Tópicos Especiais em Eletrônica

Lista 2

Resolução

Ivan Carlos

October 2021

a)

O primeiro passo é escrever nossa função abaixo.

$$F6(x,y) = 0.5 - \frac{(\sin\sqrt{x^2+y^2})^2 - 0.5}{(1+(0.0001)\cdot(x^2+y^2)))^2} \tag{1}$$

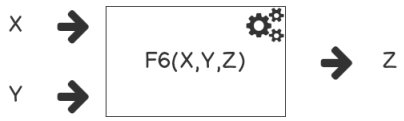

Figura 1: Modelo da função F6

Dentro do diretório de trabalho criamos o arquivo “F6.m”, conforme mostrado abaixo.

Algorítmo 1: Arquivo F6.m

```
% Name: F6.m
% Description: Implenatção da função F6
%                                    2    2  2
%                           sin(sqrt(x  + y ))  - 0.5
%   z = F6(x,y) =  0.5 - -------------------------
%                                      2    2  2
%                          (1 + 0.001 (x  + y ))
%
%
%   F6(0,0);
%                    1.000000000
function z = F6(x,y)
    % inputs:
    %   x - é um escalar
    %   y - é um escalar
```

```
16      % output:
17      %    z - é um escalar
18    z = 0.5 - (sin(sqrt(x^2 + y^2))^2 - 0.5)/(1 + 0.001 * (x^2 + y^2)
          )^2;
19  end
```

Para invocar a função basicamente digitamos *F6* munido de suas entradas.
Dessa forma testamos as entradas $X = 0, Y = 0$ mencionadas no item e verificamos se a saída é $Z = 1$.

Algorítmo 2: Prompt de comando rodando F6(0,0)

```
1 octave:1> X=0; Y=0;
2 octave:2> F6(X,Y)
3 ans = 1
4 octave:3>
```

Após essa análise inicial, voltamos a nossa função *F6*, mas levando em consideração que esta possui uma quantidade elevada de picos e vales no $\mathbb{R}^3$ necessitando de pelo menos três abordagens para plotar essa função com os recursos presentes no MATLAB[1].
Mas antes de começar a análise com o matlab jogamos a função no WolframAlpha[2] para entender o que nós espera.

3D plot	$0.5 - \dfrac{\sin^2\left(\sqrt{x^2 + y^2}\right) - 0.5}{\left(1 + 0.001\left(x^2 + y^2\right)\right)^2}$

Figura 2: Equação

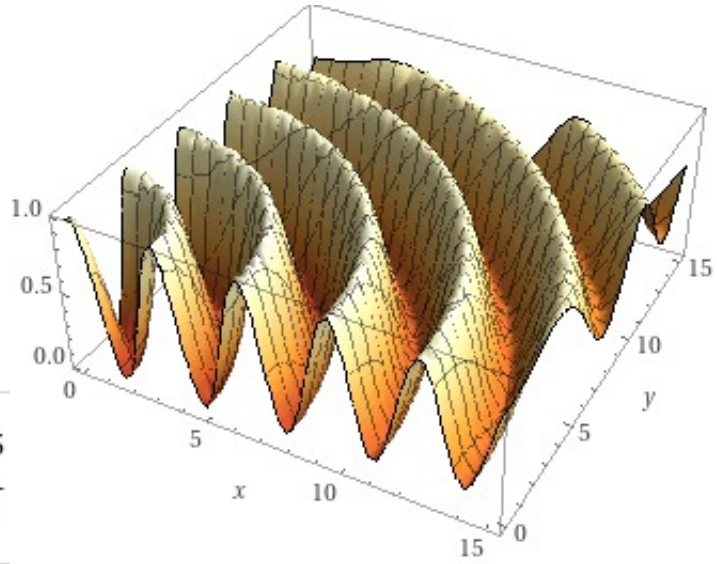

Figura 3: WolframAlpha[2]

Primeira abordagem: plotando pontos em intervalos discretos

Definimos x e y como arrays que definem a faixa de intervalo $[-100..100]$ ao passo de .5 por exemplo.

Algorítmo 3: Plot 1

```
% Name: code/myplot1.m
% Description:  Primeira tentativa

% Nossos vetores X e Y são compostos pelos valores de uma
% progressão aritimética iniciada em -100, terminada em 100 com
% uma razão de 0.5.
% a[inicial] = -100
% a[final]   = 100
% r = 0.5
x = [-100:0.5:100];
y = [-100:0.5:100];
z = arrayfun(@(x,y) F6(x,y), x, y)
%   ^^^^^^^^ ^^^^   ^^^^^^   ^  ^
%   |        |      |        |  |
%   |        |      |        |  +-a segunda entrada vetorial
%   |        |      |        +-a primeira entrada vetorial
%   |        |      +-a função que será mapeada
%   |        +-definição de entradas
%   +-função mapeadora "MAP"
figure
plot3(x,y,z)
xlabel('x')
ylabel('y')
zlabel('z')
```

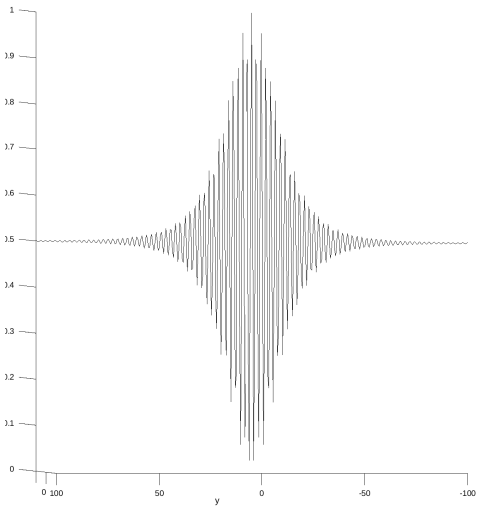

Figura 4: Plot 1

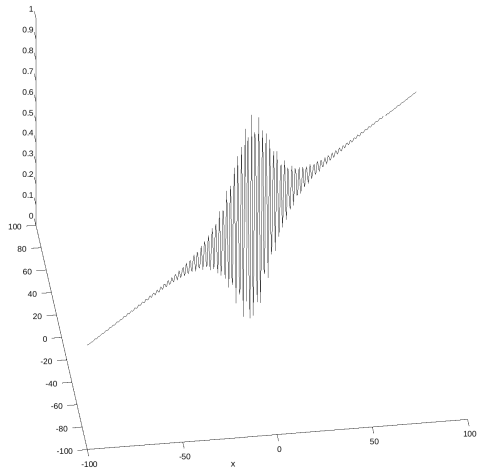

Figura 5: Plot 1 visualizando os 3 eixos

Segunda abordagem: plotando pontos em intervalos discretos com redução do intervalo e um aumento substancial de pontos

Definimos x e y como arrays que definem a faixa de intervalo $[-1..1]$ ao passo de .00005 por exemplo.

Algorítmo 4: Plot 2

```
% Name: code/myplot2.m
% Description:  Segunda tentativa

% Nossos vetores X e Y são compostos pelos valores de uma
% progressão aritimética iniciada em -1, terminada em 1 com
% uma razão de 0.00005.
% a[inicial] = -1
% a[final]   = 1
% r = 0.5
x = [-1:0.00005:1];
y = [-1:0.00005:1];
z = arrayfun(@(x,y) F6(x,y), x, y)
%   ^^^^^^^^ ^^^^   ^^^^^^   ^  ^
%   |        |      |        |  |
%   |        |      |        |  +-a segunda entrada vetorial
%   |        |      |        +-a primeira entrada vetorial
%   |        |      +-a função que será mapeada
%   |        +-definição de entradas
%   +-função mapeadora "MAP"

figure
hold on
plot3(x,y,z)
xlabel('x')
ylabel('y')
zlabel('z')
grid on
%axis('equal')
```

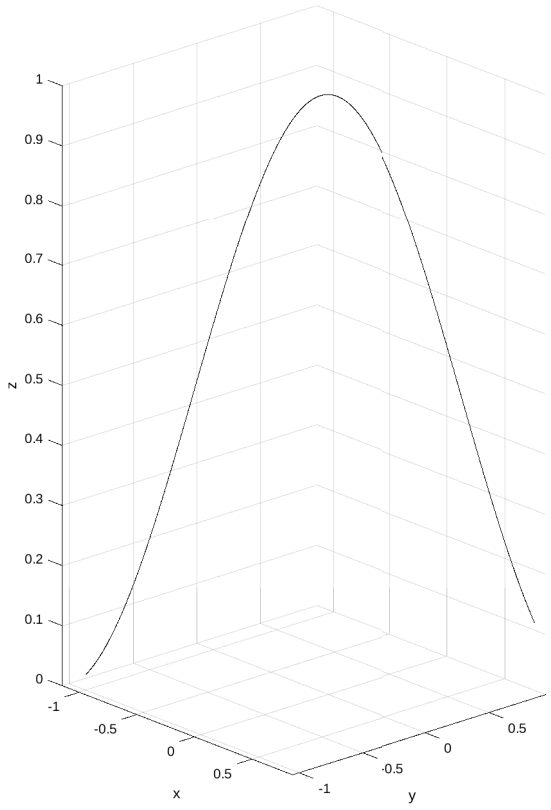

Figura 6: Plot 2

Definitivamente essa metodologia não diz muito sobre os picos e vales de nossa função *F6*, os unicos pontos positívos até o momento são a observação da sua faixa de relevância e que o ponto de máximo se encontra na vizinhança do ponto $(0, 0, z)$.

Terceira abordagem: plotando a função como uma curva de superfície em formato de malha em intervalos discretos com redução do intervalo de relevância

Mantemos o intervalo de relevância entre $[-100, 100]$, todavia mantendo o número de pontos em torno de 1000 pontos.

$x = linspace(-100, 100, 1000);$

Algorítmo 5: Plot 3

```
% Name: code/myplot3.m
% Description:  terceira tentativa
x = linspace(-100, 100, 1000);
y = x;

[X,Y] = meshgrid(x,y)
% A função meshgrid é usada para criar uma grade retangular a partir
% de duas matrizes unidimensionais fornecidas que representam a
% indexação cartesiana ou indexação de matriz.
%
% Considere a figura acima com o eixo X variando de -4 a 4 e o eixo
% Y variando de -2 a 2. Portanto, há um total de (9 * 5) = 45
% pontos marcados na figura, cada um com uma coordenada X e uma
% coordenada Y. Para qualquer linha paralela ao eixo X, as
% coordenadas X dos pontos marcados respectivamente são -4, -3, -2,
% -1, 0, 1, 2, 3, 4. Por outro lado, para qualquer linha
% paralelo ao eixo Y, as coordenadas Y dos pontos marcados de
% baixo para cima são -2, -1, 0, 1, 2.
% A meshgrid retorna duas matrizes bidimensionais que representam as
% coordenadas X e Y de todos os pontos.
%
%
%                        Y
%                        ^
%                      2 +    *--*--*--*--*--*--*--*--*
%                        |    |  |  |  |  |  |  |  |  |
%                      1 +    *--*--*--*--*--*--*--*--*
% 2+2+1 = 5 pontos       |    |  |  |  |  |  |  |  |  |
%                      0 +    *--*--*--*--*--*--*--*--*
%                        |    |  |  |  |  |  |  |  |  |
%                     -2 +    *--*--*--*--*--*--*--*--*
%                        |
%                        |
%                        +----+--+--+--+--+--+--+--+--+---> X
%                            -4 -3 -2 -1  0  1  2  3  4
%
%                            4+4+1 = 9 pontos
```

```
38 %
39 %         grade de 5*9 = 45 pontos
40 %
41 %
42 %
43 Z = arrayfun(@(x,y) F6(x,y), X, Y)
44 %   ^^^^^^^^ ^^^^   ^^^^^^   ^  ^
45 %   |        |      |        |  |
46 %   |        |      |        |  +-a segunda entrada vetorial
47 %   |        |      |        +-a primeira entrada vetorial
48 %   |        |      +-a função que será mapeada
49 %   |        +-definição de entradas
50 %   +-função mapeadora "MAP"
51
52 figure
53 mesh(X,Y,Z)
54 hold on
55 xlabel('x')
56 ylabel('y')
57 zlabel('z')
58 grid on
```

Mesmo com essa nova forma de calcular os pontos e conectar os pequenos planos, “mesh” ou grades, não ficou claro conforme é visto no gráfico abaixo os máximos e mínimos de *F6*.

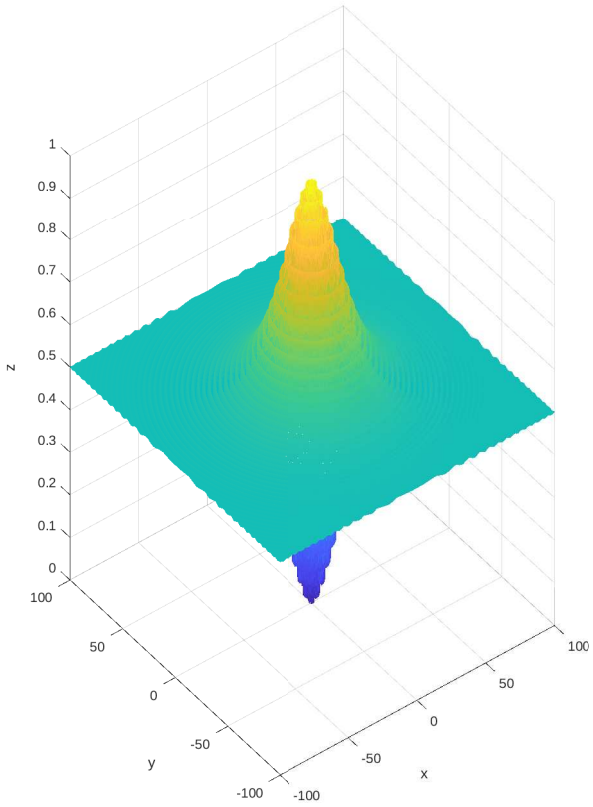

Figura 7: $X, Y \in [-100, 100]$

Nesse ponto reduzimos o intervalo de relevância, o domínio, de forma drástica para $[-1, 1]$ e mantemos o número de pontos em torno de 1000 pontos.

$x = linspace(-1, 1, 1000);$

Algorítmo 6: Plot 4

```
% Name: code/myplot4.m
% Description:  quarta tentativa
x = linspace(-1, 1, 1000);
y = x;

[X,Y] = meshgrid(x,y)
% A função meshgrid é usada para criar uma grade retangular a partir
% de duas matrizes unidimensionais fornecidas que representam a
% indexação cartesiana ou indexação de matriz.
%
% Considere a figura acima com o eixo X variando de -4 a 4 e o eixo
% Y variando de -2 a 2. Portanto, há um total de (9 * 5) = 45
% pontos marcados na figura, cada um com uma coordenada X e uma
% coordenada Y. Para qualquer linha paralela ao eixo X, as
% coordenadas X dos pontos marcados respectivamente são -4, -3, -2,
% -1, 0, 1, 2, 3, 4. Por outro lado, para qualquer linha
% paralelo ao eixo Y, as coordenadas Y dos pontos marcados de
% baixo para cima são -2, -1, 0, 1, 2.
% A meshgrid retorna duas matrizes bidimensionais que representam as
% coordenadas X e Y de todos os pontos.
%
%
%                       Y
%                       ^
%                     2 +    *--*--*--*--*--*--*--*--*
%                       |    |  |  |  |  |  |  |  |  |
%                     1 +    *--*--*--*--*--*--*--*--*
% 2+2+1 = 5 pontos      |    |  |  |  |  |  |  |  |  |
%                     0 +    *--*--*--*--*--*--*--*--*
%                       |    |  |  |  |  |  |  |  |  |
%                    -2 +    *--*--*--*--*--*--*--*--*
%                       |
%                       |
%                       +----+--+--+--+--+--+--+--+--+---> X
%                           -4 -3 -2 -1  0  1  2  3  4
%
%                           4+4+1 = 9 pontos
%
%        grade de 5*9 = 45 pontos
%
%
%
Z = arrayfun(@(x,y) F6(x,y), X, Y)
%   ^^^^^^^^ ^^^^   ^^^^^^   ^  ^
%   |        |      |        |  |
%   |        |      |        |  +-a segunda entrada vetorial
```

```
47 %   |         |      |        +-a primeira entrada vetorial
48 %   |         |      +-a função que será mapeada
49 %   |         +-definição de entradas
50 %   +-função mapeadora "MAP"
51
52 figure
53 mesh(X,Y,Z)
54 hold on
55 xlabel('x')
56 ylabel('y')
57 zlabel('z')
58 grid on
```

Mesmo seguindo essas dicas, não obtivemos sucesso na análise gráfica.

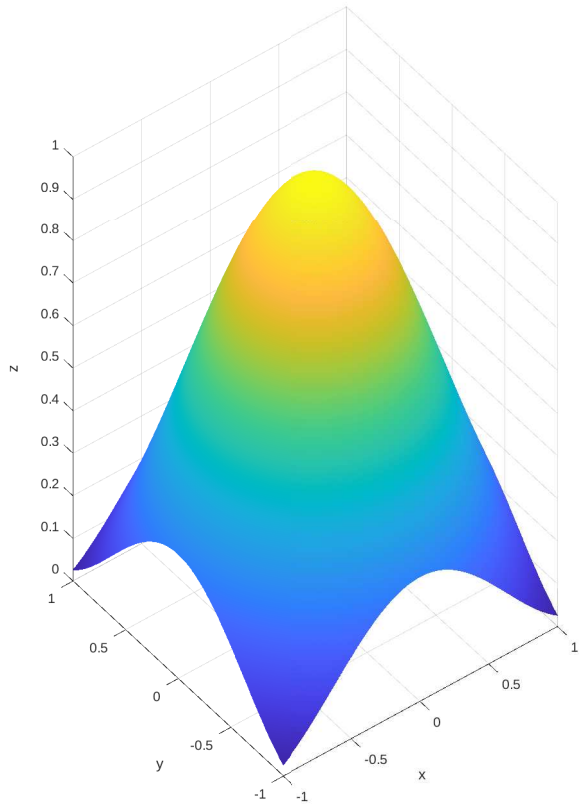

Figura 8: $X, Y \in [-1, 1]$

Sendo menos preciosista e após vários testes chegamos a uma possível interpretação da curva de superfície com o algorítmos subsequente. Para tanto alteramos o domíneo até o ponto desse ser mais substancialmente indicativo da estrutura espacial de *F6*.

Algorítmo 7: Plot 5

```
1 % Name: code/myplot5.m
```

```
% Description:  quinta tentativa
x = linspace(-20, 20, 1000);
y = x;

[X,Y] = meshgrid(x,y)

Z = arrayfun(@(x,y) F6(x,y), X, Y)

figure
mesh(X,Y,Z)
hold on
xlabel('x')
ylabel('y')
zlabel('z')
grid on
```

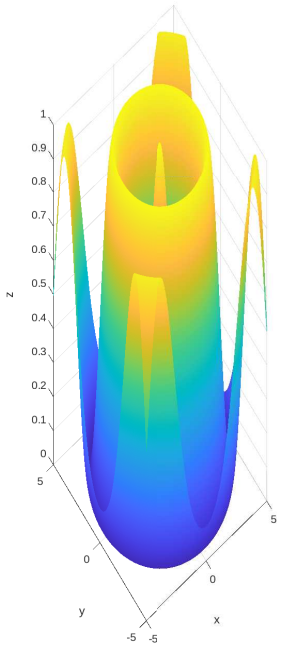

Figura 9: $X, Y \in [-5, 5]$

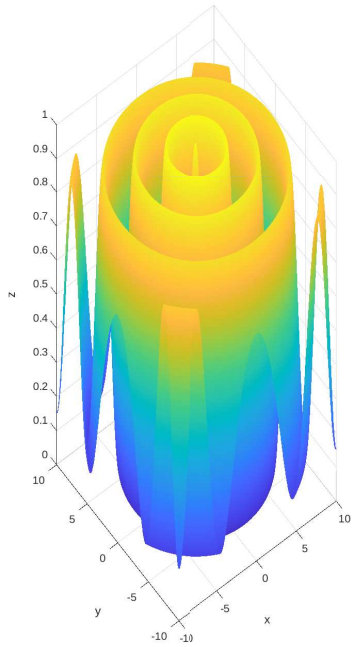

Figura 10: $X, Y \in [-10, 10]$

“ Sendo assim, podemos afirmar que *F6* possui vários máximos locais. Formalmente, um ponto máximo local é um ponto no espaço de entrada tal que todas as outras entradas em uma pequena região perto desse ponto produzem valores menores quando aplicadas na função multivariável. ”

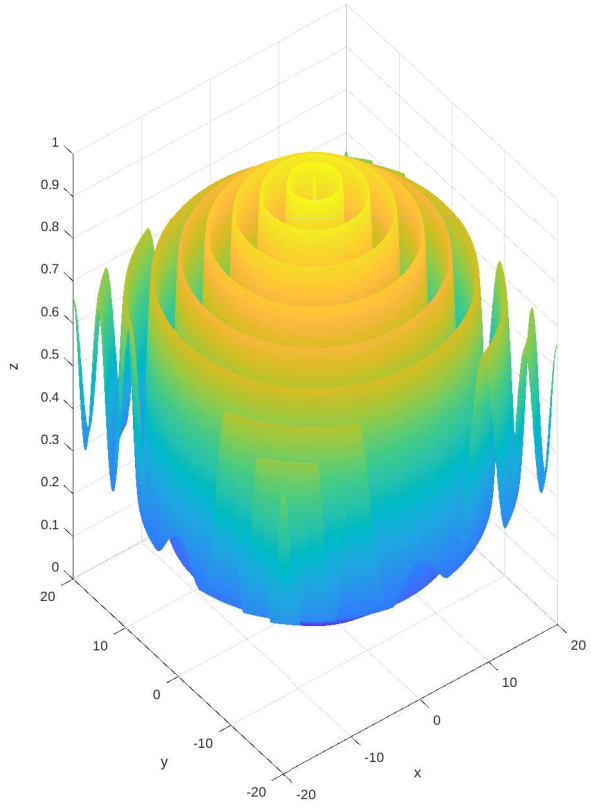

Figura 11: $X, Y \in [-20, 20]$

b)

Baseando-se no exemplo fornecido no gaot (binaryExample.m), implemente um programa baseado em GA (GA1) para maximizar a função f6 com as seguintes características:

- representação binária (binário codificando real);
- domínio de x e y $[-100, 100]$;
- epsilon = $1e-06$ (precisão);
- aptidão é a avaliação;
- População = 100;
- Total de indivíduos = 40.000;
- Taxa de Crossover = 80%;
- Taxa de Mutação = 1%;
- função de seleção (*roulette weel*)

Antes de mais nada, precisamos instalar a biblioteca no MATLAB[1], no nosso caso consiste em adicionar a pasta no caminho de leitura do programa, conforme mostrado abaixo.

Algorítmo 8: Instalando biblioteca gaot

```
% Add folders to search path
addpath('gaot')
```

Reescrevemos a função *F6* para que ela seja compatível com a biblioteca *GAOT*.

Algorítmo 9: Reimplementação de F6

```
% Name: F6.m
% Description: A função F6 foi rescrita de modo a atender as
    espectativas da
%
%                                    2    2  2
%                           sin(sqrt(x  + y ))  - 0.5
%   z = F6(x,y) =  0.5 - -------------------------
%                                      2    2  2
%                            (1 + 0.001 (x  + y ))
% biblioteca gaot
% Input:
%  sol: input [x y]
%  options: input vector
%  sol: output [x y]
```

```
%  val: output z
%
function [sol,val] = F6(sol,options)

x=sol(1);
y=sol(2);
val = 0.5 - (sin(sqrt(x^2 + y^2))^2 - 0.5)/(1 + 0.001 * (x^2 + y^2)
    )^2;
```

Implementamos abaixo, o maximizador de *F6* escrevendo o programa "maximizaF6.m" seguindo o modelo apresentado no exemplo "binaryExample.m".

Algorítmo 10: Realiza Maximização da função *F6*

```
% Name: maximizaF6.m
% Description: Maximizes the function F6
%

% Definindo a semente no início para fins de comparação
rand('seed',0)

% domínio de x e y [-100, 100]
dominio = [-100 100; -100 100];
%          ^^^^^^^^   ^^^^^^^^
%          x          y
%

% Definindo precisação
% epsilon = 1e-06 (precisão);
epsilon  = 1e-06;
precisao = epsilon;

% Função de avaliação
% è a nossa função proposta no problema, levemente modificada para a
    API da
% biblioteca GAOT
evalFn  = 'F6';
evalOps = [];

% Operação de Crossover
taxa_de_crossover = .8;      % taxa de Crossover de 80%
xFns  = 'simpleXover';       % função de crossover selecionada
xOpts = [taxa_de_crossover]; % parâmetros da função de crossover

% Operação de mutação
taxa_de_mutacao = 0.01;    % taxa de Mutação de 1%
```

```
mFns = 'binaryMutation';   % função de mutação selecionada
mOpts = [taxa_de_mutacao]; % parâmetros da função de mutação

% Função de parada
termFns = 'maxGenTerm';
termOps = [200]; % 200 Generations

% Função de seleção (roulette weel)
selectFn = 'roulette'
selectOps = [];

% GA Options [epsilon float/binar display]
ga_opts=[precisao 0 1];

% Gera uma população inicial de tamanho 100
populacao = 100;
populacao_inicial = initializega(populacao, dominio,'F6', [], [
    precisao 0]);

[x endPop bestPop trace]=ga( ...
        dominio,             ...
        evalFn,              ...
        evalOps,             ...
        populacao_inicial,   ...
        ga_opts,             ...
        termFns,             ...
        termOps,             ...
        selectFn,            ...
        selectOps,           ...
        xFns,                ...
        xOpts,               ...
        mFns,                ...
        mOpts                ...
);

% x é a melhor solução encontrada
x
```

Após rodar o programa teremos a seguinte saída:

$$\begin{aligned} x &= -3.7253e-07 = -3.7253 * 10^{-7} = -0.0000003725 \\ y &= -2.0675e-04 = -2.0675 * 10^{-4} = 0.0002067500 \\ z &= 1.000000 \end{aligned} \tag{2}$$

Algorítmo 11: Saida do programa *maximizaF6.m*

```
51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
     80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104
    105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124
    125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
    145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164
    165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184
    185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
200 1.000000
x =

  -3.7253e-07  -2.0675e-04
```

Nossa resposta corresponde exatamente a proposição inicial, onde x e y estão no entrorno de $(0, 0)$ e $z = 1$. Conforme verificamos mais de uma vez ao efetuarmos $F6(0, 0)$ e confirmar que tem imagem de valor 1.

b.1)

Quantos bits tem o cromossomo? Explique.
Utilizando o programa *maximizaF6.m* obtemos um total de 28 bits. Um binário também pode representar um número real X_R $[X_{min}X_{mx}]$, com precisão de p casas decimais. Para isso são necessários K bits, sendo K calculado pela inequação:

$$
\begin{aligned}
& 2^K \leq (X_{max} - X_{min}) \times 10^p \\
& \Rightarrow log(2^K) \leq log((X_{max} - X_{min}) \times 10^p) \\
& \Rightarrow log(2^K) \leq log((100 - (-100)) \times 10^6) \\
& \Rightarrow log(2^K) \leq log(200 \times 10^6) \\
& \Rightarrow K \times log2 \leq log(2 \times 10^8) \\
& \Rightarrow K \times log2 \leq log2 + log10^8 \\
& \Rightarrow K \times log2 \leq log2 + 8 \times log10 \\
& \Rightarrow K \times log2 \leq log2 + 8 \\
& \Rightarrow K \leq 1 + \frac{8}{log2} \\
& \Rightarrow K \leq 1 + \frac{8}{0.301} \\
& \Rightarrow K \leq 1 + 26.6 \\
& \Rightarrow K \leq 27.6 \\
& \Rightarrow K \leq 28 \\
& \Rightarrow K = 28
\end{aligned}
\tag{3}
$$

Algorítmo 12: Função que obtem o número de bits em *"maximizaF6.m"*

```
% cálculo do número de bits
%
%                K                      p
% equação:      2  = (X_max - X_min) 10
% X_max = 100
% X_min = -100
% p     = 6      epsilon (expoente)
%
%  >   K = 9 + (8 log(5))/log(2)
%  >>>>> K <<<<
%
bits=calcbits(dominio,epsilon)
```

Algorítmo 13: Saida do programa *maximizaF6b.m*

```
51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
     80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104
    105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124
```

```
    125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
    145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164
    165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184
    185 186 187 188 189 190 191 192 193 194 195 196 197 198 199
200 1.000000
x =

  -3.7253e-07  -2.0675e-04

bits =

   28   28
```

b.2)

Gere as curvas referentes ao melhor indivíduo de cada geração e à média dos indivíduos de cada geração. Comente os resultados obtidos.
Um dos retornos da função *ga* é a variável "trace" que consiste numa matriz de quatro colunas, onde a primeira coluna é o número da geração por linha, para uma geração igual a 200 teremos uma coluna com valores de 1 a 200, a segunda coluna contém o melhor avaliado a cada geração e a terceira coluna a média dos indivíduos por geração.

Algorítmo 14: Programa que obtem os gráficos de melhor indivíduo e média por geração em *"maximizaF6b2.m"*

```
69 trace
70
71 plot(trace(:,1),trace(:,2));
72
73 plot(trace(:,1),trace(:,3));
```

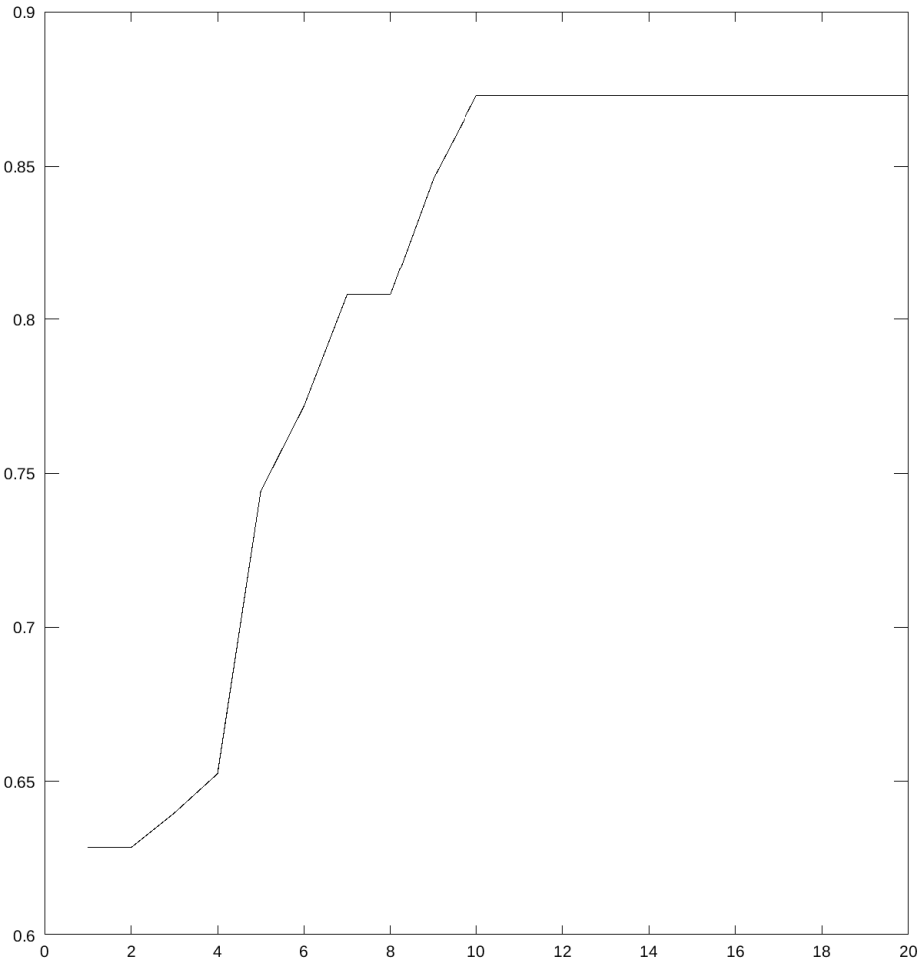

Figura 12: Melhor indivíduo de cada geração

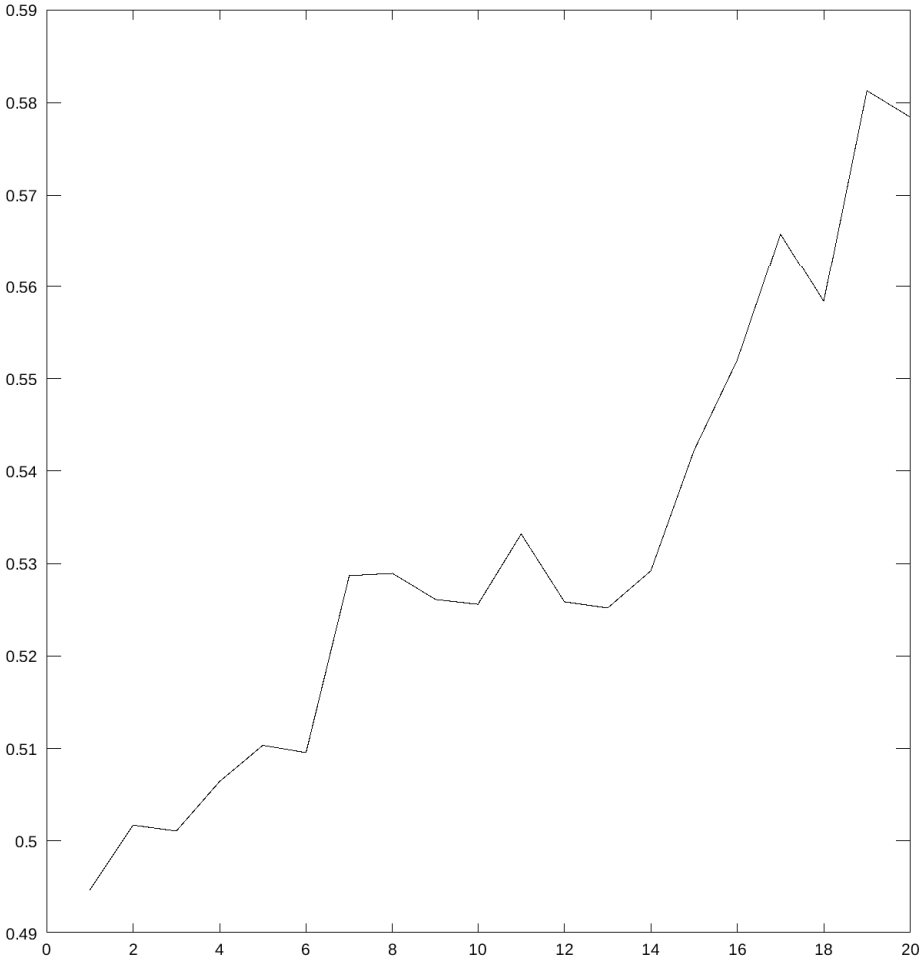

Figura 13: Média dos indivíduos de cada geração

c)

Repita o item (b.2) (Programa GA2) para normalized geometric select (q=0,20). Comente os resultados obtidos.

Algorítmo 15: Programa que obtem os gráficos de melhor indivíduo e média por geração em *"maximizaF6c.m"*

```
% Função de seleção (roulette weel)
% selectFn = 'roulette'
% selectOps = [];
% Selection Function
% Função de seleção (geométrica normalizada)
selectFn = 'normGeomSelect';
selectOps = [0.2];
```

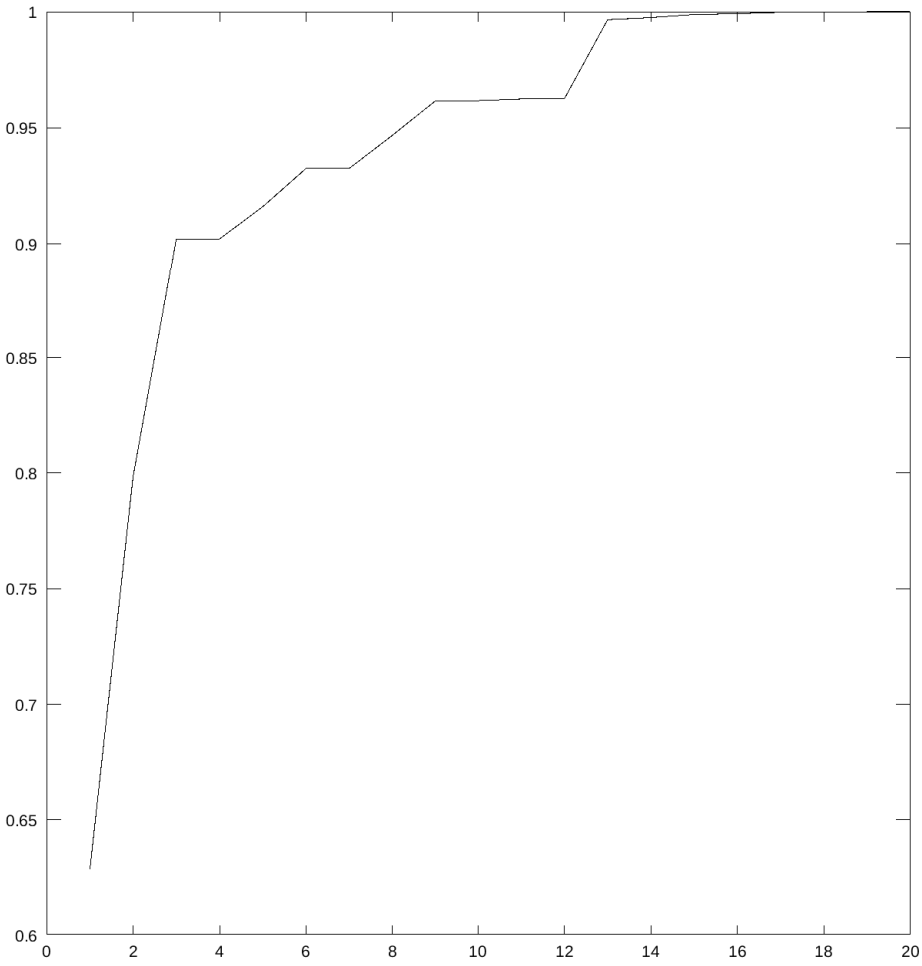

Figura 14: Melhor indivíduo de cada geração utilizando seleção por normalização geométrica

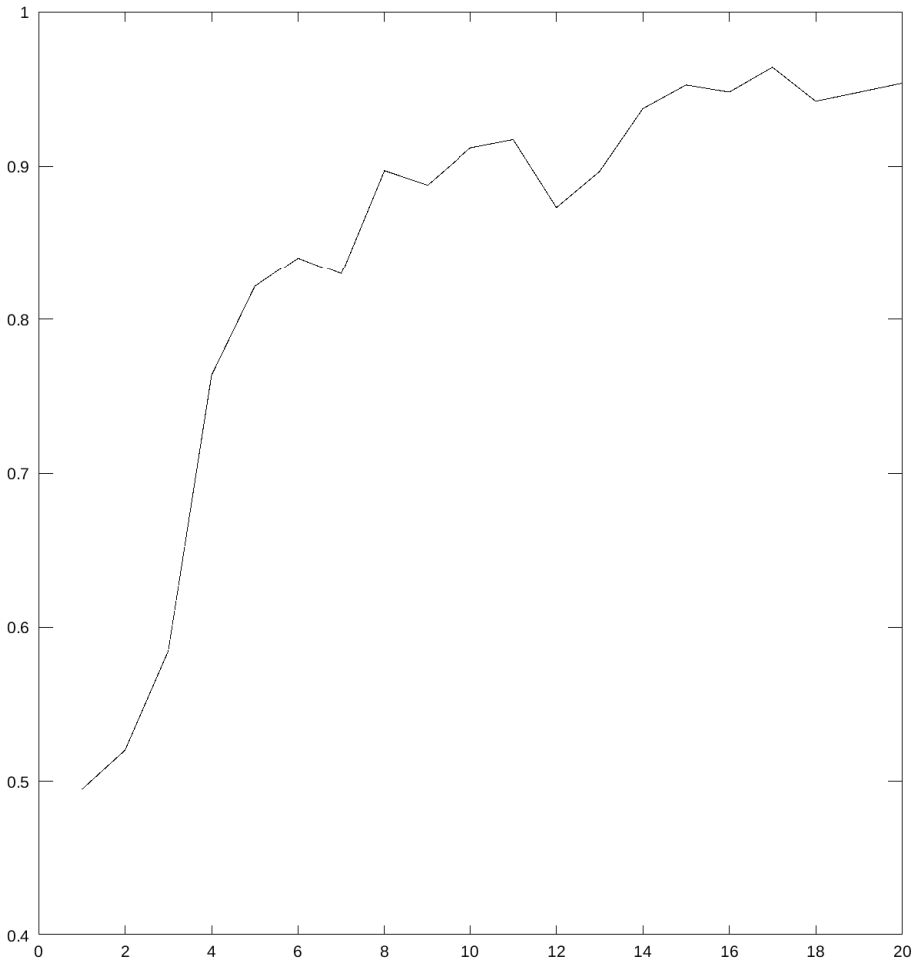

Figura 15: Média dos indivíduos de cada geração utilizando seleção por normalização geométrica

Ao utilizando a seleção por normalização geométrica conseguimos um ganho superior pois a média é maior por geração e atingimos a maximização mais rapidamente.

Referências

[1] The Mathworks, Inc., Natick, Massachusetts. *MATLAB version 9.9.0.1467703 (R2017b)*, 2017.

[2] Eric W. Weisstein. wolframalpha. `https://www.wolframalpha.com/input/?i=plot+3d+0.5+-+%28sin%28sqrt%28x%5E2+%2B+y%5E2%29%29%5E2+-+0.5%29%2F%281+%2B+0.001+*+%28x%5E2+%2B+y%5E2%29+%29%5E2`.