MOUNTAINS OF THE MOON UNIVERSITY
FACULTY OF SCIENCE TECHNOLOGY AND INNOVATION
DEPARTMENT OF COMPUTER SCIENCE
COURSE UNIT: LINEAR PROGRAMMING
LECTURER: MR.OCEN SAMUEL

NAMONYE KENNETH
REG NO. 2023/U/MMU/BCS/01668

February 2024

1 BASIC RESOURCE ALLOCATION

```
from pulp import*
import matplotlib.pyplot as plt
import numpy as np
#creating LpProblem
model = LpProblem(name="cost_Minimization", sense =LpMinimize)

#decision variables
x=LpVariable("x",0)
y=LpVariable("y",0)

#objective function
model += 4* x + 5 *y

#constraints
model += 2*x + 3*y >=10, "CPU"
model += x + 2*y >=5, "MEMORY"
model += 3*x + y >=8, "STORAGE"
#Solving
model.solve()
#display
optimal_x = x.varValue
optimal_y = y.varValue
optimal_value = model.objective.value()
print("optimal solution: ")
print("x:", optimal_x)
print("y:",optimal_y)
print("minimum cost is:", optimal_value)

#plotting feaseble region
x_values=np.linspace(0,10,200)
y1_values=(10-2*x_values)/3
y2_values=(5-x_values)/2
y3_values=(8-3*x_values)

plt.plot(x_values,y1_values,label=2*x + 3*y >=10)
plt.plot(x_values,y2_values,label=x + 2*y >=5)
plt.plot(x_values,y3_values,label=3*x + y >=8)

x=[0,0,5,2,1]
y=[5,0,0,2,5]
plt.fill(x,y,color='grey',label='unwanted region')

plt.xlim(0,5)
plt.ylim(0,5)
plt.xlabel('x')
plt.ylabel('y')
plt.grid(True)
plt.title('Graph showing the unwanted region')
plt.scatter(optimal_x,optimal_y,color="red",label='Optimal Solution')
plt.legend()
plt.show()

optimal solution:
x: 2.0
y: 2.0
optimal value: 18.0
```

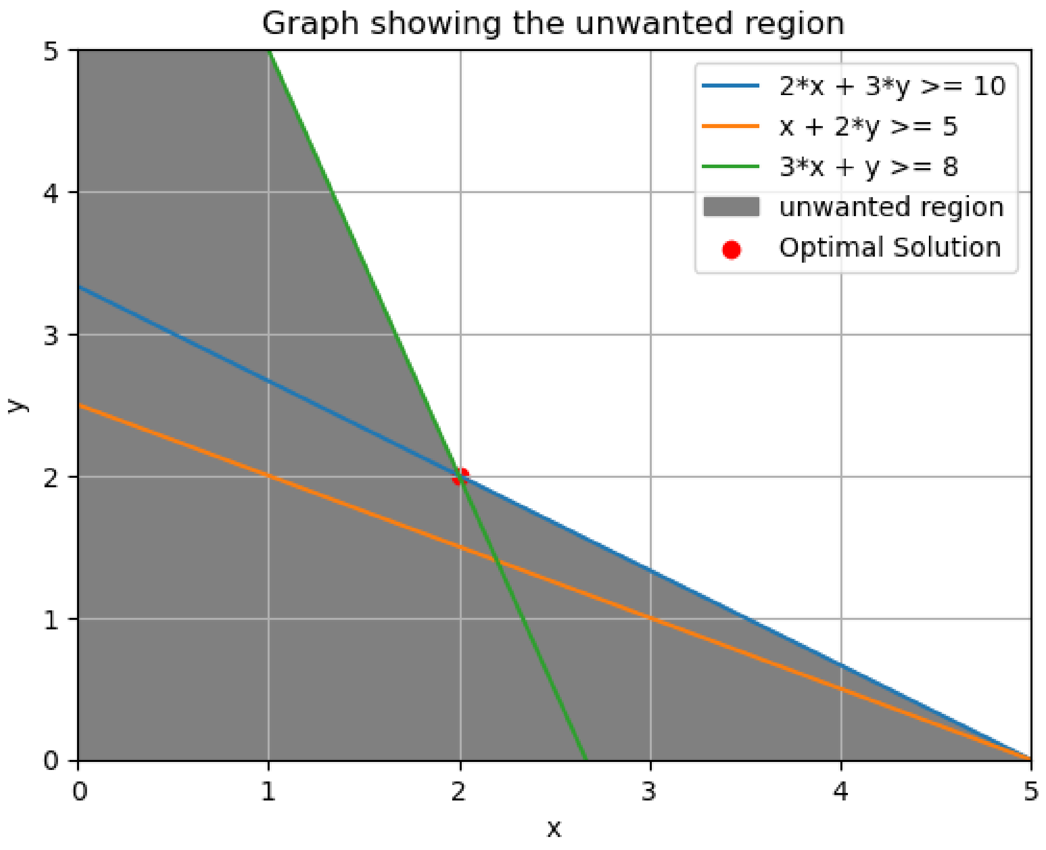

Figure 1: This graph shows the unwanted region

2 LOAD BALANCING

```
    from pulp import*
import matplotlib.pyplot as plt
import numpy as np
#creating LpProblem
model = LpProblem("Optimize_distribution_of _workloads", sense=LpMinimize)

#decision variables
x =LpVariable (name="x", lowBound=0)
y = LpVariable(name="y", lowBound=0)
#objective function
model += 5*x + 4*y
#constraints
model+= 2*x + 3*y <= 20, "sever1"
model+=4*x + 2*y <= 15, "sever2"
#solving
model.solve()
#disply results
optimal_x = x.varValue
optimal_y = y.varValue
optimal_value = model.objective.value()
print("optimal_solution: ")
print("x:",optimal_x)
print("y:",optimal_y)
print("objective value:", optimal_value)

#plotting
x_values = np.linspace(0,16,2000)
y1_values = (20-2*x_values)/3
y2_values = (15-4*x_values)/2

plt.plot(x_values,y1_values,label=2*x + 3*y<=20)
plt.plot(x_values,y2_values, label=4*x + 2*y<=15)

x=[0,0,10,10,3.75,0.625]
y=[6.67,10,10,0,0,6.25]
plt.fill(x,y,color='grey',label='unwanted region')
plt.xlim(0,10)
plt.ylim(0,10)
plt.grid(True)
plt.scatter(optimal_x,optimal_y,color='red',label='optimal solution')
plt.title("A graph showing the unwanted region")
plt.legend()
plt.show()
Optimal solution:
```

```
x: 0.0
y: 0.0
Optimal value: 0.0
```

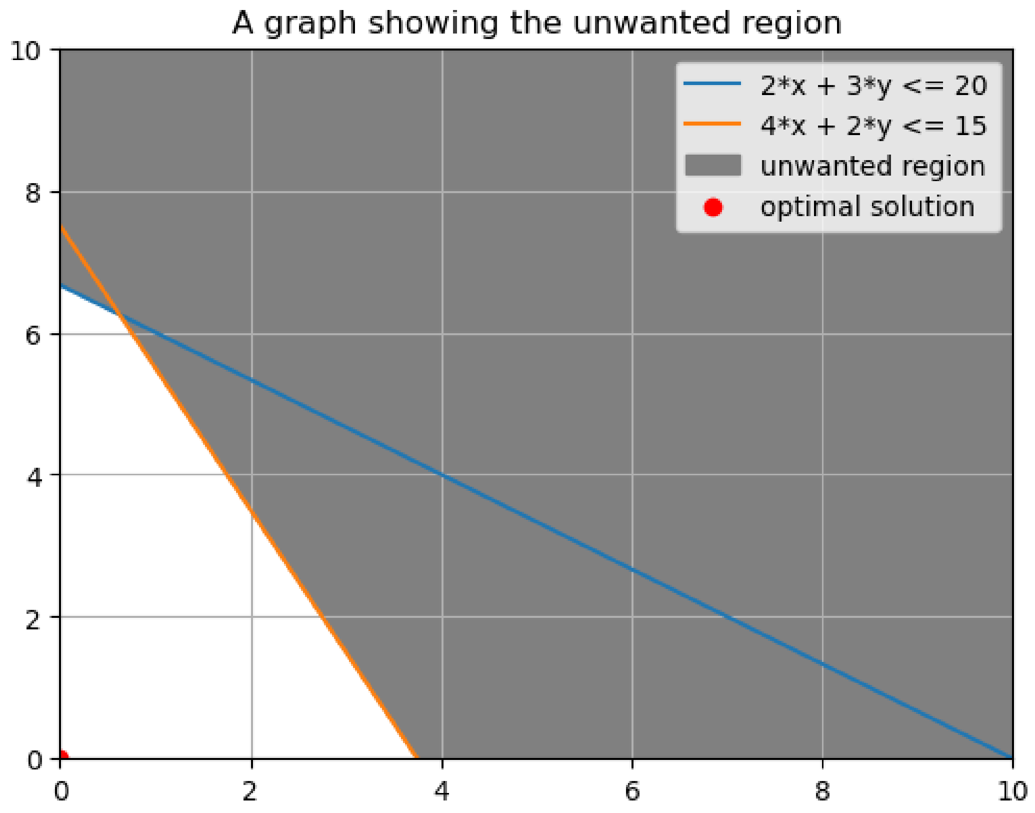

Figure 2: This graph shows the unwanted region

3 ENERGY EFFICIENT RESOURCE ALLOCATION

```
from pulp import*
import matplotlib.pyplot as plt
import numpy as np
#creting LpProblem
model=LpProblem(name="Energy_Consumption",sense=LpMinimize)
#decision variables
x=LpVariable("x",0)
y=LpVariable("y",0)
#objective function
model+= 3*x + 2*y
#constraints
model+= 2*x + 3*y>=15,"CPU_allocation"
model+= 4*x +2*y>=10,"Memory_allocation"
model+= x>=0
model +=y>=0
#solving
model.solve()
#display results
optimal_x = x.varValue
optimal_y = y.varValue
optimal_value = model.objective.value()
print('optimal solution:')
print("x:", optimal_x)
print("y:",optimal_y)
print("objective value :",optimal_value)
import numpy as np
import matplotlib.pyplot as plt

x_values = np.linspace(0, 10, 100)
y1_values = (15 - 2 * x_values) / 3
y2_values = (10 - 4 * x_values) / 2

x=[0,0,3,3]
y=[5,0,0,3]
plt.fill(x,y,color="grey",label='unwanted region')

plt.plot(x_values, y1_values, label='2x + 3y >= 15')
plt.plot(x_values, y2_values, label='4x + 2y >= 10')

plt.title('Graph showing the unwanted region ')
plt.scatter(optimal_x,optimal_y,color='red',label='optimal solution')
plt.xlim(0, 3)
plt.ylim(0, 6)
plt.grid(True)
plt.legend()
plt.show()
Optimal solution:
x: 0.0
y: 5.0
optimal value: 10.0
```

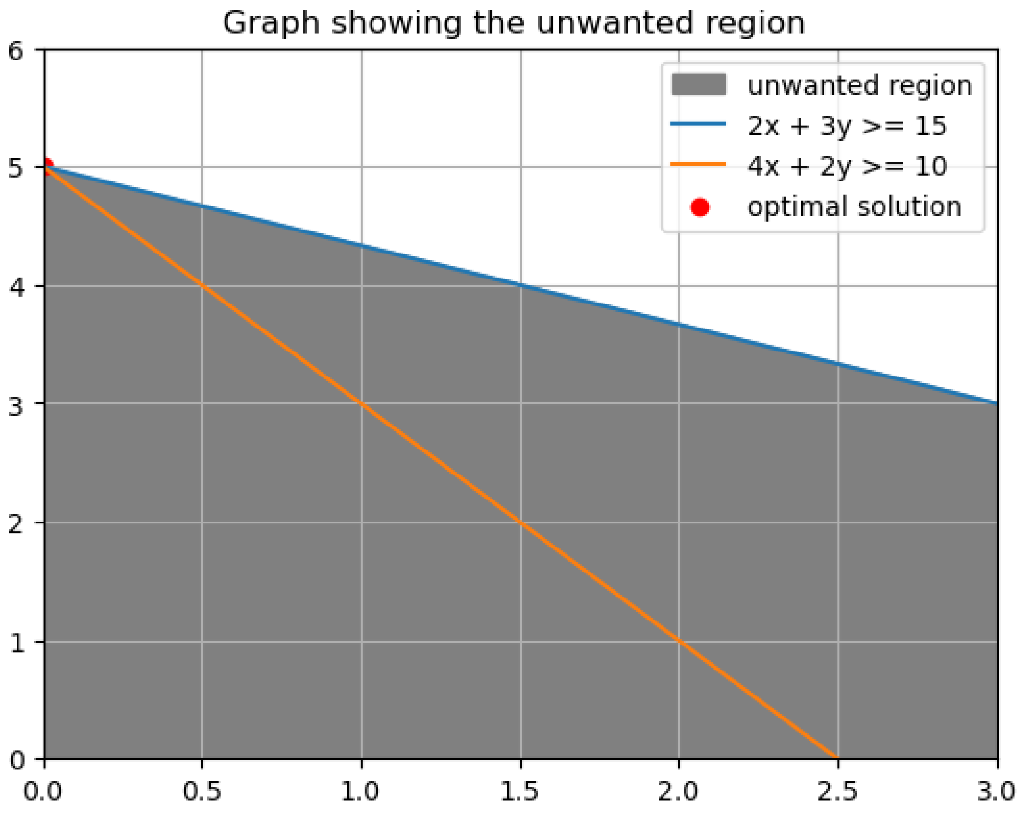

Figure 3: This graph shows the unwanted regions

4 MULTI-TENANT RESOURCE SHARING

```
from pulp import*
import matplotlib.pyplot as plt
import numpy as np
#LpProblem
model= LpProblem(name="Multi_tenant", sense=LpMinimize)
#variables
x=LpVariable("x",0)
y=LpVariable("y",0)
#objective function
model += 5*x +4*y
#constraints
model +=2*x +3*y >=12,"tenant1"
model +=4*x + 2*y >=18,"tenant2"
#solving
model.solve()
#display results
optimal_x = x.varValue
optimal_y= y.varValue
optimal_value= model.objective.value()
print("optimal solution:")
print("x:",optimal_x)
print("y:", optimal_y)
print("z:", optimal_value)
#ploting
x_values= np.linspace(0,10,100)
y1_values=(12-2*x_values)/3
y2_values=(18-4*x_values)/2

plt.plot(x_values,y1_values,label=2*x + 3*y >=12)
plt.plot(x_values,y2_values,label=4*x + 2*y >=1)

x=[0,3.75,6,0]
y=[9,1.5,0,0]
plt.fill(x,y,color="grey")

plt.xlim(0,10)
plt.ylim(0,10)
plt.xlabel('X')
plt.ylabel('Y')
plt.title('Graph showing the unwanted region')
plt.scatter(optimal_x,optimal_y,color='red', label='optimal solution')
plt.grid(True)
plt.legend()
plt.show()
```

```
Optimal solution:
x: 3.75
y: 1.5
optimal value: 24.75
```

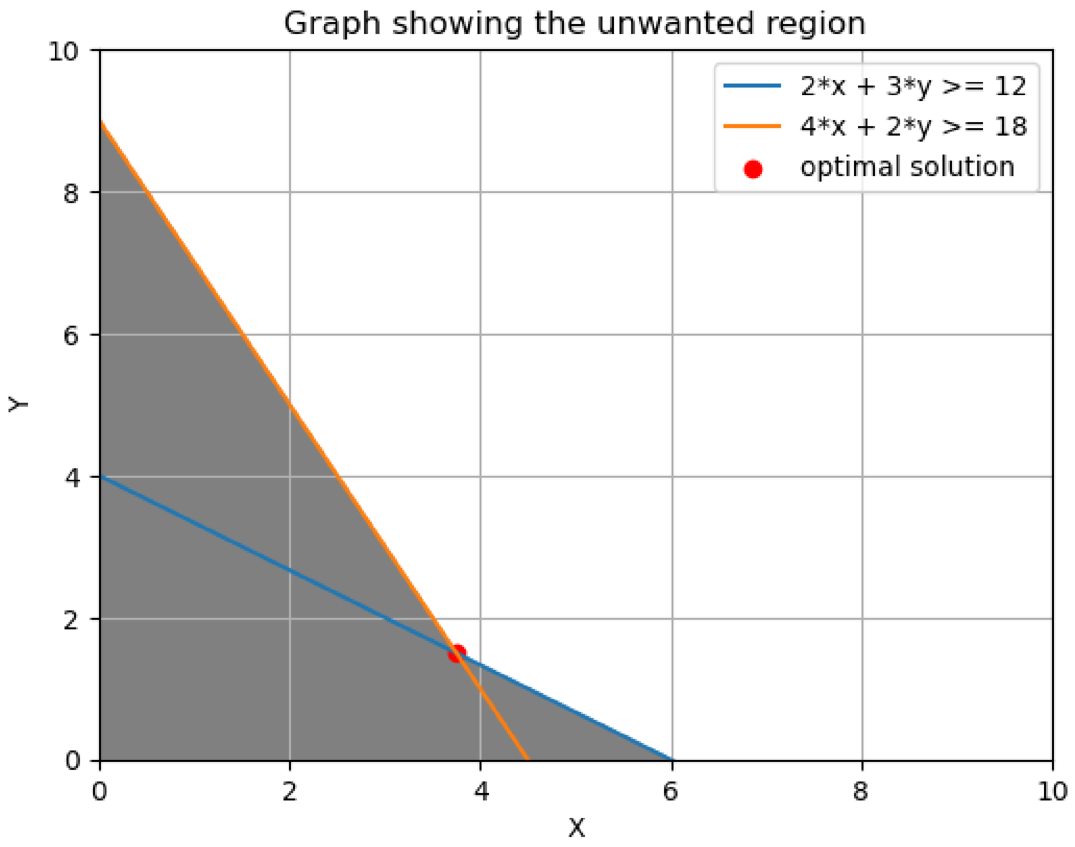

Figure 4: This graph shows the unwanted region

5 PRODUCTION PLANNING

```
    import numpy as np
import matplotlib.pyplot as plt
from pulp import LpMaximize, LpProblem, LpVariable

# Define the problem
problem = LpProblem("Maximize_Objective", LpMaximize)

# Define decision variables
x = LpVariable('x', lowBound=0)
y = LpVariable('y', lowBound=0)

# Define objective function
problem += 5*x + 3*y, "Objective Function"

# Define constraints
problem += 2*x + 3*y <= 60
problem += 4*x + 2*y <= 80

# Solve the problem
problem.solve()

# Optimal values
optimal_x = x.varValue
optimal_y = y.varValue
optimal_value = problem.objective.value()

print("Optimal Solution:")
print("x =", optimal_x)
print("y =", optimal_y)
print("Objective Value =", optimal_value)

# Plotting the feasible region
x_values = np.linspace(0, 40, 400)
y1_values = (60 - 2*x_values) / 3
y2_values = (80 - 4*x_values) / 2

plt.plot(x_values, y1_values, label='2x + 3y <= 60')
plt.plot(x_values, y2_values, label='4x + 2y <= 80')
x=[0,15,20,20]
y=[20,10,0,20]
plt.fill(x,y,color='grey',label='unwanted region')
plt.xlim(0,20)
plt.ylim(0,20)
plt.grid(True)
plt.title('A graph showing the unwanted region')
plt.scatter(optimal_x,optimal_y,color='red',label='Optimal solution')
plt.xlabel('x_axis')
plt.ylabel('y_axis')
plt.legend()
plt.show()
Optimal solution:
x: 15.0
y: 10.0
optimal value: 105.0
```

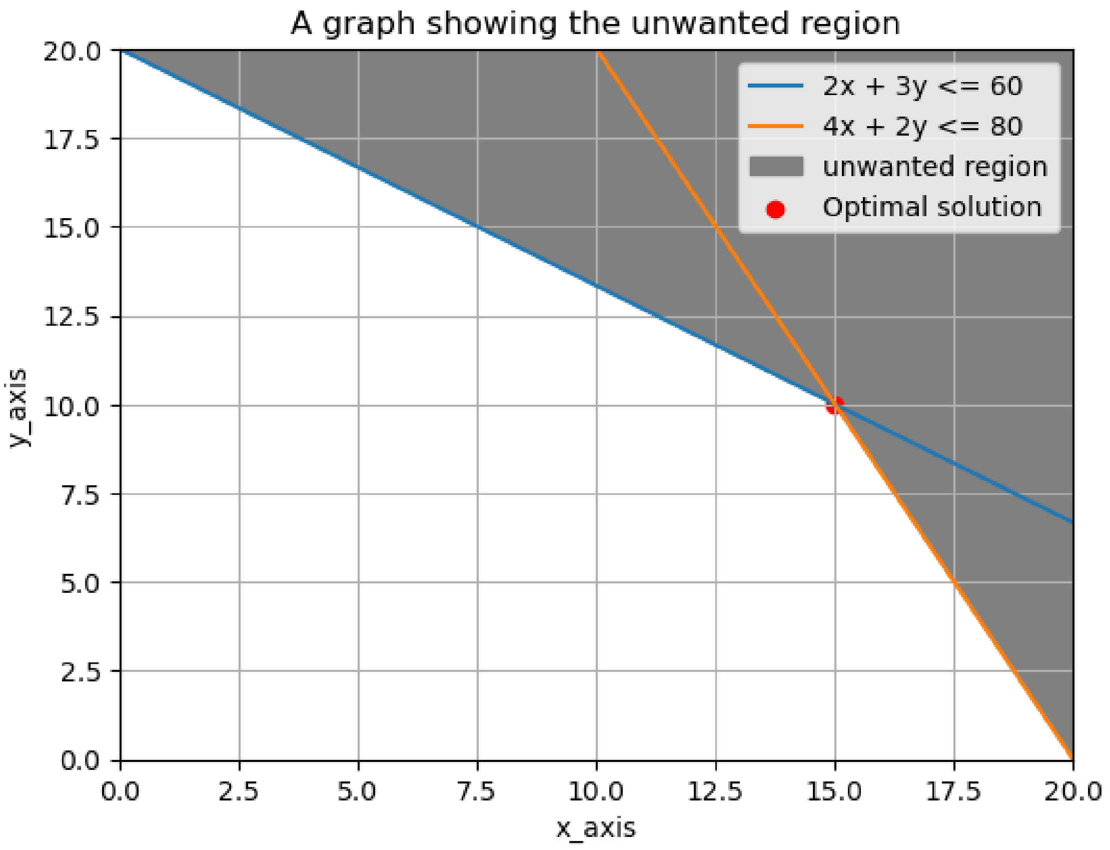

Figure 5: This graph shows the unwanted region

6 DIET OPTIMIZATION

```
    from pulp import*
import matplotlib.pyplot as plt
import numpy as np
#lpProblem
model = LpProblem(name="diet_optimization", sense=LpMinimize)
#LpVariables
x=LpVariable("x",0)
y=LpVariable("y",0)
#objective function
model+=3*x + 2*y
#constraints
model+= 2*x + y >= 20
model+=3*x + 2*y>= 25
#solving
model.solve()
#display results
x=x.varValue
y=y.varValue
optimal_value= model.objective.value()
print("status:",LpStatus[model.status])
print("x",x)
print("y",y)
print("optimal", optimal_value)

import matplotlib.pyplot as plt
import numpy as np

# Define the constraints
def constraint1(x):
    return 20 - 2*x

def constraint2(x):
    return (25 - 3*x) / 2

# Create x range for plotting
x_values = np.linspace(0, 15, 400)

# Plot constraints
plt.plot(x_values, constraint1(x_values), label='2x + y >= 20')
plt.plot(x_values, constraint2(x_values), label='3x + 2y >= 25')

# Shade unwanted region
x=[0,0,15]
y=[20,-10,-10]
```

```
plt.fill(x,y,color='gray',label='unwanted region')


# Set labels and legend
plt.xlabel('x')
plt.ylabel('y')
plt.title('A graph showing the unwanted region')
plt.scatter(x,y,color='red',label='optimal solution')
plt.legend()

# Show plot
plt.grid(True)
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)
plt.show()
x: 10.0
y: 0.0
optimal value: 30.0
```

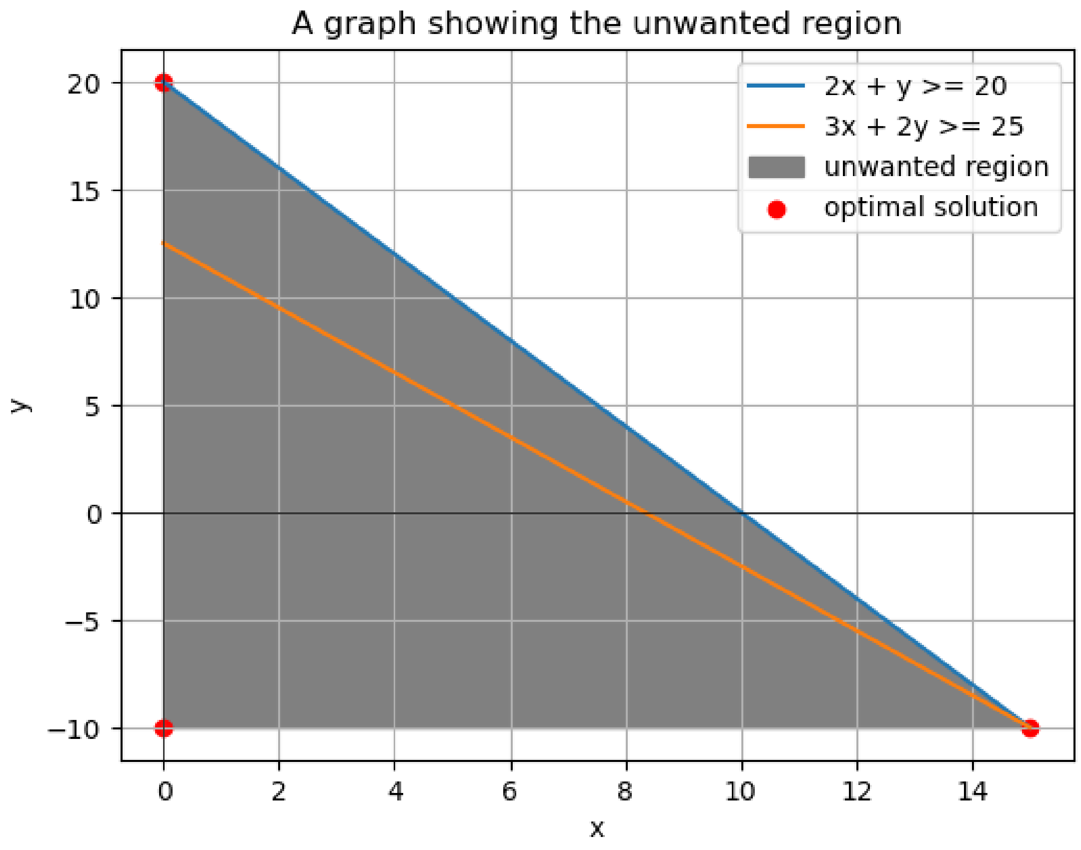

Figure 6: This graph shows the unwanted region

7 PRODUCTION PLANNING IN MANUFACTURING

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pulp import LpVariable, LpMinimize, LpProblem

#lpProblem
problem = LpProblem(name="Production_minimizing", sense=LpMinimize)

#decision variables
x1 = LpVariable(name="x", lowBound=0)
x2 = LpVariable(name="y", lowBound=0)
x3 = LpVariable(name="z", lowBound=0)

# Define the objective function coefficients
problem += 5*x1 + 3*x2 + 4*x3, "objective"

# Coefficients of the inequality constraints
problem += 2*x1 + 3*x2 + x3 <= 1000, "raw_materials"
problem += 4*x1 + 2*x2 + 5*x3 <= 120, "labour"
problem += x1 >= 200
problem += x2 >= 300
problem += x3 >= 150


# Solve linear programming problem
problem.solve()

# Display the results
print("OPTIMAL SOLUTION:")
print(f"X: {x1.varValue}")
print(f"Y: {x2.varValue}")
print(f"Z: {x3.varValue}")
print(f"Minimum cost: {problem.objective.value()}")


# plotting the graph

# Create a meshgrid for x1, x2, and x3
x1_vals = np.linspace(0, 400, 50)
x2_vals = np.linspace(0, 400, 50)
x1_grid, x2_grid = np.meshgrid(x1_vals, x2_vals)
```

```
# Calculate the corresponding z-values (objective function)
z_vals = 5 * x1_grid + 3 * x2_grid + 4 * (1950 - x1_grid - x2_grid)

# Create the 3D plot
fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(111, projection='3d')

# Plot the feasible region (constraints)
ax.plot([200, 200], [0, 400], [0, 0], color='red', linestyle='--', linewidth=2, label='Raw M
ax.plot([0, 400], [300, 300], [0, 0], color='green', linestyle='--', linewidth=2, label='Lab
ax.plot([0, 400], [0, 400], [1950, 1950], color='blue', linestyle='--', linewidth=2, label='

# Highlight the optimum point
optimum_x1 = 200
optimum_x2 = 300
optimum_z = 1950
ax.scatter(optimum_x1, optimum_x2, optimum_z, color='purple', s=100, label='Optimum Point')

# Set labels and title
ax.set_xlabel('x1')
ax.set_ylabel('x2')
ax.set_zlabel('x3')
ax.set_title('Production Cost Minimization')

# Add a legend
ax.legend()

# Show the plot
plt.show()
OPTIMAL SOLUTION:
X: 200.0
Y: 300.0
Z: 0.0
Minimum cost: 1900.0
```

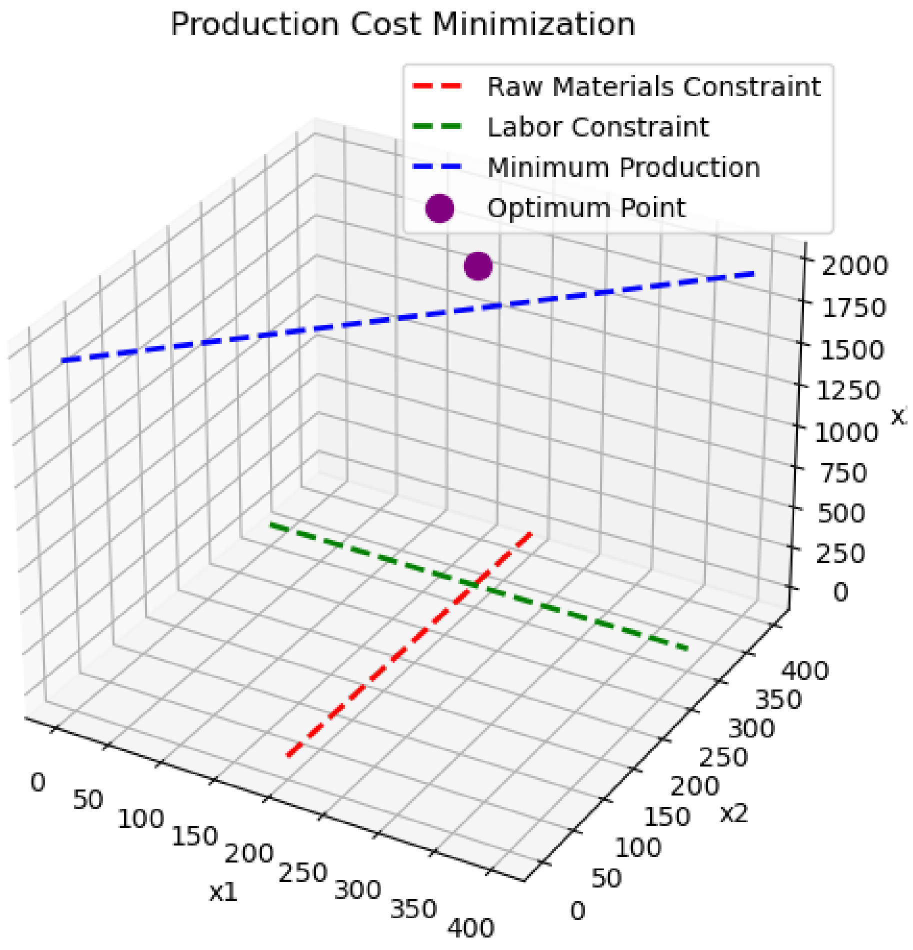

Figure 7: Caption

8 FINANCIAL PORTFOLIO OPTIMIZATION

```
    import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pulp import LpVariable, LpMaximize, LpProblem

#the problem
problem = LpProblem(name="Financial_portfolio_optimization", sense=LpMaximize)

#decision variables
x1 = LpVariable(name="x", lowBound=0)
x2 = LpVariable(name="y", lowBound=0)
x3 = LpVariable(name="z", lowBound=0)

# Define the objective function coefficients
problem += 0.08*x1 + 0.1*x2 + 0.12*x3, "objective"

# Coefficients of the inequality constraints
problem += 2*x1 + 3*x2 + x3 <= 10000, "budget"
problem += x1 >= 2000
problem += x2 >= 1500
problem += x3 >= 1000


# Solve linear programming problem
problem.solve()

# Display the results
print("OPTIMAL SOLUTION:")
print(f"X: {x1.varValue}")
print(f"Y: {x2.varValue}")
print(f"Z: {x3.varValue}")
print(f"Minimum cost: {problem.objective.value()}")

# Create a meshgrid for A, B, and C
A_vals = np.linspace(0, 2500, 50)
B_vals = np.linspace(0, 2000, 50)
A_grid, B_grid = np.meshgrid(A_vals, B_vals)

# Calculate the corresponding z-values (ROI function)
C_vals = (10000 - 2 * A_grid - 3 * B_grid)  # Budget constraint
ROI_vals = 0.08 * A_grid + 0.1 * B_grid + 0.12 * C_vals

# Create the 3D plot
fig = plt.figure(figsize=(10, 6))
```

```
ax = fig.add_subplot(111, projection='3d')

# Plot the feasible region (constraints)
ax.plot([2000, 2000], [0, 2000], [0, 470], color='red', linestyle='--', linewidth=2, label=
ax.plot([0, 2500], [1500, 1500], [0, 470], color='green', linestyle='--', linewidth=2, label
ax.plot([0, 2500], [0, 2000], [470, 470], color='blue', linestyle='--', linewidth=2, label=

# Highlight the optimum point
optimum_A = 2000
optimum_B = 1500
optimum_ROI = 470
ax.scatter(optimum_A, optimum_B, optimum_ROI, color='purple', s=100, label='Optimum Point')

# Set labels and title
ax.set_xlabel('A')
ax.set_ylabel('B')
ax.set_zlabel('ROI')
ax.set_title('3D Plot: ROI Maximization')

# Add a legend
ax.legend()

# Show the plot
plt.show()
OPTIMAL SOLUTION:
X: 2000.0
Y: 1500.0
Z: 1500.0
Minimum cost: 490.0
```

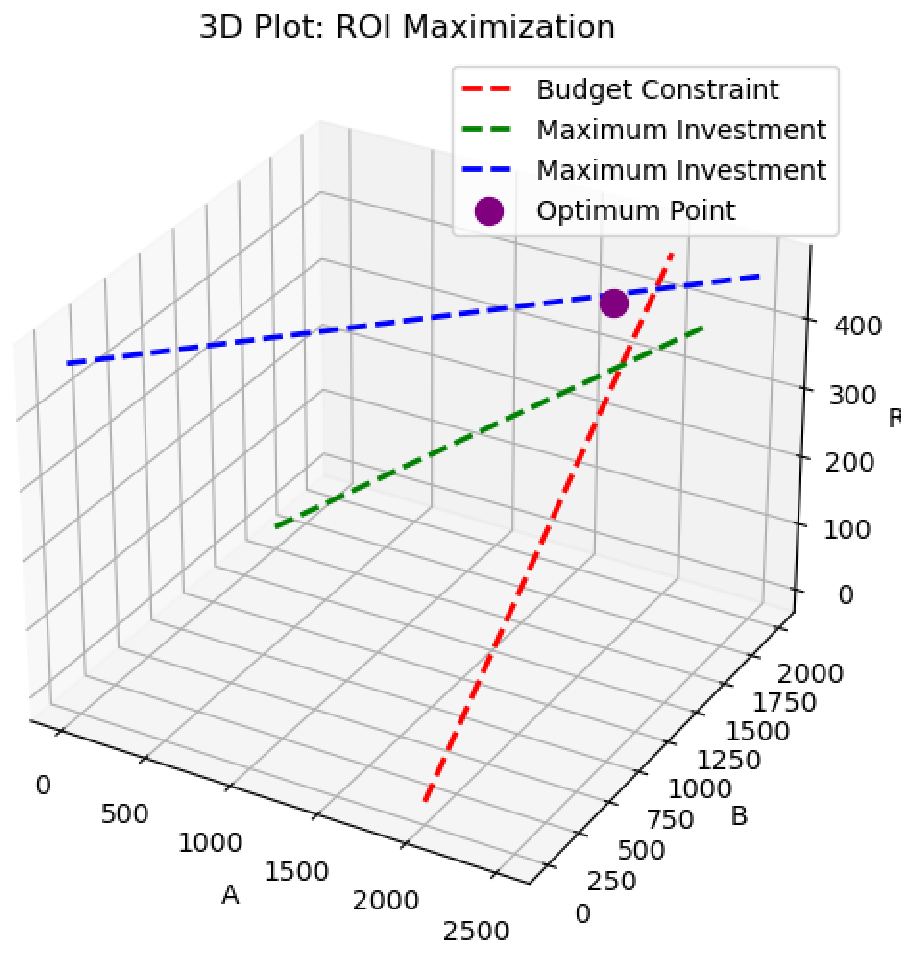

Figure 8: Caption