EJE2CUASI-NEWTON

April 3, 2023

1 Método Cuasi-Newton

[2]:
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from scipy.optimize import minimize
```

[3]:
```
#Primero definimos nuestra funcion f.
```

[4]:
```
def f(xy):
    x, y = xy
    return np.exp(x**2)*y**2 + 2*np.exp(y**2)*x**2 + 4*x*y + 2*x**2 + 4*x - 2*y
```

[5]:
```
#Calculamos las derivadas de nuestra función.
```

[6]:
```
def df(xy):
    x, y = xy
    df_dx = 2*x*np.exp(x**2)*y**2 + 4*np.exp(y**2)*x + 4*y + 4*x + 4
    df_dy = 2*y*np.exp(x**2)*x**2 + 4*np.exp(y**2)*y + 4*x - 2
    return np.array([df_dx, df_dy])
```

[7]:
```
#Definimos nuestro punto semilla.
```

[8]:
```
x0 = np.array([25, -30])
```

[9]:
```
#Aplicamos nuestro método de CUASI-NEWTON.
```

[10]:
```
res = minimize(f, x0, method='BFGS', jac=df, tol=0.00001)
print(res)
```

```
      fun: nan
 hess_inv: array([[1, 0],
       [0, 1]])
      jac: array([nan, nan])
  message: 'Desired error not necessarily achieved due to precision loss.'
     nfev: 112
      nit: 1
     njev: 112
   status: 2
```

```
  success: False
        x: array([nan, nan])
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_26208\176315060.py:3: RuntimeWarning:
overflow encountered in exp
  return np.exp(x**2)*y**2 + 2*np.exp(y**2)*x**2 + 4*x*y + 2*x**2 + 4*x - 2*y
C:\Users\ASUS\AppData\Local\Temp\ipykernel_26208\1256377086.py:4:
RuntimeWarning: overflow encountered in exp
  df_dy = 2*y*np.exp(x**2)*x**2 + 4*np.exp(y**2)*y + 4*x - 2
C:\Users\ASUS\anaconda3\lib\site-packages\scipy\optimize\linesearch.py:153:
RuntimeWarning: invalid value encountered in double_scalars
  alpha1 = min(1.0, 1.01*2*(phi0 - old_phi0)/derphi0)
C:\Users\ASUS\anaconda3\lib\site-packages\scipy\optimize\linesearch.py:403:
RuntimeWarning: invalid value encountered in double_scalars
  alpha1 = min(1.0, 1.01*2*(phi0 - old_phi0)/derphi0)
```

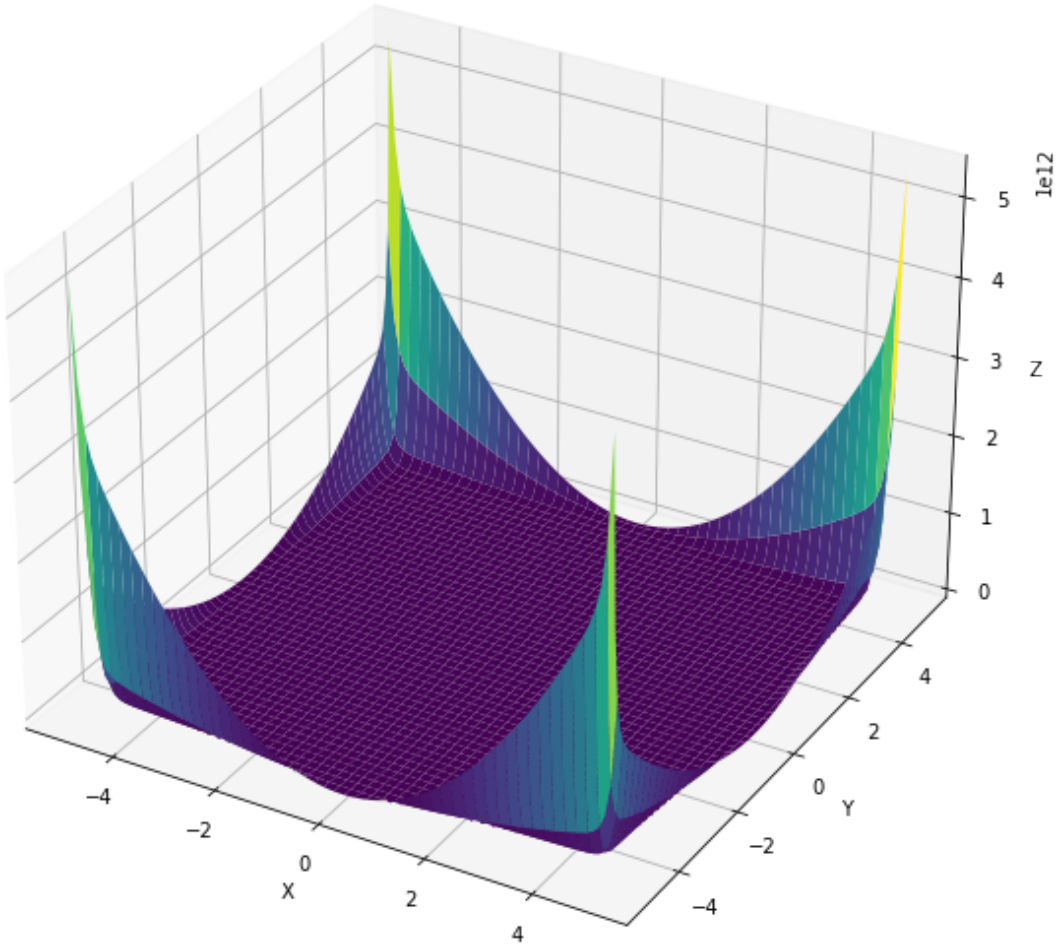
[11]:
```
#Ahora para graficar.
```

[12]:
```
x_vals = np.linspace(-5, 5, 100)
y_vals = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x_vals, y_vals)
Z = f([X, Y])

fig = plt.figure(figsize=(10, 10))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z, cmap='viridis')
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')

ax.scatter(res.x[0], res.x[1], f(res.x), c='r', s=100, marker='o')

plt.show()
```


1e12
5
4
3
Z
2
1
0
4
2
0
Y
−2
−4
−4
−2
0
X
2
4

[ ]: