Instituto Tecnológico de Aeronáutica - ITA
Inteligência Artificial para Robótica Móvel - CT213
Aluno: Pedro Pinheiro Borges

Relatório do Laboratório 3 - Otimização com Métodos de Busca Local

1 Breve Explicação em Alto Nível da Implementação

1.1 Descida do Gradiente

A Descida do Gradiente é implementado com a equação 1

$$\theta = \theta - \alpha \frac{\partial J(\theta)}{\partial \theta} \tag{1}$$

onde α é um hiperparâmetro ajustado manualmente.

Além disso, sempre que o valor de θ é atualizado, um *array* chamado *history* é atualizado com esse novo valor inserido nele.Finalmente, o critério de parada do algoritmo é ou se a quantidade de iterações atingir um máximo pré-definido ou se o valor da função de custo atingir um valor mínimo pré-definido.

1.2 *Hill Climbing*

O *Hill Climbing* é implementado analisando os custos de "vizinhos" do conjunto de parâmetros a ser otimizado. Tais vizinhos são definidos como 8 vetores de parâmetros igualmente espaçados em uma circunferência de raio Δ centrada no vetor de parâmetros original.

Se o custo de um vizinho for menor que o custo do original, então atualiza-se o vetor de parâmetros para esse novo valor. Além disso, o *history* e o critério de parada são os mesmos do algoritmo Descida do Gradiente.

1.3 *Simulated Annealing*

O *Simulated Annealing* é implementado utilizando um conceito de "temperatura", o qual decai seguindo a equação 2

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

onde T_0 e β são hiperparâmetros e i é a iteração atual.

Inicialmente, escolhe-se um vizinho aleátorio do vetor de parâmetros e analisa-se se o custo desse vizinho é menor ou não, se for, o vetor de parâmetros é atualizado com o valor do vizinho, se não for, analisa-se se a diferença de custo entre o vetor de parâmetros e o vizinho, representado por ΔE, satisfaz 3

$$r \leq exp(\frac{\Delta E}{T}) \tag{3}$$

onde r é um número aleátorio entre 0 e 1. Caso ΔE satisfaça essa inequação, o valor do vetor de parâmetros é atualizado com o valor do vizinho.

Finalmente, o *history* e o critério de parada são os mesmos do algoritmo Descida do Gradiente.

2 Figuras Comprovando Funcionamento do Código

2.1 Descida do Gradiente

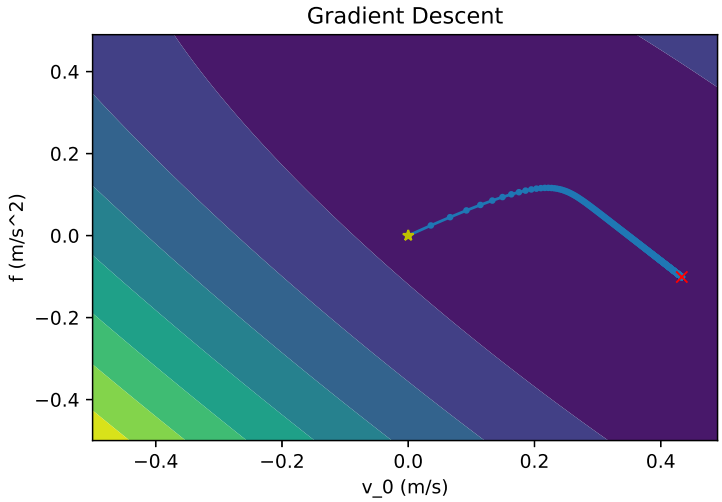

Figura 1: Curvas de nível da função de custo para o algoritmo Descida do Gradiente

2.2 *Hill Climbing*

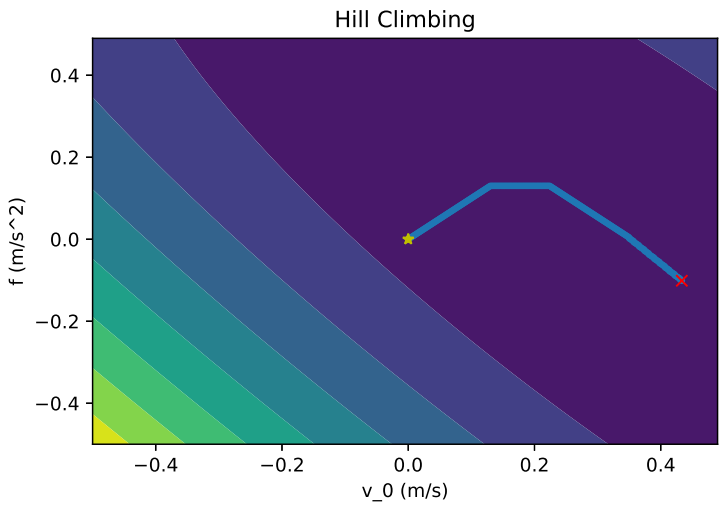

Figura 2: Curvas de nível da função de custo para o algoritmo *Hill Climbing*

2.3 *Simulated Annealing*

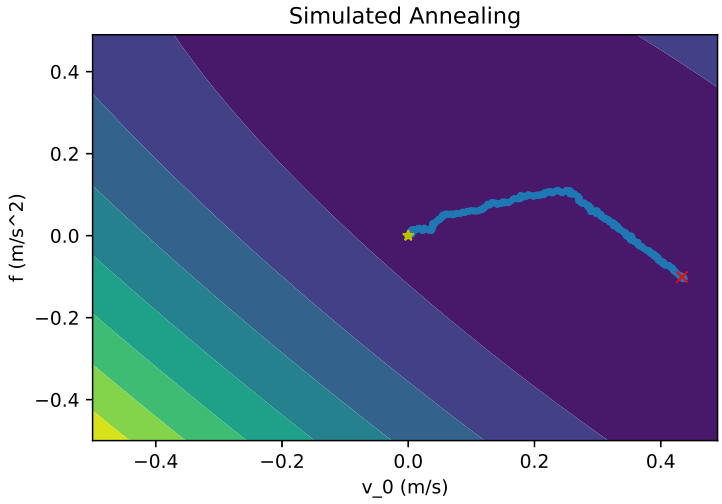

Figura 3: Curvas de nível da função de custo para o algoritmo *Simulated Annealing*

3 Comparação entre os métodos

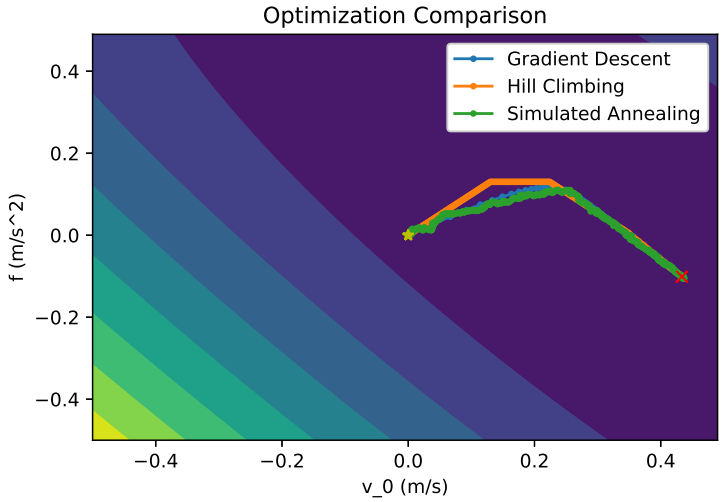

Figura 4: Comparação entre os três algoritmos de optimização

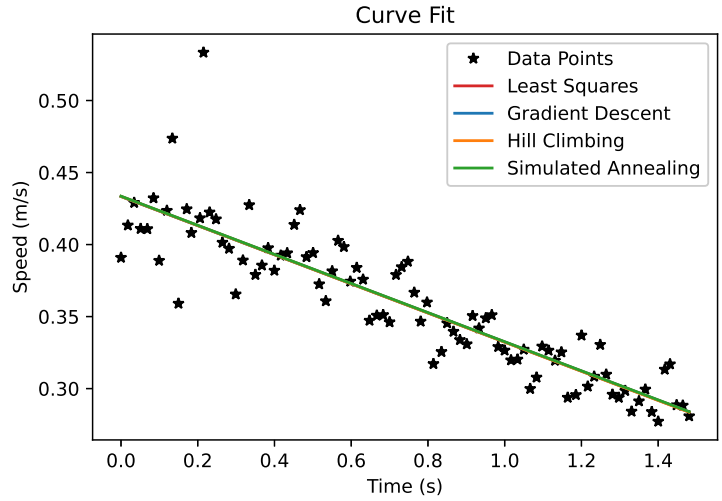

Figura 5: Retas ajustadas pelos três algoritmos implementados e pelo MMQ

Tabela 1 com a comparação dos parâmetros da regressão linear obtidos pelos métodos de otimização.

Tabela 1: parâmetros da regressão linear obtidos pelos métodos de otimização.

Método	v_0	f
MMQ	0.433373	-0.101021
Descida do gradiente	0.433371	-0.101018
Hill climbing	0.433411	-0.101196
Simulated annealing	0.433587	-0.101028