Gradiente Descendente

Felipe Maia Meiga – 2011460

Introdução

O tema deste trabalho é otimização, com o objetivo de implementar e comparar duas variações do método de Gradiente Descendente, usado para encontrar o mínimo de funções com múltiplas variáveis. A primeira variação define o passo por uma constante, enquanto a segunda utiliza o Método de Interpolação Parabólica Sucessiva (MIPS) para determinar o passo.

Desenvolvimento

Neste projeto, foi implementado e comparado o método de Gradiente Descendente com duas variações: passo constante e passo definido pelo Método de Interpolação Parabólica Sucessiva (MIPS). A implementação foi realizada em Python, utilizando bibliotecas como NumPy e Matplotlib para manipulação de dados e visualização dos resultados. A seguir, descrevemos os detalhes de cada método e as estratégias utilizadas.

Algoritmo de Gradiente Descendente com Passo Constante

O método de Gradiente Descendente com passo constante atualiza as variáveis em cada iteração na direção oposta ao gradiente, multiplicado por um passo fixo *a*. O algoritmo continua até que a norma do gradiente seja menor que uma tolerância pré definida *TOL* ou até atingir um número máximo de iterações *MAX_STEPS*.

```
def descgrad_constant(n, f, v, a)
```

Algoritmo de Gradiente Descendente com IPS

O método de Gradiente Descendente com MIPS ajusta o passo em cada iteração utilizando uma interpolação parabólica baseada em três estimativas iniciais *r*, *s* e *t*. O novo passo é determinado pelo mínimo da parábola que passa por esses três pontos. Dependendo do critério escolhido, a nova estimativa pode substituir a estimativa menos recente ou a pior estimativa.

```
def descgrad_ips(n, r, s, t, c, f, v)
```

Cálculo do Gradiente

O cálculo do gradiente é uma parte fundamental do método de gradiente descendente. O gradiente de uma função multivariada é um vetor que aponta na direção de maior aumento da função. Para encontrar o mínimo de uma função, movemos na direção oposta ao gradiente.

Derivada Parcial

Para calcular o gradiente de uma função *f* em relação a um vetor de variáveis **v**, primeiro precisamos calcular as derivadas parciais da função em relação a cada variável. A derivada parcial de *f* em relação à variável *vi* é aproximada numericamente usando uma pequena perturbação *h*:

$$\partial f/\partial vi \approx (f(v1, ..., vi + h, ..., vn) - f(v1, ..., vi - h, ..., vn)) / 2h$$

O código abaixo realiza esse cálculo:

```
def partial_derivative(f, v, h, i)
```

Função para Calcular o Gradiente

A função **"compute_gradient"** calcula o vetor gradiente da função *f* em um ponto *v*. Esta função utiliza a função **"partial_derivative"** para calcular cada componente do gradiente. Após calcular todas as derivadas parciais, normalizamos o gradiente se a sua norma for maior que 1, para garantir que os passos de atualização não sejam excessivamente grandes. O código abaixo é a função em questão:

```
def compute_gradient(f, v, h)
```

A partir deste ponto a implementação do passo constante é simples porque só precisamos saber o tamanho do passo para andar na direção contrária a do gradiente até que se aproxime de zero, ponto esse que podemos assumir que a resposta atingiu a convergência.

Se queremos determinar o valor do passo para minimizar a função *f* (x) na direção oposta ao gradiente, utilizamos o método de Interpolação Parabólica Sucessiva (MIPS). Este método usa três estimativas iniciais *r*, *s* e *t* para criar uma parábola que passa pelos pontos *f* (r), *f* (s) e *f* (t). O novo passo é então calculado como o ponto mínimo dessa parábola conforme a fórmula abaixo:

$$u = \frac{r+s}{2} + \frac{(f(s)-f(r))\cdot(t-r)\cdot(t-s)}{2\cdot[(s-r)\cdot(f(t)-f(s))-(f(s)-f(t))\cdot(t-s)]}$$

Como já explicitado anteriormente, dependendo do critério (representado pela variável *c* na função **descgrad_ips**) escolhido, podemos substituir a pior e a menos recente estimativa pela nova. No primeiro caso, a fórmula abaixo representa a pior estimativa.

pior estimativa = max(max(f (r), f (s)), f (t))

Resultados e Análise

Os resultados dos experimentos utilizando as funções de gradiente descendente com passo constante e com IPS foram analisados para duas funções bidimensionais (2D) e uma função tridimensional (3D).

Para a função f

$$f(x, y) = x^4 + y^4 + 2x^2y^2 + 6xy - 4x - 4y + 1$$

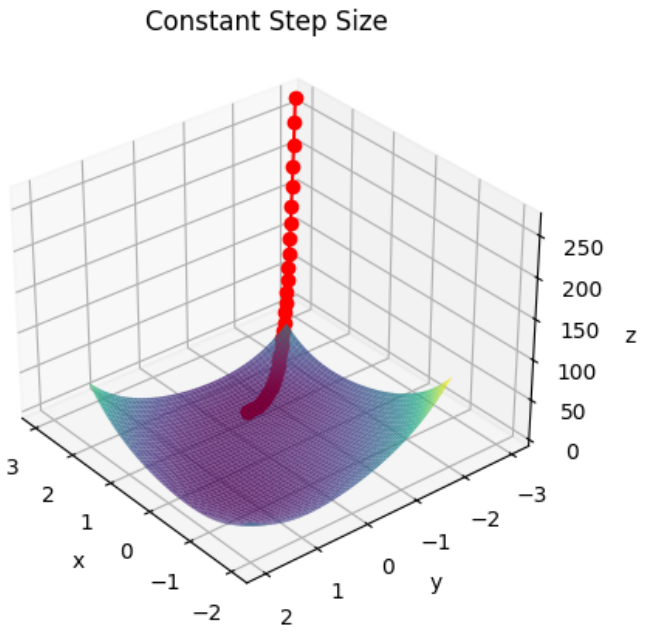
foram utilizadas estimativas iniciais v[0] = 3 v[1] = −3.

Resultados com Passo Constante

- Número de Iterações: 34
- Mínimo Global Estimado: [1.1340195, -0.46685381]

Resultados com IPS

- Estimativa Menos Recente
  - Número de Iterações: 6
  - Mínimo Global Estimado: [1.1333065, -0.467231]
- Pior Estimativa
  - Número de Iterações: 6
  - Mínimo Global Estimado: [1.13326513, -0.4672342]



Os resultados mostram que o método IPS convergiu muito mais rapidamente do que o método com passo constante, necessitando de apenas 6 iterações para ambas as estratégias de IPS. O método com passo constante, por outro lado, precisou de 34 iterações. Ambos os métodos de IPS forneceram estimativas de mínimos globais muito próximas, indicando a robustez da abordagem.

Figura 1: Resultado da função f com passo constante e ponto de início (3,-3)

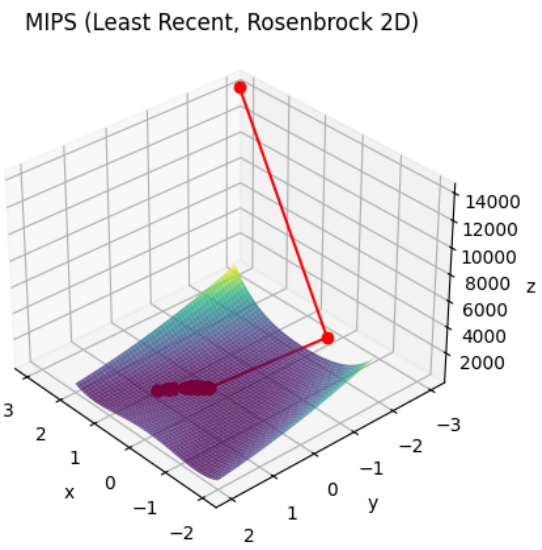
Para a função **Rosenbrock 2D** foram utilizadas as mesmas estimativas iniciais v[0] = 3 e v[1] = −3.

Resultados com Passo Constante

- Número de Iterações: 9085
- Mínimo Global Estimado: [0.92197675, 0.8497913]

Resultados com IPS

- Estimativa Menos Recente
  - Número de Iterações: 107
  - Mínimo Global Estimado: [0.9440238, 0.89109531]
- Pior Estimativa
  - Número de Iterações: 123
  - Mínimo Global Estimado: [0.92627192, 0.8572276]



Novamente, o método IPS superou significativamente o método com passo constante em termos de número de iterações. O método com passo constante precisou de 9085 iterações para convergir, enquanto o método IPS com substituição pela estimativa menos recente precisou de 107 iterações e com substituição pela pior estimativa, 123 iterações. Ambos os métodos IPS encontraram mínimos próximos, mas o método de substituição pela estimativa menos recente convergiu ligeiramente mais rápido.

Figura 2: Resultado da função Rosenbrock 2D com MIPS Least Recent e ponto de início (3,-3)

Para a função **Rosenbrock 3D**, foram utilizadas as estimativas iniciais v[0]=3, v[1]=−3, e v[2]=3.

Resultados com Passo Constante

- Número de Iterações: 7580
- Mínimo Global Estimado: [1.0142576, 1.02885938, 1.05880147]

Resultados com IPS

- Estimativa Menos Recente
  - Número de Iterações: 754
  - Mínimo Global Estimado: [0.99783833, 0.99509477, 0.98964639]
- Pior Estimativa
  - Número de Iterações: 1123
  - Mínimo Global Estimado: [0.96287465, 0.92735021, 0.85946339]

Para a função Rosenbrock 3D, o método com passo constante precisou de 7580 iterações para convergir. O método IPS foi novamente mais eficiente, necessitando de 754 iterações com substituição pela estimativa menos recente e 1123 iterações com substituição pela pior estimativa. A substituição pela estimativa menos recente demonstrou ser a estratégia mais eficiente.

Como podemos ver nas tabelas abaixo, os métodos tiveram uma boa convergência para diferentes estimativas iniciais, os mínimos globais foram muito próximos.

Função *f*

Estimativa Inicial	Método de Otimização	Número de Iterações	Mínimo Global Estimado
[3, -2]	Passo Constante	27	[1.13441785, -0.46654554]
	MIPS (Estimativa Menos Recente)	6	[1.13351268, -0.46715237]
	MIPS (Pior Estimativa)	6	[1.13350225, -0.46716635]
[5, -1]	Passo Constante	42	[1.13455628, -0.46615374]
	MIPS (Estimativa Menos Recente)	4	[1.13179955, -0.46550046]
	MIPS (Pior Estimativa)	4	[1.13179476, -0.46553317]
[6, -7]	Passo Constante	85	[1.13177887, -0.46651989]
	MIPS (Estimativa Menos Recente)	6	[1.13312874, -0.46697909]
	MIPS (Pior Estimativa)	6	[1.13303521, -0.46700929]

Função Rosenbrock 2D

Estimativa Inicial	Método de Otimização	Número de Iterações	Mínimo Global Estimado
[3, -2]	Passo Constante	8041	[0.92199058, 0.84981687]
	MIPS (Estimativa Menos Recente)	143	[0.99333543, 0.98668407]
	MIPS (Pior Estimativa)	167	[0.95584874, 0.91352654]
[5, -1]	Passo Constante	8467	[0.92197632, 0.84979051]

	MIPS (Estimativa Menos Recente)	121	[0.95391687, 0.9098082]
	MIPS (Pior Estimativa)	154	[0.92629181, 0.8572618]
[6, -7]	Passo Constante	15767	[0.92197152, 0.84978164]
	MIPS (Estimativa Menos Recente)	76	[0.94548672, 0.89385632]
	MIPS (Pior Estimativa)	72	[0.93311295, 0.86955108]

Função Rosenbrock 3D

Estimativa Inicial	Método de Otimização	Número de Iterações	Mínimo Global Estimado
[1, 2, 3]	Passo Constante	9377	[1.01425656, 1.02885726, 1.0587971]
	MIPS (Estimativa Menos Recente)	3494	[0.96271798, 0.92702051, 0.85891297]
	MIPS (Pior Estimativa)	1803	[0.96846909, 0.93838992, 0.88057547]
[5, 2, 7]	Passo Constante	26930	[1.01425755, 1.02885929, 1.05880128]
	MIPS (Estimativa Menos Recente)	990	[0.9664524, 0.93423798, 0.87238657]
	MIPS (Pior Estimativa)	216	[0.95913485, 0.9200996, 0.84552021]
[-10, 5, 3]	Passo Constante	19528	[0.96299319, 0.92729987, 0.85963527]
	MIPS (Estimativa Menos Recente)	19	[0.96802526, 0.93718149, 0.87797291]
	MIPS (Pior Estimativa)	16	[0.97021348, 0.94113838, 0.88566566]

Conclusão

Os resultados obtidos demonstram que o método de Interpolação Parabólica Sucessiva (IPS) é significativamente mais eficiente em termos de número de iterações em comparação com o método de passo constante, especialmente para funções de maior complexidade como a Rosenbrock 2D e 3D. Entre as estratégias de IPS, a substituição pela estimativa menos recente apresentou melhor desempenho geral.

Uma curiosidade que encontramos durante a realização dos experimentos foi que a função f (x, y) é simétrica, e por isso, todos os algoritmos utilizados falham ao encontrar o mínimo global quando f (x, x), uma vez que o gradiente sempre vai apontar na mesma direção, sendo assim encontraremos um mínimo local nestes casos, que se encontra no ponto (0.5, 0.5).

Resultado da função f com ponto de partida (1, 1) e passo constante:

- Número de Iterações: 9
- Mínimo Global Estimado: [0.50018984, 0.50018984]