Inteligência Artificial para Robótica Móvel: CT-213

Instituto Tecnológico de Aeronáutica (ITA)

Relatório do Laboratório 3 - Otimização com Métodos de Busca Local

Leonardo Peres Dias

5 de abril de 2025

Sumário

1 Breve Explicação em Alto Nível da Implementação

1.1 Descida do Gradiente

A implementação da descida do gradiente parte de uma função de custo e seu gradiente em relação aos parâmetros. aprendizado $\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \alpha \nabla J(\boldsymbol{\theta}_i)$, de forma a reduzir progressivamente o valor da função de custo. A iteração é realizada até que o número máximo de iterações seja atingido ou que a função de custo torne-se menor que um valor de tolerância pré-definido.

1.2 Hill Climbing

Na implementação do Hill Climbing, a cada iteração, o algoritmo gera um vetor de parâmetros $\boldsymbol{\theta}$ vizinhos. Para isso a função `neighbors` calcula os n vizinhos igualmente distantes do ponto atual e espaçados por um ângulo $\frac{2\pi}{n}$. Em seguida, o algoritmo avalia cada vizinho e seleciona o que apresenta o menor valor da função de custo, considerando o θ atual, pois se nenhum vizinho tem custo menor, já se está no mínimo local. O processo é repetido até que o número máximo de iterações seja atingido ou que a função de custo torne-se menor que um valor de tolerância pré-definido.

1.3 Simulated Annealing

O algoritmo inicia com uma solução inicial $\boldsymbol{\theta}_0$ e percorre o espaço de busca explorando vizinhos aleatórios, permitindo ocasionalmente aceitar soluções piores com o objetivo de escapar de mínimos locais.

Os vizinhos são gerados usando `random_neighbor`, que retorna um ponto ao redor da solução atual, sobre uma circunferência de raio fixo Δ. A direção dessa ponto é definida por um ângulo amostrado de uma distribuição uniforme no intervalo $[-\pi, \pi]$.

A temperatura T a cada iteração é determinada pela função `schedule`, que segue o resfriamento:

$$T(i) = \frac{T_0}{1 + \beta i^2},$$

A cada iteração, um vizinho aleatório é calculado. Se esse vizinho apresenta uma função de custo menor que a atual, ele se torna o novo $\boldsymbol{\theta}$. Caso contrário, ele pode ainda

ser aceito com uma probabilidade que decresce de acordo com:

$$P = \exp\left(-\frac{\Delta E}{T}\right),$$

O algoritmo é encerrado quando a função de custo atinge um valor abaixo de um limiar ϵ ou após um número máximo de iterações.

2 Figuras Comprovando Funcionamento do Código

2.1 Descida do Gradiente

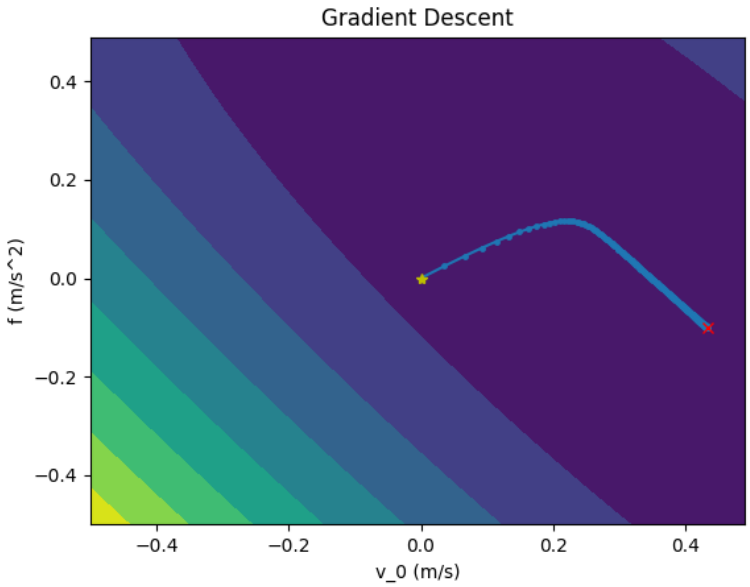

Figura 1: Trajetória de otimização da descida do gradiente.

2.2 Hill Climbing

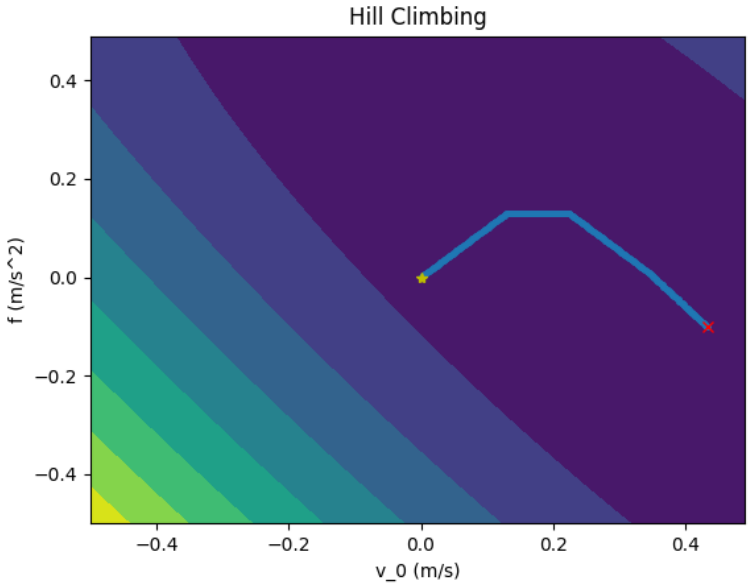

Figura 2: Trajetória de otimização do Hill Climbing.

2.3 Simulated Annealing

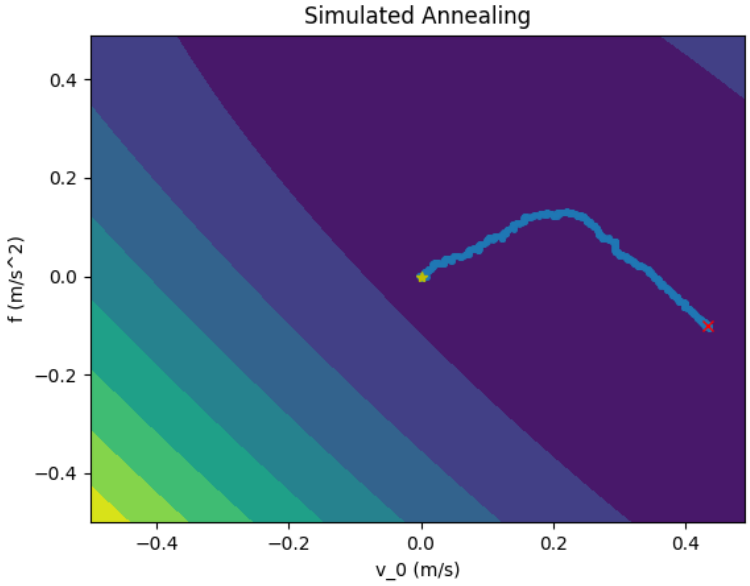

Figura 3: Trajetória de otimização do Simulated Annealing.

3 Comparação entre os Métodos

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	0.432951	-0.100354

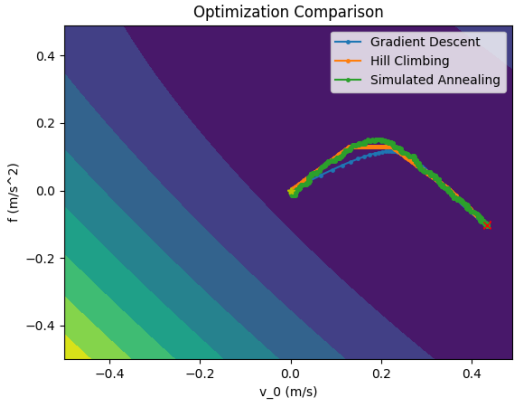

Figura 4: Comparação entre as trajetórias dos métodos de otimização.

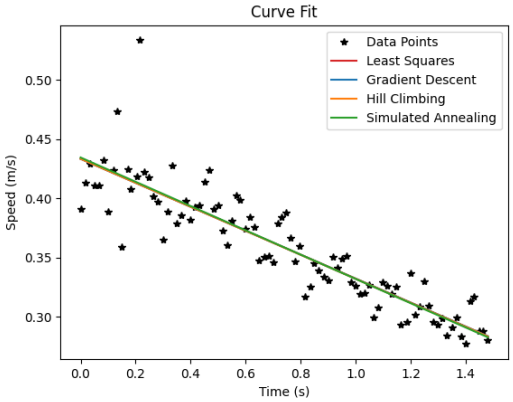

Figura 5: Curvas de regressão linear obtidas pelos métodos de otimização.