IF824 - Otimização
Otimização não-linear irrestrita: Algoritmos

Gurvan Huiban

1 Otimização unidimensional: Busca de intervalo

Dada uma função $f : \mathbb{R}^+ \to \mathbb{R}$ contínua e decrescente em $x = 0$, o algoritmo 1 busca um intervalo $[c_1, c_2]$ contendo um mínimo local de f.

Data: Função f, passo inicial δ
Result: Intervalo $[c_1, c_2]$
$c_1 \leftarrow 0$;
if $f(c_1 + \delta) \leqslant f(c_1)$ **then**
    $x_m \leftarrow c_1 + \delta$;
    $\delta \leftarrow 2\delta$;
    while $f(x_m) > f(x_m + \delta)$ **do**
        $c_1 \leftarrow x_m$;
        $x_m \leftarrow x_m + \delta$;
        $\delta \leftarrow 2\delta$
    end
    $c_2 \leftarrow x_m + \delta$;
else
    $c_2 \leftarrow c_1 + \delta$;
    $\delta \leftarrow \frac{\delta}{2}$;
    while $f(c_1 + \delta) > f(c_1)$ **do**
        $c_2 \leftarrow c_1 + \delta$;
        $\delta \leftarrow \frac{\delta}{2}$;
    end
    $x_m \leftarrow c_1 + \delta$;
end

Algorithm 1: Busca de intervalo incluindo um mínimo local

2 Otimização unidimensional: Seção áurea

Dada uma função $f : \mathbb{R}^+ \to \mathbb{R}$ contínua e dado um intervalo $[c_1, c_2]$ contendo um mínimo local de f, o algoritmo da seção áurea (algoritmo 2) calcula um mínimo local de f sobre $[c_1, c_2]$ com precisão ϵ.

Data: Função f, intervalo $[c_1, c_2]$, precisão ϵ
Result: Mínimo local de f sobre $[c_1, c_2]$
$x_1 \leftarrow c_2 - \alpha (c_2 - c_1)$;
$x_2 \leftarrow c_1 + \alpha (c_2 - c_1)$;
while $\frac{(c_2 - c_1)}{2} > \epsilon$ **do**
  if $f(x_1) \leqslant f(x_2)$ **then**
    $c_2 \leftarrow x_2$;
    $x_2 \leftarrow x_1$;
    $x_1 \leftarrow c_2 - \alpha (c_2 - c_1)$;
  else
    $c_1 \leftarrow x_1$;
    $x_1 \leftarrow x_2$;
    $x_1 \leftarrow c_1 + \alpha (c_2 - c_1)$;
  end
end

Algorithm 2: Otimização unidimensional com seção áurea

3 Otimização multidimensional: Algoritmos de descida

Seja uma função $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável e seja o problema de otimização irrestrita seguinte:

$$\min_{X \in \mathbb{R}^n} f(X)$$

Dado um ponto $X_0 \in \mathbb{R}^n$, um algoritmo de descida vai convergir para um mínimo local da bacia de atração contendo X_0, caso houver. A forma padrão de um algoritmo de descida é dada pelo algoritmo 3.

Data: Função f, ponto inicial X_0
Result: Aproximação numérica de um mínimo local de f
$k \leftarrow 0$;
repeat
  Calcular d_k;
  Resolver o problema de otimização unidimensional: $\alpha_k = \arg\min_{\alpha \geqslant 0} f(X_k + \alpha_k d_k)$;
  $X_{k+1} \leftarrow X_k + \alpha_k d_k$;
  $k \leftarrow k + 1$;
until *Condição de parada*;

Algorithm 3: Algoritmo de descida padrão

3.1 Algoritmo do gradiente

No caso do algoritmo do gradiente, temos:

- $d_k \leftarrow -\nabla f(X_k)$

$\nabla f(X_k)$ representa o vetor gradiente de f em X_k.

3.2 Algoritmo de Newton

O algoritmo de Newton padrão é definido pelo algoritmo 4. O algoritmo de Newton só pode ser usado com funções duas vezes diferenciáveis. $\nabla^2 f(X_k)$ representa a matriz hessiana de f em X_k.

O algoritmo de Newton pode ser modificado de forma a seguir o padrão dos algoritmos de descida. Neste caso temos:

- $d_k \leftarrow -\left[\nabla^2 f(X_k)\right]^{-1} \nabla f(X_k)$

Data: Função f, ponto inicial X_0
Result: Aproximação numérica de um mínimo local de f
$k \leftarrow 0$;
repeat
  $X_{k+1} \leftarrow X_k - \left[\nabla^2 f(X_k)\right]^{-1} \nabla f(X_k)$;
  $k \leftarrow k+1$;
until *Condição de parada*;

Algorithm 4: Algoritmo de Newton

3.3 Algoritmos quasi-Newton

Os algoritmos quasi-Newton aproximam a hessiana de f em X_k por uma sequência de matrizes. Neste caso temos:

- $d_k \leftarrow -H_k \nabla f(X_k)$ onde:
  - $H_0 \leftarrow I$
  - $H_{k+1} = H_k + C_k$
  - C_k é calculado a partir das fórmulas:

$$\text{BFGS:} \quad C_k = \left(1 + \frac{r_k^T H_k r_k}{v_k^T r_k}\right) \frac{v_k v_k^T}{v_k^T r_k} - \frac{v_k r_k^T H_k + H_k r_k v_k^T}{v_k^T r_k} \tag{1}$$

$$\text{DFP:} \quad C_k = \frac{v_k v_k^T}{v_k^T r_k} - \frac{H_k r_k r_k^T H_k}{r_k^T H_k r_k} \tag{2}$$

onde $v_k = X_k - X_{k-1}$ e $r_k = \nabla f(X_k) - \nabla f(X_{k-1})$

3.4 Algoritmos do gradiente conjugado

O algoritmo do gradiente conjugado padrão é definido para as funções quadráticas. Uma forma de aplicá-lo para funções não quadráticas é fazendo uma aproximação quadrática da função. Neste caso, a função deve ser duas vezes diferenciável. O algoritmo 5 mostra o que seria o algoritmo resultante.

Data: Função f, ponto inicial X_0
Result: Aproximação numérica de um mínimo local de f
$d_0 \leftarrow -\nabla f(X_0)$;
$k \leftarrow 0$;
repeat
  $X_{k+1} \leftarrow X_k + \alpha_k d_k$ com $\alpha_k = -\frac{[\nabla f(X_k)]^T d_k}{d_k^T \nabla^2 f(X_k) d_k}$;
  $k \leftarrow k+1$;
  if $k \equiv 0 \bmod n$ **then**
    $d_k = -\nabla f(X_k) + \beta_{k-1} d_{k-1}$ com $\beta_{k-1} = \frac{[\nabla f(X_k)]^T \nabla^2 f(X_{k-1}) d_{k-1}}{d_{k-1}^T \nabla^2 f(X_{k-1}) d_{k-1}}$;
  else
    $d_k = -\nabla f(X_k)$;
  end
until *Condição de parada*;

Algorithm 5: Algoritmo do gradiente conjugado

O algoritmo do gradiente conjugado pode ser modificado de forma a seguir o padrão dos algoritmos de descida, trocando o uso da hessiana com o uso de uma matriz aproximada da hessiana. Neste caso temos:

- Se $k \equiv 0 \mod n$, $d_k = -\nabla f(X_k)$
- Senão $d_k = -\nabla f(X_k) + \beta_{k-1} d_{k-1}$
- β_k é calculado a partir das fórmulas:

$$\text{Fletcher-Reeves} \quad \beta_k = \frac{\left[\nabla f(X_{k+1})\right]^T \nabla f(X_{k+1})}{\left[\nabla f(X_k)\right]^T \nabla f(X_k)} \tag{3}$$

$$\text{Polak-Ribière} \quad \beta_k = \frac{\left[\nabla f(X_{k+1}) - \nabla f(X_k)\right]^T \nabla f(X_{k+1})}{\left[\nabla f(X_k)\right]^T \nabla f(X_k)} \tag{4}$$