Resultados dos Problemas de Otimização - Método da Descida por Coordenadas

Análise Computacional

16 de setembro de 2025

1 Introdução

A **Descida por Coordenadas** (também conhecida como **Coordinate Descent**) é um algoritmo de otimização que minimiza uma função objetivo multivariável minimizando sequencialmente em relação a uma coordenada por vez, mantendo todas as outras coordenadas fixas. Este método é particularmente eficaz para problemas onde a minimização unidimensional é computacionalmente mais simples que a minimização multidimensional.

2 Conceitos Fundamentais

2.1 Ideia Principal

O algoritmo da descida por coordenadas baseia-se na seguinte estratégia:

1. Dado um ponto atual $x^{(k)}$, escolha uma coordenada i
2. Minimize $f(x)$ em relação à coordenada x_i, mantendo todas as outras coordenadas fixas
3. Atualize apenas a coordenada i com o valor ótimo encontrado
4. Repita o processo para a próxima coordenada

2.2 Formulação Matemática

Para um problema de minimização:

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1}$$

O algoritmo da descida por coordenadas funciona da seguinte forma:

1. **Inicialização:** Escolha um ponto inicial $x^{(0)}$
2. **Iteração k:** Para cada coordenada $i = 1, 2, \ldots, n$:

$$x_i^{(k+1)} = \arg\min_{x_i} f(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k)}, \ldots, x_n^{(k)}) \tag{2}$$

3. **Convergência:** Pare quando $\|x^{(k+1)} - x^{(k)}\| < \epsilon$

3 Algoritmo da Descida por Coordenadas

3.1 Algoritmo Básico

[label=**Passo 1:**]

1. **Inicialização:** Escolha um ponto inicial x_0 e tolerância ϵ.
2. **Iteração k:**
   [label=(b)]
   (a) Para cada coordenada $i = 1, 2, \ldots, n$:
   [label=i]
   i. Defina a função unidimensional: $g_i(t) = f(x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n)$
   ii. Resolva: $t^* = \arg\min_t g_i(t)$
   iii. Atualize: $x_i = t^*$
   (b) Verifique convergência: se $\|x^{(k+1)} - x^{(k)}\| < \epsilon$, pare.

3.2 Minimização Unidimensional

Para cada coordenada, precisamos resolver um problema de minimização unidimensional. No nosso algoritmo, utilizamos o método de Brent:

$$t^* = \arg\min_{t \in \mathbb{R}} g_i(t) \tag{3}$$

onde $g_i(t) = f(x_1, \ldots, x_{i-1}, t, x_{i+1}, \ldots, x_n)$.

4 Implementação no Código

4.1 Estrutura Principal

O algoritmo implementado segue a seguinte estrutura:

```
def descida_por_coordenadas(f, x0, max_iter, tol):
    x = x0
    n = len(x)
    fo = f(x0)  # Valor inicial da função

    for k in range(max_iter):
        x_anterior = x.copy()

        # Atualizar cada coordenada sequencialmente
        for i in range(n):
            x[i] = minimizar_coordenada(f, x, i)

        # Verificar convergência
        if ||x - x_anterior|| < tol:
            break

        fo = f(x)

    return x, fo, k
```

4.2 Minimização de Coordenada

Para cada coordenada, o algoritmo resolve um problema unidimensional:

```
def minimizar_coordenada(f, x, i):
    def funcao_1d(xi):
        x_temp = x.copy()
        x_temp[i] = xi
        return f(x_temp)

    resultado = minimize_1d(funcao_1d, x0=x[i])
    return resultado.x
```

4.3 Métodos de Minimização 1D

O algoritmo implementa três métodos para minimização unidimensional:

1. **Método de Brent:** Busca linear eficiente
2. **Método de Newton 1D:** Usa primeira e segunda derivadas
3. **Fallback:** Método de Brent como padrão

4.3.1 Método de Newton 1D

Para o método de Newton, utilizamos:

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \tag{4}$$

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} \tag{5}$$

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} \tag{6}$$

5 Vantagens da Descida por Coordenadas

- **Simplicidade:** Cada iteração resolve apenas problemas unidimensionais
- **Eficiência:** Pode ser mais rápido que métodos de ordem superior para problemas específicos
- **Paralelização:** Coordenadas podem ser atualizadas em paralelo (em algumas versões)
- **Memória:** Requer menos memória que métodos que armazenam gradientes completos
- **Robustez:** Menos sensível a problemas de condicionamento

6 Desvantagens e Limitações

- **Convergência lenta:** Pode convergir lentamente para problemas mal condicionados
- **Dependência da ordem:** A ordem de atualização das coordenadas pode afetar a convergência
- **Problemas não separáveis:** Menos eficaz para funções com forte acoplamento entre variáveis
- **Mínimos locais:** Pode ficar preso em mínimos locais

7 Parâmetros do Algoritmo

No código implementado, os parâmetros utilizados são:

- **Máximo de iterações:** 1000 (padrão)
- **Tolerância:** 10^{-6} (padrão)
- **Precisão das derivadas:** $h = 10^{-6}$ (diferenças finitas)
- **Método 1D:** Brent (padrão)

8 Problemas Testados

O algoritmo foi testado em 16 problemas de otimização da coleção Liu-Nocedal: A tabela 1 apresenta os problemas de otimização não-linear resolvidos usando o método da Descida por Coordenadas e o número de variáveis de cada problema.

Tabela 1: Problemas de otimização e número de variáveis

Problema	**Número de Variáveis**
ROSENBROCK	10
PENALTY	5
TRIGONOMETRIC	10
EXTENDED ROSENBROCK	10
EXTENDED POWELL	12
QOR	50
GOR	50
PSP	50
TRIDIAGONAL	10
ENGGVAL1	10
LINEAR MINIMUM SURFACE	9
SQUARE ROOT 1	16
SQUARE ROOT 2	16
FREUDENTHAL ROTH	10
SPARSE MATRIX SQRT	10
ULTS0	64

9 Resultados de Convergência

A tabela 2 apresenta os resultados de convergência para cada problema, incluindo o número de iterações necessárias, o valor mínimo da função objetivo encontrado e a precisão da solução (norma do gradiente).

Tabela 2: Resultados de convergência dos problemas de otimização

Problema	**Iterações**	**Valor Mínimo**	**Precisão** ($||\nabla f(x^*)||$)	**Tempo (s)**
ROSENBROCK	1	0.000000e+00	3.999688e-10	0.000s
PENALTY	3	9.147106e-02	8.413248e-10	0.017s
TRIGONOMETRIC	64	4.218634e-05	5.394831e-07	0.214s
EXTENDED ROSENBROCK	0	0.000000e+00	8.943575e-10	0.000s
EXTENDED POWELL	999	4.570395e-07	7.860834e-05	2.100s
QOR	46	1.175472e+03	5.356569e-06	3.846s
GOR	342	1.381118e+03	1.831276e-05	31.875s
PSP	293	2.020485e+02	4.221645e-05	17.742s
TRIDIAGONAL	36	1.452112e-12	1.968393e-06	0.040s
ENGGVAL1	12	9.177470e+00	9.951075e-07	0.031s
LINEAR MINIMUM SURFACE	3	1.000000e+00	2.220446e-09	0.006s
SQUARE ROOT 1	236	5.734431e-11	7.268811e-06	0.680s
SQUARE ROOT 2	999	9.984211e-04	9.520171e-04	3.165s
FREUDENTHAL ROTH	62	1.014064e+03	3.499695e-05	0.186s
SPARSE MATRIX SQRT	65	2.512124e-01	1.108387e-06	0.316s
ULTS0	999	7.589275e-03	2.054497e-01	109.267s

10 Soluções Encontradas (Primeiras 5 Variáveis)

A tabela 3 apresenta as primeiras 5 variáveis da solução encontrada para cada problema. Para problemas com menos de 5 variáveis, apenas as variáveis disponíveis são mostradas.

Tabela 3: Primeiras 5 variáveis das soluções encontradas

Problema	**x1**	**x2**	**x3**	**x4**	**x5**
ROSENBROCK	1.000000e+00	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
PENALTY	9.815760e-01	9.815760e-01	9.815760e-01	9.815760e-01	9.815760e-01
TRIGONOMETRIC	5.391346e-02	5.549715e-02	5.728647e-02	5.933727e-02	6.173081e-02
EXTENDED ROSENBROCK	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
EXTENDED POWELL	1.551461e-02	-1.551323e-03	8.742480e-03	8.743722e-03	1.551461e-02
QOR	5.923351e-01	-7.111490e-01	6.286106e-02	-2.651204e+00	1.583513e+00
GOR	-1.798017e+00	-3.441810e-01	-3.039579e+00	-1.059181e-01	7.362649e+00
PSP	4.999680e+00	4.943251e+00	5.002957e+00	2.903055e+00	4.995767e+00
TRIDIAGONAL	1.000001e+00	5.000007e-01	2.500005e-01	1.250003e-01	6.250021e-02
ENGGVAL1	9.010300e-01	5.458807e-01	6.512110e-01	6.240700e-01	6.319669e-01
LINEAR MINIMUM SURFACE	3.123433e+00	4.328209e+00	3.123433e+00	4.328209e+00	3.123433e+00
SQUARE ROOT 1	7.749120e-01	-5.116510e-01	-6.544902e-01	-2.698334e-01	-4.149487e-01
SQUARE ROOT 2	9.060716e-01	2.969159e-01	2.524937e-01	-2.378508e-01	-2.227805e-01
FREUDENTHAL ROTH	1.226913e+01	-8.318617e-01	-1.506923e+00	-1.534671e+00	-1.535798e+00
SPARSE MATRIX SQRT	-5.309442e-01	7.083652e-01	1.447271e-01	2.497762e-03	1.399359e-02
ULTS0	1.781476e-02	-2.549321e-02	-5.020470e-02	-7.421915e-02	-8.752060e-02