Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУК «Информатика и управление»

КАФЕДРА ИУК4 «Программное обеспечение ЭВМ, информационные технологии»

Лабораторная работа №2

«ГРАФИЧЕСКИЙ МЕТОД РЕШЕНИЯ ЗАДАЧ МАТЕМАТИЧЕСКОГО ПРОГРАММИРОВАНИЯ»

ДИСЦИПЛИНА: «Моделирование»

Выполнил: студент гр. ИУК4-72Б ______________ (Калашников А.С.)
(подпись) (Ф.И.О.)

Проверил: ______________ (Никитенко У.В.)
(подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2023

Цели: изучение математического аппарата математического программирования на примере задач небольшой размерности, допускающих графическое решение

Задачи: представить графическое решение, реализованное на языке высокого уровня

Вариант №6

Решить задачу нелинейного программирования графическим методом.

$$z = 2x_1 + x_2 \rightarrow (\max, \min)$$

при ограничениях

$$\begin{cases} (x_1 - 2)^2 + (x_2 - 1)^2 \geq 4, \\ (x_1 - 2)^2 + (x_2 - 1)^2 \leq 9, \\ x_1 + x_2 \geq 3; \end{cases}$$

$$x_1 \geq 0, x_2 \geq 0.$$

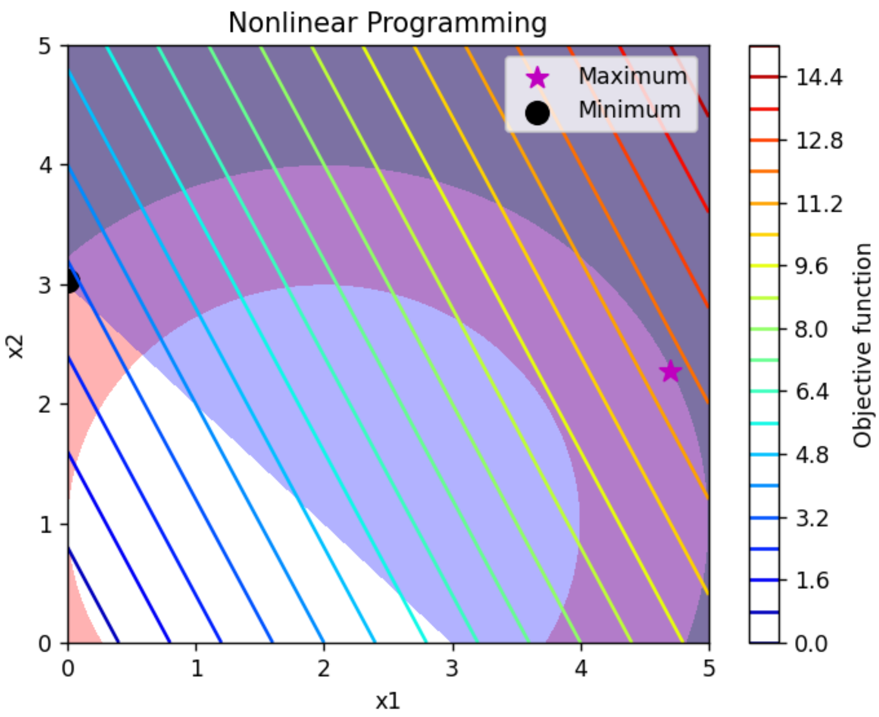

Рис.1 Результаты

Задание №2

Найти условный экстремум функции методом множителей Лагранжа

$Z=x_1+2x_2$->extr При условии (x_1)^2+(x_2)^2=1

Решение:

Экстремум достигается в точке:

x1 = -0.894427190999916

x2 = 0.447213595499958

Значение функции в экстремуме:

z = 0

Вывод: в ходе выполнения работы были изучены математические аппараты математического программирования на примере задач небольшой размерности, допускающих графическое решение

ПРИЛОЖЕНИЯ

Листинг программы

Ex. 1.6

```
import numpy as np
import matplotlib.pyplot as plt

def objective(x):
    return 2*x[0] + x[1]

def constraint1(x):
    return (x[0]-2)**2 + (x[1]-1)**2 - 4

def constraint2(x):
    return (x[0]-2)**2 + (x[1]-1)**2 - 9

def constraint3(x):
    return x[0] + x[1] - 3

# Задаем область значений x и y
x = np.linspace(0, 5, 100)
y = np.linspace(0, 5, 100)

# Создаем сетку значений x и y
X, Y = np.meshgrid(x, y)

# Вычисляем значение ограничений для каждой точки сетки
Z1 = constraint1([X, Y])
Z2 = constraint2([X, Y])
Z3 = constraint3([X, Y])

# Построение графиков ограничений
plt.contourf(X, Y, Z1, [0, np.inf], colors='r', alpha=0.3, label='Constraint
1')
plt.contourf(X, Y, Z2, [0, np.inf], colors='g', alpha=0.3, label='Constraint
2')
plt.contourf(X, Y, Z3, [0, np.inf], colors='b', alpha=0.3, label='Constraint
3')

# Построение графика целевой функции
plt.contour(X, Y, objective([X, Y]), 20, cmap='jet')

# Отображение графика
plt.xlabel('x1')
plt.ylabel('x2')
plt.title('Nonlinear Programming')
plt.colorbar(label='Objective function')
plt.legend()

# Нахождение максимума и минимума в области пересечения ограничений
intersection = np.logical_and(np.logical_and(Z1>=0, Z2<=0), Z3>=0)
x_intersection = X[intersection]
y_intersection = Y[intersection]
objective_intersection = objective([x_intersection, y_intersection])
max_index = np.argmax(objective_intersection)
min_index = np.argmin(objective_intersection)
max_x = x_intersection[max_index]
max_y = y_intersection[max_index]
```

```
min_x = x_intersection[min_index]
min_y = y_intersection[min_index]
plt.scatter(max_x, max_y, color='m', marker='*', s=100, label='Maximum')
plt.scatter(min_x, min_y, color='k', marker='o', s=100, label='Minimum')

plt.legend()
plt.show()
```

Ex. 2.6

```
from sympy import symbols, Eq, cos, sin, solve

# Определение символов
x1, x2, l = symbols('x1 x2 l')

# Определение функции и ограничения
f = x1 + 2*x2
constraint = x1**2 + x2**2 - 1

# Определение уравнений с помощью метода множителей Лагранжа
equation1 = Eq(f - l * constraint, 0)
equation2 = Eq(constraint, 0)

# Решение системы уравнений
solution = solve((equation1, equation2), (x1, x2, l))

# Вывод результатов

# Проверка существования экстремума
if not solution:
    print("Экстремум не существует.")
else:
    # Итерация по всем найденным решениям
    for i in range(len(solution)):
        # Вывод результата
        print("Экстремум достигается в точке:")
        print("x1 =", solution[i][0].evalf())
        print("x2 =", solution[i][1].evalf())
        print("Значение функции в экстремуме:")
        print("z =", f.subs({x1: solution[i][0], x2:
solution[i][1]}).evalf())
```