Otimização de Funções com Métodos de Pesquisa

Protocolo de Trabalho Prático 2

INTELIGÊNCIA ARTIFICIAL

4 de dezembro de 2022

Criado por: Francisco Conceição al73819

utad

Inteligência Artificial
Licenciatura em Engenharia Informática
ECT - 2022/2023

Otimização de Funções com Métodos de Pesquisa

Protocolo de Trabalho Prático 2

Introdução

No âmbito da Unidade Curricular de Inteligência Artificial, do curso de Engenharia Informática foi proposto pelos regentes da mesma a realização de um protocolo sobre os algoritmos Hill-Climbing (Subida da Colina), Hill-Climbing with Multiple Re-Start (Subida da Colina com Reinicialização Múltipla) e o Simulated Annealing (SA). Este relatório é dividido em duas partes, a primeira parte aborda os conceitos teóricos destes algoritmos e a segunda parte trata dos resultados de cada algoritmo.

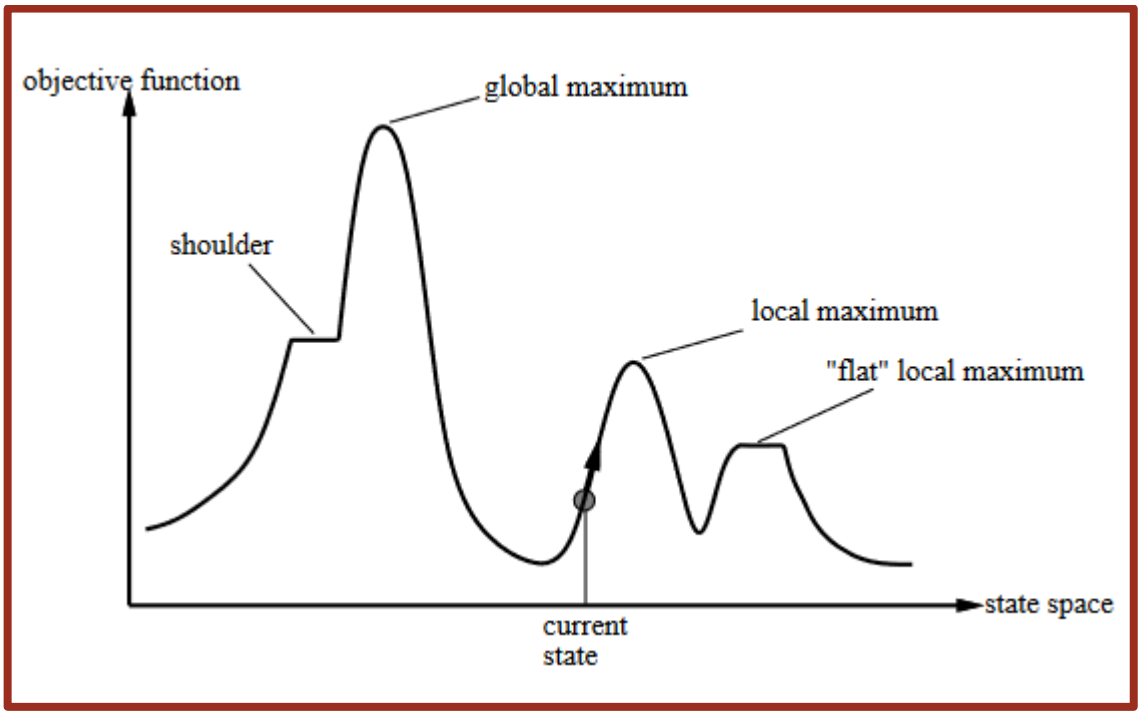
Objetivos do Trabalho

O objetivo principal deste trabalho é o desenvolvimento da aquisição de conhecimentos e de competências relativas aos algoritmos citados na introdução deste relatório. Usando o software Matlab e as funções bidimensionais indicadas pelos regentes realiza-se os algoritmos de modo a atingir este objetivo.

Conceitos Teóricos

O Hill-Climbing e o Simulated Annealing são métodos de pesquisa que integram a maioria dos cursos de Inteligência Artificial. Estes algoritmos servem como introdução para metas heurísticas baseadas em estocásticas e probabilísticas. O Simulated Annealing pode ser considerado uma variante do Hill-Climbing com uma decisão de probabilidade. Enquanto o SA pode ser considerado um algoritmo simples, na prática pode ser muito difícil de parametrizar.

Estes algoritmos procuram o máximo de uma função onde podem existir vários máximos, existindo o *global maximum* tal como o nome indica é o máximo global de uma função e o l*ocal maximum*, identificados na figura a seguir.



Algoritmo Random-Search

O algoritmo Random-Search é o mais simples dos outros algoritmos mencionados pois usa ou uma aleatoriedade ou uma probabilidade na definição do método. Os algoritmos Random-Search costumam ser uteis em problemas de otimização onde a função objetivo pode ser não convexa, não diferenciável e possivelmente descontínuo sobre um domínio continuo, discreto ou contínuo-discreto.

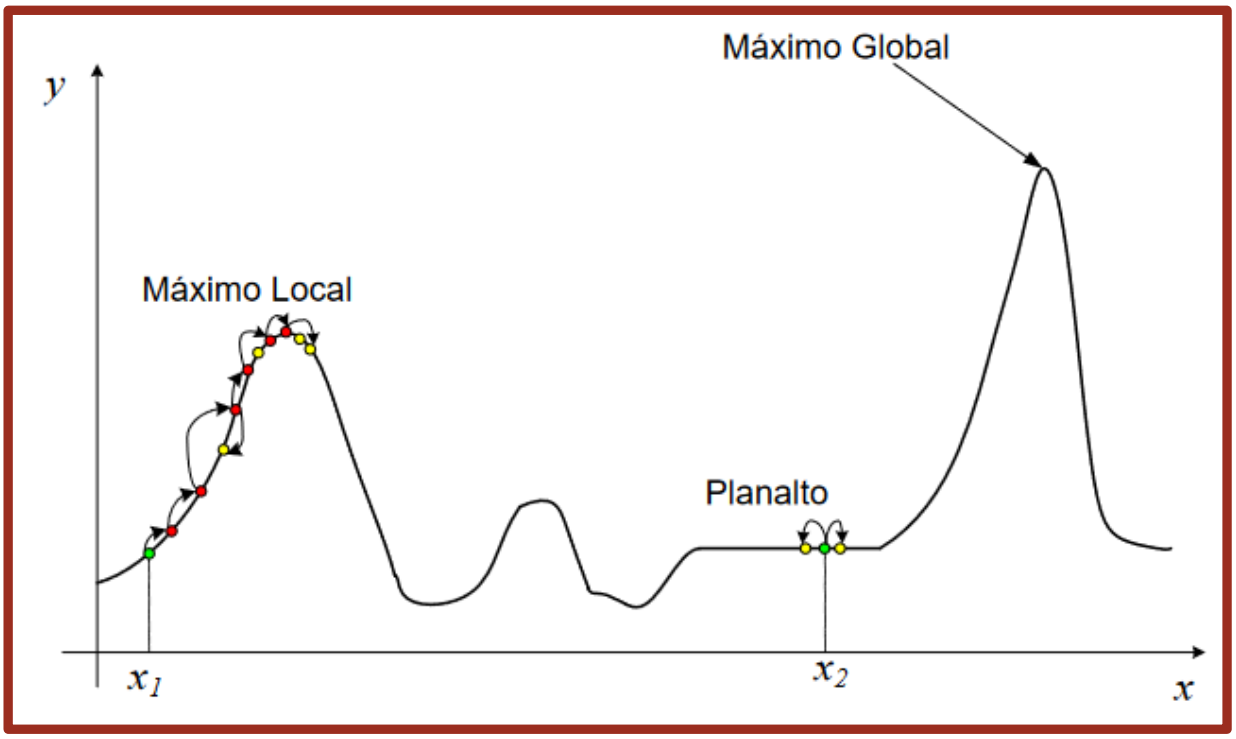
Este algoritmo é dividido em 3 pontos principais:

1. Seleção de um ponto inicial aleatório;
2. Alteração local do estado atual;
3. Repetição do passo 2 até o objetivo ou o critério seja atingido.

Algoritmo Hill-Climbing (Subida da Colina)

O algoritmo Hill-Climbing (Subida da Colina) como já falado é uma técnica de busca local, com o objetivo de propor uma configuração e modificá-la até a obtenção de uma solução. Este algoritmo é um tipo de algoritmo de busca local, ou seja, em cada momento o algoritmo considera somente os estados imediatamente acessíveis a partir do estado atual.

Assumindo que o tipo de função é de maximização, o seu funcionamento consiste em subir a "colina" até ao encontro da solução ótima (Cortez, 2014). De uma forma iterativa, vai procurando sempre novas soluções na vizinhança e caso essa seja melhor escolhe essa como a nova, caso contrário é escolhido um outro vizinho e é verificado se a nova solução é a melhor. A execução deste algoritmo termina quando se atinge um critério de paragem, tal como não se encontrar uma melhoria na melhor solução ao fim de algumas iterações ou quando passa um dado tempo limite de execução (Michalewicz et al., 2006).

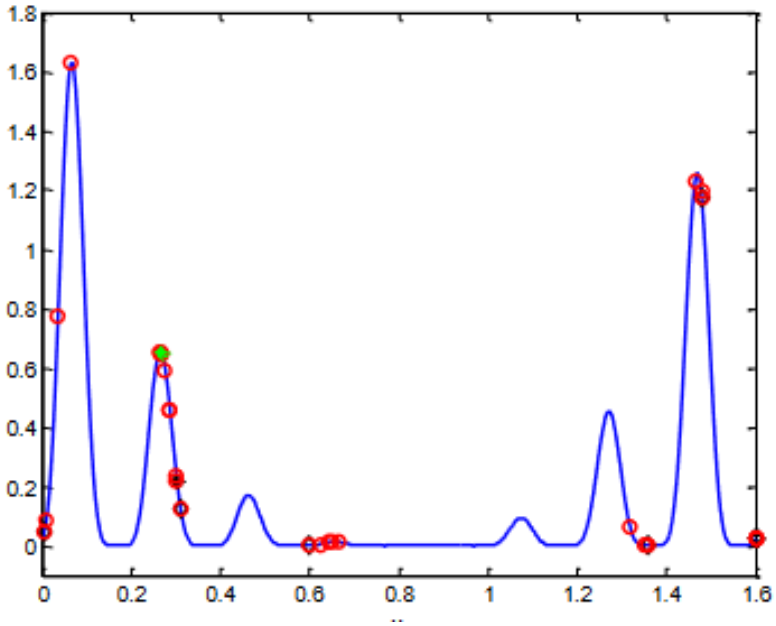


Pseudocódigo do Hill Climbing

```
t = 0
initialize x(t) randomly
while(!(termination criterion))
   generate a new random solution x_new
   if f(x_new) > f(x(t))
      x(t)=x_new
      f(x(t))=f(x_new)
   end if
   t = t + 1
end while
```

Algoritmo Hill-Climbing with Multiple Re-Start (Subida da Colina com Reinicialização Múltipla)

O algoritmo Hill-Climbing with Multiple Re-Start (Subida da Colina com Reinicialização Múltipla) é uma variação do algoritmo de Hill-Climbing onde executa uma série de buscas Hill-Climbing a partir de estados iniciais aleatórios. O ciclo só acaba quando terminar um número finito de iterações ou continuar até não conseguir melhorar o melhor valor encontrado, salvando o melhor resultado obtido até então, geralmente uma solução boa pode ser encontrada em um número pequeno de iterações.



Pseudocódigo do Hill Climbing with Multiple Re-Start

```
t = 0
initialize x(t) randomly
best_x=x(t)
best_f=f(t)
while(!(termination criterion))
   generate a new random solution x_new
   if f(x_new) > f(x(t))
      x(t)= x_new
      f(x(t))=f(x_new)
      if f(x(t)) > f(best_x)
         best_x=x(t)
         best_f=f(x(t))
      end if
   end if
   evaluate search stagnation criterion
   if search stagnated
      initialize x(t) randomly
      update f(x(t))
   end if
   t = t + 1
end while
```

Algoritmo Simulated Annealing

O Simulated Annealing (SA), ou em português Arrefecimento Simulado é um dos primeiros algoritmos de otimização moderna (C. Blum & Roli, 2003). É um método que é semelhante ao Hill Climbing, sendo este inspirado no arrefecimento de metais, onde se assume que um metal é inicialmente aquecido e depois é realizado um arrefecimento de uma forma controlada até a obtenção de um metal num estado solido (Cortez, 2014).

O seu funcionamento consiste em escolher um valor alto para a variável T (frequentemente referido como temperatura) e uma solução aleatória inicial. Em cada iteração, faz-se uma procura de Hill Climbing, mas com um resultado probabilístico que depende da temperatura. Assim, a escolha da solução a adotar é inicialmente mais aleatória para elevadas temperaturas, mas à medida que o processo evoluí, e a temperatura arrefece, o processo tende a ser mais determinístico (Michalewicz et al., 2006).

Resultados

O problema proposto pelos regentes da Unidade Curricular foi através da função bidimensional apresentada na figura a seguir implementar os diversos algoritmos aqui apresentados utilizando a ferramenta Matlab.

$$f_1(x,y) = a(1-x^2)exp(-x^2-(y+1)^2),$$
$$f_2(x,y) = -b\left(\frac{x}{5} - x^3 - y^5\right)exp(-x^2-y^2),$$
$$f_3(x,y) = -\frac{1}{3}exp(-(x+1)^2-y^2),$$
$$f(x,y) = |f_1(x,y) + f_2(x,y) + f_3(x,y)|,$$

Ao longo deste relatório iremos designar a função 1 que contém o ("a = 3 e b = 10") e a função 2 a que contém o ("a = 9 e b = 5"). Relembrando que o intervalo das variáveis das duas funções são:

$$-3 \leq x,y \leq +3,$$

Algoritmo Random Search

Este algoritmo foi executado uma vez para cada função com 1000 iterações apresentando os seguintes resultados nas duas funções:

Função 1

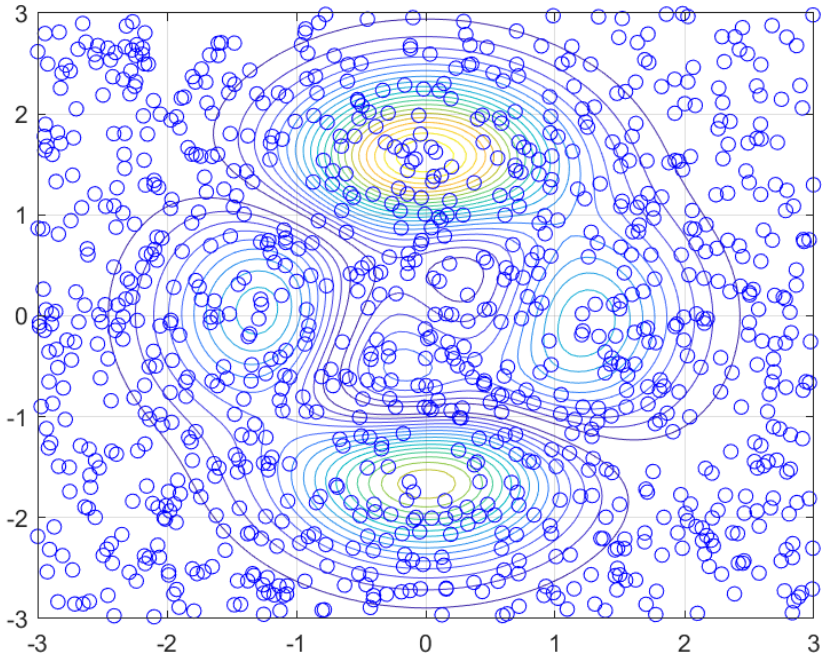
Como é possível observar nas duas figuras o teste foi bem-sucedido pois foi atingido o *global maximum*¸ atingindo também dois *local maximum*.

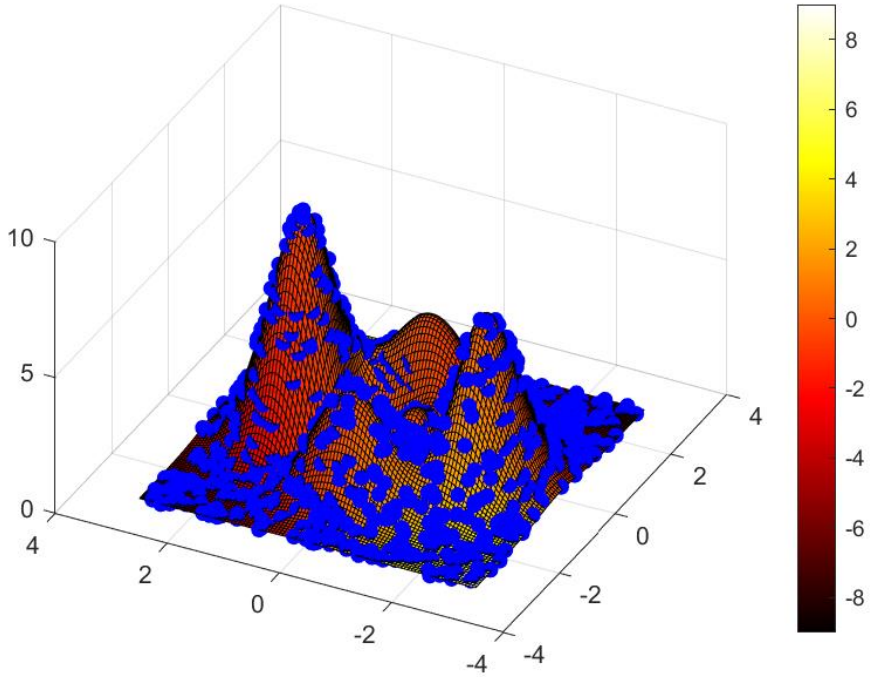
Função 2

Nas outras duas figuras da outra função constata-se que o teste foi parcialmente bem-sucedido visto que neste teste só foi atingido a vizinhança do *global maximum* e não atingindo nenhum *local maximum*.

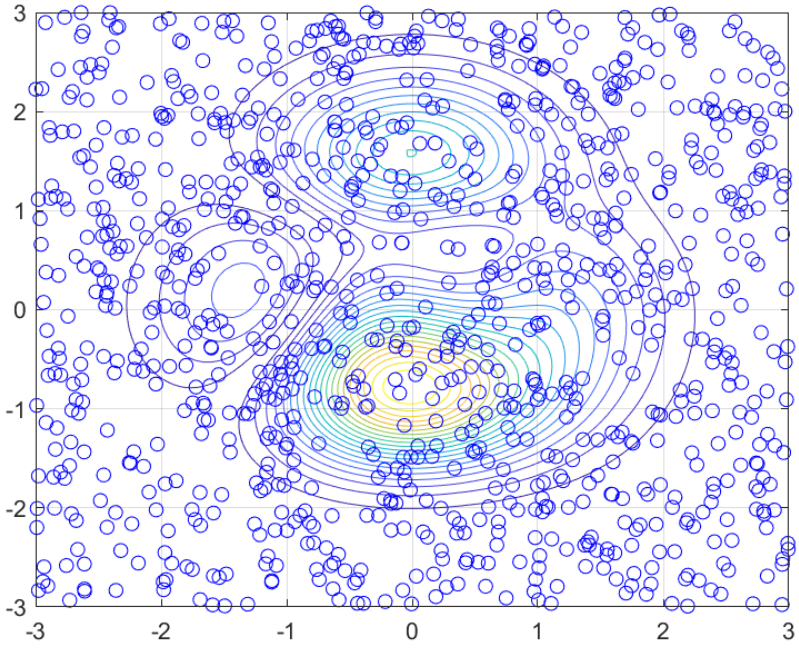
Assim podemos concluir que este algoritmo é instável e é dependente do numero de iterações que o ciclo contém pois com um número de iterações elevado, maior serão as chances do teste ser bem-sucedido.

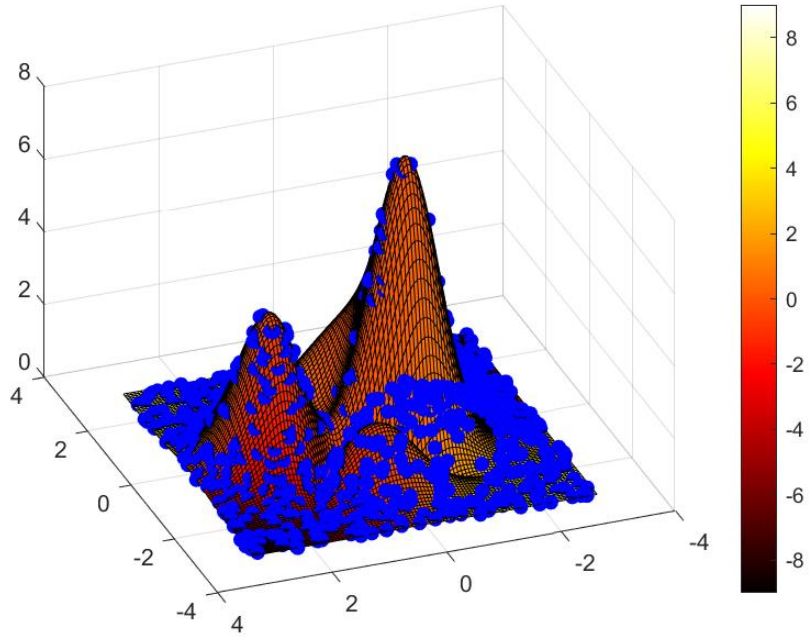
Função 1





Função 2





Algoritmo Hill-Climbing (Subida da Colina)

Este algoritmo irá ter dois testes para cada função tendo 1000 iterações para a função 1 e 500 iterações para a função 2.

Função 1 (1ºTeste)

No primeiro teste realizado à Função 1 conclui-se que o teste não foi bem-sucedido pois este não atingiu o *global maximum* atingindo assim um dos *local maximum* que com ajuda do terceiro gráfico tem coordenadas X=0.014 Y=-1.661 e Z = 6.080.

Função 1 (2ºTeste)

Neste segundo teste, com base nos gráficos podemos caracterizar o teste como bem-sucedido tendo este atingido o *global maximum* com coordenadas X=0 Y=1.577 e Z = 8.106.

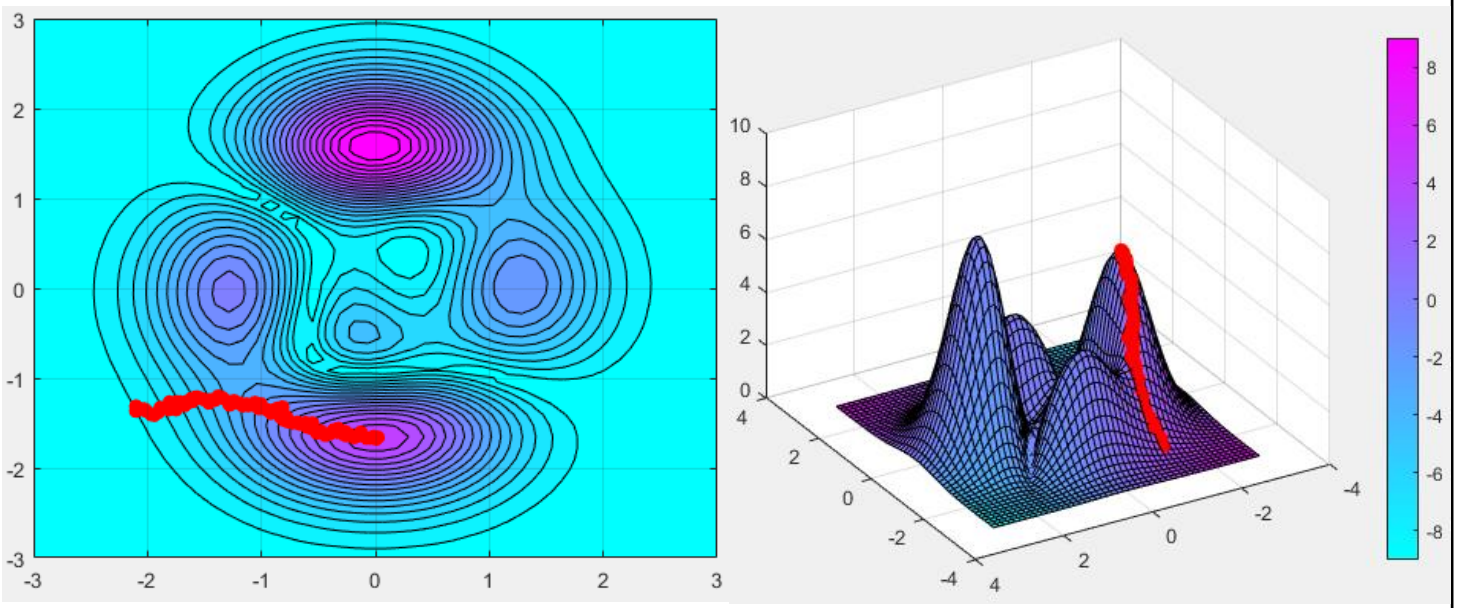
Função 2 (1ºTeste)

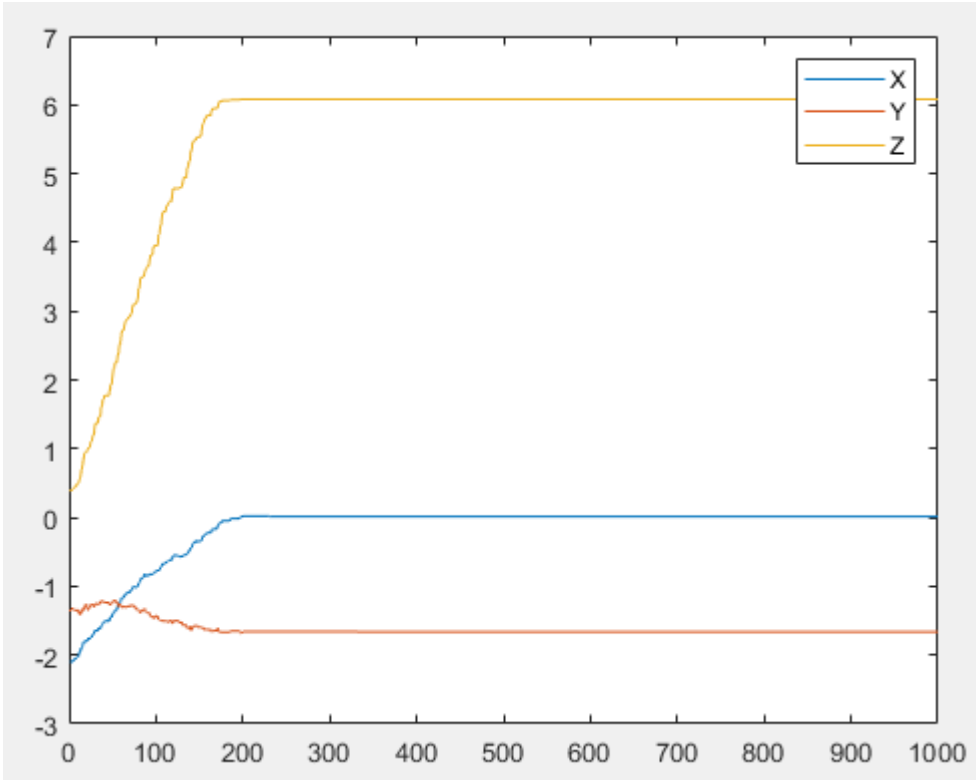
Tal como o primeiro teste da Função 1, o teste não foi bem-sucedido pois este atingiu um dos *local maximum* com coordenadas X=-0.01 Y=1.582 e Z = 4.057.

Função 2 (2ºTeste)

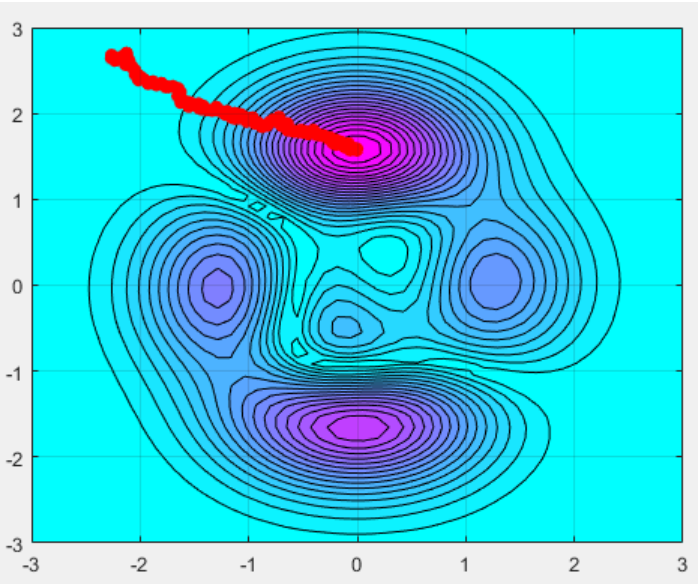
O segundo teste foi falhado para além de não atingir o *global maximum* o algoritmo não atingiu nenhum dos *local maximum*. Conclui-se que o número de iterações foi muito limitado neste teste e as coordenadas do ponto inicial aleatório estão à beira do limite da função.

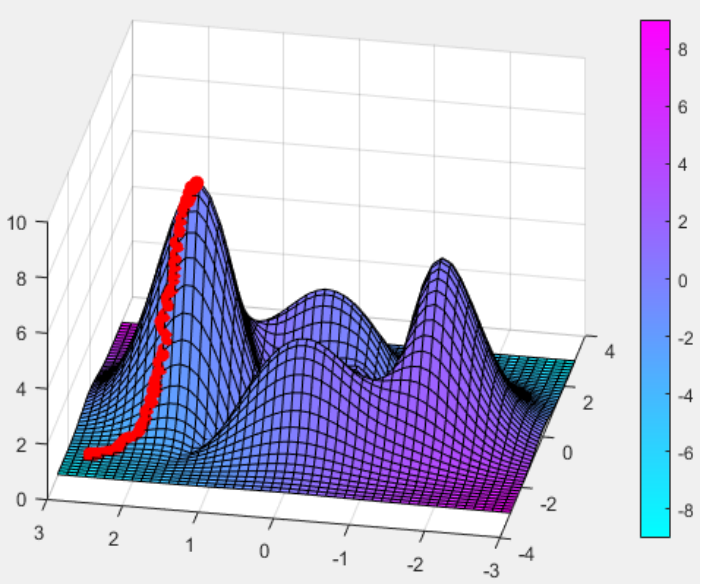
Função 1 (1ºTeste)

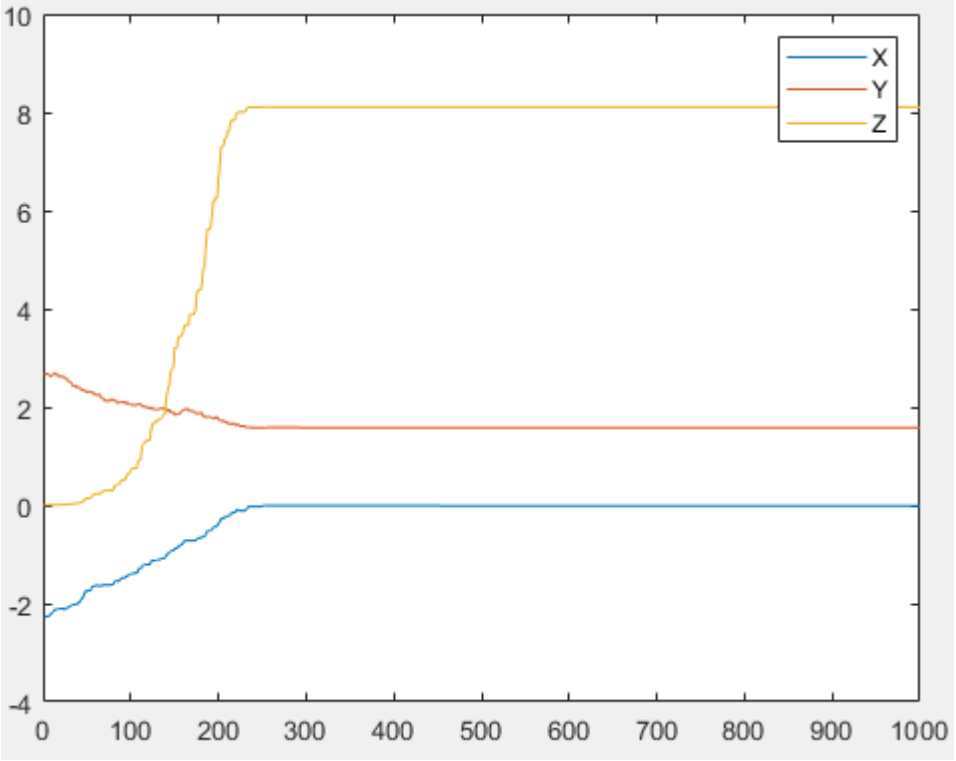




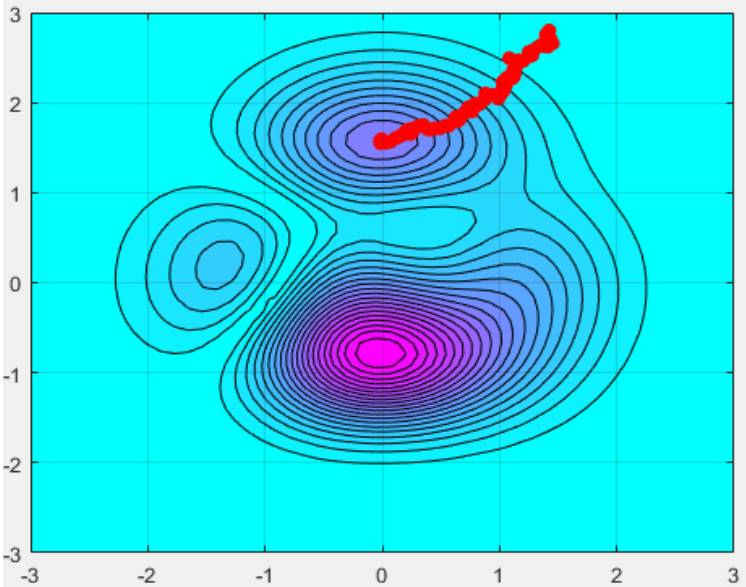
Função 1 (2ºTeste)

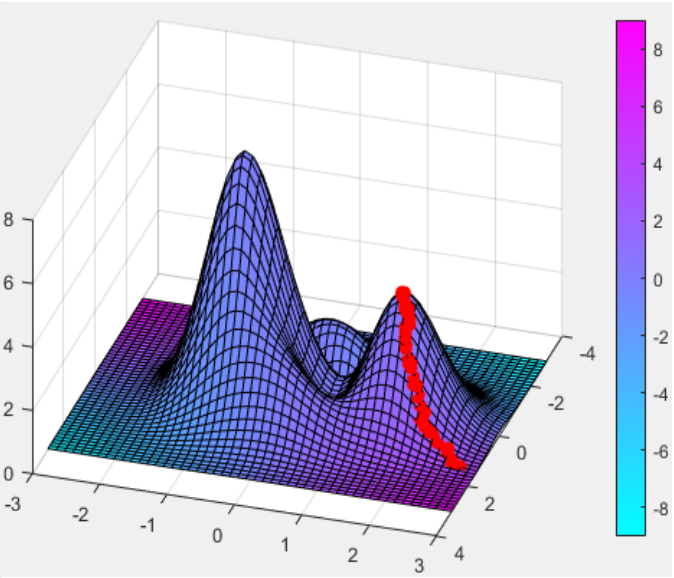


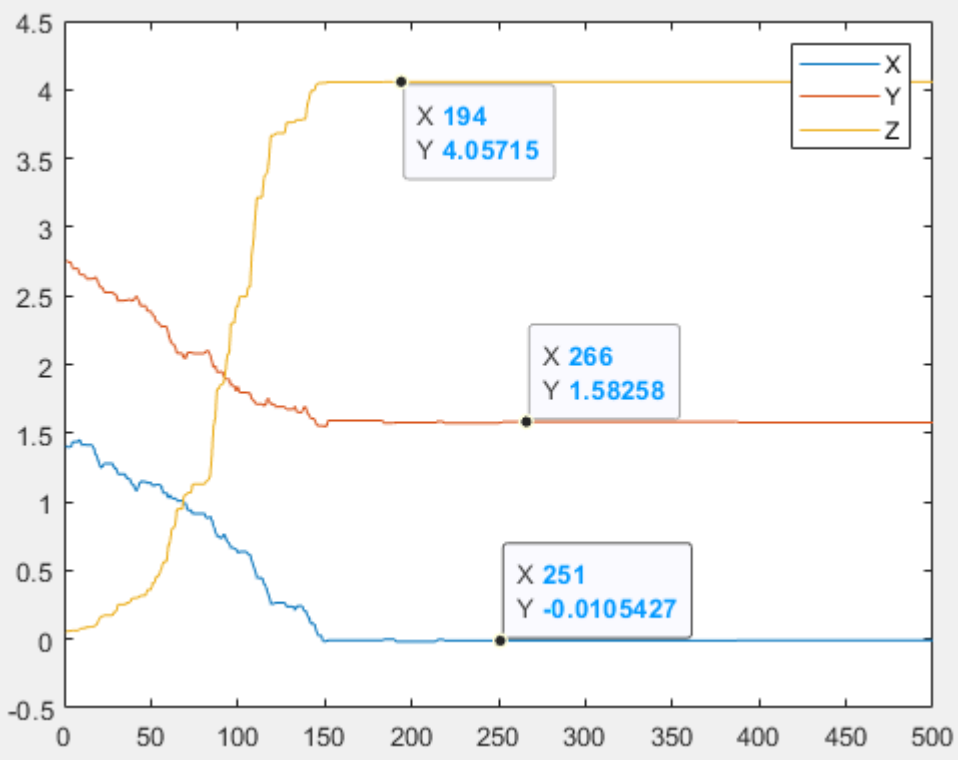




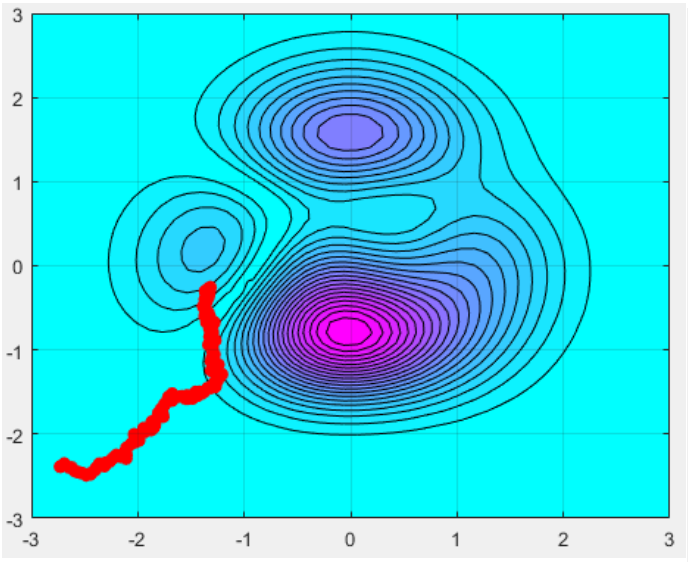
Função 2 (1ºTeste)

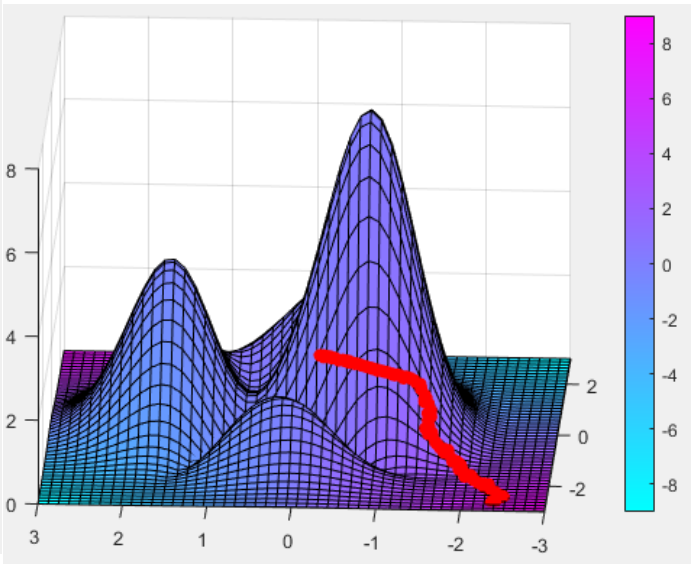


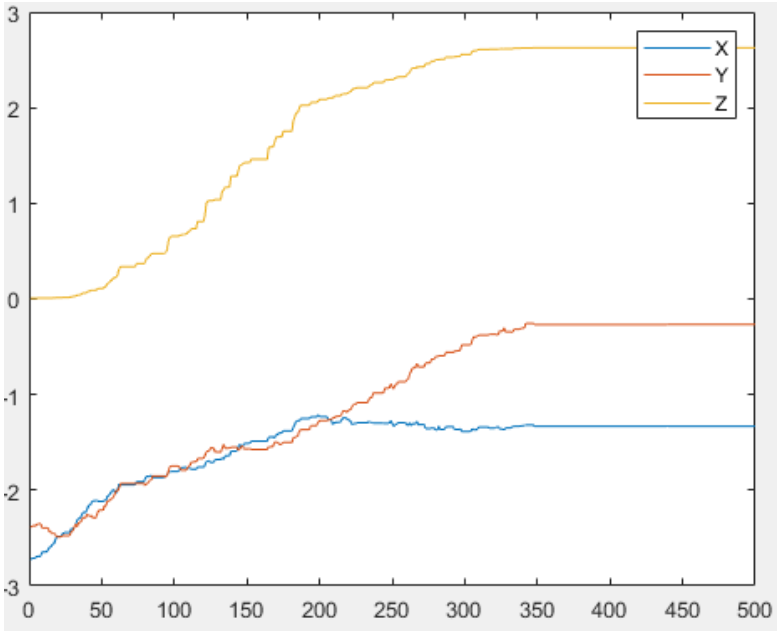




Função 2 (2ºTeste)







Algoritmo Hill-Climbing with Multiple Re-Start (Subida da Colina com Reinicialização Múltipla)

Este algoritmo sendo variante do algoritmo Hill-Climbing foi feito com dois ciclos, para a Função 1 o primeiro ciclo contém 10 iterações e o segundo ciclo 100 iterações, para a Função 2 o primeiro ciclo contém 10 iterações e o segundo ciclo 50 iterações, concluindo assim que iremos ter 10 resultados para cada função. Foram executados dois testes para cada função.

Função 1(1º Teste)

No primeiro teste da Função 1 foi parcialmente bem-sucedido visto que nas 10 iterações só uma é que atingiu o *global maximum* e oito atingido o *local maximum*.

Função 1(2º Teste)

O segundo teste correu melhor que o 1ºteste pois este atingiu duas vezes o *global maximum* e três vezes o *local maximum*.

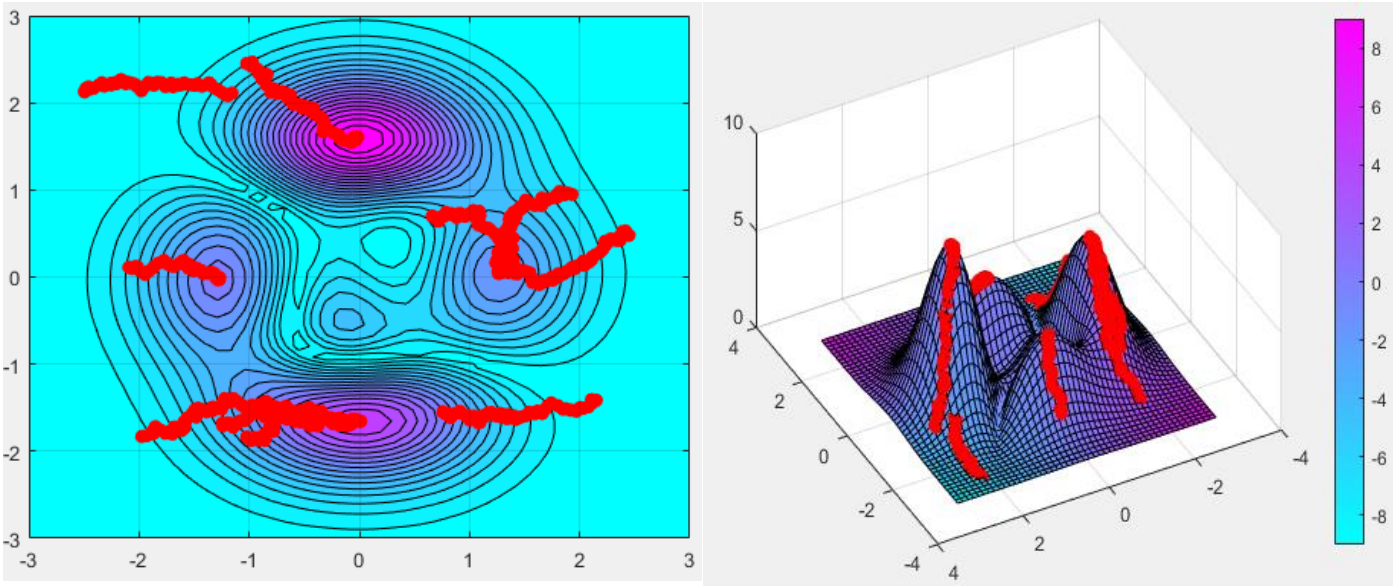
Função 2(1º Teste)

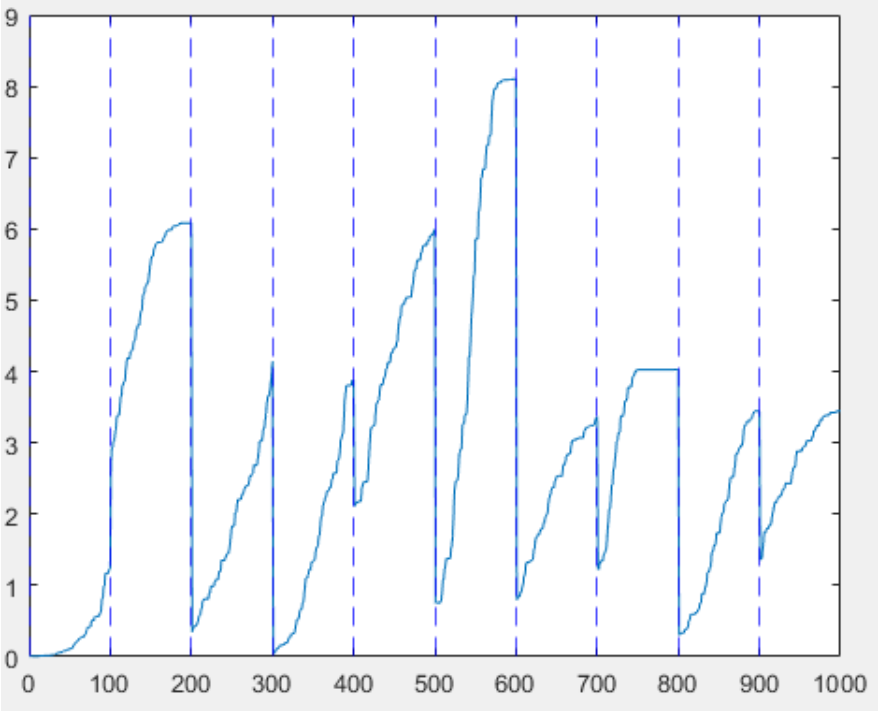
O primeiro teste foi parcialmente bem-sucedido visto que nas iterações só uma é que atingiu o *global maximum* e sete atingido o *local maximum*.

Função 2(2º Teste)

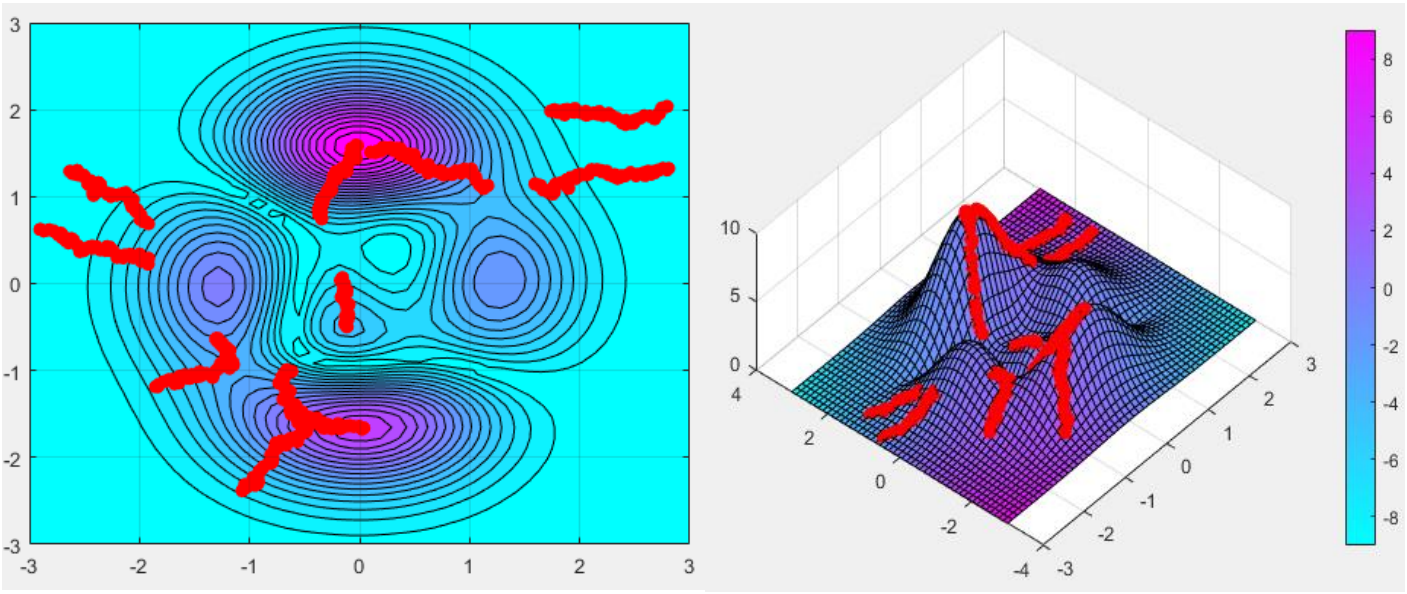
O segundo teste correu pior que o 1ºteste desta função porque este teste não atingiu o global maximum e três vezes atingindo o local maximum.

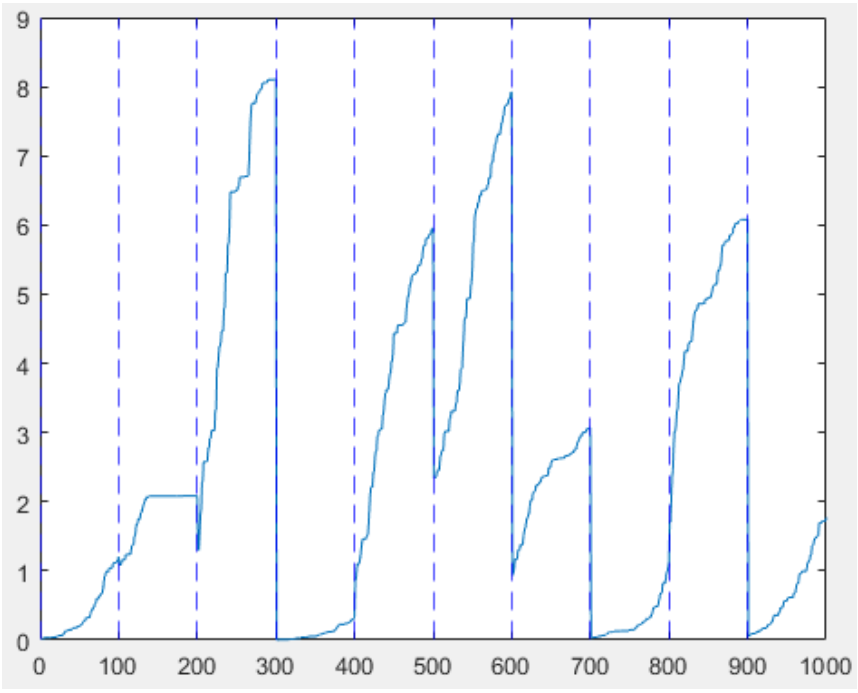
Função 1(1º Teste)



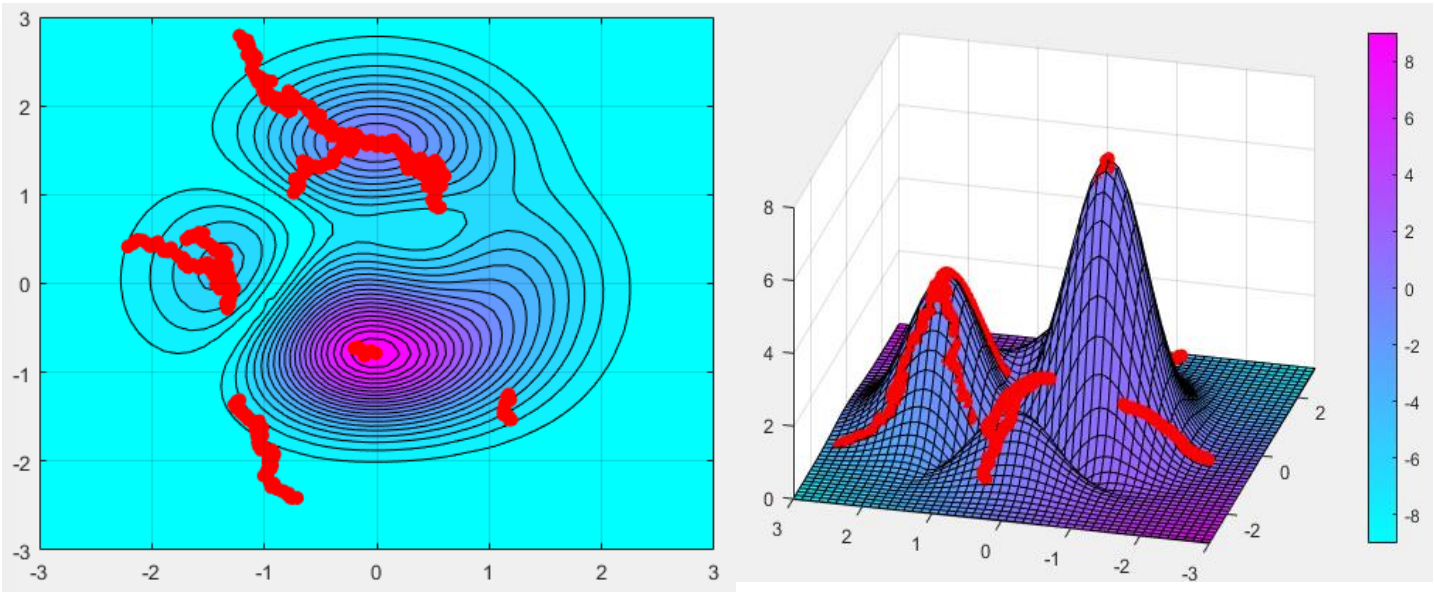


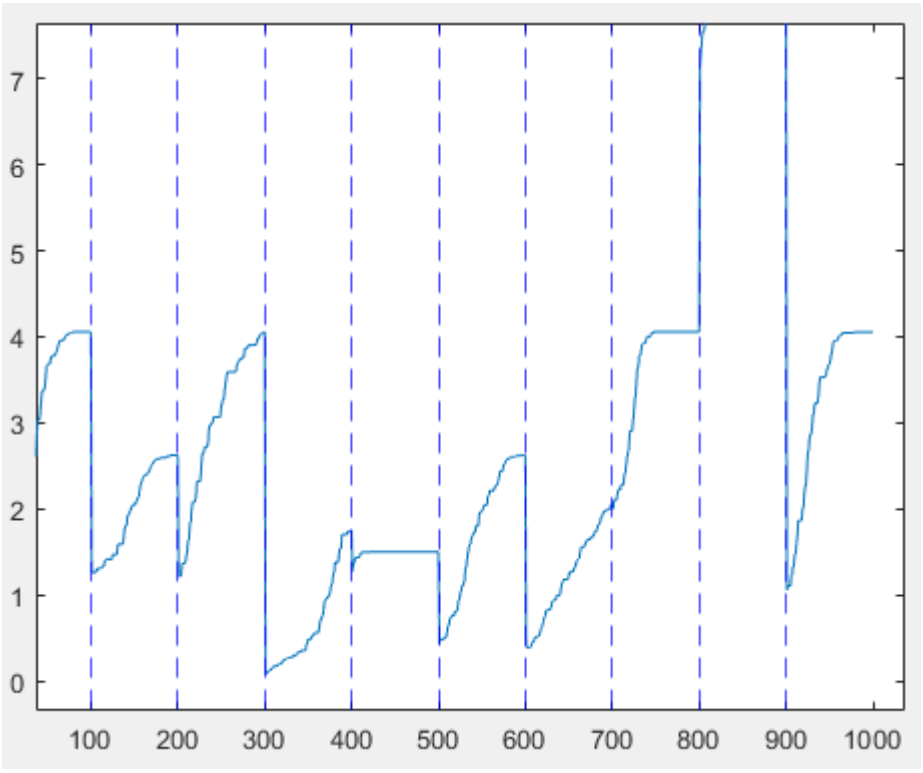
Função 1(2º Teste)



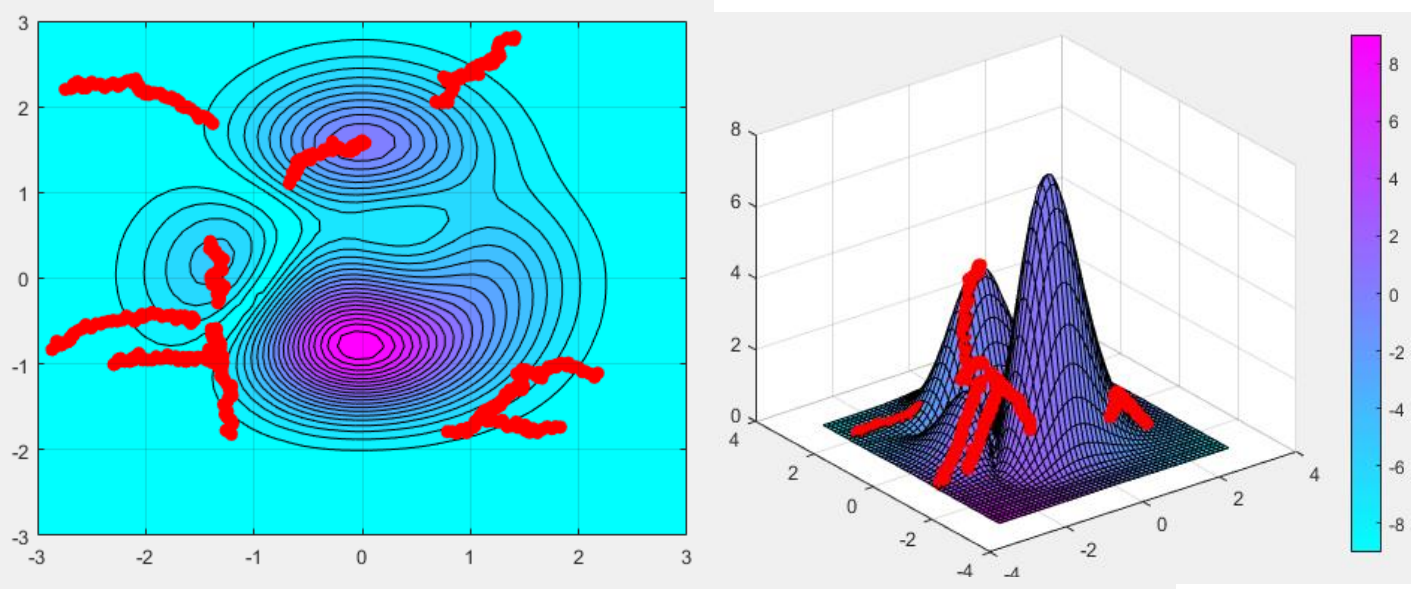


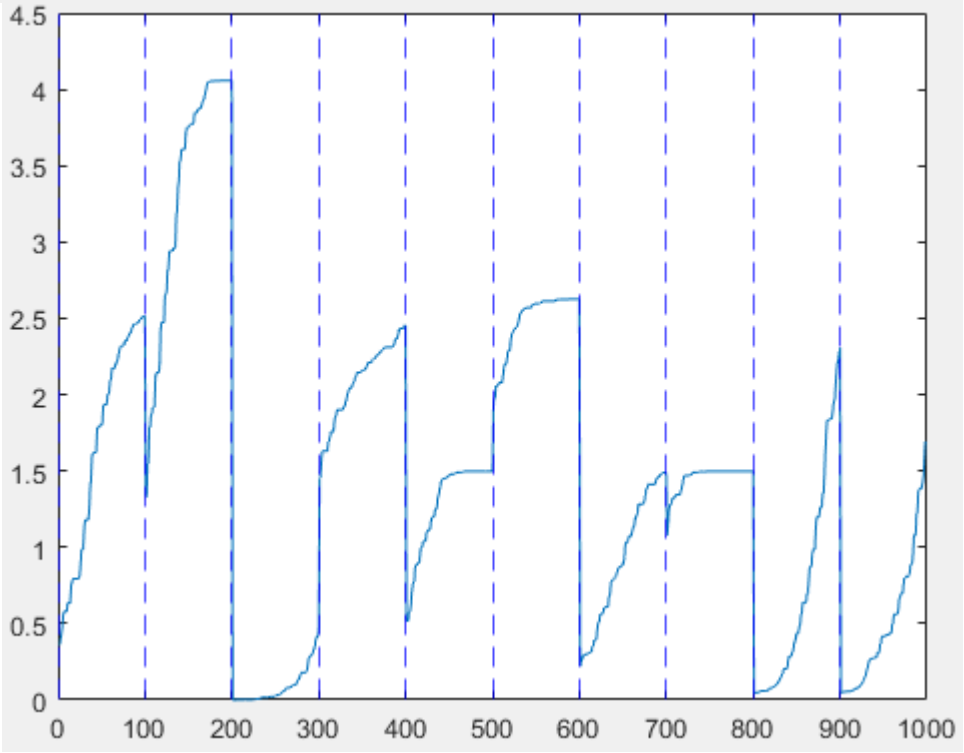
Função 2(1º Teste)





Função 2(2º Teste)





Algoritmo Simulated Annealing com Restart

Este algoritmo sendo variante do algoritmo Hill-Climbing foi feito com dois ciclos, para a Função 1 o primeiro ciclo contém 10 iterações e o segundo ciclo 100 iterações, para a Função 2 o primeiro ciclo contém 10 iterações e o segundo ciclo 50 iterações, concluindo assim que iremos ter 10 resultados para cada função. Foram executados dois testes para cada função, sendo o gráfico a azul a temperatura ao longo das 10 iterações e o gráfico a vermelho a probabilidade ao longo das 10 iterações.

Função 1(1º Teste)

No primeiro teste da Função 1 foi parcialmente bem-sucedido visto que nas 10 iterações só uma é que atingiu o global maximum e uma atingido o local maximum.

Função 1(2º Teste)

O segundo teste correu melhor que o primeiro teste pois este nas 10 iterações duas delas atingiram o global maximum e uma atingiu o local maximum.

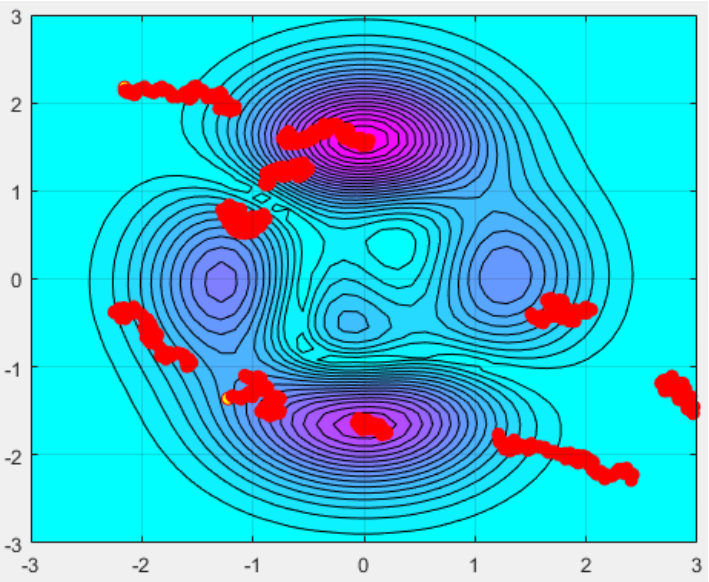
Função 2(1º Teste)

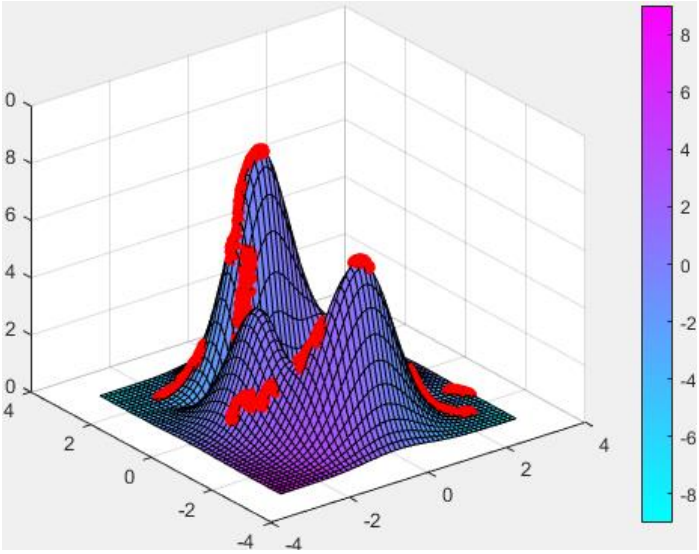
O primeiro teste foi parcialmente bem-sucedido visto que nas iterações só uma é que atingiu o *global maximum* e nenhuma atingiu o *local maximum*.

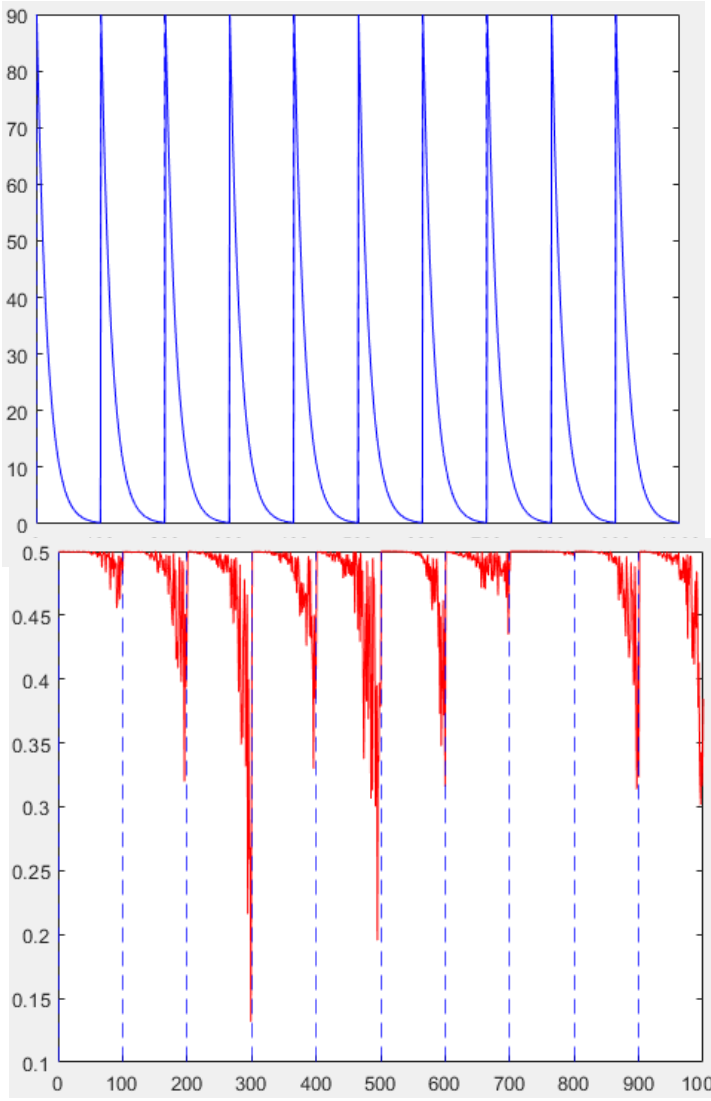
Função 2(2º Teste)

O segundo teste correu pior que o 1ºteste desta função porque este teste não atingiu o global maximum e uma vez atingindo o local maximum.

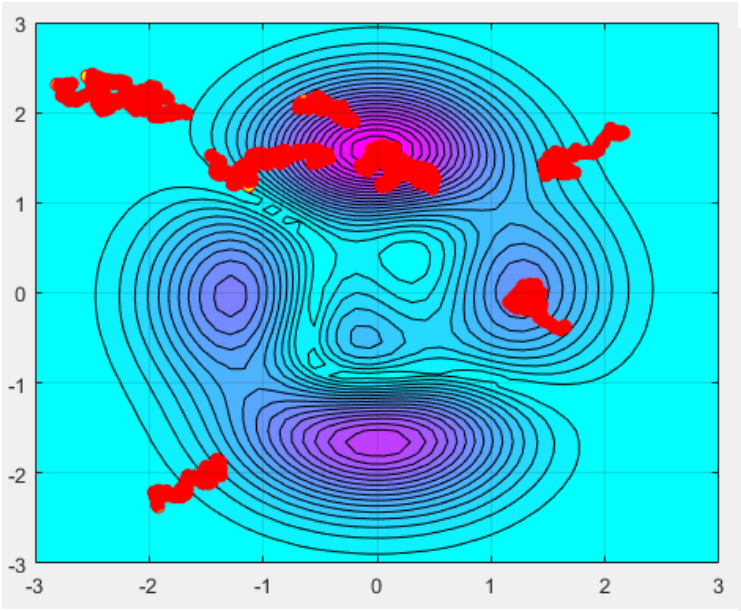
Função 1(1º Teste)

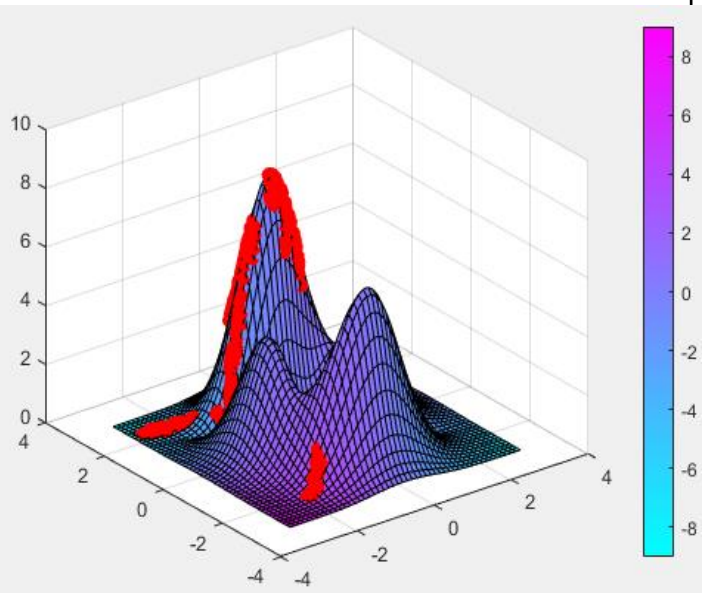


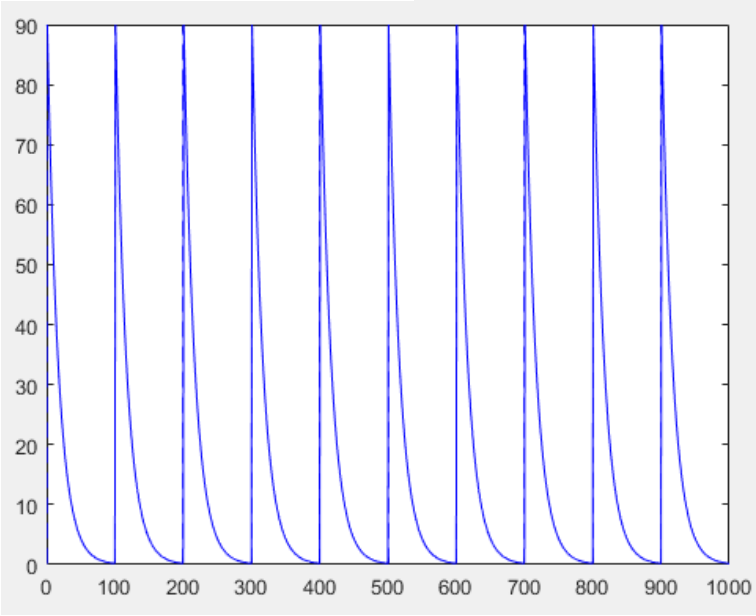


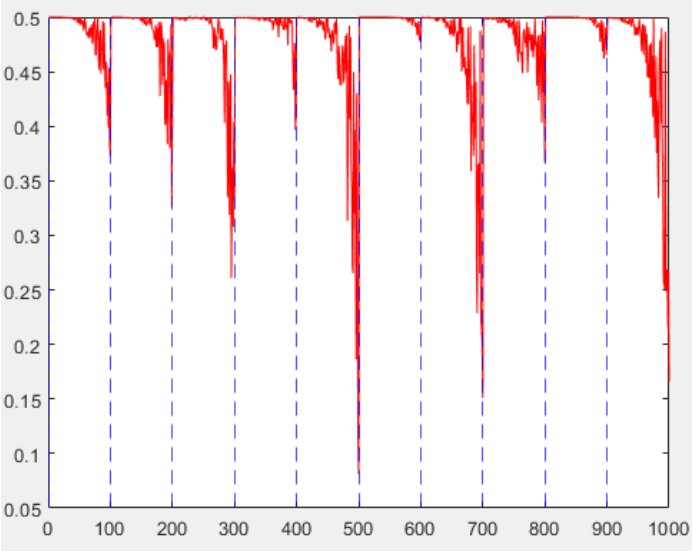


Função 1(2º Teste)

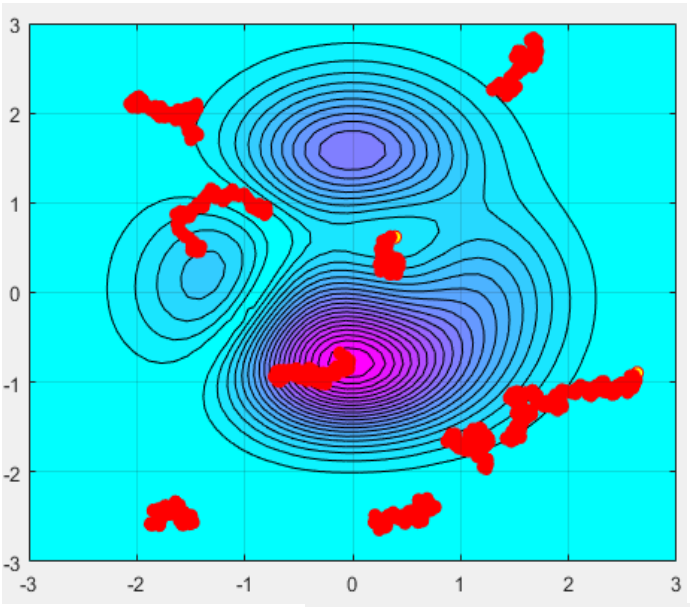


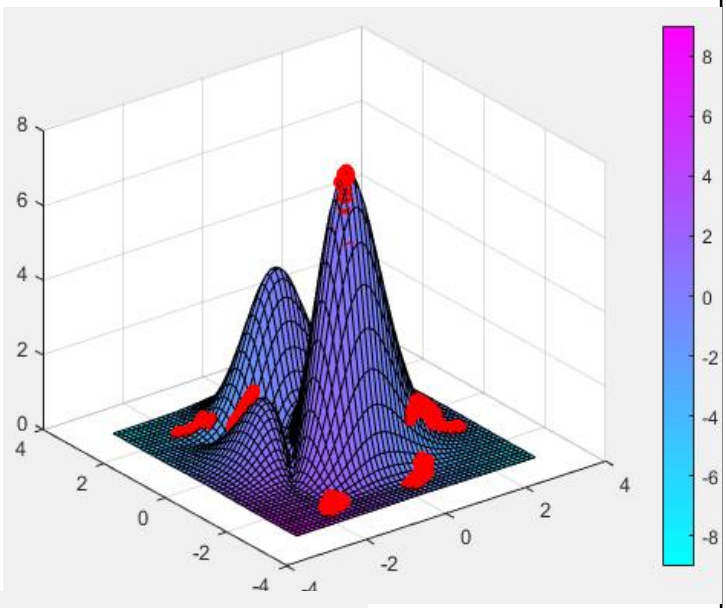


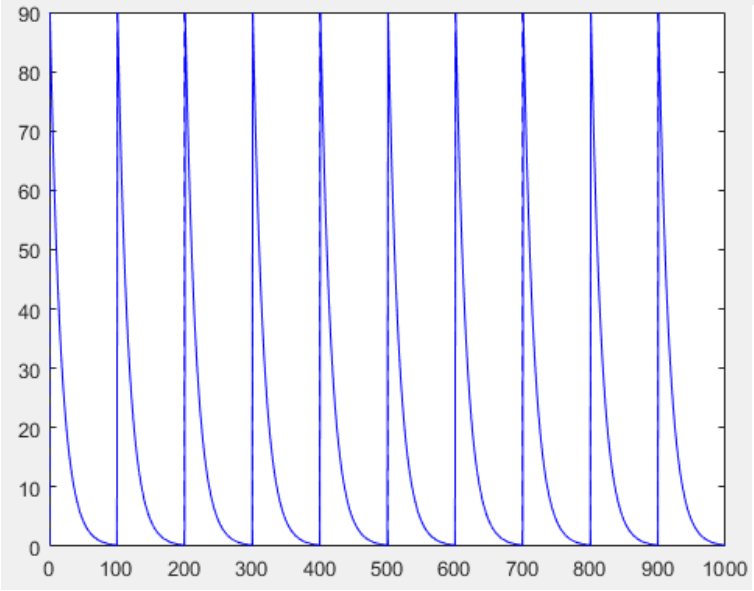


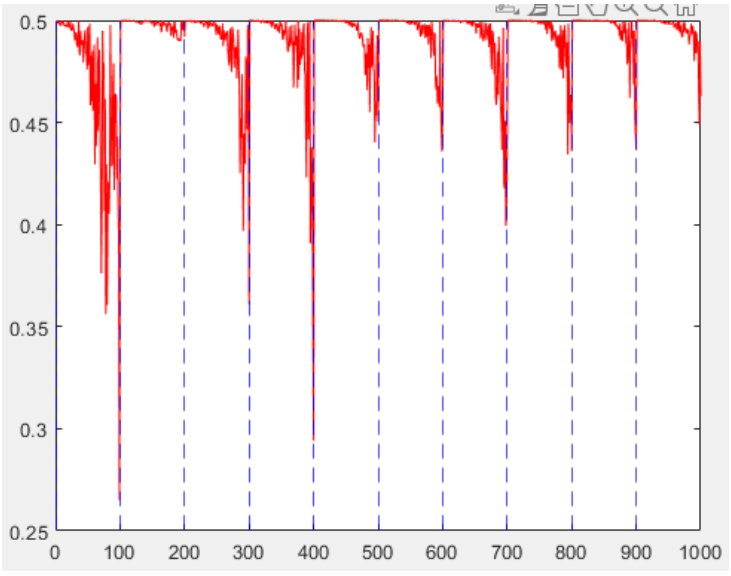


Função 2(1º Teste)

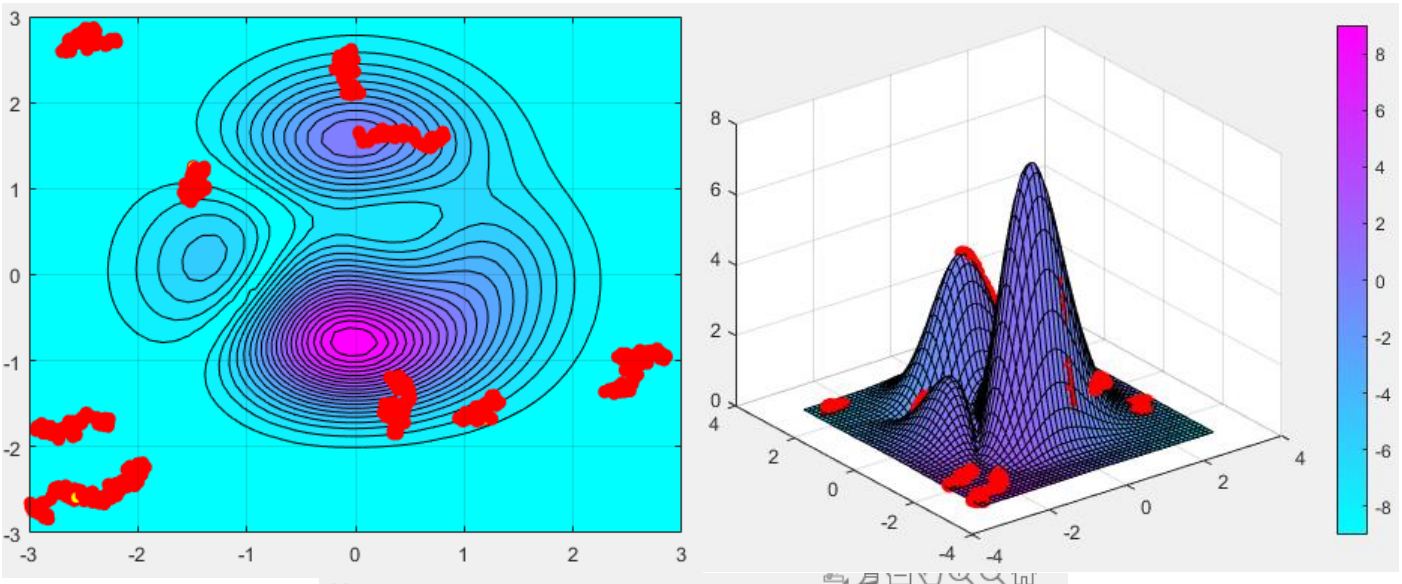


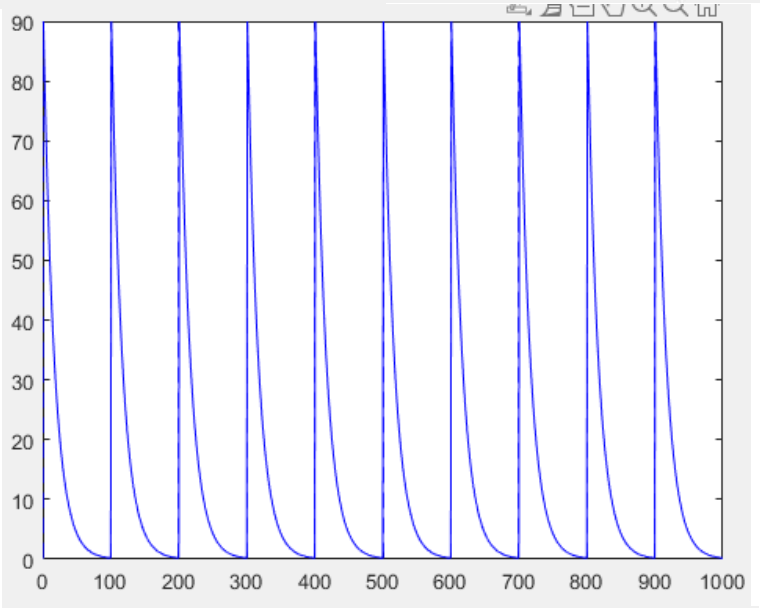


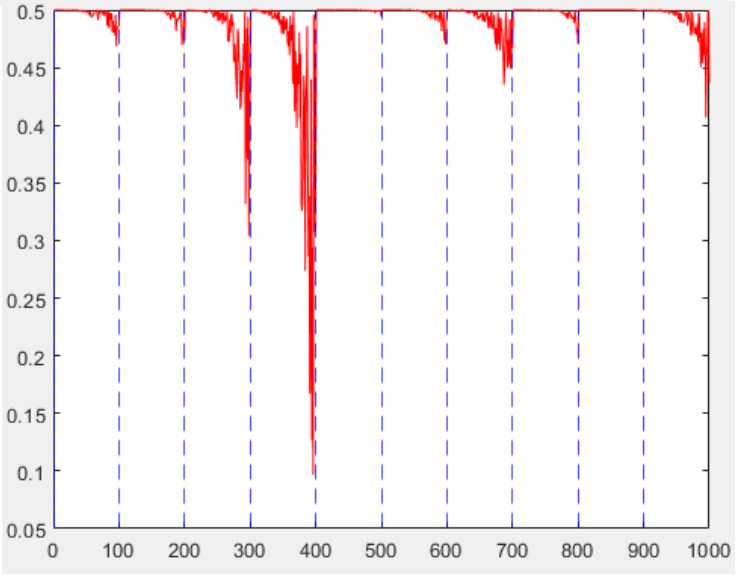




Função 2(2º Teste)







Conclusão

Concluindo este relatório pode admitir-se que foi atingido o objetivo inicial proposto no relatório. Neste projeto aprendemos no software Matlab a conhecer melhor os dois algoritmos. Em relação ao Hill-Climbing posso concluir que é um algoritmo que contém falhas quando este nem sempre consegue atingir o *global maximum*, mas recorrendo à sua variante da Reinicialização Múltipla foi possível solucionar este problema. No caso do SA posso constatar que é um código mais difícil de implementar do que o Hill-Climbing, onde só podemos atingir o global maximum e obter uma maior chance de um teste bem-sucedido se tivermos um elevado número de iterações no ciclo.

Bibliografia

Michalewicz, Z., Schmidt, M., Michalewicz, M., & Chiriac, C. (2006). Adaptive business intelligence. Adaptive Business Intelligence.

Blum, C., & Roli, a. (2003). Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Computing Surveys,

Cortez, P. (2014). Modern Optimization with R. Guimarães: Springer.