Constrained Nonlinear Optimisation

Optimisation
Project Report

Serkan Burak Örs[*]

[*]orss19@itu.edu.tr

Contents

1 Introduction

In this project, given constrained optimisation problem is solved with using Lagrange multipliers and the solution is verified graphically.

$$\text{Maximise } F(x_1, x_2) = 2x_1 + x_2 - x_1^2 - x_2^2 - 2 \tag{1.1}$$

$$\text{Subject to: } 2x_1 + x_2 \geq 4 \tag{1.2}$$

$$x_1 + 2x_2 \geq 4 \tag{1.3}$$

2 Solution

Firstly, we can write the given problem as

$$\text{Minimise } F(x_1, x_2) = -(2x_1 + x_2 - x_1^2 - x_2^2 - 2) \tag{2.1}$$

$$\text{Subject to: } g_1(x_1, x_2) = 4 - 2x_1 - x_2 \leq 0 \tag{2.2}$$

$$g_2(x_1, x_2) = 4 - x_1 - 2x_2 \leq 0 \tag{2.3}$$

Then, we can define the Lagrangian as,

$$L = F + \lambda^T g_i = -(2x_1 + x_2 - x_1^2 - x_2^2 - 2) + \lambda_1(4 - 2x_1 - x_2) + \lambda_2(4 - x_1 - 2x_2) \tag{2.4}$$

Necessary conditions:

$$\frac{\partial L}{\partial x_1} = -2 + 2x_1 - 2\lambda_1 - \lambda_2 = 0 \tag{2.5}$$

$$\frac{\partial L}{\partial x_2} = -1 + 2x_2 - \lambda_1 - 2\lambda_2 = 0 \tag{2.6}$$

$$\lambda_1 \frac{\partial L}{\partial \lambda_1} = \lambda_1(4 - 2x_1 - x_2) = 0 \tag{2.7}$$

$$\lambda_2 \frac{\partial L}{\partial \lambda_2} = \lambda_2(4 - x_1 - 2x_2) = 0 \tag{2.8}$$

Now we need to assign active of inactive states to Lagrange multipliers.

- Assume all Lagrange multipliers are inactive: $\lambda_1 = \lambda_2 = 0$

$$\frac{\partial L}{\partial x_1} = -2 + 2x_1 = 0 \tag{2.9}$$

$$\frac{\partial L}{\partial x_2} = -1 + 2x_2 = 0 \tag{2.10}$$

  gives solution of $(x_1, x_2) = (1, 0.5)$ which does not satisfy the both of inequality constraints.

- Assume $\lambda_1 \geq 0$ (active) and $\lambda_2 = 0$ (inactive)

$$\frac{\partial L}{\partial x_1} = -2 + 2x_1 - 2\lambda_1 = 0 \tag{2.11}$$

$$\frac{\partial L}{\partial x_2} = -1 + 2x_2 - \lambda_1 = 0 \tag{2.12}$$

$$\lambda_1 \frac{\partial L}{\partial \lambda_1} = \lambda_1 (4 - 2x_1 - x_2) = 0 \tag{2.13}$$

  gives solution $(x_1, x_2) = (1.6, 0.8)$ and $\lambda_1 = 0.6$ which does not satisfy the second constraint.

- Assume $\lambda_2 \geq 0$ (active) and $\lambda_1 = 0$ (inactive)

$$\frac{\partial L}{\partial x_1} = -2 + 2x_1 - \lambda_2 = 0 \tag{2.14}$$

$$\frac{\partial L}{\partial x_2} = -1 + 2x_2 - 2\lambda_2 = 0 \tag{2.15}$$

$$\lambda_2 \frac{\partial L}{\partial \lambda_2} = \lambda_2 (4 - x_1 - 2x_2) = 0 \tag{2.16}$$

  gives solution $(x_1, x_2) = (1.4, 1.3)$ and $\lambda_2 = 0.8$ which satisfy all the constraints.

Hence the maximum value and the maximum value of the function is

$$F^*(x_1^*, x_2^*) = F^*(1.4, 1.3) = -1.55 \tag{2.17}$$

Also, we can visualize the objective function, constraints and maximum point at the Fig.[1]

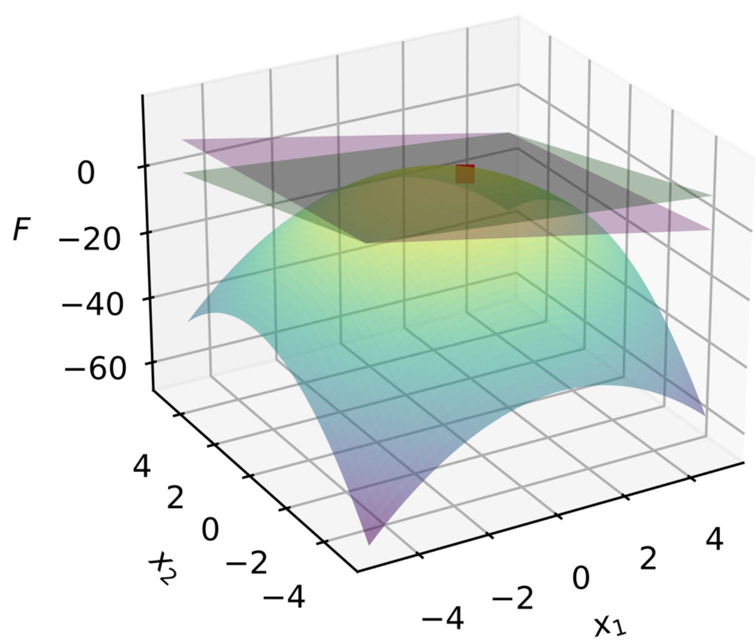

Figure 1: Graphical representation of problem and the solution of problem

3 Appendix A: Python Code

```
import numpy as np
from scipy.optimize import minimize
import matplotlib.pyplot as plt
def objective(x):
return -(2*x[0] + x[1] - x[0]**2 - x[1]**2 - 2)
def constraint1(x):
return  2*x[0] + x[1] - 4
def constraint2(x):
return  x[0] + 2*x[1] - 4
# Define the range of the variables
xx = np.arange(-5, 5, 0.1)
# Calculate the objective function values
FF = np.zeros((xx.size, xx.size))
for ii in range(xx.size):
for jj in range(xx.size):
FF[ii, jj] = -(2*xx[ii] + xx[jj] - xx[ii]**2 - xx[jj]**2 - 2)
# Define the constraints as surfaces
X, Y = np.meshgrid(xx, xx)
g1 = 4 - 2*X - Y  # first constraint
g2 = 4 - X - 2*Y  # second constraint
# Plot the objective function and the constraint surfaces
fig = plt.figure(1,dpi=1200)
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, -FF, cmap='viridis', alpha=0.5,label='surface')
ax.plot_surface(X, Y, g1, color='m', alpha=0.3,label='constraint')
ax.plot_surface(X, Y, g2, color='g', alpha=0.3,label='constraint')
# Set labels and view angle
ax.set_xlabel('$x_1$')
ax.set_ylabel('$x_2$')
ax.set_zlabel('$F$')
ax.view_init(elev=25, azim=-120)
# Solve the optimization problem
cons = [{'type': 'ineq', 'fun': constraint1}, {'type': 'ineq', 'fun': ...
    constraint2}]
res = minimize(objective, [1,1], method='SLSQP', constraints=cons)
# Plot the optimal solution point
ax.plot([res.x[0]], [res.x[1]], [-res.fun], 'rs', markersize=5, ...
    markerfacecolor='r')
# Show the plot and print the output
```

```
plt.show()
print("Optimal value of the objective function is {:.4f}".format(-res.fun))
print("Optimal point is ({:.4f}, {:.4f})".format(res.x[0], res.x[1]))
```

4 References

[1] Kirk D.E. *Optimal Control Theory*. Dover Publications. 1998.

[2] Chong, Edwin K. P., and Stanislaw H. Zak. *Introduction to Optimization*. John Wiley & Sons. 2013.