TALLER 1: OPTIMIZACIÓN DE FUNCIONES

Presentado por:

Luis Fernando Rios Zapata
Andrés Grisales Gonzalez

Curso de optimización

Profesor: RONALD AKERMAN ORTIZ GARCIA

Universidad de Antioquia

Facultad de Ingeniería

Departamento de Sistemas

2022

Nota: todos los algoritmos de Python realizados se encuentran en el siguiente repositorio público de Github:

https://github.com/lfernandorios/optimizacion-funciones

1. Teniendo en cuenta la siguiente función:

$f(x,\ y)\ =\ 3x\ +\ 5y$
s.a: $g(x,y)\ =\ 3x^2\ +\ y^2\ =\ 4$

Encuentre los puntos críticos usando multiplicadores de Lagrange. Use alguna de las herramientas para graficar el problema.

Solución

(1) $f(x)\ =\ \lambda g(x)\ \rightarrow\ 3\ =\ \lambda 6x\ \rightarrow\ 1\ =\ \lambda 2x$ (1)
(2) $f(y)\ =\ \lambda g(y)\ \rightarrow\ 5\ =\ \lambda 2y$
(3) $g(x,y)\ =\ 0\ \rightarrow\ 3x^2\ +\ y^2 - 4 =\ 0$

de (1): $1\ =\ \lambda 2x\ \rightarrow \lambda\ =\ \frac{1}{2x}$
de (2): $5\ =\ \lambda 2y\ \rightarrow\ \lambda\ =\ \frac{5}{2y}$

Como $\lambda\ =\ \lambda$
$\frac{1}{2x} = \frac{5}{2y}\ \rightarrow 2y\ =\ 10x\ \rightarrow\ y\ =\ 5x$

de (3): $3x^2\ +\ (5x)^2 - 4\ =\ 0\ \rightarrow\ 28x^2 - 4\ =\ 0\ \rightarrow\ 7x^2 = 1\ \rightarrow\ x = \sqrt{\frac{1}{7}}\ =\ \pm\ 0.377964473$

como $y\ =\ 5x\ \rightarrow\ y\ =\ \pm\ 1.889$

Los puntos obtenidos son: (0.38 , 1.89) ; (0.38, -1.89) ; (-0.38, 1.89) ; (-0.38, -1.89) y los evaluamos en la función:

f(0.38 , 1.89) = 10.59
f(0.38, -1.89) = -8.31

f(-0.38, 1.89) = 8.31
f(-0.38, -1.89) = -10.59

R: Por lo tanto los puntos críticos son (0.38 , 1.89) y (-0.38, -1.89)

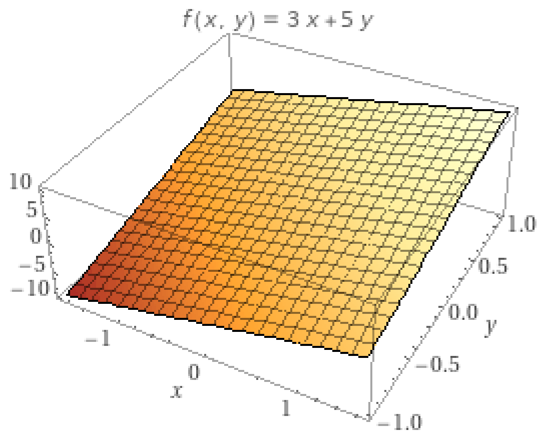


Figura 1. Función donde gráficamente se observan los máximos.

2. Para cada uno de los siguientes ejercicios debe

Resolverlo analíticamente (principios de primera y segunda derivada)

Programar en el lenguaje de programación que maneje, el ejercicio asignado.

Solución:

a) $f(x) \; = \; -1x^2 + 8x - 12, \; con \; x_0 = 0 \, ; \; x_1 = 2 \, ; \; x_2 = 6$

Criterio primera y segunda derivada

$f(x) = -1x^2 + 8x - 12$

$f'(x) = -2x + 8 = 0$ → (el valor que hace cero a esta función es x = 4)

evaluando intervalos:

$(-\infty, 4) \rightarrow x = 2 \rightarrow f'(2) = -4 + 8 = 4$ {es positivo}

$(4, \infty) \rightarrow x = 5 \rightarrow f'(5) = -10 + 8 = -2$ {es negativo}

de positivo a negativo, hay un máximo relativo

$f''(x) = -2$

Interpolación Cuadrática

```
InterpolacionCuadratica
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/InterpolacionCuadratica.py
Puntos:  [0, 2, 6]  |Tolerancia:  0.0001
Iteración:  1  |X3: 4.0  |f(X3):  4.0  |Error:  2.0
Puntos:  [0, 2, 6]
Iteración:  2  |X3: 4.0  |f(X3):  4.0  |Error:  2.0
Puntos:  [2, 4.0, 6]

Process finished with exit code 0
```

Figura 2. Respuesta por consola utilizando el algoritmo de la Interpolación cuadrática. Nota: se rompe el algoritmo cuando el valor del medio es igual al valor obtenido, ya que para está función tenemos una excepción al algoritmo donde se queda indefinidamente sin poder cambiar el valor intermedio. El algoritmo en las demás funciones sí corre de manera correcta.

b) Encuentre el valor de x que maximiza $f(x) = -2,5x^5 - 2x^4 + 12x$ en el problema con el uso de la búsqueda de la sección dorada. Emplee valores iniciales de $xl = 0$ y $xu = 2$ tolerancia de 0.005.

Criterio primera y segunda derivada

$f(x) = -2,5x^5 - 2x^4 + 12x$

$f'(x) = -12,5x^4 - 8x^3 + 12$ → (los valores que hacen cero a esta función son x = -1.1981 y x = 0.8614)

$f''(x) = -50x^3 - 24x^2$

Tomamos la función principal, hallamos la primera derivada,y luego la segunda derivada. Obtenemos los valores que hacen 0 a la primera derivada (en este caso -1.1981 y 0.8614), luego evaluamos estos valores en la segunda derivada para determinar cuál corresponde a un mínimo y cuál a un máximo. En este caso el máximo corresponde a 0.8614 y es el valor en x que usamos para evaluar en la función principal y obtener el máximo (8.0499).

El máximo lo encontramos cuando x = 0.8614, donde obtenemos que la función evaluada en ese punto da 8.0499, como se observa en la siguiente gráfica.

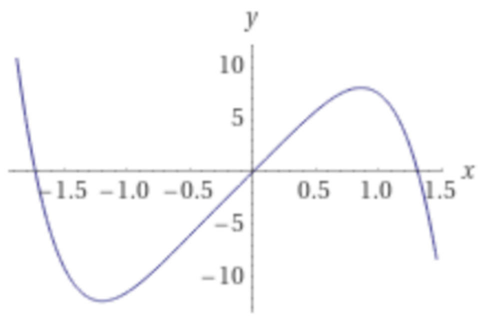


Figura 3. Función donde gráficamente se observa que el máximo corresponde a 8.0499 cuando x vale 0.8614

Sección o razón dorada

Resolviendo de manera manual el ejercicio utilizando el criterio de la primera y segunda derivada,

Mediante el algoritmo de la Sección dorada desarrollado en Python se obtienen los siguientes resultados utilizando una tolerancia de 0.0005, el valor de x entonces sería el resultado de promediar el último x lower con el último x upper, que en este caso corresponde a 0.8614,

```
RazonDorada
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/RazonDorada.py
X lower:  0  | X upper:  2  | Tolerancia:  0.0005
Iteración:  0  | xl:  0 , | xu:  2 , | error:  0.472
Iteración:  1  | xl:  0 , | xu:  1.236 , | error:  0.29169599999999996
Iteración:  2  | xl:  0.472152 , | xu:  1.236 , | error:  0.18026812800000003
Iteración:  3  | xl:  0.7639419359999999 , | xu:  1.236 , | error:  0.11140570310399989
Iteración:  4  | xl:  0.7639419359999999 , | xu:  1.055673819552 , | error:  0.06884872451827206
Iteración:  5  | xl:  0.7639419359999999 , | xu:  0.944232240035136 , | error:  0.0425485117522921 3
Iteración:  6  | xl:  0.8328128321414219 , | xu:  0.944232240035136 , | error:  0.026294980262916434
Iteración:  7  | xl:  0.8328128321414219 , | xu:  0.9016700262197371 , | error:  0.01625029780248244
Iteración:  8  | xl:  0.8328128321414219 , | xu:  0.8753665780818207 , | error:  0.010042684041934091
Iteración:  9  | xl:  0.8490683630906543 , | xu:  0.8753665780818207 , | error:  0.006206378737915297
Iteración:  10  | xl:  0.8490683630906543 , | xu:  0.8653206599551951 , | error:  0.0038355420600315604
Iteración:  11  | xl:  0.8552767404929089 , | xu:  0.8653206599551951 , | error:  0.0023703649930996162
Iteración:  12  | xl:  0.8591135177275022 , | xu:  0.8653206599551951 , | error:  0.0014648855657354298
Iteración:  13  | xl:  0.8591135177275022 , | xu:  0.8629495316242164 , | error:  0.0009052992796245718
Iteración:  14  | xl:  0.860578875036047 , | xu:  0.8629495316242164 , | error:  0.0005594749548079037
Iteración:  15  | xl:  0.860578875036047 , | xu:  0.8620439408075357 , | error:  0.00034575552207127824

Process finished with exit code 0
```

Figura 4. Respuesta por consola del algoritmo de la sección o razón dorada (se promedia el último xl con el último xu)

c) Tenga en cuenta la siguiente función: $f(x) = -x^4 - 2x^3 - 8x^2 - 5x$

Interpolación cuadrática (x0 = –2, x1 = –1, x2 = 1)

Método de Newton ($x0 = -1$). Tolerancias de 0.05 y 0.0001

Criterio de primera y segunda derivada

$$f(x) = -x^4 - 2x^3 - 8x^2 - 5x$$

$$f'(x) = -4x^3 - 6x^2 - 16x - 5$$

$$f''(x) = -12x^2 - 12x - 16$$

Tomamos la función principal, hallamos la primera derivada,y luego la segunda derivada. Obtenemos los valores que hacen 0 a la primera derivada (en este caso -0.3472), luego evaluamos este valor en la segunda derivada para determinar si corresponde a un máximo o un mínimo. En este caso corresponde a un máximo que es -0.3472 y es el valor en x que usamos para evaluar en la función principal y obtener el máximo (0.8407).

El máximo lo encontramos cuando x = -0.3472, donde obtenemos que la función evaluada en ese punto da 0.8407, como se observa en la siguiente gráfica.

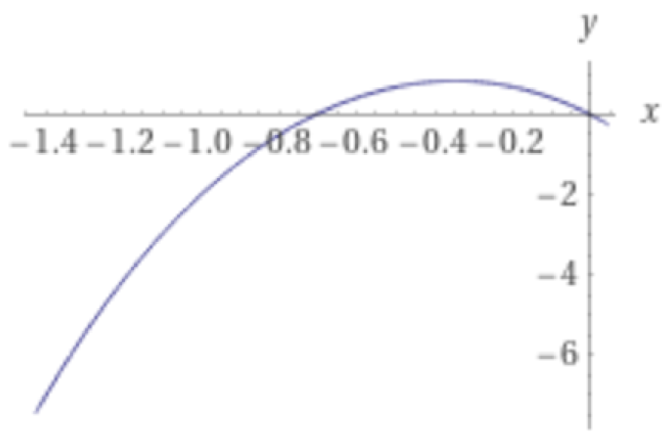
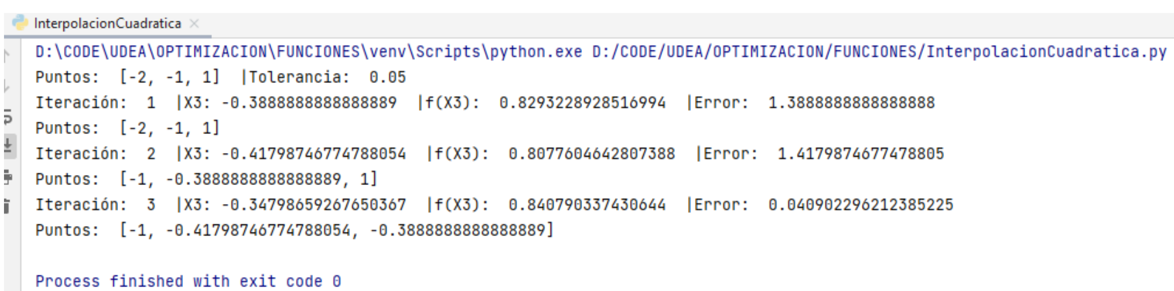


Figura 5. Función donde gráficamente se observa que el máximo corresponde a 0.8407 cuando x vale -0.3472

Interpolación cuadrática $(x0 = -2, x1 = -1, x2 = 1)$

Utilizando una tolerancia de 0.05



```
InterpolacionCuadratica
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/InterpolacionCuadratica.py
Puntos:  [-2, -1, 1]  |Tolerancia:  0.05
Iteración:  1  |X3: -0.3888888888888889  |f(X3):  0.8293228928516994  |Error:  1.3888888888888888
Puntos:  [-2, -1, 1]
Iteración:  2  |X3: -0.41798746774788054  |f(X3):  0.8077604642807388  |Error:  1.4179874677478805
Puntos:  [-1, -0.3888888888888889, 1]
Iteración:  3  |X3: -0.34798659267650367  |f(X3):  0.840790337430644  |Error:  0.040902296212385225
Puntos:  [-1, -0.41798746774788054, -0.3888888888888889]

Process finished with exit code 0
```

Figura 6. Respuesta por consola del algoritmo del algoritmo Interpolación cuadrática (Iteración 3 donde se observa el valor de x3 y el resultado al evaluarlo en la función)

Utilizando una tolerancia de 0.0001

```
InterpolacionCuadratica
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/InterpolacionCuadratica.py
Puntos:  [-2, -1, 1]  |Tolerancia:  0.0001
Iteración:  1  |X3: -0.3888888888888889  |f(X3):  0.8293228928516994  |Error:  1.3888888888888888
Puntos:  [-2, -1, 1]
Iteración:  2  |X3: -0.41798746774788054  |f(X3):  0.8077604642807388  |Error:  1.4179874677478805
Puntos:  [-1, -0.3888888888888889, 1]
Iteración:  3  |X3: -0.34798659267650367  |f(X3):  0.840790337430644  |Error:  0.040902296212385225
Puntos:  [-1, -0.41798746774788054, -0.3888888888888889]
Iteración:  4  |X3: -0.3471357500322807  |f(X3):  0.8407938478961895  |Error:  0.041753138856608196
Puntos:  [-0.41798746774788054, -0.34798659267650367, -0.3888888888888889]
Iteración:  5  |X3: -0.34724935828812825  |f(X3):  0.8407939352704734  |Error:  0.041639530600760644
Puntos:  [-0.34798659267650367, -0.3471357500322807, -0.3888888888888889]
Iteración:  6  |X3: -0.3472504704813446  |f(X3):  0.8407939352786288  |Error:  0.00011472044906390977
Puntos:  [-0.34798659267650367, -0.34724935828812825, -0.3471357500322807]
Iteración:  7  |X3: -0.34725046840445756  |f(X3):  0.8407939352786288  |Error:  1.1101163293059102e-06
Puntos:  [-0.34798659267650367, -0.3472504704813446, -0.34724935828812825]

Process finished with exit code 0
```

Figura 7. Respuesta por consola del algoritmo del algoritmo Interpolación cuadrática (Iteración 7 donde se observa el valor de x3 y el resultado al evaluarlo en la función)

Metodo de Newton

Utilizando una tolerancia de 0.05

```
InterpoladorNewton
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/InterpoladorNewton.py
X0:  -1  |Tolerancia:  0.05
Iteración:  1  |X1:  -0.437500000000000  |fx1:  0.787094116210938  |Error:  0.562500000000000
Iteración:  2  |X1:  -0.346556886227545  |fx1:  0.840790740880911  |Error:  0.0909431137724551
Iteración:  3  |X1:  -0.347250399956576  |fx1:  0.840793935278599  |Error:  0.000693513729030860

Process finished with exit code 0
```

Figura 8. Respuesta por consola del algoritmo del algoritmo Interpolación cuadrática (Iteración 3 donde se observa el valor de x1 y el resultado al evaluarlo en la función).

Utilizando una tolerancia de 0.0001

```
InterpoladorNewton
D:\CODE\UDEA\OPTIMIZACION\FUNCIONES\venv\Scripts\python.exe D:/CODE/UDEA/OPTIMIZACION/FUNCIONES/InterpoladorNewton.py
X0:  -1  |Tolerancia:  0.0001
Iteración:  1  |X1:  -0.437500000000000  |fx1:  0.787094116210938  |Error:  0.562500000000000
Iteración:  2  |X1:  -0.346556886227545  |fx1:  0.840790740880911  |Error:  0.0909431137724551
Iteración:  3  |X1:  -0.347250399956576  |fx1:  0.840793935278599  |Error:  0.000693513729030860
Iteración:  4  |X1:  -0.347250466543088  |fx1:  0.840793935278629  |Error:  6.65865120593606e-8

Process finished with exit code 0
```

Figura 9. Respuesta por consola del algoritmo del algoritmo Interpolación cuadrática (Iteración 4 donde se observa el valor de x1 y el resultado al evaluarlo en la función).