Laboratório 3 - Otimização com Métodos de Busca Local

Marcelo Buga Martins da Silva

CT-213 - Professor Marcos Ricardo Omena de Albuquerque Máximo

31/03/2021

1 Implementação

Três diferentes algoritmos foram implementados para a otimização dos parâmetros v_0 e f, que representam a velocidade inicial de um experimento com uma bolinha e a aceleração provocada pelo atrito nela, respectivamente. A equação 1 representa o problema:

$$v(t) = v_0 + ft \tag{1}$$

Os algoritmos implementados foram *Gradient Descent*, *Hill Climbing* e *Simmulated Annealing*. Como base para comparação, foi utilizado o método dos mínimos quadrados que, para esse caso, fornece solução analítica ao problema. O detalhamento de cada implementação é dado a seguir

1.1 *Gradient Descent*

A função Gradient Descent parte da utilização do vetor gradiente para ir no sentido contrário ao gradiente da função para diminuir ao máximo o valor da função para chegar a um mínimo rapidamente. Como a função de erro (Equação 2) é facilmente derivável (Equação 3), é possível utilizar esse método sem grandes problemas para implementação, sendo apenas necessário limitar o número de iterações e um limite inferior de custo para a convergência do algoritmo.

$$J(v_0, f) = \frac{1}{2m} \sum_i (v_0 + f t_i - v_i)^2 \tag{2}$$

$$\nabla J = \frac{1}{m} \left(\sum_i (v_0 + f t_i - v_i) \, , \sum_i (v_0 + f t_i - v_i) \, t_i \right) \tag{3}$$

O maior problema dessa implementação é a possibilidade de se convergir para um mínimo local, em vez do global. Entretanto, isso não ocorreu para esse problema. O caminho percorrido para essa otimização é apresentado visualmente pela Figura 1.

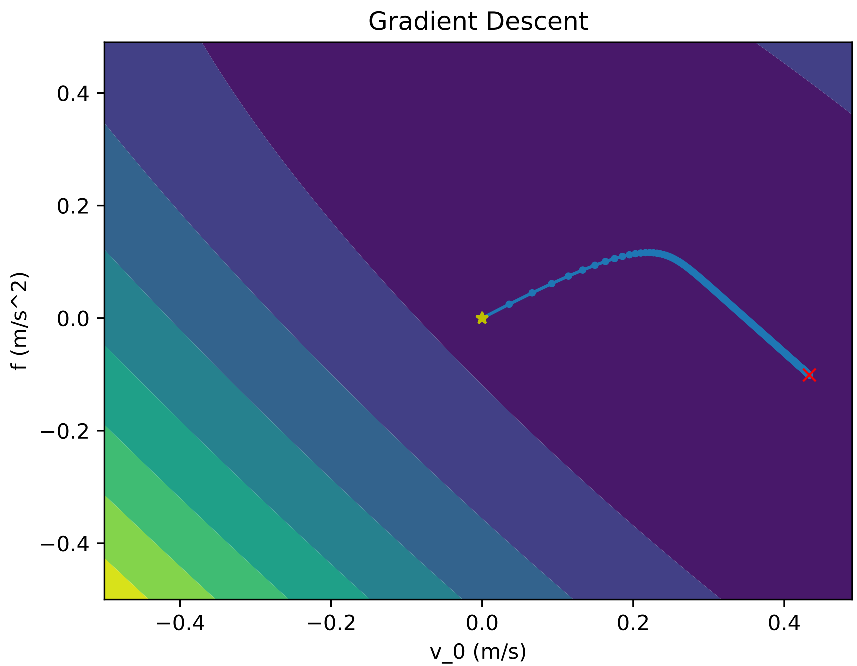

Figura 1: Caminho de otimização *Gradien Descent*

1.2 *Hill Climbing*

Para a função *Hill Climbing* foi necessário, também, criar a função *neighbors* que lista os vizinhos de um ponto (v_0, f) a uma distância Δ. Foi escolhido o modelo 8 conectado, de forma que os vizinhos horizontais, verticais e diagonais em 45° são listados.

A função *Hill Climbing*, então, basicamente encontra o vizinho de menor custo entre os vizinhos de um certo ponto, passa para ele e avalia os vizinhos novamente. A função também foi implementada com limites de iterações e de custo para a convergência do algoritmo.

Esse algoritmo sofre do mesmo problema que *Gradient Descent*: existe a possibilidade da execução ficar presa em um mínimo local. Entretanto, isso não ocorreu, como pode ser evidenciado pela Figura 2.

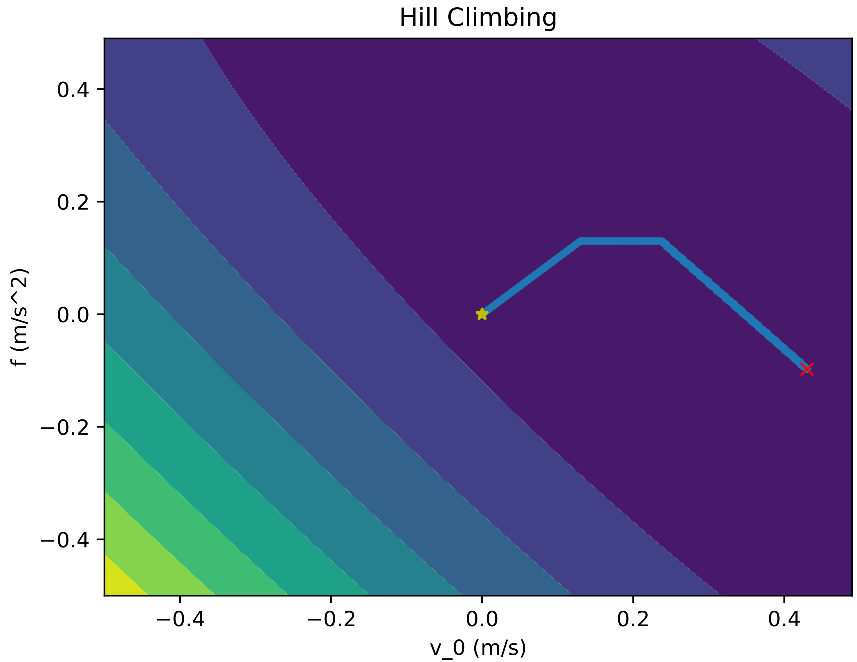

Figura 2: Caminho de otimização *Hill Climbing*

1.3 *Simulated Annealing*

Por fim, o algoritmo *Simmulated Annealing* muito se assemelha com *Hill Climbing*, porém utiliza um fator randômico para evitar se limitar a mínimos locais. Para essa implementação, foram necessárias as criações das funções *random_neighbor*, que escolhe um vizinho aleatório a uma distância Δ do ponto atual, e *schedule*, que determina a "temperatura"de acordo com a Equação 4, em que i é o número da iteração.

$$T = \frac{T_0}{1 + \beta i^2} \tag{4}$$

Basicamente, esse algoritmo pega um vizinho aleatório do ponto atual e vai para ele caso seu custo seja inferior ao custo atual (como *Hill CLimbing*) ou caso um número tomado aleatoriamente entre 0 e 1 seja superior a um número que decresce com o decrescimento da temperatura, fazendo com que essa mudança seja cada vez mais improvável. A função tem a mesma implementação de limites descrito nos outros dois algoritmos. Uma representação do caminho percorrido por essa otimização é apresentada na Figura 3, sendo o comportamento aleatório (mais evidente no início) representado pela Figura 4.

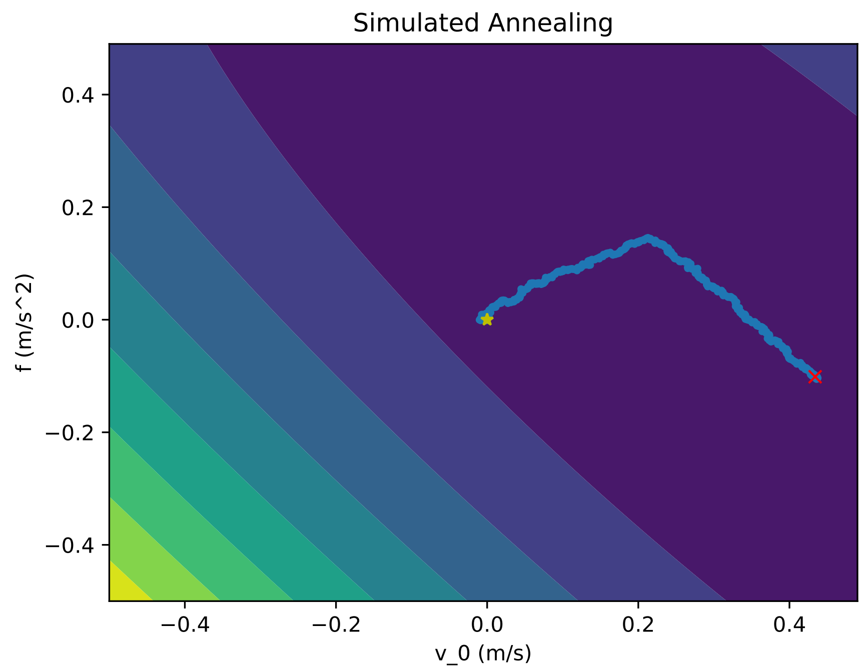

Figura 3: Caminho de otimização *Simulated Annealing*

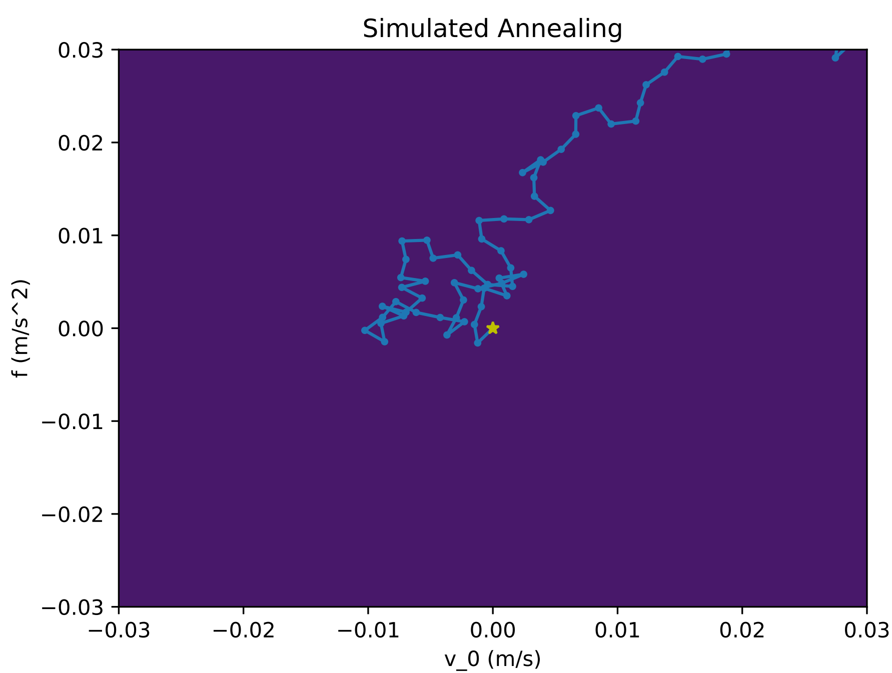

Figura 4: Zoom no início do caminho de otimização *Simulated Annealing*

2 Comparação

A Tabela 1 compara os valores otimizados de v_0 e f para cada algoritmo, assim como pelo método dos mínimos quadrados.

Tabela 1: Comparação de performance dos algoritmos

Algoritmo	v_0	f
Mínimos Quadrados	0.43337277	-0.10102096
Gradient Descent	0.43337067	-0.10101846
Hill Climbing	0.43010765	-0.09789235
Simulated Annealing	0.43397656	-0.10134529

Pode-se preceber que *Gradient Descent* teve o resultado mais próximo do calculado analiticamente pelo método dos mínimos quadrados, enquanto *Hill Climbing* teve o resultado mais distante. As Figuras 5 e 6 mostram os gráficos das funções traçadas por cada algoritmo. Percebe-se que *Gradient Descent* se sobrepões a Mínimos Quadrados até sobre o zoom da Figura 7.

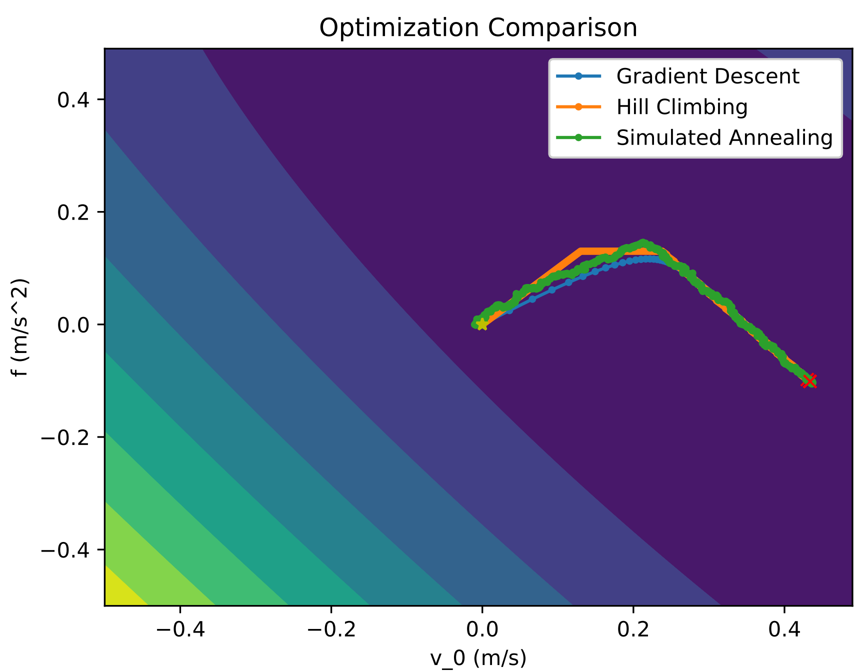

Figura 5: Comparação dos caminhos de otimização

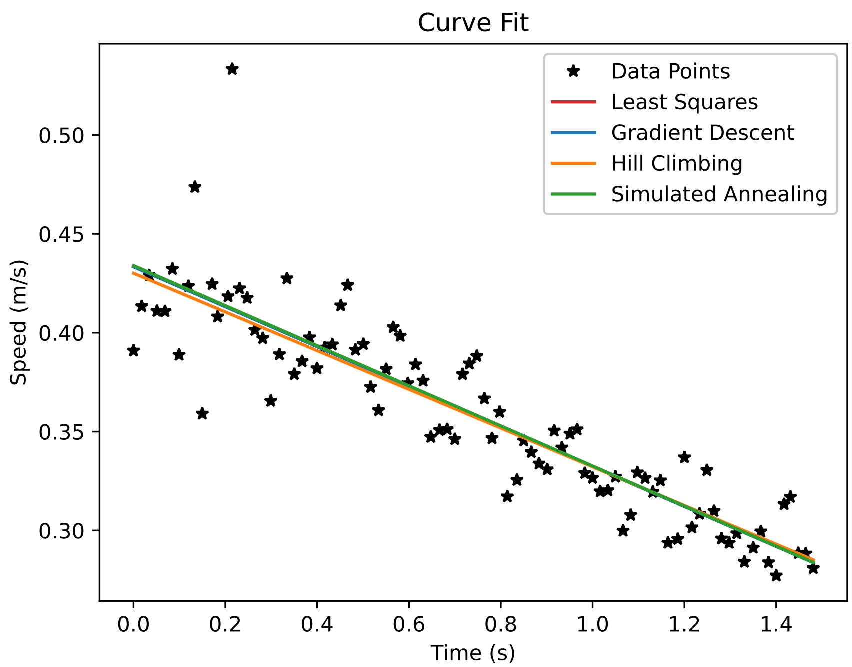

Figura 6: Gráfico das funções encontradas

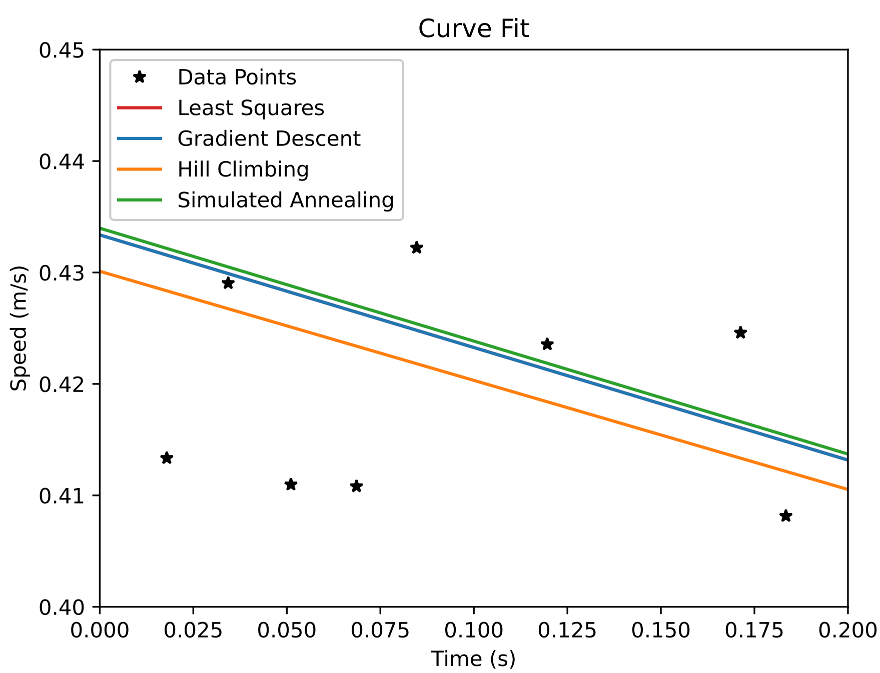

Figura 7: Zoom no início do gráfico das funções encontradas