Ramanujan College(University of Delhi)
Kalkaji Delhi-110019

Numerical Optimization
Practical File

Submitted To:
Mr.Akash Singh

Submitted By:
AMAN,
B.Sc(H) Computer science
20221406

Q1-WAP for finding optimal solution using Line Search method.

File Edit Format Run Options Window Help

```python
'''
Name-Aman
Examination roll number -22020570024
Class Roll number -20221406
Course- B.Sc(H) computer science
'''

import pulp as p
import matplotlib.pyplot as plt
import numpy as np

# Define the Linear Programming Problem
lp_problem = p.LpProblem("Maximize_Profit", p.LpMaximize)

# Define decision variables
x = p.LpVariable("x", lowBound=0)  # x >= 0
y = p.LpVariable("y", lowBound=0)  # y >= 0

# Objective function
lp_problem += 3 * x + 2 * y, "Z"

# Constraints
lp_problem += x + y <= 4
lp_problem += 2 * x + y <= 7

# Solve the LP problem
lp_problem.solve()
```

Output:

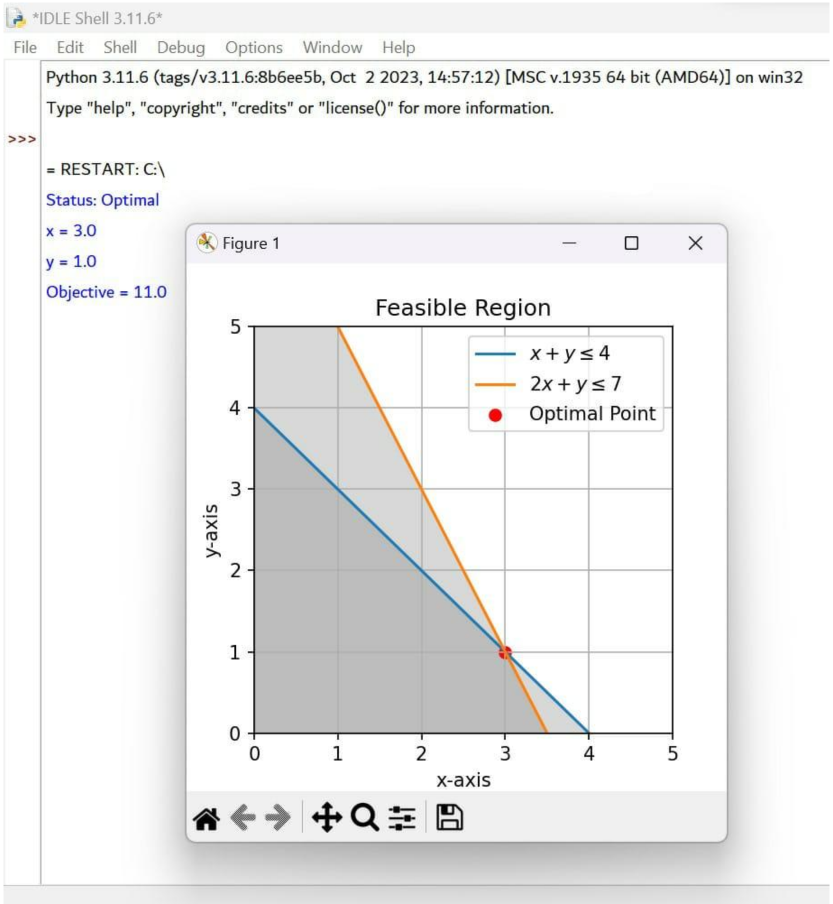

IDLE Shell 3.11.6
File Edit Shell Debug Options Window Help
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct 2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\
Optimal Solution: [0.25]
Optimal Value: [0.875]
>>>

Q2-WAP to solve a LPP graphically.

File Edit Format Run Options Window Help

```python
'''
Name-Aman
Examination roll number -22020570024
Class Roll number -20221406
Course- B.Sc(H) computer science
'''

import numpy as np
from scipy.optimize import minimize_scalar

# Define your objective function here
def objective_function(x):
    return 2 * x**2 - x + 1  # Example function: f(x) = 2 * x**2 - x + 1

def line_search_method(initial_guess, search_direction):

    # Define the objective function to be minimized along the search direction
    def objective_along_direction(alpha):
        x = initial_guess + alpha * search_direction
        return objective_function(x)

    # Perform a line search to find the optimal step size (alpha)
    result = minimize_scalar(objective_along_direction)
    optimal_alpha = result.x

    # Calculate the optimal solution
    optimal_solution = initial_guess + optimal_alpha * search_direction
```

Output-


IDLE Shell 3.11.6
File Edit Shell Debug Options Window Help
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct 2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\
Status: Optimal
x = 3.0
y = 1.0
Objective = 11.0
Figure 1
Feasible Region
x + y ≤ 4
2x + y ≤ 7
Optimal Point
y-axis
x-axis

Q3-WAP to compute the gradient and Hessian of the function
$f(x)$ = 100(x2 − x1)^2 + (1 − x1)^2

File Edit Format Run Options Window Help

```
Name-Aman
Examination roll number -22020570024
Class Roll number -20221406
Course- B.Sc(H) computer science
'''
import numpy as np
from scipy.optimize import approx_fprime, check_grad

# Define the function
def f(x):
  return 100 * (x[1] - x[0]**2)**2 + (1 - x[0])**2

# Define the point at which to compute the gradient and Hessian
x0 = np.array([0, 0])

# Compute the gradient numerically
gradient = approx_fprime(x0, f, epsilon=1e-8)

# Compute the Hessian numerically
hessian = np.zeros((len(x0), len(x0)))
for i in range(len(x0)):
  hessian[i, :] = approx_fprime(x0, lambda x: approx_fprime(x, f, epsilon=1e-8)[i], epsilon=1e-8)

# Print the results
print("Gradient:")
print(gradient)
```

Output-

```
IDLE Shell 3.11.6
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct  2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\.
Gradient:
[-2.00000000e+00  9.99200722e-07]

Hessian Matrix:
[[  4.4408921    0.        ]
 [  0.        199.84014443]]
>>>
```

Q4-WAP to find Global Optimal Solution of a function $f(x) = -10Cos(\pi x - 2.2) + (x + 1.5)x$ algebraically.

File Edit Format Run Options Window Help

```python
'''
Name-Aman
Examination roll number -22020570024
Class Roll number -20221406
Course- B.Sc(H) computer science
"""
import numpy as np
from scipy.optimize import minimize

# Define the objective function
def objective_function(x):
    return -10 * np.cos(np.pi * x - 2.2) + (x + 1.5) * x

# Set the initial guess
initial_guess = 0.0

# Use the minimize function from scipy.optimize to find the global minimum
result = minimize(objective_function, initial_guess, method='BFGS')

# Print the result
if result.success:
    print("Global optimal solution found at x =", result.x[0])
    print("Optimal function value =", -result.fun)
else:
    print("Optimization failed:", result.message)
```

Output-


IDLE Shell 3.11.6
File Edit Shell Debug Options Window Help
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct 2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\
Global optimal solution found at x = 0.6714379292644944
Optimal function value = 8.500986423557436
>>>

Q5- WAP to find Global Optimal Solution of a function $f(x) = -10Cos(\pi x - 2.2) + (x + 1.5)x$ graphically.

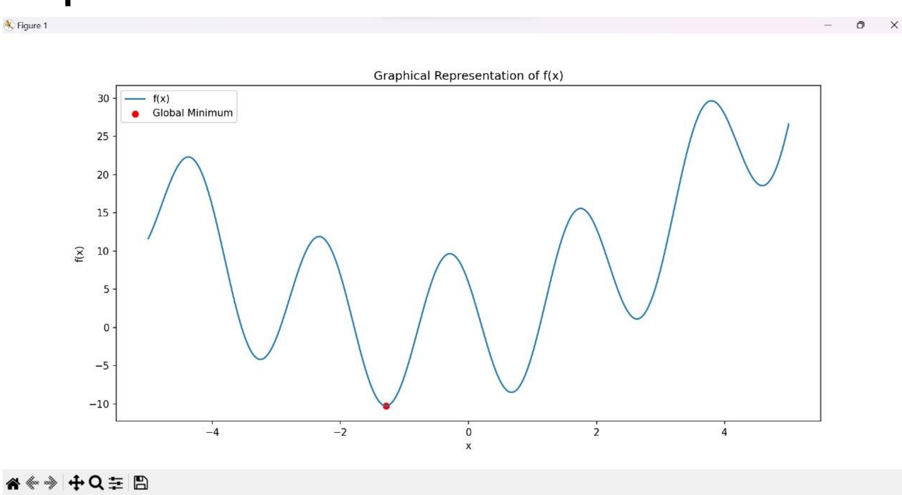
```
"""
 Name-Aman
 Examination roll number -22020570024
 Class Roll number -20221406
 Course- B.Sc(H) computer science
 '''
import numpy as np
import matplotlib.pyplot as plt

# Rest of the code remains unchanged



# Define the objective function
def objective_function(x):
    return -10 * np.cos(np.pi * x - 2.2) + (x + 1.5) * x

# Generate x values
x_values = np.linspace(-5, 5, 1000)

# Calculate corresponding y values
y_values = objective_function(x_values)

# Plot the function
plt.plot(x_values, y_values, label='f(x)')

# Find the minimum of the function
min_x = x_values[np.argmin(y_values)]
min_y = objective_function(min_x)
```

Prashant Mishra(20221481)-5.py - C:\Prashant mishra(20221481)\Prashant Mishra(20221481)-5.py (3.11.6)

File Edit Format Run Options Window Help

```python
y_values = objective_function(x_values)

# Plot the function
plt.plot(x_values, y_values, label='f(x)')

# Find the minimum of the function
min_x = x_values[np.argmin(y_values)]
min_y = objective_function(min_x)

# Mark the minimum point on the plot
plt.scatter(min_x, min_y, color='red', label='Global Minimum')

# Add labels and title
plt.xlabel('x')
plt.ylabel('f(x)')
plt.title('Graphical Representation of f(x)')

# Display the legend
plt.legend()

# Show the plot
plt.show()

# Print the result
print("Global optimal solution found at x =", min_x)
print("Optimal function value =", min_y)
```

Output-



Q6-WAP to solve constraint optimization problem.

File Edit Format Run Options Window Help

```
'''
 Name-Aman
 Examination roll number -22020570024
 Class Roll number -20221406
 Course- B.Sc(H) computer science
 '''
from scipy.optimize import minimize

# Objective function to minimize
def objective_function(x):
    return x[0]**2 + x[1]**2  # Example: minimize x^2 + y^2

# Constraint function
def constraint_function(x):
    return x[0] + x[1] - 1  # Example: constraint x + y = 1

# Initial guess
initial_guess = [0, 0]

# Define the constraint
constraint = {'type': 'eq', 'fun': constraint_function}

# Solve the optimization problem
result = minimize(objective_function, initial_guess, constraints=constraint)

# Print the result
print("Optimal solution:", result.x)
print("Optimal value of the objective function:", result.fun)
```

Output-

```
IDLE Shell 3.11.6
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct  2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\
Optimal solution: [0.5 0.5]
Optimal value of the objective function: 0.5
>>>
```