes propostos na Tabela 5 mostram tem o objetivo de avaliar cada um dos 4 parâmetros do algoritmo de morcegos: n° de iterações, n° de morcegos, taxa de decréscimo de amplitude α e a taxa de emissão de pulso λ.

Tabela – Testes propostos para o algoritmo de morcegos.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Teste | Iterações | N | α | λ | Objetivos  (Impactos na FOB e otimalidade) |
| 1 | 50 | 40 | 50% | 30% | Observar o impacto do aumento do n° de iterações (\*) |
| 2 | 100 |
| 3 | 150 |
| 4 | 200 |
| 5 | 100 | 10 | 50% | 30% | Observar o impacto do aumento do n° de morcegos. |
| 6 | 40 |
| 7 | 70 |
| 8 | 100 |
| 9 | 100 | 40 | 25% | 30% | Observar o aumento da proporção da taxa de decréscimo de amplitude. |
| 10 | 50% |
| 11 | 75% |
| 12 | 100 | 40 | 50% | 15% | Observar o aumento da taxa de emissão de pulso. |
| 13 | 30% |
| 14 | 45% |
| (\*) | Quando n° de iterações varia, α e λ variam também, mantendo a proporção. | | | | |
|

Inicialmente os testes foram feitos nos limites propostos de [3,10] para as variáveis A, B, e C. No entanto, após discussões com o professor e alunos da disciplina, achou-se por bem ampliar o limite inferior, sendo assim, serão avaliados também os resultados dos mesmos testes porém com limites [-10, 10] para os coeficientes.

## Testes para limites [-10,10]

A Tabela 6 mostra os resultados encontrados para os limites [-10,10]. Cada um dos 14 testes foi executado 50 vezes, o melhor valor de FOB foi escolhido. A fim de refletir a otimalidade é apresentado também o valor médio da FOB, assim como o tempo médio de simulação.

Tabela – Resultado dos testes para o BAT Algorithm, limites [3,10].

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | Iterações | N | α | λ | A | B | C | FOB | Tempo | FOB média | Tempo médio |
| 1 | 50 | 40 | 50% | 30% | 3,9231 | 3,3371 | -6,2364 | 0,0750 | 0,2948 | 0,0782 | 0,3125 |
| 2 | 100 | 3,9147 | 3,3393 | -6,2385 | 0,0750 | 0,7200 | 0,0760 | 0,7600 |
| 3 | 150 | 3,8684 | 3,3484 | -6,2425 | 0,0748 | 0,9325 | 0,0755 | 0,9625 |
| 4 | 200 | 3,8714 | 3,3581 | -6,2582 | 0,0750 | 1,2293 | 0,0756 | 1,2570 |
| 5 | 100 | 10 | 50% | 30% | 4,0697 | 3,2875 | -6,1997 | 0,0754 | 0,1568 | 0,0896 | 0,1529 |
| 6 | 40 | 3,9085 | 3,3381 | -6,2358 | 0,0749 | 0,8187 | 0,0757 | 0,6299 |
| 7 | 70 | 3,9920 | 3,3036 | -6,2031 | 0,0750 | 1,1801 | 0,0756 | 1,1157 |
| 8 | 100 | 3,8747 | 3,3477 | -6,2456 | 0,0749 | 1,5441 | 0,0752 | 1,5731 |
| 9 | 100 | 40 | 25% | 30% | 3,8617 | 3,3533 | -6,2466 | 0,0748 | 0,6216 | 0,0760 | 0,6470 |
| 10 | 50% | 3,9140 | 3,3316 | -6,2383 | 0,0752 | 0,7350 | 0,0759 | 0,6283 |
| 11 | 75% | 4,0314 | 3,2983 | -6,2098 | 0,0754 | 0,6091 | 0,0766 | 0,6179 |
| 12 | 100 | 40 | 50 | 15% | 3,8116 | 3,3869 | -6,2879 | 0,0750 | 0,6377 | 0,0757 | 0,6293 |
| 13 | 30% | 3,9093 | 3,3238 | -6,2111 | 0,0751 | 0,6203 | 0,0758 | 0,6374 |
| 14 | 45% | 3,8638 | 3,3677 | -6,2679 | 0,0750 | 0,7369 | 0,0758 | 0,6737 |

### Variação do número de iterações

A Figura 3 mostra o resultado dos testes 1 a 4 nos quais se aumenta o número de iterações de 50, 100, 150 e 200 respectivamente. Pode-se perceber que o melhor valor de cada teste não se altera muito nos 4 testes, enquanto que o valor médio (se traduzindo em otimalidade) diminui consideravelmente. O tempo de simulação, obviamente, sobe com o número de iterações.



Figura – Análise do aumento do número de iterações, limite [-10,10].

A Figura 4Figura 3 mostra a evolução da FOB para os quatro valores de iterações escolhidos, considerando os 5 primeiros testes. Em todos os quadros os valores chegam bem próximos aos valores ótimos antes do número de iterações acabar. A configuração eleita como a melhor do conjunto foi a (2) com 100 iterações, na qual se observa uma boa relação: precisão, otimalidade e eficiência computacional.



Figura – Evolução da FOB para os números de iteração selecionados, limite [-10,10].

### Variação do número de morcegos

O gráfico da Figura 5 apresenta a evolução da FOB para os testes de 5 a 8 com o aumento do número de morcegos de 10, 40, 70 e 100. Da mesma forma, o melhor valor da bateria de testes de cada configuração não teve muita alteração, porém, o valor médio caiu consideravelmente de 10 para 40 morcegos. No entanto, a queda não continuou proporcionalmente com o aumento de morcegos diferentemente do tempo.



Figura – Análise do aumento do número de morcegos, limite [-10,10].

A Figura 6 mostra a evolução da FOB para os 5 primeiros valores dos testes. Evidentemente a pior evolução é a com 10 morcegos, para os outros números a evolução é boa, chegam ao ponto ótimo entre 40 e 60 iterações. A única desvantagem de um grande número de iterações então, foi o tempo de simulação. Logo, a configuração eleita no grupo com a melhor performance (precisão, otimalidade e eficiência computacional) foi a (6) com 40 morcegos.



Figura – Evolução da FOB para o número de morcegos selecionados, limite [-10,10].

### Variação de α

Os testes de 9 a 11 têm objetivo de avaliar a variação α em 25%, 50% e 75% do número de iterações, que usando a Equação (4) chega-se à , e . A Figura 7 mostra dois gráficos, o primeiro (esquerda) apresenta a evolução da amplitude sonora com o número de iterações considerando os valores de *α* do teste. Nota-se que para o mecanismo de busca entra, de maneira geral, menos vezes na busca global, enquanto que para ocorre o oposto, a busca global é realizada mais vezes.



Figura – Análise da variação de α (esquerda), resultados dos testes 9 a 11 (direita), limite [-10,10].

O segundo gráfico da Figura 7 apresenta a evolução da FOB durante os testes 9 a 11. Com o aumento de α o valor da melhor solução piora (azul). No entanto o valor médio melhora, ou seja, as soluções estão mais próximas umas das outras mostrando que α e λ têm um bom compromisso. O tempo de simulação diminui com o aumento de α, pois com isso o mecanismo entra menos vezes na busca global.

A Figura 8 mostra a evolução da FOB para os três valores de α para o teste. O menor valor de α chega mais rápido ao valor ótimo, já os maiores valores permitem mais tempo de busca global e a evolução da FOB é mais lenta. Escolheu-se o teste (10) como configuração de melhor desempenho, mantendo o compromisso (precisão, otimalidade e eficiência computacional).



Figura – Evolução da FOB para valores de α selecionados, limite [-10,10].

### Variação de λ

Os testes de 12 a 14 têm objetivo de avaliar a variação λ em 15%, 30% e 45% do número de iterações, que usando a Equação (4) chega-se à , e . A Figura 9 mostra dois gráficos, o primeiro (esquerda) apresenta a evolução da taxa de emissão com o número de iterações considerando os valores de λ do teste. Nota-se que para o mecanismo de busca entra, de maneira geral, menos vezes na busca local, enquanto que para ocorre o oposto, a busca global é realizada mais vezes.



Figura – Análise da variação de λ (esquerda), resultados dos testes 12 a 14 (direita), limite [-10,10].

O segundo gráfico da Figura 9Figura 7 apresenta a evolução da FOB durante os testes 12 a 14. Com a diminuição de λ o valor da melhor solução melhora (azul). Isso porque de maneira geral o método tem mais tempo de fazer uma melhor procura global, assim, a busca local fica mais refinada. Analogamente, o valor médio quase não se altera, mostrando que otimalidade é quase constante.

A Figura 10 mostra a evolução da FOB para os três valores de λ para o teste. De forma geral, chega mais rápido à solução, enquanto tem uma maior diferença nas primeiras iterações. Escolheu-se o teste (13) como configuração de melhor desempenho, mantendo o compromisso (precisão, otmalidade e eficiência computacional).



Figura – Evolução da FOB para valores de λ selecionados, limite [-10,10].

De forma geral, a melhor solução encontrada foi a do teste (3) com FOB de 0,0748. A Figura 10 mostra a curva de original dada pela Tabela 1 e a curva com ajuste do teste (3), percebe-se que o erro é relativamente pequeno.

|  |  |
| --- | --- |
|  |  |
| Figura – Curva do problema e curva de ajuste  (visão a) [-10,10]. | Figura – Curva do problema e curva de ajuste  (visão b) [-10,10]. |

## Testes para limites [3,10]

A Tabela 7 mostra os resultados encontrados para os limites [3,10]. Cada um dos 14 testes foi executado 50 vezes, o melhor valor de FOB foi escolhido. A fim de refletir a otimalidade é apresentado também o valor médio da FOB, assim como o tempo médio de simulação.

Tabela – Resultado dos testes para o BAT Algorithm, limites [3,10].

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | Iterações | N | α | λ | A | B | C | FOB | Tempo | FOB média | Tempo médio |
| 1 | 50 | 40 | 50% | 30% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,4854 | 1,9219 | 0,2880 |
| 2 | 100 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5496 | 1,9219 | 0,5380 |
| 3 | 150 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,8312 | 1,9219 | 0,8029 |
| 4 | 200 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 1,1290 | 1,9219 | 1,0703 |
| 5 | 100 | 10 | 50% | 30% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,1287 | 1,9229 | 0,1361 |
| 6 | 40 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5137 | 1,9219 | 0,5362 |
| 7 | 70 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,9673 | 1,9219 | 0,9620 |
| 8 | 100 | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 1,6653 | 1,9219 | 1,3645 |
| 9 | 100 | 40 | 25% | 30% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5532 | 1,9219 | 0,5462 |
| 10 | 50% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5447 | 1,9219 | 0,5370 |
| 11 | 75% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5308 | 1,9219 | 0,5407 |
| 12 | 100 | 40 | 50 | 15% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,6091 | 1,9219 | 0,5573 |
| 13 | 30% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,5378 | 1,9219 | 0,5411 |
| 14 | 45% | 3,0000 | 3,0000 | 3,0000 | 1,9219 | 0,4973 | 1,9219 | 0,5291 |

Os valores encontrados para os coeficientes A, B e C foram exatamente iguais para todos os testes, obviamente os valores de FOB também. Apenas o tempo sofreu alteração com a variação de parâmetros.

### Variação do número de iterações

A Figura 13 mostra a evolução da FOB para os testes (1) a (4) para o aumento do número de iterações de 50, 100, 150 e 200. Apenas se pode ver a alteração no tempo de simulação, pois, nem os coeficientes, nem a FOB ou seu valor médio sofreram iterações. Obviamente, com um maior número de iterações o tempo de simulação aumenta.

de

Figura – Análise do aumento do número de iterações, limite [3,10]

A Figura 14 mostra a evolução da FOB para os 4 valores de iteração escolhidos. O método encontra a solução ótima em poucas iterações para todos os testes.



Figura – Evolução da FOB para os números de iteração selecionados, limite [3,10].

### Variação do número de morcegos

O gráfico da Figura 15 Figura 5apresenta a evolução da FOB para os testes de 5 a 8 com o aumento do número de morcegos de 10, 40, 70 e 100. Da mesma forma, a FOB não teve alterações, apenas o tempo de simulação. Quanto maior o número de morcegos maior o tempo de simulação.



Figura – Análise do aumento do número de morcegos, limite [3,10].

A Figura 16 mostra a evolução da FOB para os 5 primeiros valores dos testes. O método encontra o valor ótimo em poucas iterações para todos os testes. Para 10 morcegos, no entanto, algumas vezes a solução demora mais, mas ainda assim encontra a solução.



Figura – Evolução da FOB para o número de morcegos selecionados, limite [3,10].

### Variação de α

Os testes de 9 a 11 têm objetivo de avaliar a variação α em 25%, 50% e 75% do número de iterações, que usando a Equação (4) chega-se à , e . A Figura 17 mostra dois gráficos, o primeiro (esquerda) apresenta a evolução da amplitude sonora com o número de iterações considerando os valores de *α* do teste. Nota-se que para o mecanismo de busca entra, de maneira geral, menos vezes na busca global, enquanto que para ocorre o oposto, a busca global é realizada mais vezes.



Figura – Análise da variação de α (esquerda), resultados dos testes 9 a 11 (direita), limite [3,10].

O segundo gráfico da Figura 17 mostra que a FOB não se alterou durante os testes 9 a 11. Porém o tempo melhorou para o teste (10), no qual, acontece a melhor proporção entre α e λ.

A Figura 18 mostra a evolução da FOB para os três valores de α para o teste. Não se percebe alteração visível entre as curvas.



Figura – Evolução da FOB para valores de α selecionados, limite [3,10].

### Variação de λ

Os testes de 12 a 14 têm objetivo de avaliar a variação λ em 15%, 30% e 45% do número de iterações, que usando a Equação (4) chega-se à , e . A Figura 19 mostra dois gráficos, o primeiro (esquerda) apresenta a evolução da taxa de emissão com o número de iterações considerando os valores de λ do teste. Nota-se que para o mecanismo de busca entra, de maneira geral, menos vezes na busca local, enquanto que para ocorre o oposto, a busca global é realizada mais vezes.



Figura – Análise da variação de λ (esquerda), resultados dos testes 12 a 14 (direita), limite [3,10].

O segundo gráfico da Figura 19 apresenta a evolução da FOB durante os testes 12 a 14. Com a diminuição de λ o tempo de simulação também diminui, porém, a o melho valor é idêntico durante a simulação.

A Figura 20 mostra a evolução da FOB para os três valores de λ para o teste. Não se percebe alteração visível entre as curvas.



Figura – Evolução da FOB para valores de λ selecionados, limite [3,10].

Impondo-se o limite [3,10] a curva não pode ser ajustada de forma adequada e o método não encontra melhor resultado que [3, 3 ,3], com erro médio quadrático de 1,9219. A Figura 21 mostra a curva de original dada pela Tabela 1 e a curva com ajuste, percebe-se que o erro é relativamente pequeno.

|  |  |
| --- | --- |
|  |  |
| Figura – Curva do problema e curva de ajuste  (visão a) [3,10]. | Figura – Curva do problema e curva de ajuste  (visão b) [3,10]. |

# Algoritmos Genéticos



## Parâmetros a serem alterados

A toolbox de Algoritmos Genéticos do MatLab possibilita uma grande gama de variações nos parâmetros do método. A Tabela 8 mostra as opções que serão alteradas nos testes propostos.

Tabela - Principais parâmetros da toolbox de Algoritmos Genéticos do MatLab.

|  |  |  |
| --- | --- | --- |
| Parâmetro | Função | Descrição |
| Tamanho da População | N | Maior: mais precisão maior tempo computacional;  Menor: Menos precisão menor tempo computacional. |
| Critério de parada | MaxGenerations  MaxTime  MaxStallGenerations  FunctionTolerance | N° de gerações;  Tempo máximo de computação;  Número de iterações estagnadas.  Média da variação da FOB nas gerações estagnadas. |
| Opções de reprodução | *EliteCount* | Especifica o número de indivíduos que são garantidos para sobreviver à próxima geração |
| CrossoverFraction | Fração da próxima geração que se reproduzirá. |
| Seleção | Stochastic uniform  (@selectionstochunif) | Cada pai corresponde a uma seção da linha de comprimento proporcional à sua expectativa. |
| Remainder (@selectionremainder) | Designa os pais deterministicamente da parte inteira do valor escalado de cada indivíduo e então usa a seleção de roleta na parte fracionária restante. |
| *Uniform (@selectionuniform)* | Seleciona os pais aleatoriamente a partir de uma distribuição uniforme (teste do algoritmo genético). |
| *Roulette (@selectionroulette)* | Simula uma roleta com a área de cada segmento proporcional à sua expectativa. Sorteio aleatório uma probabilidade igual à área. |
| *Tournament (@selectiontournament)* | Seleciona possíveis pais aleatoriamente, (número de participantes do torneio), em seguida escolhe-se o melhor indivíduo para ser um pai. |
| Cruzamento | *Scattered (@crossoverscattered)* | O cruzamento é realizado gene a gene por uma sequência aleatória. |
| *Single point (@crossoversinglepoint)* | Concatena genes dos pais segundo uma posição aleatória. |
| *Two point (@crossovertwopoint)* | Concatena genes dos pais segundo duas posições aleatórias. |
| *Intermediate (@crossoverintermediate)* | Cria filhos por uma média ponderada aleatória dos pais. |
| *Heuristic (@crossoverheuristic)* | Cria filhos aleatórios entre os pais. Mais perto do pai com melhor FOB e mais distante do pai com pior FOB. |
| *Arithmetic (@crossoverarithmetic)* | Filhos com média ponderada dos pais. |
| Mutação | *Uniform (mutationuniform)* | Mutação uniforme aleatória em uma taxa especificada da população. |
| *Adaptive Feasible (@mutationadaptfeasible)* | Mutações em direção à última geração bem-sucedida, respeitando os limites e as restrições lineares. |

## Testes iniciais com limites [-10,10]

Inicialmente, é necessário estabelecer uma configuração geral para aplicação dos testes. Para isso foi feita uma bateria de testes iniciais para definir: (*i*) tamanho da população, (*ii*) critério de parada e (*iii*) opções de reprodução. As configurações propostas para os testes iniciais estão na Tabela 9. Para os critérios de seleção, cruzamento e mutação são definidos os valores padrão:

* Seleção: *Roulette*;
* Cruzamento: *Single* *point*;
* Mutação: *Adaptive* *Feasible*, taxa 1%.

Tabela - Configurações propostas para os testes iniciais.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teste | Individuos | Gerações | Elitismo | Crossover |
| 1 | 10 | 50 | 5% | 70% |
| 2 | 40 |
| 3 | 70 |
| 4 | 100 |
| 5 | 40 | 30 | 5% | 70% |
| 6 | 50 |
| 7 | 75 |
| 8 | 40 | 50 | 0% | 70% |
| 9 | 5% |
| 10 | 10% |
| 11 | 40 | 50 | 5% | 50% |
| 12 | 70% |
| 13 | 90% |

Cada um dos 13 testes foi simulado 50 vezes, com limites de [-10,10] e os melhores resultados estão na Tabela 10. São mostrados também os valores médios da FOB e do tempo. O valor médio da FOB funciona como uma medida da otimalidade da configuração, já que o sistema é continuo e a probabilidade de resultados iguais é baixa. Quando *Flag* é ZERO quer dizer que o critério de parada foi o número máximo de iterações, quando é UM o método foi interrompido pois a FOB ficou estagnada.

Tabela – Resultados dos testes iniciais, limite [-10,10].

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | A | B | C | FOB | Tempo | Flag | FOB média | Tempo Médio |
| 1 | 3,5647 | 3,5860 | -6,5201 | 0,0761 | 0,2094 | 0 | 0,1190 | 0,2682 |
| 2 | 4,1770 | 3,2633 | -6,1893 | 0,0758 | 0,2091 | 0 | 0,1222 | 0,3333 |
| 3 | 3,9783 | 3,3069 | -6,2049 | 0,0749 | 0,5588 | 0 | 0,0840 | 0,3526 |
| 4 | 3,8907 | 3,3346 | -6,2246 | 0,0747 | 0,3565 | 0 | 0,0773 | 0,3891 |
| 5 | 3,9219 | 3,3258 | -6,2185 | 0,0749 | 0,1247 | 0 | 0,1142 | 0,1227 |
| 6 | 3,8299 | 3,3776 | -6,2739 | 0,0749 | 0,3461 | 0 | 0,1119 | 0,3981 |
| 7 | 4,9446 | 3,0176 | -6,0126 | 0,0780 | 0,4114 | 0 | 0,1016 | 0,5672 |
| 8 | 3,9255 | 3,3247 | -6,2192 | 0,0748 | 0,3058 | 0 | 0,1087 | 0,3232 |
| 9 | 3,8830 | 3,3442 | -6,2382 | 0,0748 | 0,3010 | 0 | 0,0960 | 0,3183 |
| 10 | 3,8634 | 3,3727 | -6,2760 | 0,0751 | 0,2689 | 0 | 0,1084 | 0,2866 |
| 11 | 3,8828 | 3,3387 | -6,2287 | 0,0747 | 0,3198 | 1 | 0,1156 | 0,2712 |
| 12 | 3,3753 | 3,7181 | -6,6714 | 0,0767 | 0,2875 | 0 | 0,1155 | 0,4143 |
| 13 | 4,7735 | 3,0454 | -6,0188 | 0,0793 | 0,2034 | 0 | 0,1135 | 0,2336 |
| 14 | 3,3700 | 3,7238 | -6,6790 | 0,0767 | 0,2091 | 0 | 0,1140 | 0,2169 |
| 15 | 3,7591 | 3,4308 | -6,3359 | 0,0752 | 0,3964 | 0 | 0,1076 | 0,3950 |
| 16 | 3,9818 | 3,3055 | -6,2037 | 0,0749 | 0,2150 | 0 | 0,1145 | 0,2218 |
| 17 | 3,7777 | 3,4168 | -6,3196 | 0,0751 | 0,2140 | 0 | 0,0887 | 0,2545 |
| 18 | 3,7530 | 3,4379 | -6,3452 | 0,0752 | 0,1888 | 0 | 0,1082 | 0,2543 |
| 19 | 2,9937 | 4,0644 | -7,1579 | 0,0822 | 0,1948 | 0 | 0,1397 | 0,2316 |

### Variação de indivíduos

A Figura 23 mostra a evolução da FOB e do tempo de simulação para os testes de 1 a 4 onde se varia o número de indivíduos entre 10, 40, 70 e 100. Pode-se notar que quanto maior o número de indivíduos melhor são os valores da FOB e o valor médio, evidenciando a melhora da otimalidade, no entanto, maior fica o tempo de simulação.



Figura – Variação do número de indivíduos no AG, testes 1 a 4, limite [-10,10].

As Figura 24 e Figura 25 mostram, a título de exemplo, a evolução da FOB para 10 e 100 indivíduos. Nestes casos é evidente a rapidez que o método com mais indivíduos chega ao resultado.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 10 indivíduos. | Figura – Evolução da FOB para 100 indivíduos. |

### Variação de gerações

A Figura 26 mostra a evolução da FOB e do tempo de simulação para os testes de 5 a 7 onde se varia o número de gerações entre 30, 50 e 75. Pode-se notar que, para este caso, a FOB não teve melhoras significativas, mas, o valor médio (otimalidade) melhorou razoavelmente, no entanto, como é de se esperar, o tempo de simulação fica maior.



Figura – Variação do número de gerações no AG, testes 5 a 7, limite [-10,10].

As Figura 27 e Figura 28 mostram, a título de exemplo, a evolução da FOB para 30 e 75 gerações. Devido às características do problema, aproximadamente na iteração 30 é que o método começa a se estabilizar. Logicamente, escolhendo-se poucas iterações o método tem uma baixa otimalidade.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 30 gerações. | Figura – Evolução da FOB para 75 gerações. |

### Variação do elitismo

A Figura 29 mostra a evolução da FOB e do tempo de simulação para os testes de 8 a 10 onde se varia a taxa da população que permanece, ou seja, o elitismo entre 0, 5 e 10%. Indo no contrassenso, a FOB e o valor médio pioraram ligeiramente, como trabalho futuro se propõe maiores investigações a fim de explicar o fato. O tempo de simulação melhorou, talvez porque menos indivíduos são cruzados no processo.



Figura – Variação do elitismo no AG, testes 8 a 10, limite [-10,10].

As Figura 30 e Figura 31 mostram, a título de exemplo, a evolução da FOB para 0 e 10% de elitismo. Devido às características do problema, pouca houve pouca melhora na rapidez com que o método chega à solução ótima.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 0% de elitismo. | Figura – Evolução da FOB para 10% de elitismo. |

### Variação do crossover

A Figura 32 mostra a evolução da FOB e do tempo de simulação para os testes de 11 a 13 onde se varia a taxa da população que é criada pela função crossover, no caso *@crossoversinglepoint*, entre 50, 70 e 90%. A FOB apresenta um ponto ótimo da FOB e do valor médio no teste (12) com 70% da população usando a função definida. Nos valores extremos a FOB e seu valor médio são maiores. O tempo de simulação diminui quase linearmente com o aumento da taxa de crossover.



Figura – Variação da porcentagem de cruzamento no AG, testes 11 a 13, limite [-10,10].

As Figura 33 e Figura 34 mostram, a título de exemplo, a evolução da FOB para 50 e 90% de da população usando a função crossover. Para 90% da população, a evolução da FOB é notavelmente mais lenta, enquanto que para 50% o método chega mais rápido ao valor ótimo.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 50% crossover. | Figura – Evolução da FOB para 90% de crossover. |

De forma geral, a melhor solução encontrada foi a do teste (11) com FOB de 0,0747. A Figura 37 mostra a curva de original dada pela Tabela 1 e a curva com ajuste do teste (11), percebe-se que o erro é relativamente pequeno.

|  |  |
| --- | --- |
|  |  |
| Figura – Curva do problema e curva de ajuste  (visão a) [-10,10]. | Figura – Curva do problema e curva de ajuste  (visão b) [-10,10]. |

## Testes com limites [3, 10]

A Tabela 11 Tabela 7mostra os resultados encontrados para os limites [3,10]. Cada um dos 14 testes foi executado 50 vezes, o melhor valor de FOB foi escolhido. A fim de refletir a otimalidade é apresentado também o valor médio da FOB, assim como o tempo médio de simulação.

Tabela – Resultado dos testes o AG, limites [3,10].

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | A | B | C | FOB | Tempo | Flag | FOB média | Tempo Médio |
| 1 | 3,0039 | 3,0032 | 3,0001 | 1,9231 | 0,1235 | 0 | 1,9610 | 0,1735 |
| 2 | 3,0003 | 3,0001 | 3,0003 | 1,9220 | 0,2754 | 0 | 1,9227 | 0,2484 |
| 3 | 3,0001 | 3,0001 | 3,0001 | 1,9219 | 0,3303 | 0 | 1,9222 | 0,3296 |
| 4 | 3,0002 | 3,0000 | 3,0001 | 1,9219 | 0,4302 | 0 | 1,9222 | 0,4188 |
| 5 | 3,0076 | 3,0005 | 3,0013 | 1,9226 | 0,1658 | 0 | 1,9273 | 0,1515 |
| 6 | 3,0021 | 3,0000 | 3,0001 | 1,9220 | 0,2176 | 0 | 1,9223 | 0,2399 |
| 7 | 3,0000 | 3,0000 | 3,0001 | 1,9219 | 0,3362 | 0 | 1,9219 | 0,3694 |
| 8 | 3,0002 | 3,0000 | 3,0001 | 1,9219 | 0,2230 | 0 | 1,9225 | 0,2320 |
| 9 | 3,0005 | 3,0001 | 3,0002 | 1,9220 | 0,2420 | 0 | 1,9230 | 0,2346 |
| 10 | 3,0021 | 3,0001 | 3,0000 | 1,9220 | 0,2218 | 0 | 1,9226 | 0,2318 |
| 11 | 3,0007 | 3,0000 | 3,0003 | 1,9220 | 0,1820 | 1 | 1,9237 | 0,1913 |
| 12 | 3,0000 | 3,0001 | 3,0000 | 1,9219 | 0,2257 | 0 | 1,9235 | 0,2257 |
| 13 | 3,0004 | 3,0003 | 3,0001 | 1,9220 | 0,2180 | 0 | 1,9225 | 0,2262 |
| 14 | 3,0006 | 3,0002 | 3,0003 | 1,9220 | 0,2314 | 0 | 1,9232 | 0,2402 |
| 15 | 3,0003 | 3,0001 | 3,0002 | 1,9219 | 0,3080 | 0 | 1,9225 | 0,2698 |
| 16 | 3,0004 | 3,0001 | 3,0000 | 1,9219 | 0,2401 | 0 | 1,9232 | 0,2760 |
| 17 | 3,0008 | 3,0000 | 3,0001 | 1,9219 | 0,2388 | 0 | 1,9224 | 0,2628 |
| 18 | 3,0009 | 3,0001 | 3,0000 | 1,9219 | 0,2219 | 0 | 1,9227 | 0,2438 |
| 19 | 3,0017 | 3,0003 | 3,0015 | 1,9224 | 0,1945 | 0 | 1,9255 | 0,2104 |

Os valores encontrados para os coeficientes A, B e C foram praticamente iguais para todos os testes, obviamente os valores de FOB também. Apenas o tempo sofreu alteração com a variação de parâmetros

### Variação de indivíduos

A Figura 37 mostra a evolução da FOB e do tempo de simulação para os testes de 1 a 4 onde se varia o número de indivíduos entre 10, 40, 70 e 100. Praticamente não houve alteração do valor da FOB, exceto seu valor médio que diminuiu com o aumento de indivíduos. O tempo de simulação como esperado, aumenta quase que linearmente.



Figura – Variação do número de indivíduos no AG, testes 1 a 4, limite [3,10].

As Figura 38 e Figura 39 mostram, a título de exemplo, a evolução da FOB para 10 e 100 indivíduos. Nestes casos é evidente a rapidez que o método com mais indivíduos chega ao resultado.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 30 gerações. | Figura – Evolução da FOB para 75 gerações. |

### Variação de gerações

A Figura 40 mostra a evolução da FOB e do tempo de simulação para os testes de 5 a 7 onde se varia o número de gerações entre 30, 50 e 75. Praticamente não houve alteração do valor da FOB, exceto seu valor médio que diminuiu com o aumento de gerações. O tempo de simulação como esperado, aumenta quase que linearmente.



Figura – Variação do número de gerações no AG, testes 5 a 7, limite [3,10].

As Figura 41 e Figura 42 mostram, a título de exemplo, a evolução da FOB para 30 e 75 gerações. A estabilização neste caso ocorre por volta de 30 iterações aproximadamente.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 30 gerações. | Figura – Evolução da FOB para 75 gerações. |

### Variação do elitismo

A Figura 43mostra a evolução da FOB e do tempo de simulação para os testes de 8 a 10 onde se varia a taxa da população que permanece, ou seja, o elitismo entre 0, 5 e 10%. A FOB melhora ligeiramente com o elitismo, enquanto o valor médio tem um ponto ótimo no teste (9. O tempo de simulação aumenta com o elitismo.



Figura – Variação do elitismo no AG, testes 8 a 10, limite [3,10].

As Figura 44 Figura 45 mostram, a título de exemplo, a evolução da FOB para 0 e 10% de elitismo. Praticamente não houve alteração na evolução da FOB.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 0% de elitismo. | Figura – Evolução da FOB para 10% de elitismo. |

### Variação do crossover

A Figura 46 mostra a evolução da FOB e do tempo de simulação para os testes de 11 a 13 onde se varia a taxa da população que é criada pela função crossover, no caso *@crossoversinglepoint*, entre 50, 70 e 90%. A FOB piora ligeiramente para altas taxas de crossover, enquanto o tempo de simulação diminui.



Figura – Variação da porcentagem de cruzamento no AG, testes 11 a 13, limite [3,10].

As Figura 47 e Figura 48 mostram, a título de exemplo, a evolução da FOB para 50 e 90% de da população usando a função crossover. Para 90% da população, a evolução da FOB é notavelmente mais rápida, enquanto que para 50% o método chega mais lento ao valor ótimo.

|  |  |
| --- | --- |
|  |  |
| Figura – Evolução da FOB para 50% crossover. | Figura – Evolução da FOB para 90% de crossover. |

Impondo-se o limite [3,10] a curva não pode ser ajustada de forma adequada e o método não encontra melhor resultado que [3, 3 ,3], com erro médio quadrático de 1,9219. A Figura 49 mostra a curva de original dada pela Tabela 1 e a curva com ajuste, percebe-se que o erro é relativamente pequeno.

|  |  |
| --- | --- |
|  |  |
| Figura – Curva do problema e curva de ajuste  (visão a) [3,10]. | Figura – Curva do problema e curva de ajuste  (visão b) [3,10]. |

## Testes com seleção

Esta seção tem por finalidade apresentar os resultados dos testes de variação do tipo de seleção de pais para a próxima geração. A Tabela 12 mostra os tipos de seleções escolhidas.

Tabela - Configurações propostas para os testes de seleção.

|  |  |  |
| --- | --- | --- |
| Teste | Seleção | OBS |
| 1 | selectionstochunif |  |
| 2 | selectionremainder |  |
| 3 | selectionuniform |  |
| 4 | selectionroulette |  |
| 5 | selectiontournament | 4 participantes |

Cada um dos 5 testes foi simulado 50 vezes, com limites de [-10,10] e os melhores resultados estão na Tabela 13. São mostrados também os valores médios da FOB e do tempo. O valor médio da FOB funciona como uma medida da otimalidade da configuração, já que o sistema é continuo e a probabilidade de resultados iguais é baixa. Quando *Flag* é ZERO quer dizer que o critério de parada foi o número máximo de iterações, quando é UM o método foi interrompido pois a FOB ficou estagnada

Tabela – Resultados dos testes de seleção, limite [-10,10].

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | A | B | C | FOB | Tempo | Flag | FOB média | Tempo Médio |
| 1 | 3,9243 | 3,3227 | -6,2152 | 0,0748 | 0,1600 | 0 | 0,0778 | 0,1908 |
| 2 | 3,9865 | 3,3055 | -6,2049 | 0,0750 | 0,1621 | 0 | 0,0987 | 0,1890 |
| 3 | 3,2147 | 3,9816 | -7,0643 | 0,0807 | 0,1552 | 0 | 0,1688 | 0,1880 |
| 4 | 3,9002 | 3,3358 | -6,2291 | 0,0748 | 0,1703 | 0 | 0,0996 | 0,2030 |
| 5 | 3,3222 | 3,7663 | -6,7300 | 0,0770 | 0,3665 | 0 | 0,1010 | 0,2950 |

A Figura 51 mostra a evolução da FOB e do tempo de simulação para os testes. O valor da FOB variou pouco para os tipos de seleção, no entanto o valor médio teve variações razoáveis. A função de seleção com melhor compromisso entre precisão e otimalidade foi a (*@selectionstochunif*), que escolhe os pais segundo um ranking, muito parecido com a roleta, mas que prioriza ainda mais os pais melhores. A pior função de seleção foi a (*@selectionuniform*), na qual, os pais são escolhidos aleatoriamente. O pior tempo de simulação foi do torneio dado as operações extras a serem feitas. Os melhores tempos ficaram entre os testes (1), (2) e (3), empatando tecnicamente.



Figura – Variação do tipo de seleção no AG, testes 1 a 5, limite [-10,10].

## Testes com cruzamento

Esta seção tem por finalidade apresentar os resultados dos testes de variação do tipo de cruzamento de pais para a próxima geração. A Tabela 14 mostra os tipos de cruzamento escolhidos.

Tabela - Configurações propostas para os testes de cruzamento.

|  |  |
| --- | --- |
| Teste | Cruzamento |
| 1 | **crossoverscattered** |
| 2 | **crossoversinglepoint** |
| 3 | **crossovertwopoint** |
| 4 | **crossoverintermediate** |

Para a função (*@crossoverintermediate*) o novo indivíduo vindo do cruzamento é definido pela expressão:

|  |  |
| --- | --- |
|  | () |

Onde é o novo indivíduo, é o indivíduo da iteração , é o indivíduo da iteração e é um valor aleatório.

Cada um dos 4 testes foi simulado 50 vezes, com limites de [-10,10] e os melhores resultados estão na Tabela 15. São mostrados também os valores médios da FOB e do tempo. O valor médio da FOB funciona como uma medida da otimalidade da configuração, já que o sistema é continuo e a probabilidade de resultados iguais é baixa. Quando *Flag* é ZERO quer dizer que o critério de parada foi o número máximo de iterações, quando é UM o método foi interrompido pois a FOB ficou estagnada

Tabela – Resultados dos testes de cruzamento, limite [-10,10].

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Teste | A | B | C | FOB | Tempo | Flag | FOB média | Tempo Médio |
| 1 | 3,9582 | 3,3125 | -6,2084 | 0,0749 | 0,1782 | 0 | 0,0990 | 0,1904 |
| 2 | 3,6268 | 3,5290 | -6,4512 | 0,0757 | 0,1793 | 0 | 0,0844 | 0,2123 |
| 3 | 3,9375 | 3,3186 | -6,2124 | 0,0748 | 0,1783 | 0 | 0,1055 | 0,2207 |
| 4 | 3,8788 | 3,3465 | -6,2397 | 0,0748 | 0,2389 | 0 | 0,0772 | 0,2761 |

A Figura 52 mostra a evolução da FOB e do tempo de simulação para os testes. O valor final da FOB variou pouco para os tipos de cruzamento, no entanto o valor médio teve uma flutuação. A função de seleção com melhor compromisso entre precisão e otimalidade foi a (*@crossoverintermediate*), que forma o indivíduo segundo a Equação (8), no entanto, teve a menor eficiência computacional. A pior função de cruzamento em termos de otimalidade foi a (*@crossovertwopoint*).



Figura – Variação do tipo de cruzamento no AG, testes 1 a 4, limite [-10,10].

# Conclusão

Neste relatório foi descrita uma série de testes em algoritmos de otimização para solução de um problema de ajuste dos coeficientes de uma curva. Os algoritmos escolhidos foram o BAT Algorithm e o Algoritmo Genéticos. O primeiro teve sua implementação discutida passo a passo, o segundo foi avaliado segundo sua toolbox no MatLab.

Baterias de testes foram conduzidas para a avaliação do desempenho dos algoritmos variando por exemplo: o número de morcegos/população, o número de iterações/gerações entre outros. A otimização por ecolocalização de morcegos tem a grande vantagem da fácil implementação devido a pequena quantidade de parâmetros, praticamente apenas 4. Os números de iterações e de indivíduos tiveram influência positiva otimalidade e negativa na eficiência computacional como se esperava.

O valor de da taxa de decréscimo na amplitude, *α*, por sua vez teve comportamento inesperado. Quando seu valor era mais baixo, ou seja, com menor probabilidade de busca global, o ponto ótimo melhor, como se observa na Figura 7. Já para a taxa de emissão de pulso, , o comportamento foi o esperado. Quanto maior a probabilidade de buscas locais, melhor o ponto ótimo, Figura 9.

Já o Algoritmo Genético tem uma gama muito maior de parâmetros que o concorrente, como tipos de seleção, tipos de cruzamento, variações na população entre outros. Assim como no algoritmo BAT, o aumento de indivíduos e gerações melhoram a precisão do ponto ótimo mais pioram consideravelmente a eficiência computacional. A consideração do elitismo teve efeito contrário ao esperado, piorando a evolução da função de aptidão e seu valor final, talvez pela característica do problema.

Fez-se também testes na função de seleção, nos quais, a melhor função foi (@*selectionstochunif*) que escolhe os pais segundo um ranking, muito parecido com a roleta, mas que prioriza ainda mais os melhores pais. A função de cruzamento mais eficiente foi a (@*crossoverintermediate*) que faz uma combinação linear dos pais para gerar o novo individuo.

De modo geral, para o problema proposto, o algoritmo BAT foi mais eficiente no que refere a implementação e simplicidade. Poucos e simples parâmetros foram suficientes para encontrar a solução. Já o algoritmo genético, tem vários parâmetros a se escolher incialmente, o que pode ser confuso e trabalhoso. Em relação à eficiência computacional o AG foi muito superior (cerca de 3 vezes menor), apesar do número parecido de gerações. Talvez por ser uma toolbox do MatLab altamente especializada e construída de forma ótima.

Outra questão importante foi a consideração de limites. Inicialmente, os limites impostos entre [3,10] para os coeficientes levavam a resposta [3, 3, 3] invariavelmente, indicando que a combinação que aproximava melhor a curva caminhava contra os limites. Quando foram considerados os limites [-10,10] o método encontrou uma solução dentro do universo, indicando um melhor ponto ótimo.

# Bibliografia

|  |  |
| --- | --- |
| [1] | I. C. d. S. Junior, *Técnicas Inteligentes: Otimização por ecolocalização de morcegos,* Tópicos Especiais de Otimização: 210115-IC - PPEE/UFJF, 2017. |
| [2] | I. C. d. S. Junior, *Técnicas Inteligentes: Algoritmos Genéticos,* Tópicos Especiais de Otimização: 210115-IC - PPEE/UFJF, 2017. |
| [3] | D. C. Montgomery, Introduction to Time Series Analysis And Forecasting, Hoboken, New Jersey: John Wiley & Sons, Inc., 2015. |