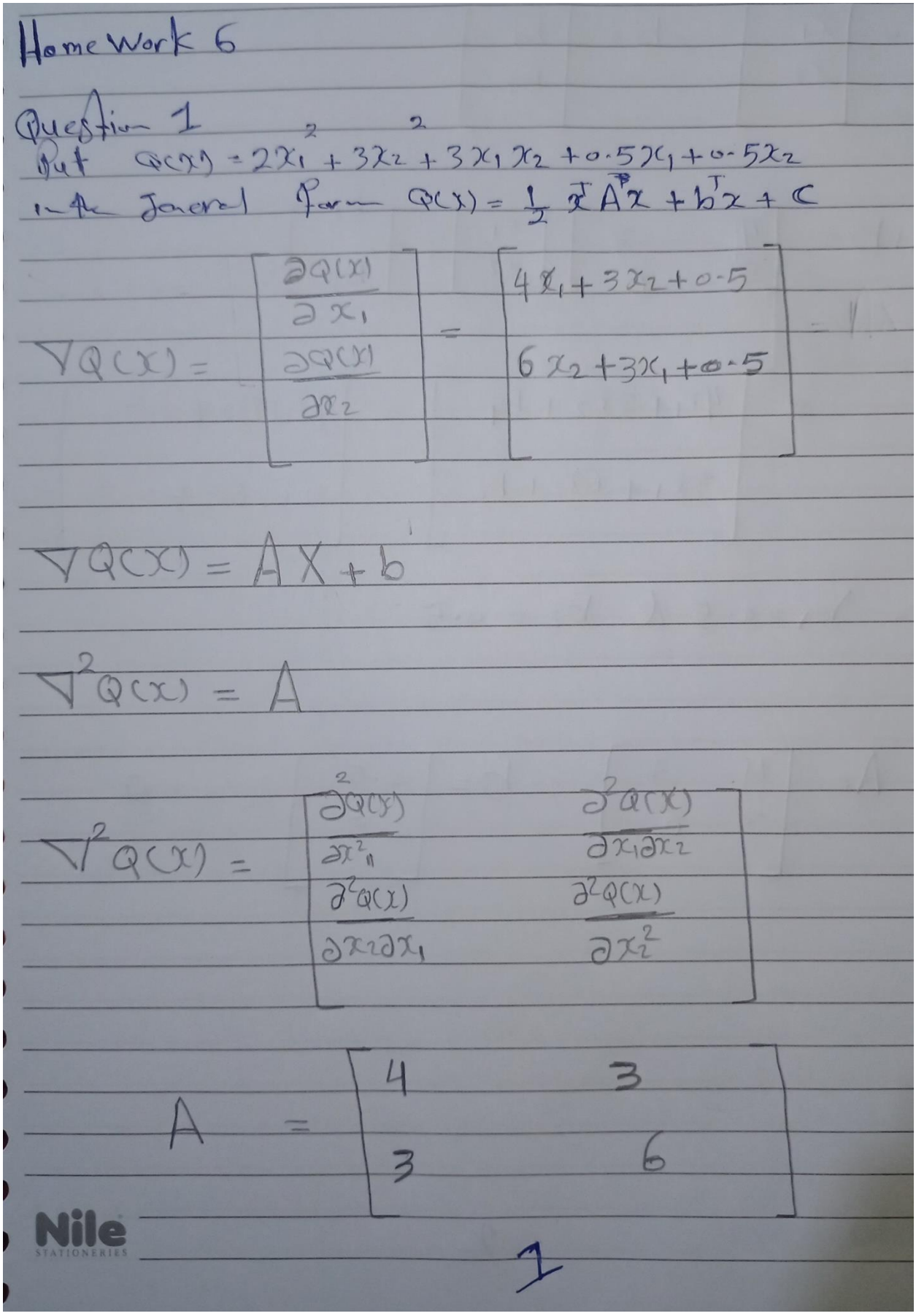
Homework 6

Name: Mohamed Khaled Ahmed Atyaa

Group No.: 3 Alex

Question 1



Homework 6

Question 1

Put $Q(x) = 2x_1^2 + 3x_2^2 + 3x_1x_2 + 0.5x_1 + 0.5x_2$

in the general form $Q(x) = \frac{1}{2}\vec{x}^T A \vec{x} + b^T x + C$

$$\nabla Q(x) = \begin{bmatrix} \frac{\partial Q(x)}{\partial x_1} \\ \frac{\partial Q(x)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 4x_1 + 3x_2 + 0.5 \\ 6x_2 + 3x_1 + 0.5 \end{bmatrix} \quad \text{--- (1)}$$

$$\nabla Q(x) = AX + b$$

$$\nabla^2 Q(x) = A$$

$$\nabla^2 Q(x) = \begin{bmatrix} \frac{\partial^2 Q(x)}{\partial x_1^2} & \frac{\partial^2 Q(x)}{\partial x_1 \partial x_2} \\ \frac{\partial^2 Q(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 Q(x)}{\partial x_2^2} \end{bmatrix}$$

$$A = \begin{bmatrix} 4 & 3 \\ 3 & 6 \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

$$Ax+b = \begin{bmatrix} 4 & 3 \\ 3 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

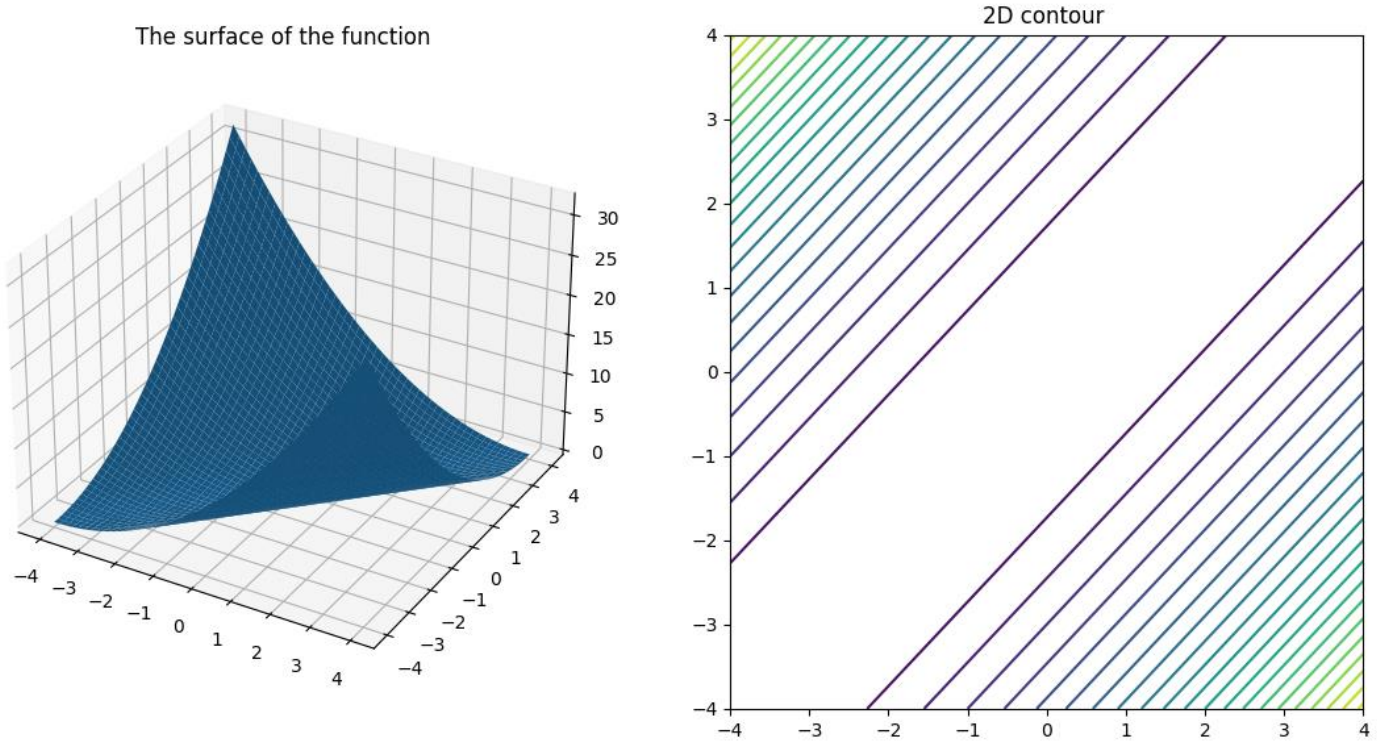
$$= \begin{bmatrix} 4x_1 + 3x_2 + b_1 \\ 3x_1 + 6x_2 + b_2 \end{bmatrix} = \begin{bmatrix} 4x_1 + 3x_2 + 0.5 \\ 3x_1 + 6x_2 + 0.5 \end{bmatrix}$$

$\therefore b_1 = 0.5$ & $b_2 = 0.5$

$$\therefore A = \begin{bmatrix} 4 & 3 \\ 3 & 6 \end{bmatrix}, \quad b = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \text{ & } S = 0$$

Question 2

```
import numpy as np
import matplotlib.pyplot as plt
def Q(x):
    x1 = x[0]
    x2 = x[1]
    return 0.5 * x1**2 - x1*x2 + 0.5*x2**2
LB = -4
UB = 4
x = np.linspace(LB, UB, 50)
y = np.linspace(LB, UB, 50)
X, Y = np.meshgrid(x, y)
Z = Q([X, Y])
fig = plt.figure(figsize=(10,8))
ax1 = fig.add_subplot(121, projection='3d')
ax1.plot_surface(X, Y, Z)
ax1.set_title("The surface of the function ")
ax2 = fig.add_subplot(122)
ax2.contour(X, Y, Z, levels=30)
ax2.set_title("2D contour ")
plt.show()
```



Question 3

Powell's method

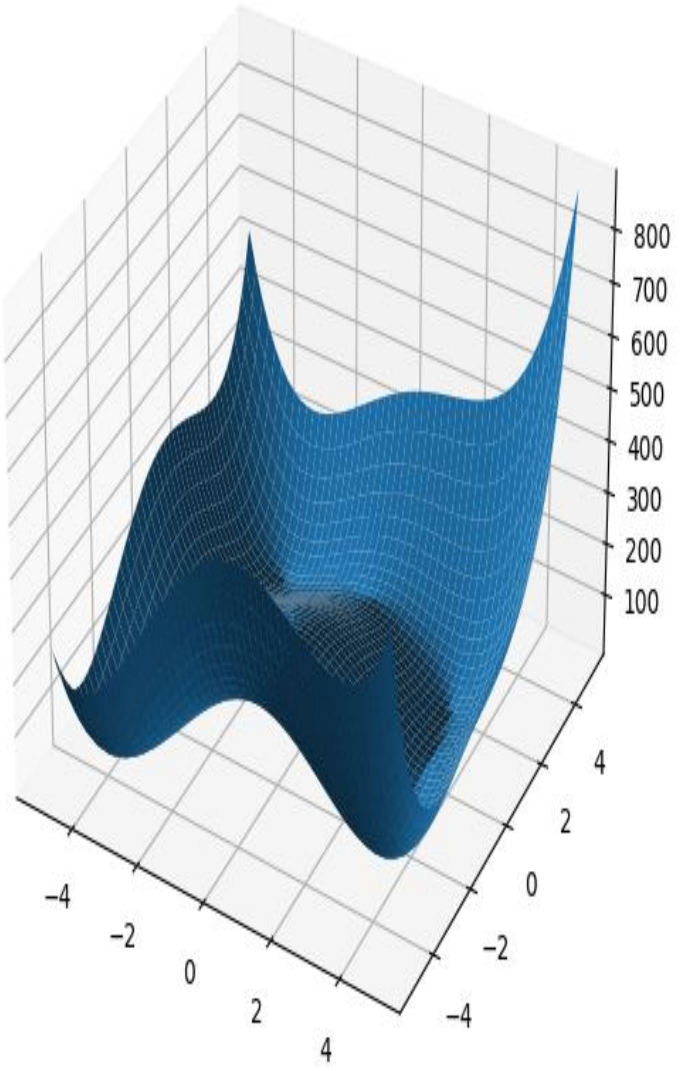
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import minimize

def F(x):
    return ((x[0]**2+x[1]-11)**2)+((x[0]+x[1]**2-7)**2)

x0 = np.array([4, -4])
res = minimize(F, x0, method='Powell', options={'disp':True,
'return_all':True})
xs = res.allvecs
LB = -5
UB = 5
x = np.linspace(LB, UB, 50)
y = np.linspace(LB, UB, 50)
X, Y = np.meshgrid(x, y)
Z = F([X, Y])
fig = plt.figure(figsize=(10,8))
ax1 = fig.add_subplot(122)
ax1.contour(X, Y, Z, levels=20)
ax1.scatter([x[0] for x in xs], [x[1] for x in xs],
c=list(range(len(res.allvecs))), marker='x')
ax1.set_title("Powell's method")
ax2 = fig.add_subplot(121, projection='3d')
ax2.plot_surface(X, Y, Z)
plt.show()
print(res.x)
```

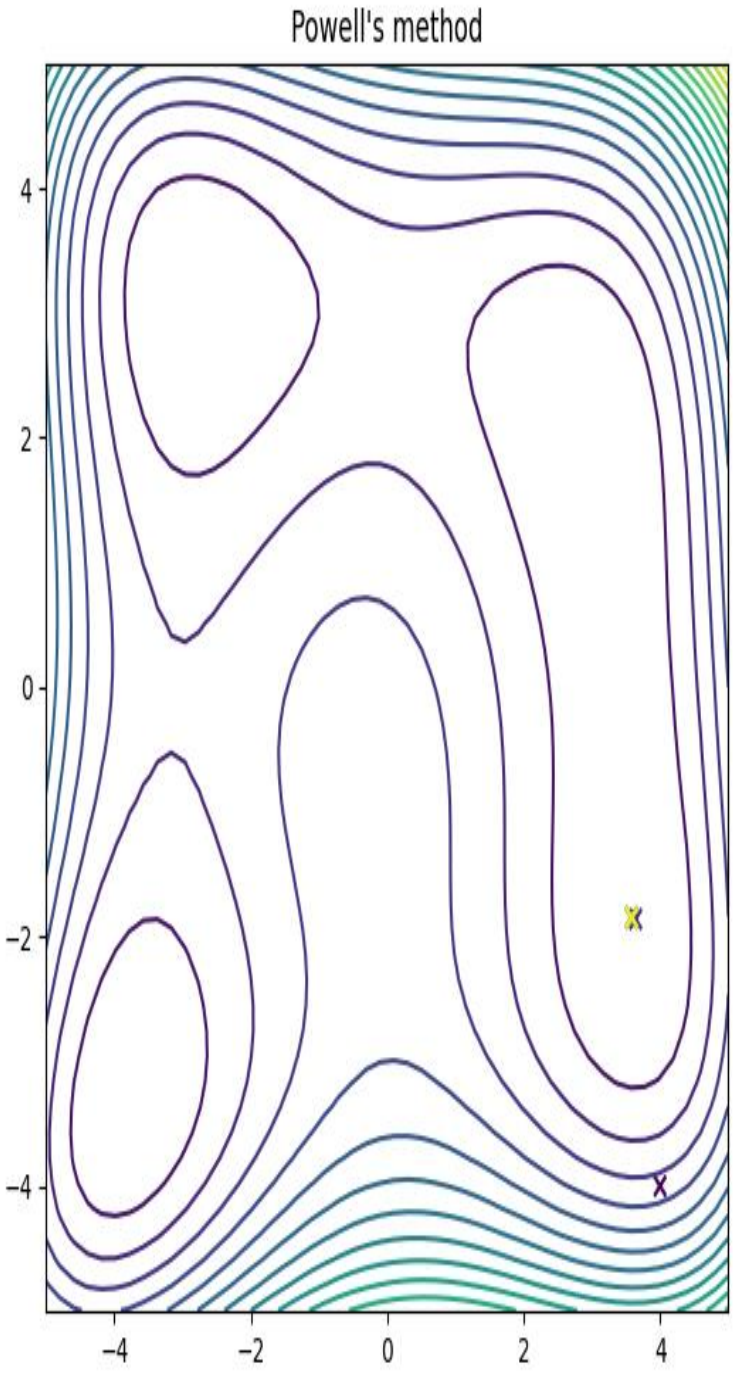
Current function value: 0.000000 #this is the min value of the function

Iterations: 6

Function evaluations: 151

the function minimize at point[3.58442834 -1.84812653]


800
700
600
500
400
300
200
100
−4
−2
0
2
4
−4
−2
0
2
4


Powell's method
4
2
0
−2
−4
−4
−2
0
2
4

Conjugate Gradient method

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import minimize

def F(x):
    return ((x[0]**2+x[1]-11)**2)+((x[0]+x[1]**2-7)**2)
def grad_F(x):
    Fx=(4*x[0]*(x[0]**2+x[1]-11)+2*(x[0]+x[1]**2-7))
    Fy=(2*(x[0]**2+x[1]-11)+2*x[1]*(x[0]+x[1]**2-7))
    return np.asarray((Fx, Fy))

x0 = np.array([4, -4])
res = minimize(F, x0, method='CG',jac=grad_F, options={'disp':True,
'return_all':True})
xs = res.allvecs
LB = -5
UB = 5
x = np.linspace(LB, UB, 50)
y = np.linspace(LB, UB, 50)
X, Y = np.meshgrid(x, y)
Z = F([X, Y])
fig = plt.figure(figsize=(10,8))
ax1 = fig.add_subplot(122)
ax1.contour(X, Y, Z, levels=20)
ax1.scatter([x[0] for x in xs], [x[1] for x in xs],
c=list(range(len(res.allvecs))), marker='x')

ax2 = fig.add_subplot(121, projection='3d')
ax2.plot_surface(X, Y, Z)
ax1.set_title ("Conjugate Gradient method")
plt.show()
print(res.x)
```

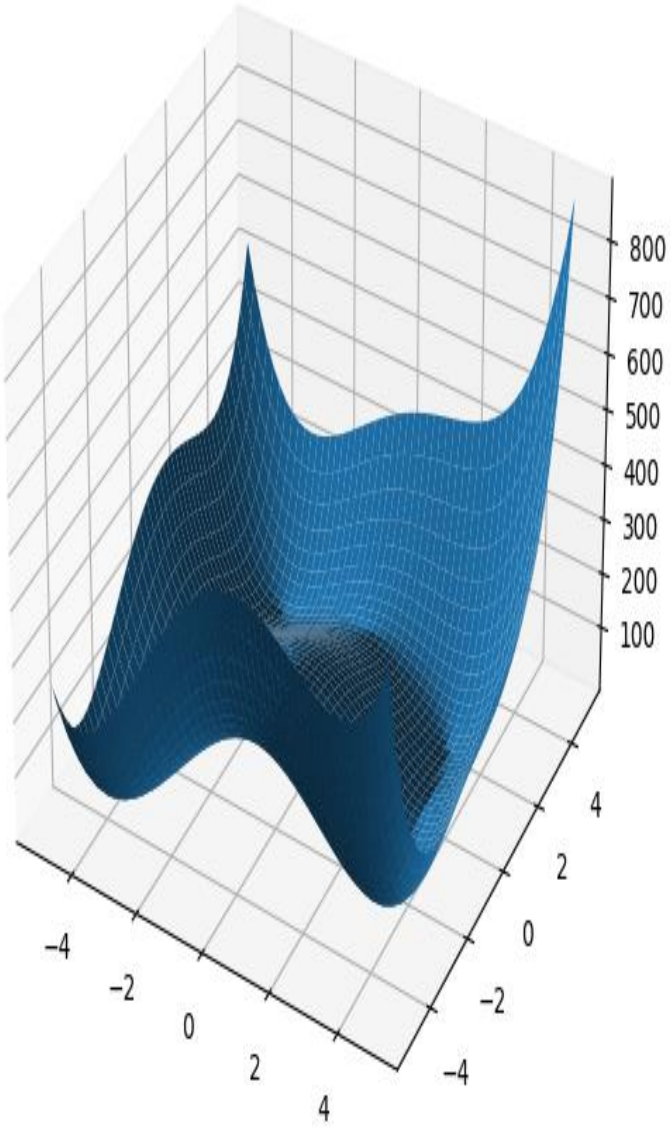
Current function value: 0.000000 #this is the min value of the function

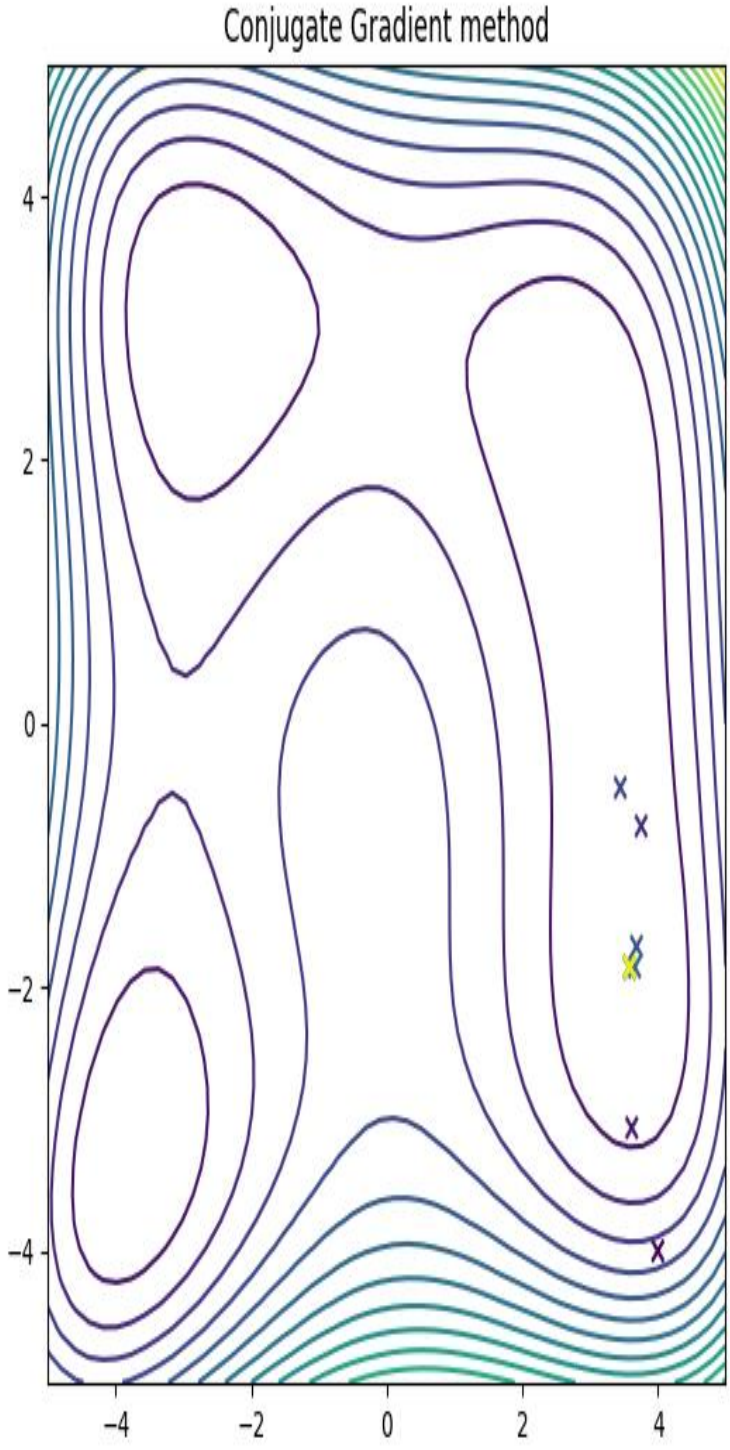
Iterations: 15

Function evaluations: 56

Gradient evaluations: 56

The function minimize at point[3.58442834 -1.84812653]


800
700
600
500
400
300
200
100
−4
−2
0
2
4
4
2
0
−2
−4


Conjugate Gradient method
4
2
0
−2
−4
−4
−2
0
2
4

BFGS method

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import minimize


def F(x):
    return ((x[0]**2+x[1]-11)**2)+((x[0]+x[1]**2-7)**2)
def grad_F(x):
    Fx=(4*x[0]*(x[0]**2+x[1]-11)+2*(x[0]+x[1]**2-7))
    Fy=(2*(x[0]**2+x[1]-11)+2*x[1]*(x[0]+x[1]**2-7))
    return np.asarray((Fx, Fy))

x0 = np.array([4, -4])
res = minimize(F, x0, method='BFGS',jac=grad_F, options={'disp':True,
'return_all':True})
xs = res.allvecs
LB = -5
UB = 5
x = np.linspace(LB, UB, 50)
y = np.linspace(LB, UB, 50)
X, Y = np.meshgrid(x, y)
Z = F([X, Y])
fig = plt.figure(figsize=(10,8))
ax1 = fig.add_subplot(122)
ax1.contour(X, Y, Z, levels=20)
ax1.scatter([x[0] for x in xs], [x[1] for x in xs],
c=list(range(len(res.allvecs))), marker='x')

ax2 = fig.add_subplot(121, projection='3d')
ax2.plot_surface(X, Y, Z)
ax1.set_title ("BFGS method")
plt.show()
print(res.x)
```

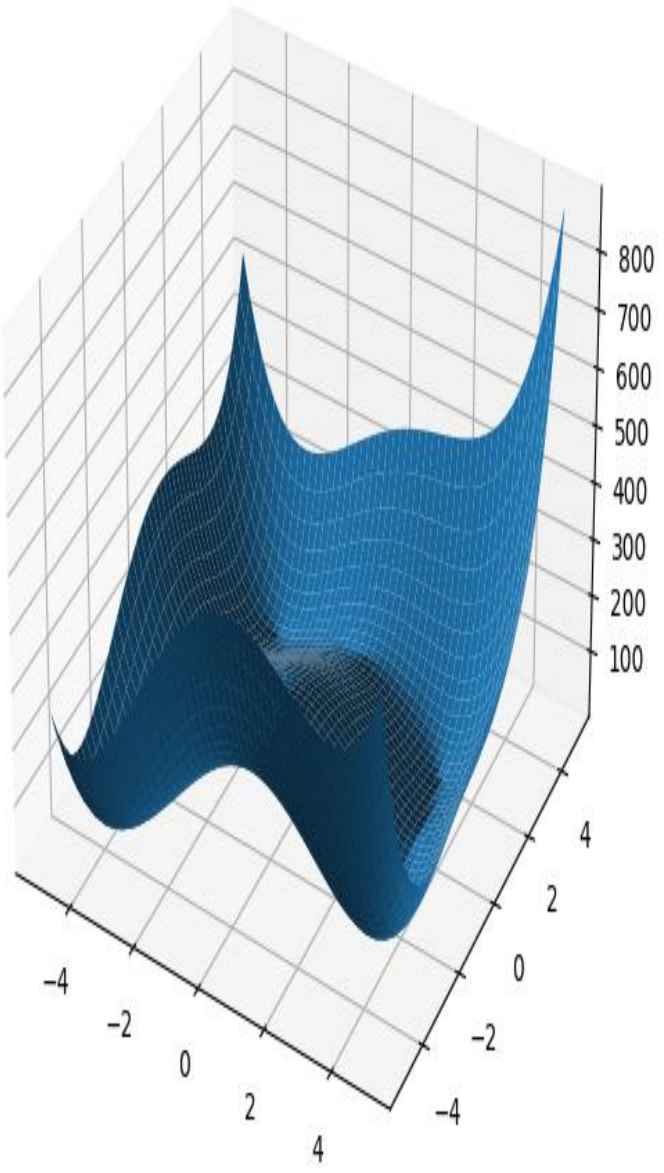
Current function value: 0.000000 #this is the min value of the function

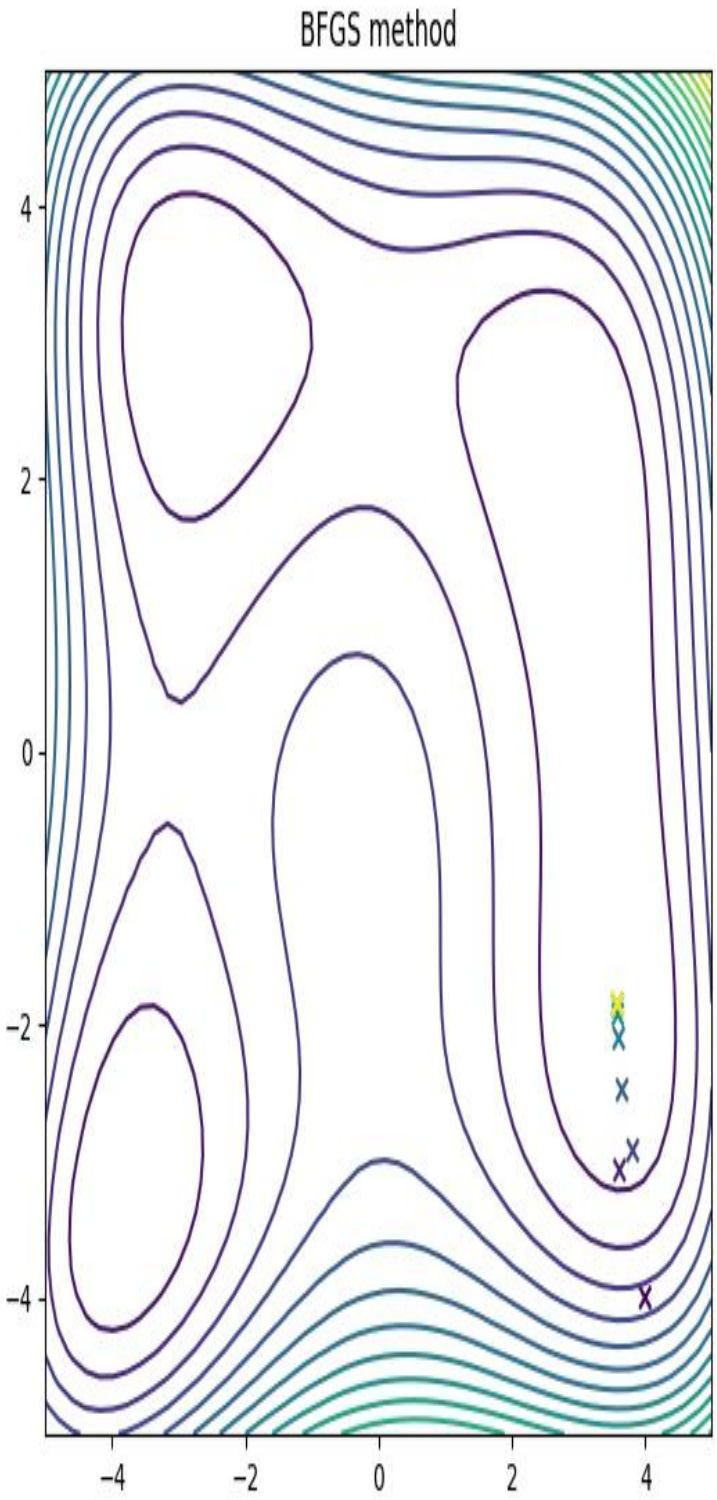
Iterations: 10

Function evaluations: 13

Gradient evaluations: 13

The function minimize at point[3.58442834 -1.84812653]


800
700
600
500
400
300
200
100
−4
−2
0
2
4
4
2
0
−2
−4


BFGS method
4
2
0
−2
−4
−4
−2
0
2
4