UNIVERSIDAD SIMÓN BOLÍVAR
CO5412: Optimización No Lineal I
Enero-Marzo 2011
Enrique Areyán

Tarea 2
Sartenejas, 7 de Febrero de 2011

1. Resolveremos el problema $min(x^2 - 3x)$ usando como intervalo inicial $[0,5; 2]$ y como criterio de parada que $|a_k - b_k| < \epsilon = 0,2$ Para ello usaremos los siguientes métodos:

   a) Método de Sección Aúrea:

$\mathbf{k}$	$\lvert\mathbf{a_k} - \mathbf{b_k}\rvert$	$\lambda_\mathbf{k} \Rightarrow \mathbf{f}(\lambda_\mathbf{k})$	$\mu_\mathbf{k} \Rightarrow \mathbf{f}(\mu_\mathbf{k})$	$[\mathbf{a_k}, \mathbf{b_k}]$
0	1,500000	1,072949 $\Rightarrow -2,067627$	1,427051 $\Rightarrow -2,244678$	[1,072949; 2,000000]
1	0,927051	1,427051 $\Rightarrow -2,244678$	1,645898 $\Rightarrow -2,228714$	[1,072949; 1,645898]
2	0,572949	1,291796 $\Rightarrow -2,206651$	1,427051 $\Rightarrow -2,244678$	[1,291796; 1,645898]
3	0,354102	1,427051 $\Rightarrow -2,244678$	1,510643 $\Rightarrow -2,249887$	[1,42705; 1,645898]
4	0,218847	1,510643 $\Rightarrow -2,249887$	1,562306 $\Rightarrow -2,246118$	[1,427051; 1,562306]
5	0,135255	-	-	-

   El método para ya que $|a_5 - b_5| = 0,135255 < \epsilon = 0,2$. Un estimado del mínimo lo podemos hallar tomando la mitad del intervalo y evaluando en la función:

$$\frac{b_4 + a_4}{2} = \frac{1,562306 + 1,427051}{2} = 1,494678$$

   Como sabemos que el mínimo de la función es 1,5, entonces podemos concluir que el método encuentra el mínimo con un error de $|1,5 - 1,494678| = 0,005322 = 0,5\,\%$

   A continuación se encuentra el código fuente de Matlab utilizado para generar los resultados presentados antes.

```
clc
f = inline('x^2-3*x');
a = 0.5;
b = 2;
a_k = a;
b_k = b;
alfa=(sqrt(5)-1)/2;
epsilon=0.2;
k=0;
parar = false;

fprintf('==================\n==METODO DE LA SECCIONAUREA==\n==================\n');
fprintf('Los siguientes son los parametros utilizados:\n a_k=%f, b_k=%f, alfa=%f, epsilon=%f\
```

```
fprintf('\nComienza la iteracion para k=%d\n',k);

while (~parar)
    distancia = abs(b_k-a_k);
    fprintf('\nk=%d\n\t%f < %f ?',k,distancia,epsilon);
    if(distancia<epsilon)
        parar = true;
    else
        lambda_k=a_k+(1-alfa)*(b_k-a_k);
        miu_k=a_k+alfa*(b_k-a_k);
        fprintf('\n\t\tlambda_k = %f\t==> f(lambda_k) = %f' , lambda_k, f(lambda_k));
        fprintf('\n\t\tmiu_k = %f\t==> f(miu_k) = %f' , miu_k, f(miu_k));
        if f(lambda_k)>f(miu_k)
            a_k=lambda_k;
            lambda_k=miu_k;
        else
            b_k=miu_k;
            miu_k=lambda_k;
        end
        fprintf('\n\n\t\tNuevo intervalo:\n\t\ta_k = %f, b_k = %f',a_k,b_k);
    end
    k=k+1;
end
fprintf('\n\n===> Termino el metodo:\n a_k = %f, b_k = %f\nf(a_k) = %f, f(b_k) = %f\n',a_k,b_
fprintf('\nMinimo aprox. =%f\n',(a_k+b_k)/2);
```

b) Método de Bisección:

$\mathbf{k}$	$\lvert\mathbf{a_k}-\mathbf{b_k}\rvert$	$\frac{\mathbf{a_k}+\mathbf{b_k}}{\mathbf{2}}$	$\mathbf{f'}(\frac{\mathbf{a_k}+\mathbf{b_k}}{\mathbf{2}})$	$[\mathbf{a_k},\mathbf{b_k}]$
0	1,500000	1,250000	$-0{,}500000$	$[1{,}250000; 2{,}000000]$
1	0,750000	1,625000	0,250000	$[1{,}250000; 1{,}625000]$
2	0,375000	1,437500	$-0{,}125000$	$[1{,}437500; 1{,}625000]$
3	0,187500	–	–	–

El método para ya que $|a_3 - b_3| = 0{,}187500 < \epsilon = 0,2$. Un estimado del mínimo lo podemos hallar tomando la mitad del intervalo y evaluando en la función:

$$\frac{b_2 + a_2}{2} = \frac{1{,}437500 + 1{,}625000}{2} = 1{,}531250$$

Como sabemos que el mínimo de la función es 1,5, entonces podemos concluir que el método encuentra el mínimo con un error de $|1{,}5 - 1{,}531250| = 0{,}03125 = 3{,}125\,\%$

A continuación se encuentra el código fuente de Matlab utilizado para generar los resultados presentados antes.

```
clc
```

```
f = inline('x^2-3*x');
f_d = inline('2*x-3');

a = 0.5;
b = 2;
a_k = a;
b_k = b;
epsilon=0.2;
%epsilon=0.0001;
k=0;
parar = false;

fprintf('==============================\n==METODO DE BISECCION==\n==========================
fprintf('Los siguientes son los parametros utilizados:\n a_k=%f, b_k=%f epsilon=%f\n',a_k, b_
fprintf('\nComienza la iteracion para k=%d\n',k);

%if(f_d(a)<0 & f_d(b)<0)


while (~parar)
    distancia = abs(b_k-a_k);
    fprintf('\nk=%d\n\t%f < %f ?',k,distancia,epsilon);
    fprintf('\n\ta_k = %f, b_k = %f',a_k,b_k);
    if(distancia<epsilon)
        parar = true;
    else
        medio = (a_k+b_k)/2;
        medio_f = f_d(medio);
        fprintf('\n\tmedio=%f,f_d(medio)=%f',medio,medio_f);
        if(medio_f<0 & f_d(a_k)<0)
            a_k=medio;
        elseif(medio_f<0 & f_d(b_k)<0)
            b_k=medio;
        elseif(medio_f>0 & f_d(a_k)>0)
            a_k=medio;
        elseif(medio_f>0 & f_d(b_k)>0)
            b_k=medio;
        end;
        fprintf('\n\ta_k=%f,b_k=%f\n\tf_d(a_k)=%f,f_d(b_k)=%f',a_k,b_k,f_d(a_k),f_d(b_k));
    end;
    k=k+1;
end
fprintf('\n\ta_k=%f,b_k=%f\n\tf_d(a_k)=%f,f_d(b_k)=%f',a_k,b_k,f_d(a_k),f_d(b_k));
fprintf('\n\n===> Termino el metodo:\n\nMinimo aprox. =%f\n',(a_k+b_k)/2);
```

Estimación del número de iteraciones

a) Sección Aúrea:
El método de la sección aúrea posee una convergencia que es de tipo Q-lineal. La sucesión

de puntos generada por el método tiene la forma:

$$|b_{k+1} - a_{k+1}| = \alpha|b_k - a_k|,$$

Donde $\alpha = \frac{\sqrt{5}-1}{2}$. Considerando que el método comienza desde a_0 y b_0 arbitrarios, tenemos que:

$$\begin{aligned}
|b_1 - a_1| &= \alpha|b_0 - a_0| \ (0)\\
|b_2 - a_2| &= \alpha|b_1 - a_1| \ (1)\\
|b_3 - a_3| &= \alpha|b_2 - a_2| \ (2)\\
&\dots\\
|b_{k+1} - a_{k+1}| &= \alpha|b_k - a_k| \ (k)
\end{aligned}$$

Reemplazando (0) en (1) queda $|b_2 - a_2| = \alpha(\alpha|b_0 - a_0|) = \alpha^2|b_0 - a_0|$, reemplazando ahora esta nueva ecuación en (2) $|b_3 - a_3| = \alpha(\alpha^2|b_0 - a_0|) = \alpha^3|b_0 - a_0|$, y así sucesivamente hasta reemplazar $(k-1)$ en (k) obtenemos que:

$$|b_{k+1} - a_{k+1}| = \alpha^k|b_k - a_k|$$

El método terminará de iterar cuando se cumpla que:

$$|b_{k+1} - a_{k+1}| = \alpha^k|b_k - a_k| < 10^{-5}$$

No conocemos b_k ni a_k, pero sabemos que la distancia entre estos dos puntos se irá acercando a cero. Cuando la distancia entre ellos es menor que 1 entonces la ecuación anterior se cumplirá necesariamente si se cumple la condición $\alpha^k < 10^{-5}$. Resolviendo para k encontramos que $\frac{\sqrt{5}-1}{2}^{24} = 0{,}0000096449 < 10^{-5} \Rightarrow k = 24$.

En conclusión, suponiendo que $|b_k - a_k| < 1$ y que $\epsilon = 10^{-5}$, entonces el método tomará como máximo 24 iteraciones antes de parar.

Nota: Utilizando la implementación de la sección aúrea en matlab para el problema 1 encontramos que el algoritmo toma 20 iteraciones.

b) Bisección:
Utilizando un razonamiento análogo al anterior llegamos a que $\alpha^k < 10^{-5}, k = \frac{1}{2} \Rightarrow k = 17$.

En conclusión, suponiendo que $|b_k - a_k| < 1$ y que $\epsilon = 10^{-5}$, entonces el método tomará como máximo 17 iteraciones antes de parar.

Nota: Utilizando la implementación del método de bisección en matlab para el problema 1 encontramos que el algoritmo toma 14 iteraciones.

2. *a*) $d = (-1, 1)^t$ será una dirección de descenso en el punto $x_0 = (1, 2)^t$ si y sólo si:

$$\nabla f(x_0)^t d < 0$$

Luego,

$$\nabla f(1,2)^t \times d = \begin{pmatrix} \frac{\partial f(1,2)}{\partial x} \\ \frac{\partial f(1,2)}{\partial y} \end{pmatrix} \times d = \begin{pmatrix} 4(1) + 2(2) - 1 \\ 2(1) - 2 - 3 \end{pmatrix} \times d = (7, -3)^t \times (-1, 1) = -7 - 3 = -10 < 0$$

Por lo tanto d es una dirección de descenso en x_0

b) Encuentre $\alpha = argmin_{\alpha>0} f(x_0 + \alpha d)$
Como $f(x,y) = 2x^2 + 2xy - 0,5y^2 - x - 3y$ es una función cuadrática, entonces la podemos escribir así:

$$f(x) = \frac{1}{2} x^t \begin{pmatrix} 4 & 2 \\ 2 & -1 \end{pmatrix} x - \begin{pmatrix} 1 \\ 3 \end{pmatrix}^t x$$

Donde $x \in \mathbb{R}^2$. Para calcular α_k mediante búsqueda lineal exacta utilizamos el hecho de que la función es caudrática y que:

$$\alpha_k = \frac{-[Qx_k - b]d_k}{d_k^t Q d_k}$$

En este caso $\alpha_k = -10$. Ocurre que el valor del α por búsqueda lineal exacta es menor que cero. En este caso la matriz Q asociada a la función cuadrática no es positiva definida. En efecto, existe un $x = (-1, 1)$ tal que $x^t Q x < 0$. Esta función no es convexa. Por lo tanto el valor de α hay que aproximarlo usando búsqueda lineal inexacta.

3. Por una parte, si M es una matriz P.D entonces se cumple que $x^t M x > 0, \forall x \in \mathbb{R}^n$.

   Por otra parte d será una dirección de descenso en $\bar{x}$ si y sólo si $\nabla f(\bar{x})^t d < 0$.

   Ahora bien, si fijamos $d = -M\nabla f(\bar{x})$, con $\bar{x} \in \mathbb{R}^n$ entonces tenemos que:

   $$\nabla f(\bar{x})^t d = \nabla f(\bar{x})^t [-M\nabla f(\bar{x})] = \nabla f(\bar{x})^t [(-1)M\nabla f(\bar{x})] = -1[(\nabla f(\bar{x}))^t M(\nabla f(\bar{x}))] < 0$$

   Ya que M es una matriz P.D.

   Por lo tanto queda demostrado que $d = -M\nabla f(\bar{x})$ es una dirección de descenso para f y cualquier $\bar{x} \in \mathbb{R}^n$

4. Aplicaremos el método de Newton y de la secante para resolver: $f_1(x) = -\frac{x^4}{4} + \frac{x^3}{3} + x^2$ y $f_2(x) = sin(x) - cos(2x)$, partiendo de $x_0 = 1$ y con criterio de parada $\epsilon = 10^{-5}$.

   Es importante destacar que los métodos utilizan las siguientes definiciones:

   $$f_1'(x) = -4 * x^3 + 3 * x^2 + 2 * x; f_1''(x) = -12 * x^2 + 6 * x + 2$$

   $$f_2'(x) = cos(x) + 2 * sin(2 * x); f_2''(x) = -sin(x) + 4 * cos(2 * x)$$

   Método de Newton:

   Para f_1, partiendo de $x_0 = 1$

k	$\|\mathbf{f'(x_k)}\|$	$\mathbf{x_k}$
0	1,000000	1,000000
1	0,625000	1,250000
2	0,053551	1,182432
3	0,000542	1,175463
4	0,000000	1,175391

   Para f_2, partiendo de $x_0 = 1$

k	$\|\mathbf{f'(x_k)}\|$	$\mathbf{x_k}$
0	2,358897	1,000000
1	1,712062	1,941278
2	0,350923	1,500414
3	0,001989	1,571194
4	0,000000	1,570796

Partiendo desde otros puntos

Para f_1, partiendo de $x_0 = -1$

k	$\|\mathbf{f}'(\mathbf{x_k})\|$	$\mathbf{x_k}$
0	5,000000	−1,000000
1	1,342773	−0,687500
2	0,313238	−0,515281
3	0,047667	−0,442057
4	0,002084	−0,426154
5	0,000005	−0,425392

Para f_2, partiendo de $x_0 = -1$

k	$\|\mathbf{f}'(\mathbf{x_k})\|$	$\mathbf{x_k}$
0	1,278293	−1,000000
1	1,015356	−2,552991
2	0,587184	−3,038985
3	0,014843	−2,892865
4	0,000022	−2,888918
5	0,000000	−2,888912

Para f_1, partiendo de $x_0 = 0$

k	$\|\mathbf{f}'(\mathbf{x_k})\|$	$\mathbf{x_k}$
0	0,000000	0,000000

Para f_2, partiendo de $x_0 = 0$

k	$\|\mathbf{f}'(\mathbf{x_k})\|$	$\mathbf{x_k}$
0	1,000000	0,000000
1	0,010061	−0,250000
2	0,000010	−0,252678
3	0,000000	−0,252680

La siguiente tabla resume la información sobre los puntos criticos encontrados con el método de newton. Según la información obtenida se determinará los puntos mínimos o máximos.

Para f_1

x_k	$f''(x_k)$	Conclusión
1.175391	-7.5262	Aproximado máximo local
-0.425392	-2.7239	Aproximado máximo local
0.000000	2	Aproximado mínimo local

Podemos ver que la función no es convexa ya que la segunda derivada no es siempre positiva.

Para f_2

x_k	$f''(x_k)$	Conclusión
1.570796	-5	Aproximado máximo local
-2.888912	3.7500	Aproximado mínimo local
-0.252680	3.7500	Aproximado mínimo local

Podemos ver que la función no es convexa ya que la segunda derivada no es siempre positiva.
Como las funciones no son convexas no podemos concluir nada sobre la globalidad de los puntos encontrados.

Método de la Secante:

Para f_1, partiendo de $x_0 = 1$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	0,999960	1,000000
1	0,624948	1,249994
2	0,157012	1,153849
3	0,016774	1,173154
4	0,000550	1,175464
5	0,000002	1,175390

Para f_2, partiendo de $x_0 = 1$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	2,358872	1,000010
1	1,712005	1,941263
2	1,126837	1,545420
3	1,009636	1,572724
4	1,000003	1,570796

Partiendo desde otros puntos

Para f_1, partiendo de $x_0 = -1$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	4,999840	−0,999990
1	1,342751	−0,687497
2	0,590232	−0,572761
3	0,183726	−0,482769
4	0,047781	−0,442095
5	0,006609	−0,427800
6	0,000311	−0,425505
7	0,000002	−0,425391

Para f_2, partiendo de $x_0 = -1$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	1,278301	−0,999990
1	1,015223	−2,553055
2	0,821504	−1,865594
3	2,224801	1,049719
4	2,426085	−3,572247
5	1,062863	−1,161248
6	2,734915	0,718537
7	1,105420	−0,635163
8	0,026971	−0,245508
9	0,007899	−0,254788
10	0,000022	−0,252686
11	0,000000	−0,252680

Para f_1, partiendo de $x_0 = 0$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	0,000020	0,000010
1	0,000000	0,000000

Para f_2, partiendo de $x_0 = 0$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	1,000040	0,000000
1	0,010060	−0,250000
2	0,000523	−0,252541
3	0,000001	−0,252680

En general los resultados conseguidos por el método de Newton y el de la secante son muy parecidos. En este caso la diferencia principal ocurrió con la función f_2 partiendo desde el punto $x_0 = -1$, en donde los métodos convergieron a diferentes puntos estacionarios. También se puede comentar que el método de la secante tarda poco más que el método de Newton en converger a la solución. En la tabla a continuación se comparan el número de iteraciones de ambos métodos:

Iteraciones	$x_0 = 1$	$x_0 = -1$	$x_0 = 0$
Newton f_1	5	6	1
Secante f_1	6	8	2
Newton f_2	5	6	4
Secante f_2	5	12	4

A continuación se encuentra el código fuente de Matlab utilizado para generar los resultados presentados antes.

Para el Método de Newton:

```
clc

%f = inline('-(1.4)*x^4+(1/3)*x^3+x^2');
%f_d = inline('-4*x^3+3*x^2+2*x');
%f_d2 = inline('-12*x^2+6*x+2');

f = inline('sin(x)-cos(2*x)');
f_d = inline('cos(x)+2*sin(2*x)');
f_d2 = inline('-sin(x)+4*cos(2*x)');

x_0 = 1;
x_k=x_0;
epsilon=10^-5;
k=0;
parar = false;

fprintf('==============================\n==METODO DE NEWTON==\n==============================\n')
fprintf('Los siguientes son los parametros utilizados:\n x_0=%f, epsilon=%f\n',x_0,epsilon);
while (~parar)
    criterio_parada = abs(f_d(x_k));
    fprintf('\nk=%d\n\t%f < %f ? x_k = %f',k,criterio_parada,epsilon,x_k);
    if(criterio_parada <epsilon)
        parar=true;
    else
        x_k = x_k-(f_d(x_k)/f_d2(x_k));
        k=k+1;
    end;
end
fprintf('\n\n===> Termino el metodo:');
fprintf('\nMinimo aprox. =%f\n',x_k);
```

Para el Método de la Secante:

```
clc

%f = inline('-(1.4)*x^4+(1/3)*x^3+x^2');
%f_d = inline('-4*x^3+3*x^2+2*x');

%f = inline('sin(x)-cos(2*x)');
f_d = inline('cos(x)+2*sin(2*x)');


x_0 = 0;
x_1 = x_0 + 10^-5; %Punto muy cercano al anterior
```

```
x_k  = x_1;
x_k_1 = x_0;

epsilon=10^-5;
k=0;
parar = false;

fprintf('============================\n==METODO SECANTE==\n============================\n');
fprintf('Los siguientes son los parametros utilizados:\n x_0=%f, epsilon=%f\n',x_0,epsilon);
while (~parar)
    criterio_parada = abs(f_d(x_k));
    fprintf('\nk=%d\n\t%f < %f ? x_k = %f',k,criterio_parada,epsilon,x_k);
    if(criterio_parada <epsilon)
        parar=true;
    else
        x_k1 = x_k-(f_d(x_k)*((x_k-x_k_1)/(f_d(x_k)-f_d(x_k_1))));
        x_k_1= x_k;
        x_k  = x_k1;
    end;
    k=k+1;
end
fprintf('\n\n===> Termino el metodo:');
fprintf('\nMinimo aprox. =%f\n',x_k);
```

5. *a*)

   b) Resolveremos con el método de newton $f_{b1}(x) = x^2 - 1 = 0$ y $f_{b2}(x) = x^2 - 2x + 1 = 0$ y $\epsilon = 10^{-5}$. Resultados del método:

Para f_{b1}, partiendo de $x_0 = 2$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	3,000000	2,000000
1	0,562500	1,250000
2	0,050625	1,025000
3	0,000610	1,000305
4	0,000000	1,000000

Para f_{b2}, partiendo de $x_0 = 2$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	1,000000	2,000000
1	0,250000	1,500000
2	0,062500	1,250000
3	0,015625	1,125000
4	0,003906	1,062500
5	0,000977	1,031250
6	0,000244	1,015625
7	0,000061	1,007812
8	0,000015	1,003906
9	0,000004	1,001953

Partiendo desde otros puntos

Para f_{b1}, partiendo de $x_0 = -2$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	3,000000	$-2,000000$
1	0,562500	$-1,250000$
2	0,050625	$-1,025000$
3	0,000610	$-1,000305$
4	0,000000	$-1,000000$

Para f_{b2}, partiendo de $x_0 = -2$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	9,000000	$-2,000000$
1	2,250000	$-0,500000$
2	0,562500	0,250000
3	0,140625	0,625000
4	0,035156	0,812500
5	0,008789	0,906250
6	0,002197	0,953125
7	0,000549	0,976562
8	0,000137	0,988281
9	0,000034	0,994141
10	0,000009	0,997070

Para f_{b1}, partiendo de $x_0 = 0,5$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	0,750000	0,500000
1	0,562500	1,250000
2	0,050625	1,025000
3	0,000610	1,000305
4	0,000000	1,000000

Para f_{b2}, partiendo de $x_0 = 0,5$

$\mathbf{k}$	$\lvert\mathbf{f}'(\mathbf{x_k})\rvert$	$\mathbf{x_k}$
0	0,250000	0,500000
1	0,062500	0,750000
2	0,015625	0,875000
3	0,003906	0,937500
4	0,000977	0,968750
5	0,000244	0,984375
6	0,000061	0,992188
7	0,000015	0,996094
8	0,000004	0,998047