

# Tarea 4: Gradiente descendente

Alumno: Ricardo De León

Realice código en Python que, recibiendo una función  $f$  dada, un valor inicial  $x_0$  y una exactitud (error) dado  $E$ , use el método de gradiente descendente (ascendente) para encontrar un mínimo (máximo) local de  $f$ . Asegúrese que cuenta el número de iteraciones realizadas.

Use su código para encontrar (si existe) los mínimos y máximos locales de cada función con precisión de **4 cifras significativas**, y exactitud de  $10^{-3}$ . En todos los casos indique el(los) valor(es) inicial(es) que utilizó y el número de iteraciones que fueron necesarias para alcanzar la respuesta.

1.  $f(x) = x^4 - 3x^3 + 2$
2.  $f(x) = 5x^6 + 21x^5 - 180x^4 + 115x^3 + 750x^2 - 1260x + 10$
3.  $f(x, y) = x^2 - 24x + y^2 - 10y$
4.  $f(x, y) = xy + \frac{1}{x} + \frac{1}{y}$
5.  $f(x, y) = \sin x + \sin y + \sin(x + y), 0 \leq x \leq 2\pi, 0 \leq y \leq 2\pi$
6.  $f(x, y, z) = x^2 + y^2 + z^2 + 1$
7.  $f(x, y, z) = 3x^2 + 4y^2 + z^2 - 9xyz$
8.  $f(x, y, z) = x^4 + y^4 + z^4 + xyz$

1.-

Entrada para gradiente descendente

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z :
Enter the value for variable X0: 1
Enter the function: x**4-3*x**3+2
```

Mínimo local:

```
Iteration 56: values = {'x': 2.25}, Y: -6.543, error: 0.0009684
```

# Tarea 4: Gradiente descendente

2.-

Entrada para gradiente descendente con un  $\alpha = .000001$

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z :
Enter the value for variable X0: .1
Enter the function: 5*x**6+21*x**5-180*x**4+115*x**3+750*x**2-1260*x+10
```

Descendente:

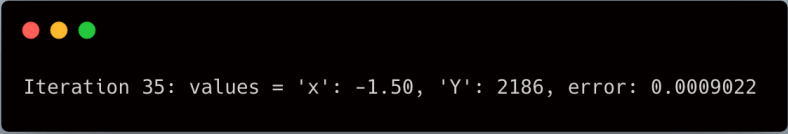
```
Iteration 186: values = {'x': -7.00}, Y: -1.907E+5, error: 0.0009256
```

Ascendente 1: Entrada con  $\alpha = .0001$

```
Enter 1 for ascent gradient, 2 for descent gradient: 1
Option will be taken as ascent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x
Enter the value for variable X0: .5
Enter the function: 5*x**6+21*x**5-180*x**4+115*x**3+750*x**2-1260*x+10
```

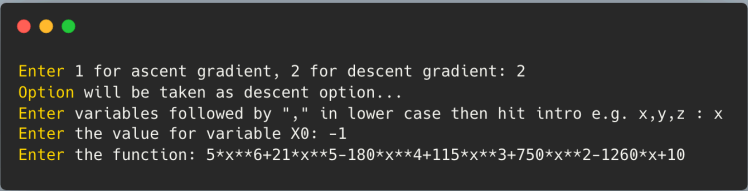
Ascendente 1: Salida

# Tarea 4: Gradiente descendente



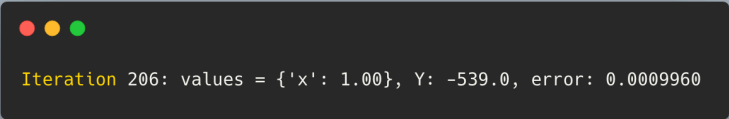
```
Iteration 35: values = 'x': -1.50, 'Y': 2186, error: 0.0009022
```

Ascendente 2: Entrada con  $\alpha = .0001$



```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x
Enter the value for variable X0: -1
Enter the function: 5*x**6+21*x**5-180*x**4+115*x**3+750*x**2-1260*x+10
```

Ascendente Salida:



```
Iteration 206: values = {'x': 1.00}, Y: -539.0, error: 0.0009960
```

3.-

Entrada para gradiente descendente

# Tarea 4: Gradiente descendente

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z :
Enter the value for variable X0: 1
Enter the value for variable Y0: 1
Enter the function: x**2 - 24*x + y**2 - 10*y
```

Descendente:

```
Iteration 498: values = {'x': 12.0, 'y': 5.00}, error: 0.0009999
```

Ascendente:

No existe

# Tarea 4: Gradiente descendente

4.-

Entrada y salida para gradiente descendente con  $\alpha = .1$

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y
Enter the value for variable X0: 2
Enter the value for variable Y0: 2
Enter the function: x*y + 1/x + 1/y
Iteration 27: values = {'x': 1.00, 'y': 1.00}, error: 0.0008591
```

Para ascendente no encontré punto

5.-

Entrada y salida para gradiente descendente

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y
Enter the value for variable X0: 1
Enter the value for variable Y0: 1
Enter the function: sin(x) + sin(y) + sin(x+y)
Iteration 436: values = {'x': -1.05, 'y': -1.05}, error: 0.0009790
```

Entrada y salida para gradiente ascendente

```
Enter 1 for ascent gradient, 2 for descent gradient: 1
Option will be taken as ascent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y
Enter the value for variable X0: -1
Enter the value for variable Y0: -1
Enter the function: sin(x) + sin(y) + sin(x+y)
Iteration 436: values = {'x': 1.05, 'y': 1.05}, Y: 2.598, error: 0.0009790
```

# Tarea 4: Gradiente descendente

6.-

Entrada y salida para gradiente descendente

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: 1
Enter the value for variable Y0: 1
Enter the value for variable Z0: 1
Enter the function: x**2 + y**2 + z**2 + 1
Iteration 404: values = {'x': 0.000280, 'y': 0.000280, 'z': 0.000280}, Y: 1.000, error: 0.00098840
```

Ascendente:

No encontré

7.-

Entrada y salida para gradiente descendente con alpha = .001

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: -.01
Enter the value for variable Y0: -.01
Enter the value for variable Z0: -.01
Enter the function: 3*x**2 + 4*y**2 + z**2 - 9*x*y*z
```

```
Iteration 1493: values = {'x': -1.22e-6, 'y': -5.32e-8, 'z': -0.000499}, Y: 2.496E-7, error: 0.0009993
```

Ascendente: No encontré

8.

Entrada y salida para gradiente descendente con alpha = .01

# Tarea 4: Gradiente descendente



```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: 1
Enter the value for variable Y0: 1
Enter the value for variable Z0: 1
Enter the function: x**4 + y**4 + z**4 + x*y*z
```



```
Iteration 3344: values = {'x': 0.0230, 'y': 0.0230, 'z': 0.0230}, Y: 0.00001300, error: 0.001000
```



```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: 6
Enter the value for variable Y0: 6
Enter the value for variable Z0: 6
Enter the function: x**4 + y**4 + z**4 + x*y*z
```



```
Iteration 1482: values = {'x': -0.252, 'y': -0.252, 'z': -0.252}, Y: -0.003904, error: 0.0009983
```



```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: -1
Enter the value for variable Y0: 1
Enter the value for variable Z0: 1
Enter the function: x**4 + y**4 + z**4 + x*y*z
```

# Tarea 4: Gradiente descendente

```
Iteration 1469: values = {'x': -0.252, 'y': 0.252, 'z': 0.252}, Y: -0.003904, error: 0.0009996
```

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: 1
Enter the value for variable Y0: -1
Enter the value for variable Z0: 1
Enter the function: x**4 + y**4 + z**4 + x*y*z
```

```
Iteration 1469: values = {'x': 0.252, 'y': -0.252, 'z': 0.252}, Y: -0.003904, error: 0.0009996
```

```
Enter 1 for ascent gradient, 2 for descent gradient: 2
Option will be taken as descent option...
Enter variables followed by "," in lower case then hit intro e.g. x,y,z : x,y,z
Enter the value for variable X0: 1
Enter the value for variable Y0: 1
Enter the value for variable Z0: -1
Enter the function: x**4 + y**4 + z**4 + x*y*z
```

```
Iteration 1469: values = {'x': 0.252, 'y': 0.252, 'z': -0.252}, Y: -0.003904, error: 0.0009996
```

Ascendente: No encontrado



# Tarea 4: Gradiente descendente

Código:

```
import sympy as sp

# Define the learning rate and number of iterations
alpha = .01
limit = 0.001
num_iterations = 10000
func_variables = []
variable_values = []

# Define the function to optimize
def get_type_asc_desc():
    global alpha
    option = int(input('Enter 1 for ascent gradient, 2 for descent gradient: '))
    if(option == 1):
        print('Option will be taken as ascent option...')
        alpha = alpha * -1
    else:
        print('Option will be taken as descent option...')

def init_func_variables():
    variable = input('Enter variables followed by "," in lower case then hit intro e.g. x,y,z : ')
    global func_variables
    func_variables = variable.split(',')
    for i in func_variables:
        i = sp.symbols(i)
    return func_variables

def create_dictionary_of_variable_and_values():
    global variable_values
    for i in func_variables:
        value = float(input(f'Enter the value for variable {i.capitalize()}0: '))
        variable_values.append(value)
    return list(zip(func_variables, variable_values))

def get_func():
    f = input('Enter the function: ')
    return f

def create_gradient(function):
    grad_lst = []
    for i in func_variables:
        grad_lst.append(sp.diff(function, i))
    return sp.Matrix(grad_lst)

get_type_asc_desc()
init_func_variables()
dict_of_variable_and_values = create_dictionary_of_variable_and_values()
f = get_func()
grad_f = create_gradient(f)

# Perform the gradient descent optimization
curr_variable_values = sp.Matrix(variable_values)
for i in range(num_iterations):

    # Evaluate the gradient at the current point
    grad = sp.Matrix([g.subs(dict_of_variable_and_values) for g in grad_f])

    # Update the current point using the gradient and learning rate
    curr_variable_values -= alpha * grad

    # Update variable values
    dict_of_variable_and_values = list(zip(func_variables, curr_variable_values))

    # Evaluate norm to break the loop
    v = sp.Matrix(grad)
    error = sp.trigsimp(v.norm())
    if(error < limit):
        break

    # Evaluate func in curr point
    derivate = sp.simplify(f).evalf(subs=dict(zip(func_variables, curr_variable_values.evalf(3))))

    print(f'Iteration {i+1}: point={dict(zip(func_variables, curr_variable_values.evalf(3)))}, val in f(): {derivate:.4}, error: {error:.4}')

print(f'Iteration {i}: values = {dict(zip(func_variables, curr_variable_values.evalf(3)))}, Y: {derivate:.4}, error: {error:.4}')
```