Universidad del Atlántico

MATUA

Revista Del Programa De Matemáticas Páginas: 40–54

**Facultad de Ciencias Básicas**
©Programa de Matemáticas
Vol. IV , Nº 1, (2017)

# APLICACIÓN DEL MÉTODO DE DESCENSO DE MAYOR PENDIENTE PARA OPTIMIZAR FUNCIONES DE VARIAS VARIABLES.

[1]Julio Cesar Romero Pabon, [2]Humberto Barrios E., [3]María J. Ortega Wilches.

[1]*Universidad del Atlántico,* [2]*Universidad Popular del Cesar,* [3]*Universidad de la Costa CUC.*

[1]*julioromero@mail.uniatlantico.edu.co,* [2]*hbarrios@unicesar.edu.co,* [3]*mortega22@cuc.edu.co*

## Resumen

Los Métodos de Optimización Basados en Derivadas, son técnicas básicas utilizadas en la solución iterativa de problemas de optimización sin restricciones. Ofrecen la forma más simple y directa de resolver estos problemas, en términos prácticos son una referencia con relación a la dificultad de implementación y velocidad de convergencia. En general, las técnicas avanzadas se comparan con estas técnicas básicas. La estructura que presentan estos métodos con:
1. Se inicia en un punto.
2. Se determina la dirección de descenso mediante una regla fija. (Primera diferencia entre Algoritmos)
3. Y luego se desplaza hacia el mínimo en esa dirección. (Búsqueda lineal).

La forma general de los métodos básicos de descenso se puede expresar como, $\vec{x}_{i+1} = \vec{x}_i + \alpha\vec{d}$.

*Palabras claves:* optimización, minimizar, maximizar y gradiente.

## Abstract

Derivative Based Optimization Methods are basic techniques used in the iterative solution of unrestricted optimization problems. They offer the simplest and most direct way of solving these problems, in practical terms they are a reference in relation to the difficulty of implementation and speed of convergence. In general, advanced techniques are compared with these basic techniques. The structure of these methods with:
1. It starts at one point.
2. The direction of descent is determined by a fixed rule. (First difference between Algorithms)
3. And then it moves towards the minimum in that direction. (Linear Search).

The general form of the basic methods of descent can be expressed as, $\vec{x}_{i+1} = \vec{x}_i + \alpha\vec{d}$

*Keywords:* optimization, minimize, maximize, and gradient.

## 1. Introducción

Las técnicas de búsqueda lineal son realmente procedimientos de optimización para una sola variable, y que es realizado repetidamente en problemas de varias variables. La elección de una dirección de búsqueda tiene un alto costo computacional, es por ello que los métodos de descenso basados en gradiente sufren modificaciones con el objeto de minimizar o reducir el número de cálculos de gradiente, Hessiano, e inversión de matrices.

La modificación fundamental consiste en reducir el problema a uno de optimización a lo largo de la dirección de descenso. Específicamente se debe resolver el sub-problema de optimización:

Encontrar $\alpha$

$$\min_{\alpha}(f(x_{i-1} - \alpha\vec{d}))$$

Donde $\vec{d}$ es la dirección de descenso. Hallado el $\alpha$ óptimo se inicia una nueva iteración de descenso.

Este sub-problema es sensiblemente más sencillo que la optimización general ya que es un problema de una dimensión con una única variable $\alpha$.

La elección de un método adecuado de búsqueda lineal es de gran importancia en un algoritmo de optimización. La búsqueda lineal es responsable de un alto porcentaje del costo de la evaluación de la función objetivo. Estos métodos pueden ser con o sin uso de derivadas.

## 2. ANÁLISIS DEL MÉTODO DE OPTIMIZACIÓN

### *2.1. MÉTODO DEL DESCENSO MÁS RÁPIDO*

Este método, denominado también método del gradiente, es una de las técnicas más antiguas para minimizar una función definida en un espacio multidimensional. Su filosofía es muy sencilla: la dirección contraria a la del vector gradiente en un punto es la dirección de más rápido decrecimiento de la función en ese punto.
El procedimiento a seguir es el siguiente:

1. Se selecciona un punto inicial $x_i$ sobre la superficie, se determina el gradiente $\nabla f(x_i)$ en ese punto, y se multiplica dicho gradiente por menos uno, para que el vector gradiente apunte hacia el mínimo de la función y no hacía el máximo de la misma.

2. Para determina un nuevo punto, se hace uso del algoritmo iterativo del método de mayor pendiente, el cual esta dado por fórmula: $\vec{x}_{i+1} = \vec{x}_i - \alpha\nabla f(x_i)$ ,donde $\alpha$ es un número positivo que minimiza la $f(x_i - \alpha\nabla f(x_i))$ . Esto significa que a partir del punto se busca en la dirección del gradiente negativo, hasta un nuevo punto $x_{i+1}$, el cual representa el mínimo de esa recta.

3. Se repite el paso 2 hasta que se encuentre un punto $x_{i+1}$ tal que $\nabla f(x_i + 1) = 0$.

El método de la máxima pendiente, es un método muy usado por su sencillez. Para ciertos tipos de funciones puede zigzaguear mucho y ser lento. Únicamente requiere primeras derivadas. Es un método de

descenso. Dado un punto crítico $x_i$ se escoge la dirección $d_i$ como mejor dirección local de descenso en $x_i$ . Sean: $\lambda$ un valor fijo, positivo y suficientemente pequeño, $d \neq 0$ una dirección tal que $\|d\|_2 = 1$ ,

$$f(x_i + \lambda d) = f(x_i) + \lambda f'(x_i)^T d + \frac{\lambda^2}{2} f''(x_i) d_i \cdots$$

La expresión anterior depende únicamente de $d$ y como $\lambda$ es pequeño se puede aproximar por

$$\psi(d) = f(x_i) + \lambda f''(x_i)^T d$$

Para escoger $d$ de manera óptima local, hay que resolver

$$\text{mín}\, \psi(d) = f(x_i) + \lambda f''(x_i)^T d$$

$$\|d\|^2 - 1 = 0$$

La función $\psi$ es afín, luego convexa y la función que define la igualdad es estrictamente convexa. Las condiciones necesarias de KKT (karush-Kuhn-Tucker) dan:

$$\lambda f'(x_i) + 2v_i d = 0$$

Si se escogen

$$d = -\frac{f'(x_i)}{\|f'(x_i)\|_2}$$

$$v_1 = \frac{\lambda \|f'(x_i)\|_2}{2} > 0$$

Se cumplen las condiciones necesarias y suficientes de KKT. Como las direcciones $-\frac{f'(x_i)}{\|f'(x_i)\|_2}$ y $-f'(x_i)$ son equivalentes se puede tomar la segunda como la dirección de $d_i$ , lo que garantiza que sea una dirección de descenso. El algoritmo se puede esquematizar así:

$$\begin{aligned}
\text{dato} : x_0, \varepsilon_g, \text{Maxit}, ... \\
\text{Para} \quad k = 0, ..., \text{Maxit} \\
\text{Si} \quad \|f'(x_i)\| \leq \varepsilon_g \quad \text{ent parar} \\
d_i = -f'(x_i) \\
\lambda_k^+ = \arg \text{mín}\, f(x_i + \lambda d_k), \lambda > 0 \\
x_{i+1} = x_i + \lambda_k^+ d_k \\
\text{Fin para} \quad k
\end{aligned}$$

No siempre este método funciona bien, ni si quiera para funciones cuadráticas. A continuación se presentan los resultados del programa realizado en Matlab para la aplicación de este método.

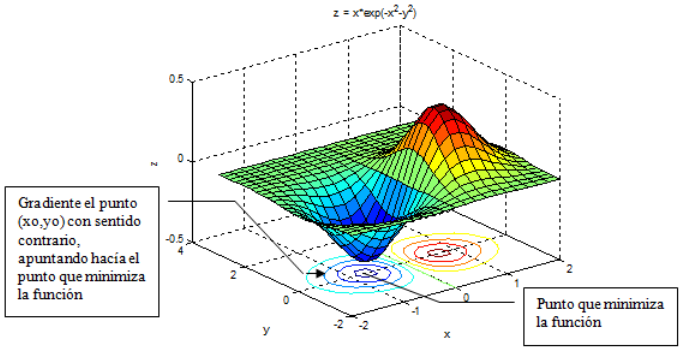
**Proposición 1.** *: Sean $f(x) = \frac{1}{2} x^T H x + c^T x$ , con $H$ definida positiva, $x^+$ el único minimizador global, $\lambda_1 > 0$ el valor propio de $H$ más grande, $\lambda_n > 0$ el valor de $H$ más pequeño, $\{x_k\}$ una sucesión determinada por el método del descenso más pendiente, $k = k_2(H) = (\rho(H^T H))^{1/2} \geq 1$ , el condicionamiento o número de condición espectral de $H$ (para matrices definidas positivas $k = \frac{\lambda_1}{\lambda_n}$. Entonces*

$$f(x_{k+1}) - f(x^+) \leq \frac{(\lambda_1 - \lambda_n)^2}{(\lambda_1 + \lambda_n)^2}(f(x_{k+1}) - f(x^+))$$
$$f(x_{k+1}) - f(x^+) \leq \frac{(k-1)^2}{(k+1)^2}(f(x_{k+1}) - f(x^+))$$
$$x_k \to x^+$$
$$(x_{k+1} - x_k)(x_k - x_{k-1}) = 0$$
$$f'(x_k)^T f'(x_{k-1}) = 0$$

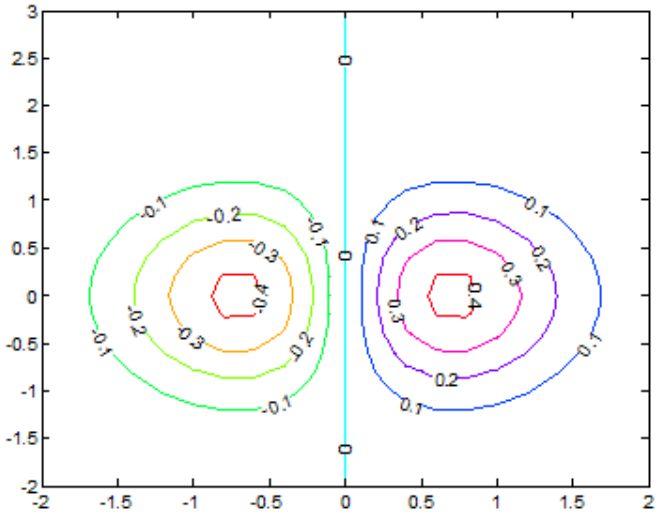
Según los resultados anteriores, el método del descenso de mayor pendiente tiene convergencia global. La sucesión $\{f(x_k)\}$ tiene convergencia lineal. La tasa de convergencia está acotada superiormente por $\frac{(k-1)^2}{(k+1)^2} = \frac{(\lambda_1 - \lambda_n)^2}{(\lambda_1 + \lambda_n)^2}$ . Es decir, entre más pequeño sea el condicionamiento (más cercano a 1), se garantiza una taza de convergencia menor, o sea, una convergencia más rápida. Dicho de otra forma, la convergencia será mejor cuando $\lambda_1$ y $\lambda_n$ sean semejantes. El caso "perfecto"se tiene cuando $\lambda_1 = \lambda_n$ , o sea $k = 1$, . Dos pasos consecutivos son ortogonales, o sea, la sucesión de puntos $\{x_k\}$ avanza formando ángulos de 90 grados.

### 2.2. *Calcular el mínimo de la función* $z = xe^{-x^2-y^2}$

Para el analizar esta situación, se procedió de la siguiente forma: Primero se elaboró la gráfica de la función en Matlab, con sus respectivas curvas de contornos:




Gráfica de curvas de contornos en le plano $xy$


3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
-0.1
-0.2
-0.3
-0.4
0.1
0.2
0.3
0.4

## 3. PROGRAMACIÓN DEL MÉTODO DE MAYOR PENDIENTE

PROGRAMACIÓN DEL MÉTODO DE MAYOR PENDIENTE

```
function mayorpendiente
clc
clear all
syms x y a
disp('                                UNIVERSIDAD DEL ZULIA')
disp('                                    OPTIMIZACION ')
disp('                                     PROYECTO 1')
disp(' ')
disp('                           METODO DE DESCENSO DE MAYOR PENDIENTE ')
disp('                       APLICADO A LA.FUNCION...F(X,Y)-X*EXP(-X^2-Y^2)
')

disp(' ')
disp('FUNCIÓN A MINIMIZAR')
disp(' ')
%f=x*exp(-x^2-y^2)
%f=2*x^2+3*y^2+x-2*y-5
f=input('f =  ')
disp(' ')
disp('Fórmula del Método de descenso de mayor pendiente:    Xi+1=Xi-
a*(transpuesta del gradiente),   con   0 < a <1')
disp(' ')

format short
disp('Punto inicial')
x0=input('x0 = ')
%x0=[-0.2 -0.05]
%x0=[1 1]
 X(1)=x0(1);
 Y(1)=x0(2);
disp('_____________________________________________________________________');
disp('Num Iter.   x     y     f(x)     dx      dy      alfa      Tiempo (seg)')
disp('_____________________________________________________________________');
%for i=[1:30]
tol=1;
i=0;
while tol>=10^-6
    i=i+1;
    if i>50
        disp(' ')
        disp('Escoja otro punto inicial')
        F(i)=00;
        X(i+1)=[00];
        Y(i+1)=[00];
        tiempo(i)=0;
        break
    end
    tic;
    %i;
    clear x y, syms x y
    f;
    d=-(jacobian(f));
    x=x0(1);
    y=x0(2);
    %fx=eval(f);
    d=eval(d);
    x=x0(1)+a*d(1);
    y=x0(2)+a*d(2);
    fa=eval(f);
```

```
    dfa=diff(fa);
    if dfa==0
        disp('No hay más iteraciones')
        n=i-1;
        if n==0
            i=1
        else
            i=n
        end
            break

    else
    A=eval(solve(dfa));
    end

    if A(1)>0
        A=A(1);
    else
        if  A(2)>0
        A=A(2);
        else
            disp('Alfa es negativo, Prueba con otro punto inicial')
            break
        end
    end

    X(i+1)=X(i)+A*d(1);
    Y(i+1)=Y(i)+A*d(2);
    xo=[X(i+1) Y(i+1)];
    %clear x y, syms x y
    x=xo(1);
    y=xo(2);
    fxl=eval(f);
   format long
disp(sprintf('  %d   %d   %d   %d   %0.6f   %0.6f   %0.6d   %0.4d', i,
xo(1),xo(2),fxl,d(1),d(2),A,toc))
disp('_____|_____|_____|_____|_____|_____|_____|_____|');
tiempo(i)=toc;
%tol=abs(fx-fxl);
 tol=norm([d(1) d(2)]);
 F(i)=fxl;
end

mini=min([F]);
tiemp=0;
for i=1:i
    tiemp=tiempo(i)+tiemp;
    if F(i)==mini
        Numero_de_iteraciones=i
        x=X(i+1);
        y=Y(i+1);
        Punto_Minimo_X_Y=[x,y]
        Minimo_de_f=mini
        Tiempo_seg=tiemp
    end
end
```

## 4. APLICACIÓN DEL MÉTODO DE DESCENSO DE MAYOR PENDIENTE PROGRAMADO

FUNCIÓN A MINIMIZAR
f= x*exp(-x^2-y^2)

Fórmula del Método de descenso de mayor pendiente: Xi+1=Xi-(alfa)*(transpuesta del gradiente), con 0 < a <1

Punto inicial: xo = [0.1 0.5]

| Iter. | x | y | f(x) | dx | dy | alfa | Tiempo (sg) |
|---|---|---|---|---|---|---|---|
| 1 | -6.781630e-001 | 5.794044e-001 | -3.060576e-001 | -0.755631 | 0.077105 | 1.029819 | 1.4100e-001 |
| 2 | -7.366823e-001 | 5.914670e-003 | -4.281274e-001 | 0.036190 | -0.354662 | 1.617003 | 7.8000e-002 |
| 3 | 6.999656e-00 | 1.406821e-001 | 4.204342e-001 | 0.049632 | -0.005064 | 28.946142 | 3.1000e-002 |
| 4 | -7.214644e-003 | 7.071048e+000 | 1.391842e-024 | -0.012071 | -0.118295 | 58.585409 | 4.6000e-002 |
| 5 | -1.145984e+000 | -6.954847e+000 | -3.034234e-022 | -0.000000 | 0.000000 | 5.903453e+021 | 3.1000e-002 |

Mínimo = -0.42812744816046

Numero_de_iteraciones= 2

Punto_Minimo_X_Y= (-0.73668234791615 0.00591466979227)

Minimo_de_f= -0.42812744816046

Tiempo_Total_seg= 0.21900000000000

### 4.1. CALCULO DEL MÍNIMO DE LA FUNCIÓN

$$f(x, y) = xe^{-x^2-y^2}$$

UTILIZANDO EL COMANDO DE MATLAB fminunc
La instrucción en Matlab para esta función fueron:

```
clc
%Punto inicial
x0 = [-1; -1];
%Opciones
options=optimset('Display','iter');
%Calcula los mínimos con el comando fminunc
f = fminunc('x(1)*exp(-x(1)^2-x(2)^2)',x0,options)
```

Los resultados fueron:

| Iteration | Func-count | f(x) | Step-size | Gradient's infinity-norm |
|---|---|---|---|---|
| 0 | 3 | -0.135335 | | 0.271 |
| 1 | 9 | -0.37501 | 3.92279 | 0.448 |
| 2 | 12 | -0.401885 | 1 | 0.258 |
| 3 | 15 | -0.426968 | 1 | 0.0554 |
| 4 | 21 | -0.428876 | 0.37464 | 0.00354 |
| 5 | 24 | -0.428882 | 1 | 0.000595 |
| 6 | 27 | -0.428882 | 1 | 1.11e-006 |

Optimization terminated: relative infinity-norm of gradient less than options.TolFun.

```
f =
  -0.7071
   0.0000
```

### 4.2. CALCULO DEL MÍNIMO DE LA FUNCIÓN

$$f(x, y) = xe^{-x^2-y^2}$$

UTILIZANDO EL COMANDO DE MATLAB fminsearch
La instrucción en Matlab para esta función fueron:

```
clc
%Punto inicial
x0 = [-1; -1];
%Opciones
options=optimset('Display','iter');
%Calcula los mínimos con el comando fminunc
P=fminsearch('x(1)*exp(-x(1)^2-x(2)^2)',x0,options)
```

| Iteration | Func-count | min f(x) | Procedure |
|---|---|---|---|
| 0 | 1 | -0.135335 | |
| 1 | 3 | -0.135335 | initial simplex |
| 2 | 5 | -0.150573 | expand |
| 3 | 7 | -0.176359 | expand |
| 4 | 9 | -0.21288 | expand |
| 5 | 11 | -0.288762 | expand |
| 6 | 12 | -0.288762 | reflect |
| 7 | 14 | -0.319814 | reflect |
| 8 | 16 | -0.381601 | expand |
| 9 | 17 | -0.381601 | reflect |
| 10 | 19 | -0.427337 | reflect |
| 11 | 21 | -0.427337 | contract inside |
| 12 | 23 | -0.427337 | contract outside |
| 13 | 24 | -0.427337 | reflect |
| 14 | 26 | -0.427337 | contract outside |
| 15 | 28 | -0.427337 | contract outside |
| 16 | 30 | -0.428613 | contract inside |
| 17 | 31 | -0.428613 | reflect |
| 18 | 33 | -0.428613 | contract inside |
| 19 | 35 | -0.428787 | contract inside |
| 20 | 37 | -0.428826 | contract inside |
| 21 | 39 | -0.428862 | contract inside |
| 22 | 41 | -0.428862 | contract inside |
| 23 | 43 | -0.428872 | contract outside |
| 24 | 45 | -0.428881 | contract inside |
| 25 | 47 | -0.428881 | contract outside |
| 26 | 49 | -0.428881 | contract inside |
| 27 | 50 | -0.428881 | reflect |
| 28 | 52 | -0.428881 | contract outside |
| 29 | 54 | -0.428882 | contract inside |
| 30 | 56 | -0.428882 | contract inside |
| 31 | 58 | -0.428882 | contract outside |
| 32 | 60 | -0.428882 | contract inside |
| 33 | 62 | -0.428882 | contract inside |
| 34 | 64 | -0.428882 | contract outside |
| 35 | 66 | -0.428882 | contract inside |
| 36 | 68 | -0.428882 | contract inside |

```
Optimization terminated:
 the current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-004
 and F(X) satisfies the convergence criteria using OPTIONS.TolFun of 1.000000e-004
F =-0.7071
```

## 5. CALCULAR EL MÍNIMO DE LA FUNCIÓN CUADRÁTICA CONSTRUIDA

Ahora se generará una función cuadrática, partiendo de $f(X) = X^T AX + b^T X + C$ , en donde $X = \begin{pmatrix} x \\ y \end{pmatrix}$ , $A$ es una matriz de 2x2 definida positiva, $b$ es un vector de 2x1, y $c$ es un número real.

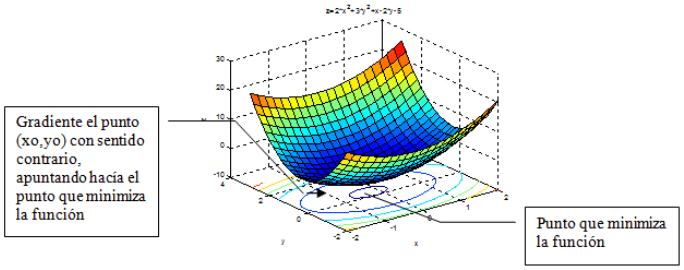
$$f(x,y) = (x,y)\begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + (1,-2)\begin{pmatrix} x \\ y \end{pmatrix} - 5$$

$$f(x,y) = (2x,3y)\begin{pmatrix} x \\ y \end{pmatrix} + (x-2y) - 5$$

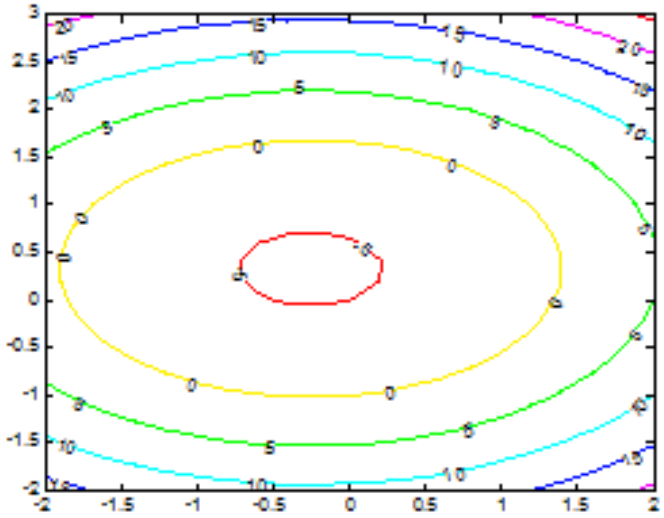
$$f(x,y) = (2x^2,3y^2) + (x-2y) - 5$$

$$f(x,y) = 2x^2 + 3y^2 + x - 2y - 5$$

Para el analizar esta situación, se procedió de la siguiente forma:
Primero se elaboró la gráfica de la función en Matlab, con sus respectivas curvas de contornos:




Gráfica de curvas de contornos en le plano $xy$



## 6. APLICACIÓN DEL MÉTODO DE DESCENSO DE MAYOR PENDIENTE PROGRAMADO

FUNCIÓN A MINIMIZAR: $f(x, y) = 2x^2 + 3y^2 + x - 2y - 5$

Punto inicial: xo = [1 1]

| Num Iter. | x | y | f(x) | dx | dy | alfa | Tiempo (sg) |
|---|---|---|---|---|---|---|---|
| 1 | -4.591837e-002 | 1.632653e-001 | -5.288265e+000 | -5.000000 | -4.000000 | 2.091837e-001 | 1.1000e-001 |
| 2 | -2.023174e-001 | 3.587641e-001 | -5.451846e+000 | -0.816327 | 1.020408 | 1.915888e-001 | 4.7000e-002 |
| 3 | -2.422151e-001 | 3.268459e-001 | -5.458086e+000 | -0.190730 | -0.152584 | 2.091837e-001 | 3.1000e-002 |
| 4 | -2.481811e-001 | 3.343034e-001 | -5.458324e+000 | -0.031140 | 0.038925 | 1.915888e-001 | 4.7000e-002 |
| 5 | -2.497030e-001 | 3.330859e-001 | -5.458333e+000 | -0.007276 | -0.005820 | 2.091837e-001 | 3.1000e-002 |
| 6 | -2.499306e-001 | 3.333703e-001 | -5.458333e+000 | -0.001188 | 0.001485 | 1.915888e-001 | 3.2000e-002 |
| 7 | -2.499887e-001 | 3.333239e-001 | -5.458333e+000 | -0.000278 | -0.000222 | 2.091837e-001 | 3.2000e-002 |
| 8 | -2.499974e-001 | 3.333347e-001 | -5.458333e+000 | -0.000045 | 0.000057 | 1.915888e-001 | 3.1000e-002 |
| 9 | -2.499996e-001 | 3.333330e-001 | -5.458333e+000 | -0.000011 | -0.000008 | 2.091837e-001 | 4.7000e-002 |
| 10 | -2.499999e-001 | 3.333334e-001 | -5.458333e+000 | -0.000002 | 0.000002 | 1.915888e-001 | 3.1000e-002 |
| 11 | -2.500000e-001 | 3.333333e-001 | -5.458333e+000 | -0.000000 | -0.000000 | 2.091837e-001 | 3.1000e-002 |

Numero_de_iteraciones = 11
Punto_Minimo_X_Y = [-0.24999998351630 0.33333331959692]
Mínimo_de_f = -5.45833333333333
Tiempo_seg = 0.47000000000000

### 6.1. CALCULO DEL MÍNIMO DE LA FUNCIÓN

$$f(x, y) = 2x^2 + 3y^2 + x - 2y - 5$$

UTILIZANDO EL COMANDO DE MATLAB fminunc
La instrucción en Matlab para esta función fueron:

```
clc
%Punto inicial
x0 = [-1; -1];
%Opciones
options=optimset('Display','iter');
%Calcula los mínimos con el comando fminunc
f = fminunc('2*x(1)^2+3*x(2)^2+x(1)-2*x(2)-5',x0,options)
```

Los resultados fueron:

| Iteration | Func-count | f(x) | Step-size | Gradient's infinity-norm |
|---|---|---|---|---|
| 0 | 3 | 6 | 10 | |
| 1 | 6 | -3 | 0.1 | 4 |
| 2 | 9 | -5.43973 | 1 | 0.357 |
| 3 | 12 | -5.4574 | 1 | 0.0682 |
| 4 | 15 | -5.45833 | 1 | 0.000923 |
| 5 | 18 | -5.45833 | 1 | 1.74e-005 |
| 6 | 21 | -5.45833 | 1 | 5.96e-008 |

Optimization terminated: relative infinity-norm of gradient less than options.TolFun.

```
f=
 -0.24999999941883
  0.33333332344343
```

### 6.2. CALCULO DEL MÍNIMO DE LA FUNCIÓN

$$f(x, y) = 2x^2 + 3y^2 + x - 2y - 5$$

UTILIZANDO EL COMANDO DE MATLAB fminsearch
La instrucción en Matlab para esta función fueron:

```
clc
%Punto inicial
x0 = [-1; -1];
%Opciones
options=optimset('Display','iter');
%Calcula los minimos con el comando fminunc
f = fminsearch ('2*x(1)^2+3*x(2)^2+x(1)-2*x(2)-5',x0,options)
```

Los resultados fueron:

```
Iteration  Func-count     min f(x)         Procedure
    0          1             6
    1          3             6         initial simplex
    2          5          4.50625      expand
    3          7          3.33281      expand
    4          9          0.878203     expand
    5         11          -1.27139     expand
    6         13          -1.33826     reflect
    7         15          -1.68404     contract outside
    8         17          -1.68404     contract inside
    9         19          -2.33154     expand
   10         21          -2.4268      expand
   11         23          -3.94214     expand
   12         24          -3.94214     reflect
   13         26          -5.26467     expand
   14         27          -5.26467     reflect
   15         29          -5.41425     contract outside
   16         31          -5.41425     contract inside
   17         33          -5.42082     contract inside
   18         35          -5.44739     contract inside
   19         37          -5.4549      contract inside
   20         39          -5.4549      contract inside
   21         41          -5.45691     contract outside
   22         43          -5.45783     contract inside
   23         45          -5.45783     contract inside
   24         47          -5.45817     contract outside
   25         49          -5.45829     contract inside
   26         51          -5.45829     contract inside
   27         53          -5.4583      contract outside
   28         55          -5.45832     contract inside
   29         57          -5.45833     contract inside
   30         58          -5.45833     reflect
   31         60          -5.45833     contract inside
   32         62          -5.45833     contract inside
   33         64          -5.45833     contract inside
   34         66          -5.45833     contract inside
   35         68          -5.45833     contract inside
   36         70          -5.45833     contract inside
   37         72          -5.45833     contract inside
   38         74          -5.45833     contract outside
   39         76          -5.45833     contract inside
   40         78          -5.45833     contract outside
   41         80          -5.45833     contract inside
   42         82          -5.45833     contract inside

Optimization terminated: the current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-004 and F(X)
satisfies the convergence criteria using OPTIONS.TolFun of 1.000000e-004
F = -0.249970833324146
     0.333343759527 4
```

## 7. Conclusiones.

El método analizado del descenso más rápido, es un método de optimización local, debido a que está basado únicamente en información proveniente de las derivadas de la función. Esto significa que si la función objetivo es multimodal, es decir, tiene varios mínimos locales y globales, el método encontrará aquel mínimo (global o local) que esté más cercano al punto inicial seleccionado.

En la práctica se utilizan estrategias de descenso que utilizan varios métodos, de la siguiente manera:

1. Se emplea el método de Newton clásico, si no hay descenso (esto lo responde búsqueda lineal).

2. Se emplea el método de Levenberg-Marquardt con un $\lambda$ inicial, $\lambda_K = 0{,}001$, si no hay descenso se incrementa en una razón, $\lambda_K = \beta\lambda_K$ .

3. Si no hay descenso después de varios intentos, se emplea el método del descenso más rápido.

## Referencias

[1] David E. Luenberger. *PROGRAMACIÓN LINEAL Y NO LINEA*. Editorial Addison-Wesley. Iberoamericana. Capítulo 7.

[2] Richard L. Burden y J. Douglas Faires. *ANALISIS NUMERICO*. Editorial Math Learning. Pág. 628-633.

[3] Héctor Manuel Mora Escobar. *3. PROGRAMACION NO LINEA Y DINAMICA*. Editorial Universidad Nacional de Colombia. Pág. 201-2010.