Optimização e Algoritmos (2004/2005)

Instituto Superior Técnico – Engenharia Electrotécnica e de Computadores

Série de Problemas 4

Minimização sem restrições
(algoritmos gradiente, Newton, quasi-Newton BFGS)

Minimização em conjuntos convexos
(algoritmo gradiente condicional)

Introdução

No problema 1 vai implementar em MATLAB os algoritmos iterativos gradiente e quasi-Newton BFGS para minimização de funções sem restrições

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

Ambos estes algoritmos operam por pesquisas em linha e obedecem ao protótipo representado na tabela 1.

1.	**inicialização**	▶	`escolher` $x_0 \in \mathbb{R}^n$ `e tolerância` $\epsilon > 0$
2.		▶	`inicializar` $k = 0$
3.	**ciclo**	▶	`calcular` $g_k = \nabla f(x_k)$
4.		▶	`se` $\|g_k\| < \epsilon$ `terminar`
5.		▶	`determinar direcção de pesquisa` d_k
6.		▶	`pesquisar na linha` d_k `para obter` x_{k+1}
7.		▶	$k \leftarrow k + 1$ `e retornar ao ciclo`

Tabela 1: Algoritmo de pesquisa em linha

Os algoritmos de gradiente e quasi-Newton BFGS apenas diferem no modo como, na linha 5, escolhem a direcção de pesquisa d_k:

- Para o método de gradiente, tem-se $d_k = -g_k$
- Para o método quasi-Newton BFGS, tem-se $d_k = -W_k g_k$ onde $W_0 = I_n$ e, para $k > 0$, usa-se a recursão

$$\begin{aligned} W_k &= W_{k-1} - \frac{sy^T W_{k-1} + W_{k-1} y s^T}{y^T s} + \left(1 + \frac{y^T W_{k-1} y}{y^T s}\right) \frac{ss^T}{y^T s} \\ y &= g_k - g_{k-1} \\ s &= x_k - x_{k-1}. \end{aligned}$$

Para ambos os algoritmos (gradiente e quasi-Newton BFGS), a linha 6 implica resolver o subproblema de optimização unidimensional

$$\min_{t>0} \phi(t), \tag{1}$$

onde $\phi(t) = f(x_k + td_k)$. Na verdade, tal como descrito na última série de problemas, basta resolver (1) *aproximadamente*. Ou seja, basta localizar um ponto "aceitável". O algoritmo para localização de pontos aceitáveis (introduzido na série de problemas anterior) está reproduzido na tabela 2.

1.	**inicialização**	▶	`escolher` $t_0 > 0$
2.		▶	`inicializar` $e = 0$, $d = +\infty$, $k = 0$
3.	**ciclo**	▶	`testar` t_k:
4.			• `se` $t_k \in \mathcal{A}$: `terminar`
5.			• `se` $t_k \in \mathcal{D}$: $d = t_k$
6.			• `se` $t_k \in \mathcal{E}$: $e = t_k$
7.		▶	`testar` d:
8.			• `se` $d = +\infty$: `escolher` $t_{k+1} > t_k$
9.			• `se` $d < +\infty$: `escolher` $t_{k+1} \in (e, d)$
10.		▶	$k \leftarrow k + 1$ `e retornar ao ciclo`

Tabela 2: Algoritmo para localização de um ponto aceitável

Comentários:

- Iremos usar sempre a regra de Wolfe

$$\begin{aligned}
\mathcal{A} &= \{\, t > 0 \,:\, \phi(t) \leq \phi(0) + \sigma\phi'(0)t \text{ e } \phi'(t) \geq \lambda\phi'(0) \,\} \\
\mathcal{D} &= \{\, t > 0 \,:\, \phi(t) > \phi(0) + \sigma\phi'(0)t \,\} \\
\mathcal{E} &= \{\, t > 0 \,:\, \phi(t) \leq \phi(0) + \sigma\phi'(0)t \text{ e } \phi'(t) < \lambda\phi'(0) \,\}
\end{aligned}$$

  com parâmetros $t_0 = 1$, $\sigma = 0.4$ e $\lambda = 0.8$;

- na linha 8 do algoritmo assuma (para simplificar) que se escolhe $t_{k+1} > t_k$ fazendo $t_{k+1} = \alpha\, t_k$ (com $\alpha = 10$);

- na linha 9 do algoritmo assuma (para simplificar) que se escolhe $t_{k+1} \in (e, d)$ fazendo $t_{k+1} = (e + d)/2$.

Problema 1. **[Algoritmo de gradiente e quasi-Newton BFGS]** Considere a função de Rosenbrock $f : \mathbb{R}^2 \to \mathbb{R}$ dada por

$$f(x_1, x_2) = 100\left(x_2 - x_1^2\right)^2 + (1 - x_1)^2 .$$

A função f tem um mínimo global em $x^* = (1, 1)$ e $f(x^*) = 0$. A figura 1 ilustra algumas curvas de nível de f na vizinhança de x^*.

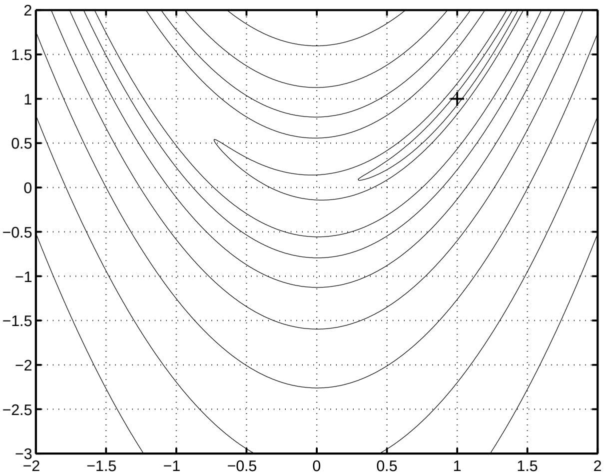

Figura 1: Curvas de nível de f [Problema 1]

(a) Implemente o algoritmo de gradiente para minimizar f. Use $x_0 = (-1, -1)$ e $\epsilon = 10^{-3}$. Quantas iterações são executadas por este algoritmo (tabela 1) ? Qual o valor de x_k na iteração final ? Represente graficamente os valores da função objectivo assumidos pelas pontos x_k gerados pelo algoritmo, isto é, $f(x_k)$ para $k = 0, 1, 2, \ldots$ até à paragem.

<u>Nota:</u> A título de exemplo (que deve tentar reproduzir) a tabela 3 contém as primeiras 4 iterações quando se parte de $x_0 = (1, -2)$. Além disso, a figura 2 exibe os valores de $f(x_k)$ correspondentes aos pontos x_k gerados pelo algoritmo.

(b) Implemente o algoritmo de quasi-Newton BFGS para minimizar f. Use $x_0 = (-1, -1)$ e $\epsilon = 10^{-3}$. Quantas iterações são executadas por este algoritmo (tabela 1) ? Qual o valor de x_k na iteração final ? Represente graficamente os valores da função objectivo assumidos pelas pontos x_k gerados pelo algoritmo, isto é, $f(x_k)$ para $k = 0, 1, 2, \ldots$ até à paragem. Qual a matriz W final ? Compare-a com a inversa da matriz Hessiana $\nabla^2 f(1, 1)$ e comente.

<u>Nota:</u> A título de exemplo (que deve tentar reproduzir) a tabela 4 contém as primeiras 4 iterações quando se parte de $x_0 = (1, -2)$. Além disso, a

k	
0	$(1.0000, -2.0000)$
1	$(0.4141, -1.7070)$
2	$(-0.1913, -0.9733)$
3	$(0.1199, -0.1843)$
4	$(0.0895, -0.0291)$

Tabela 3: Primeiras iterações do algoritmo de gradiente [Problema 1 (a)]

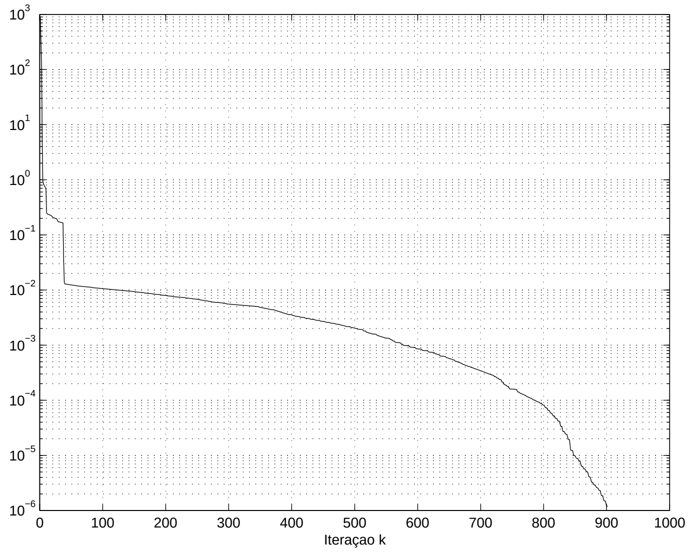

Figura 2: Função objectivo $f(x_k)$ [Problema 1(a)]

figura 3 exibe os valores de $f(x_k)$ correspondentes aos pontos x_k gerados pelo algoritmo.

Problema 2. **[Newton]** Considere a função $h : \mathbb{R}^2 \to \mathbb{R}$,

$$h(x_1, x_2) = (11 - x_1 - x_2)^2 + (1 + x_1 + 10x_2 - x_1 x_2)^2 - 40.$$

A função h tem dois minimizante globais, $x_1^* = (13, 4)$ e $x_2^* = (7, -2)$, que satisfazem $h(x_1^*) = h(x_2^*) = 0$. Tem igualmente um ponto de sela em $(10, 1)$. A figura 4 ilustra algumas curvas de nível de h em torno desses pontos de estacionariedade.

Considere a utilização do método de Newton puro para minimizar h. Ou seja, temos

$$x^{k+1} = x^k - \left(H^k\right)^{-1} g^k, \tag{2}$$

k	
0	$(1.0000, -2.0000)$
1	$(0.4141, -1.7070)$
2	$(0.9894, 0.5862)$
3	$(0.9651, 0.8788)$
4	$(0.9624, 0.9259)$

Tabela 4: Primeiras iterações do algoritmo quasi-Newton BFGS [Problema 1 (b)]

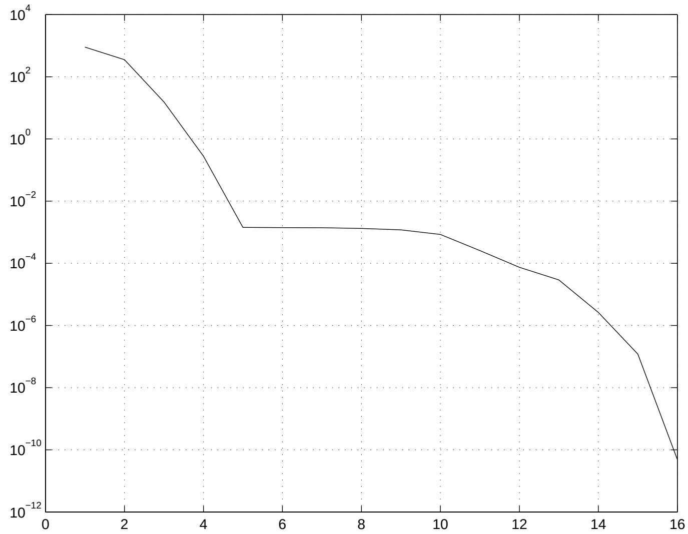

Figura 3: Função objectivo $f(x_k)$ [Problema 1(b)]

onde $g^k = \nabla h\left(x^k\right)$ e $H^k = \nabla^2 h(x^k)$.

(a) Implemente a iteração em (2) em MATLAB, tomando como condição inicial $x^0 = (18, 6)$. Para onde convergem as iterações do algoritmo ?

(b) Repita a alínea (a) tomando $x^0 = (1, -4)$ como ponto inicial. Para onde convergem as iterações do algoritmo ?

(c) Repita a alínea (a) tomando $x^0 = (9, 3)$ como ponto inicial. Para onde convergem as iterações do algoritmo ?

(d) Comente os resultados que obteve nas alíneas anteriores.

Problema 3.[Algoritmo do gradiente condicional] Considere o problema de optimização

$$\min_{x \in C} f(x), \tag{3}$$

onde f representa uma função diferenciável e C é um conjunto convexo fecha-

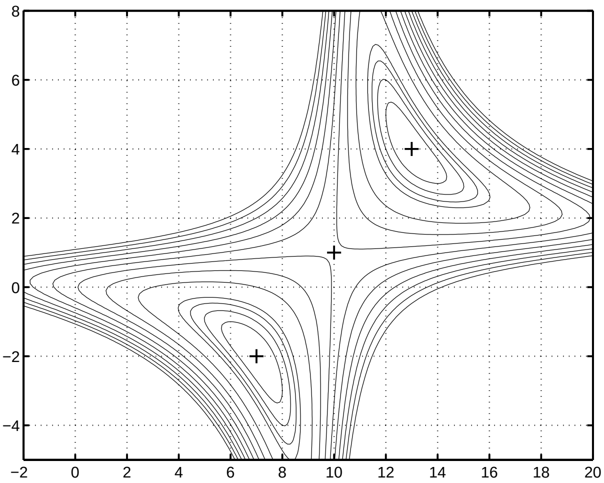

Figura 4: Curvas de nível de h [Problema 2]

do. Considere o algoritmo do gradiente condicional (com regra de Armijo)

$$\begin{aligned} x^{k+1} &= x^k + \alpha^k d^k \\ d^k &= \overline{x}^k - x^k \\ \overline{x}^k &= \arg\min_{x \in C} \nabla f\left(x^k\right)^T \left(x - x^k\right) \end{aligned}$$

com $\alpha^k = \beta^{m_k}$, onde m_k representa o primeiro inteiro não-negativo $m \in \{0, 1, 2, \ldots\}$ que satisfaz

$$f\left(x^k + \beta^m d^k\right) \leq f\left(x^k\right) + \beta^m \sigma \nabla f\left(x^k\right)^T d^k.$$

Aqui, $0 < \beta, \sigma < 1$ representam parâmetros do algoritmo. Para este problema, faça $\beta = 0.2$ e $\sigma = 0.8$.

Neste problema, consideramos a minimização da função de Rosenbrock (introduzida no problema 1) $f : \mathbb{R}^2 \to \mathbb{R}$

$$f(x_1, x_2) = 100\left(x_2 - x_1^2\right)^2 + (1 - x_1)^2.$$

O conjunto convexo C é uma "caixa"centrada em $(1, -1)$ e aresta de comprimento 1, ou seja,

$$C = \{(x_1, x_2) : \|(x_1, x_2) - (1, -1)\|_\infty \leq 0.5\}$$

A figura 5 ilustra problema.

Implemente o algoritmo do gradiente condicional, partindo de $x^0 = (1, -1)$. Para onde convergem as iterações do algoritmo ?

Problema 4.[Funções convexas: um minimizante local é um minimizante global] Seja $S \subset \mathbb{R}^n$ um conjunto convexo e $f : S \to \mathbb{R}$ uma

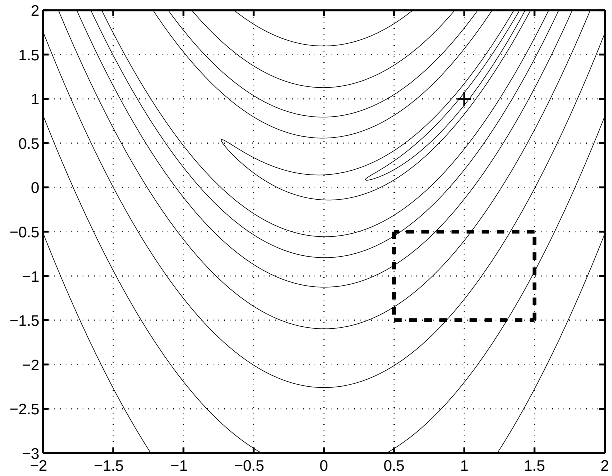

Figura 5: Curvas de nível de f e conjunto C [Problema 4]

função convexa. Mostre: se $x_0 \in S$ é um minimizante local de f, então x_0 é um minimizante global de f. [**Nota:** este resultado é muito importante! Torna as funções convexas extremamente atractivas para minimizar dado a ausência de "falsos" mínimos]