Método de Newton e Quasi-Newton

Departamento de Matemática
Universidade do Minho

Outline

Método de Newton

Considere-se, novamente o seguinte problema de otimização sem restrições:

$$\underset{w \in \mathbb{R}^d}{\text{minimizar}} \; F(w)$$

Para w numa vizinhança de w^k, o valor de $F(w)$ pode ser aproximado através da expansão quadrática de Taylor:

$$F(w) \approx m(w) := F(w^k) + \nabla F(w^k)^T (w - w^k) + \frac{1}{2}(w - w^k)^T \nabla^2 F(w^k)(w - w^k)$$

Note-se que: h é uma função quadrática cujo o minimizante global é encontrado resolvendo o sistema $\nabla m(w) = 0$, ou seja:

$$\nabla m(w) = \nabla F(w^k) + \nabla^2 F(w^k)(w - w^k) = 0. \tag{1}$$

Assumindo que a matriz $\nabla^2 F(w^k)$ é invertível e resolvendo a equação (1) em ordem a w obtém-se:

$$w = w^k - \nabla^2 F(w^k)^{-1} \nabla F(w^k).$$

A direção, $s^{(k)} = -\nabla^2 F(w^k)^{-1} \nabla F(w^k)$ é chamada por direção de Newton em w^k.

E o algoritmo de Newton apresenta-se da seguinte forma:

Algoritmo: **Método de Newton MN**

1. Dar: $w^{(1)}$
2. Fazer $k = 1$
3. **Enquanto** ($w^{(k)}$ não satisfaz o critério de paragem)
4. Calcular a direção de procura $s^{(k)} = -\nabla^2 F(w^{(k)})^{-1} \nabla F(w^{(k)})$.
5. Tamanho do comprimento do passo $\eta_k = 1$.
6. Fazer $w^{(k+1)} = w^{(k)} + \eta_k s^{(k)}$
7. Fazer $k = k + 1$
8. **Fim enquanto**

Notar que:

- O MN assume que a matriz Hessiana $\nabla^2 F(w^k)$ é invertível para cada k;
- não existe a garantia que $F(w^{k+1}) \leq F(w^k)$;
- o quinto passo do algoritmo acima MN pode ser melhorado encontrando o valor ótimo do tamanho do passo η_k, i.e., fazendo uma procura unidimensional do tamanho do passo η_k que minimiza $F(w^{(k)} + \eta s^{(k)})$ ou a condição de Armijo com backtracking.

Exercício: Utilize o Método de Newton para determinar a solução do problema

$$\underset{w \in \mathbb{R}^2}{\text{minimizar}}\; w_1^2 + 2w_2^2.$$

com $w^0 = (1,1)^T$ com tolerância $\epsilon = 10^{-8}$. Quantas iterações foram necessárias?

Teorema 1

Seja $F(w) = \frac{1}{2}w^T Qw + c^T w + d$, em que a matriz $Q \in \mathbb{R}^d \times \mathbb{R}^d$ é simétrica e definida positiva, $c \in \mathbb{R}^d$ e $d \in \mathbb{R}$. Então a sucessão de iteradas obtida pelo método de Newton aplicado à minimização de F a partir de um qualquer ponto inicial $w^{(1)} \in \mathbb{R}^d$, atinge o minimizante global de F num único passo.

Demonstração.

Temos $\nabla F(w) = Qw + c$ e $\nabla^2(w) = Q$. O facto de Q ser definida positiva garante a convexidade estrita de F e, consequentemente, que a única solução $Qw = c$ é o minimizante global w^* de F.

Aplicando o MN, tem-se que:

$$\begin{aligned} w^1 &= w^0 - \nabla^2 F(x^0)^{-1} \nabla F(w^0) \\ &= w^0 - Q^{-1}(Qw^0 + c) \\ &= w^0 - Q^{-1}(Qw^0 - Qw^*) \\ &= w^*. \end{aligned}$$

Conclui-se que o MN converge numa única iteração. $\square$

No caso mais geral em que $\nabla^2 F(w)$ é simétrica definida positiva, mas a função não é necessariamente quadrática temos o seguinte resultado:

Teorema 2

Se $\nabla^2 F(w)$ é simétrica definida positiva e a direção d de Newton em w não for nula, i.e., $s = -\nabla^2 F(w)^{-1} \nabla F(w) \neq 0$, então s é uma direção descendente, i.e., $F(w + \eta s < F(w))$ para todos os valores de η suficientemente pequenos.

Demonstração.

Basta mostrar que

$$\nabla F(w) s = -\nabla F(w) \nabla^2 F(w)^{-1} \nabla F(w) < 0.$$

É facil de verificar que esta desigualdade é verdadeira se $\nabla^2 F(w)^{-1}$ for definida positiva. Uma vez que $\nabla^2 F(w)$ é simétrica definida positiva, tem-se que

$$0 < (\nabla^2 F(w)^{-1} v)^T \nabla^2 F(w) (\nabla^2 F(w)^{-1} v) = v^T \nabla^2 F(w)^{-1} v$$

para todo o $v \in \mathbb{R}^d$, $v \neq 0$, i.e., $\nabla^2 F(w)^{-1}$ é definida positiva. $\square$

Exercício: Considere-se o problema

$$\underset{w \in \mathbb{R}^2}{\text{minimizar}} \sqrt{w_1^2 + 1 +} + \sqrt{w_2^2 + 1}$$

cuja a solução é $(0,0)$. Contudo verifique que algoritmo não converge para $w^0 = (1,1)^T$ com tolerância $\epsilon = 10^{-8}$.

É fácil de compreender o que se passa fazendo duas iterações no algoritmo de MN à mão: conclui-se que

$$\begin{array}{ll} s^{2i} = (-2,-2)^T, & w^{2i+1} = (-1,-1)^T, \\ s^{2i+2} = (2,2)^T, & w^{2i+2} = (1,1)^T, \quad \forall i \in \mathbb{N}_0. \end{array}$$

Verifique o que acontece para $w^0 = (0.5, 0.5)^T$.

Método de Quasi - Newton

Calcular $\nabla^2 F(w^k)$ e $\nabla^2 F^{-1}(w^k)$ em cada ponto w^k pode ser computacionalmente pesado quando estamos perante uma problema de grandes dimensões.

Para contornar este problema surge o método Quasi - Newton, cuja a sua forma geral é:

$$w^{k+1} = w^k - \eta_k B_k^{-1} \nabla F(w^k),$$

onde η_k é o comprimento do passo e B_k é uma aproximação à matriz hessiana $\nabla^2 F$.

Note-se que: se $\eta_k = 1$ e $B_k = \nabla^2 F(w^k)$ estamos perante o Método de Newton.

Método BFGS

Considera-se o seguinte modelo quadrático da função objetivo a cada iteração w^k:

$$m_k(w) = F(w^k) + \nabla F(w^k)^T(w - w^k) + \frac{1}{2}(w - w^k)^T B_k (w - w^k)$$

onde B_k é uma matriz simétrica definida positiva que recebe uma actualização a cada iteração.

O minimizante w de um modelo convexo quadrático, satisfaz a equação $\nabla m_k(w) = 0$, ou seja, a direção a considerar

$$B_k s^k = -\nabla F(w^k).$$

Considere-se o modelo:

$$\begin{aligned} m_{k+1}(w) = \quad & F(w^{k+1}) + \nabla F(w^{k+1})^T (w - w^{k+1}) \\ & + \frac{1}{2}(w - w^{k+1})^T B_{k+1} (w - w^{k+1}). \end{aligned}$$

que corresponde à aproximação quadrática de m_{k+1} de F em w^{k+1}

Derivando m_k em ordem a w, tem-se que:

$$\nabla m_{k+1}(w) = \nabla F(w^{k+1}) + B_{k+1}(w - w^{k+1}).$$

Pretende-se que B_{k+1} seja uma matriz em que ∇m_{k+1} seja igual ao gradiente de F em w^k e w^{k+1};

- $\nabla m_{k+1}(w^{k+1}) = \nabla F(w^{k+1})$.
- $\nabla m_{k+1}(w^k) = \nabla F(w^{k+1}) + B_{k+1}(w^k - w^{k+1}) = \nabla F(w^k)$.

A relação (condição Quasi - Newton)

$$B_{k+1} p^k = y_k \tag{2}$$

resulta da igualdade de cima com $y_k = \nabla F(w^{k+1}) - \nabla F(w^k)$ e $p^k = \eta_k s^k = w^{k+1} - w^k$.

O objetivo é fazer pequenos ajuste na matriz B_k por forma a que a equação (2) seja satisfeita garantindo que B_k é uma matriz simétrica definida positiva, vamos considerar:

$$B_{k+1} = B_k + \alpha u u^T + \beta v v^T.$$

Seja $u = y_k$ e $v = B_k p^{(k)}$ tem-se:

$$B_{k+1} = B_k + \alpha y_k y_k^T + \beta B_k p^{(k)} {p^{(k)}}^T p^{(k)} B_k.$$

Pela equação (2), sabemos que:

$$y_k = B_{k+1} p^{(k)} = B_k p^{(k)} + \alpha y_k y_k^T p^{(k)} + \beta B_k p^{(k)} {p^{(k)}}^T p^{(k)} B_k p^{(k)}$$

Deve ser satisfeito se:

$$\alpha = \frac{1}{y_k p^{(k)}} \quad \text{e} \quad \beta = -\frac{1}{p^{(k)}}^T B_k p^{(k)}$$

Logo,

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k p^{(k)}} - \frac{B_k p^{(k)} {p^{(k)}}^T p^{(k)} B_k}{p^{(k)}}^T B_k p^{(k)}.$$

Algoritmo: **Método de BFGS**

1. Dar: $w^{(1)}$
2. Dar: B_0 $(B_0 = I)$
3. Dar: Tamanho do comprimento do passo $(\eta_k = 1)$.
4. Fazer $k = 1$
5. **Enquanto** ($w^{(k)}$ não satisfaz o critério de paragem)
6. Resolver $B_k s^k = -\nabla F(w^{(k)})$
7. Fazer $w^{(k+1)} = w^{(k)} + \eta_k s^{(k)}$
8. Fazer $p^k = w^{(k+1)} - w^{(k)}$
9. Fazer $y_k = \nabla F(w^{(k+1)}) - \nabla F(w^{(k)})$
10. Fazer $B_{k+1} = B_k + \triangle B$
11. Fazer $k = k + 1$
12. **Fim enquanto**

onde $\triangle B = \frac{y_k y_k^T}{y_k p^{(k)}} - \frac{B_k p^{(k)} {p^{(k)}}^T B_k}{{p^{(k)}}^T B_k p^{(k)}}$

Em vez de utilizar uma atualização a uma aproximação da Hessiana, pode-se utilizar uma atualização uma aproximação da inversa da Hessiana.

Para calcular a inversa de B_{k+1} utilizando a formula Sherman-Moreison-WooDBury

$$(A + ab^T)^{-1} = A^{-1} - \frac{A^{-1}ab^T A^{-1}}{1 + b^T A^{-1} a}.$$

Obtendo-se

$$H_{k+1} = (I - \rho_k p^{(k)} y_k^T) H_k (I - \rho y_k p^{(k)}) + \rho_k p^{(k)} {p^{(k)}}^T$$

com $\rho_k = \frac{1}{y_k^T p^{(k)}}$.

Algoritmo: Método de BFGS

1. Dar: $w^{(1)}$
2. Dar: H_0 $(H_0 = I)$
3. Dar tamanho do comprimento do passo $(\eta_k = 1)$
4. Fazer $k = 1$
5. **Enquanto** ($w^{(k)}$ não satisfaz o critério de paragem)
6. Fazer $s^{(k)} = -H_k \nabla F(w^{(k)})$
7. Fazer $w^{(k+1)} = w^{(k)} + \eta_k s^{(k)}$
8. Fazer $p^{(k)} = w^{(k+1)} - w^{(k)}$
9. Fazer $y_k = \nabla F(w^{(k+1)}) - \nabla F(w^{(k)})$
10. Fazer $H_{k+1} = \triangle H$
11. Fazer $k = k + 1$
12. **Fim enquanto**

onde $\triangle H = (I - \rho_k p^{(k)} y_k^T) H_k (I - \rho y_k p^{(k)}) + \rho_k p^{(k)} p^{(k)^T}$.

Exercício: Utilize o Método BFGS para determinar a solução do problema

$$\underset{w\in\mathbb{R}^2}{\text{minimizar}}\, w_1^2 + 2w_2^2$$

. $w^0 = (1,1)^T$, $\eta_k = 1$ com tolerância $\epsilon = 10^{-8}$.

BFGS é implementado no Matlab com a função fminunc.

Opções de otimização, incluindo o método BFGS. options = optimset('fminunc');

Chama a função fminunc para otimizar a função usando o método BFGS. $[w^*, F(w^*)] = fminunc(fun, w^{(0)}, options)$;

Exercício: Compare os resultados obtidos anteriormente com os resultados obtidos com a utilização da rotina 'fminunc' do Matlab.

Memória L - BFGS

O BFGS usa matrizes $n \times n$ densas. Para problemas muito grandes onde o espaço é uma preocupação, a função Memória Reduzida (Memória L - BFGS) pode ser usado para aproximar BFGS.

A principal ideia deste método é usar a informação da curvatura unicamente das últimas iterações para construir a aproximação à matriz Hessiana.

A informação da curvatura das iterações anteriores, que é menos relevante para o actual comportamento da Hessiana na iteração corrente, é descartada com o objetivo de salvar armazenamento.

Note-se que: Cada passo do métdod BFGS tem a forma:

$$w^{(k+1)} = w^{(k)} - \eta_k H_k \nabla F(w^{(k)}),$$

onde η_k é o comprimento do passo e H_k é a actualização a cada iteração escrita na forma:

$$H_{k+1} = V_k^T H_k V_k + \rho_k p^{(k)} {p^{(k)}}^T, \tag{3}$$

onde

$$\rho_k = \frac{1}{y_k^T p^{(k)}}, \qquad V_k = I - \rho_k y_k p^{(k)}$$

e

$$p^{(k)} = w^{(k+1)} - w^{(k)}, \qquad y_k = \nabla F(w^{(k+1)}) - \nabla F(w^{(k)}).$$

Na iteração k, a corrente iteração é $w^{(k)}$ e o conjunto de pares de valores é dado por $\{p^{(i)}, y_i\}$ para $i = k-m, \ldots, k-1$. Escolhemos uma aproximação inicial para a matriz Hessiana em k, que denotamos por H_k^0, e repetindo a formula (3) m vezes que L - BFGS se aproxima de H_k.

$$
\begin{aligned}
H_k = \quad & (V_{k-1}^T \ldots V_{k-m}^T) H_k^0 (V_{k-m} \ldots V_{k-1}) \\
& + \rho_{k-m} (V_{k-1}^T \ldots V_{k-m+1}^T) p^{(k-m)} p^{(k-m)^T} (V_{k-m+1} \ldots V_{k-1}) \\
& + \rho_{k-m+1} (V_{k-1}^T \ldots V_{k-m+2}^T) p^{(k-m+1)} p^{(k-m+1)^T} (V_{k-m+2} \ldots V_{k-1}) \\
& \ldots \\
& + \rho_{k-1} p^{(k-1)} p^{(k-1)^T}
\end{aligned}
$$

Memória L - BFGS

Da expressão anterior, pode-se derivar um procedimento recursivo para calcular o produto $H_k \nabla F(w^k)$. $H_k \nabla F(w^k)$ é calculado como a soma produto interno dos vetores: $\nabla F(w^{(k)})$ e o par $\{p^{(i)}, y_i\}$.

Algoritmo (*): **Método de L - BFGS calculo de $H_k \nabla F(w^k)$**

1. Fazer $q = \nabla F(w^k)$
2. **De** $i = k - 1$ **até** $k - m$
3. Fazer $\alpha_i = \rho_i {p^{(i)}}^T q$
4. Fazer $q = q - \alpha_i y_i$
5. **Fim**
6. Fazer $r = H_k^0 q$
7. **De** $i = k - m$ **até** $k - 1$
8. Fazer $\beta = \rho_i y_i^T r$
9. Fazer $r = r + p^{(i)}(\alpha_i - \beta)$
10. **Fim**
11. Para o resultado com $H_k \nabla F(w^k) = r$

Memória L - BFGS

Este algoritmo tem a vantagem de que a multiplicação pela a matriz inicial H_k^0 é isolada do resto da computação, permite escolher livremente a matriz H_k^0 e variar de iteração para iteração.

Escolha de H_k^0:

$$H_k^0 = \gamma_k I \tag{4}$$

com $\gamma_k = \frac{{p^{(k-1)}}^T y_{k-1}}{y_{k-1}^T y_{k-1}}$.

Algoritmo: Método de L - BFGS

1. Dar: $w^{(1)}$
2. Dar: $m > 0$
3. Dar tamanho do comprimento do passo ($\eta_k = 1$)
4. Fazer $k = 1$
5. **Enquanto** ($w^{(k)}$ não satisfaz o critério de paragem)
6. Dar: H_k^0 (ver 4)
7. Fazer $s^{(k)} = -H_k \nabla F(w^{(k)})$ (do Algoritmo (*))
8. Fazer $w^{(k+1)} = w^{(k)} + \eta_k s^{(k)}$
9. Se $k > m$, apagar os vetor $[p^{(k-m)}, y_{k-m}]$ do armazenamento.
10. Fazer $p^{(k)} = w^{(k+1)} - w^{(k)}$
11. Fazer $y_k = \nabla F(w^{(k+1)}) - \nabla F(w^{(k)})$
12. Fazer $k = k + 1$
13. **Fim enquanto**

Método L - BFGS

O algoritmo L - BFGS é equivalente ao algoritmo BFGS se a matriz inicial H_0 é a mesma em ambos os algoritmos e se em L - BFGS escolher-se $H_k^0 = H_0$ em cada iteração.

Pode se usar a rotina “fminunc´´ do matlab e escolher o algoritmo “quasi-newton´´, nas opções “HessianApproximation´´ escolher “lbfgs´´.

Ou escolher a rotina “fminlbfgs“:

```
options = optimset('GradObj', 'on' ,'Display', 'iter', 'HessUpdate', 'bfgs',
'GoalsExactAchieve',1);
[x,fval] = fminlbfgs(@myfun, [1 1 1 1 1], options);
```

Método L - BFGS

Exercício: Utilize o Método L - BFGS para determinar a solução do problema

$$\underset{w \in \mathbb{R}^2}{\text{minimizar}}\, w_1^2 + 2w_2^2$$

. $w^0 = (1,1)^T$, $\eta_k = 1$ com tolerância $\epsilon = 10^{-8}$ e $m = 5$.

Exercício: Compare os resultados obtidos anteriormente com os resultados obtidos com a utilização da rotina 'fminunc' do e com a rotina ´fminlbfgs´ do Matlab.