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 задачи математического программирования»

ДИСЦИПЛИНА: «Моделирование»

Выполнил: студент гр. ИУК4-72Б _____ (Карельский М.К.)
(Подпись)

Проверил: _____ (Никитенко У.В.)
(Подпись)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2023

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

Задачи: представить графическое решение, реализованное на языке высокого уровня.

Вариант 7

Решить задачу нелинейного программирования графическим методом:

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

$$\begin{cases} x_1 x_2 \geq 1 \\ x_1^2 + x_2^2 \leq 9 \end{cases}$$

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

Решение:

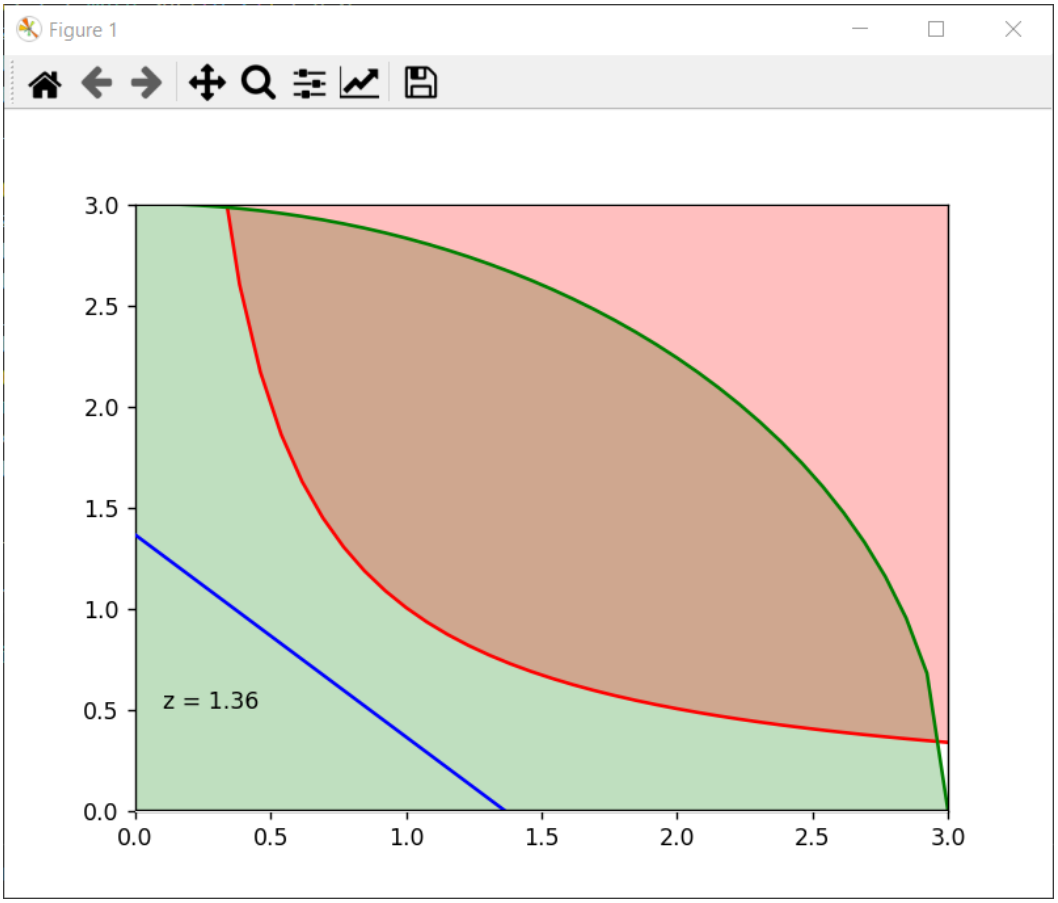

Рис. 1.1. Решение

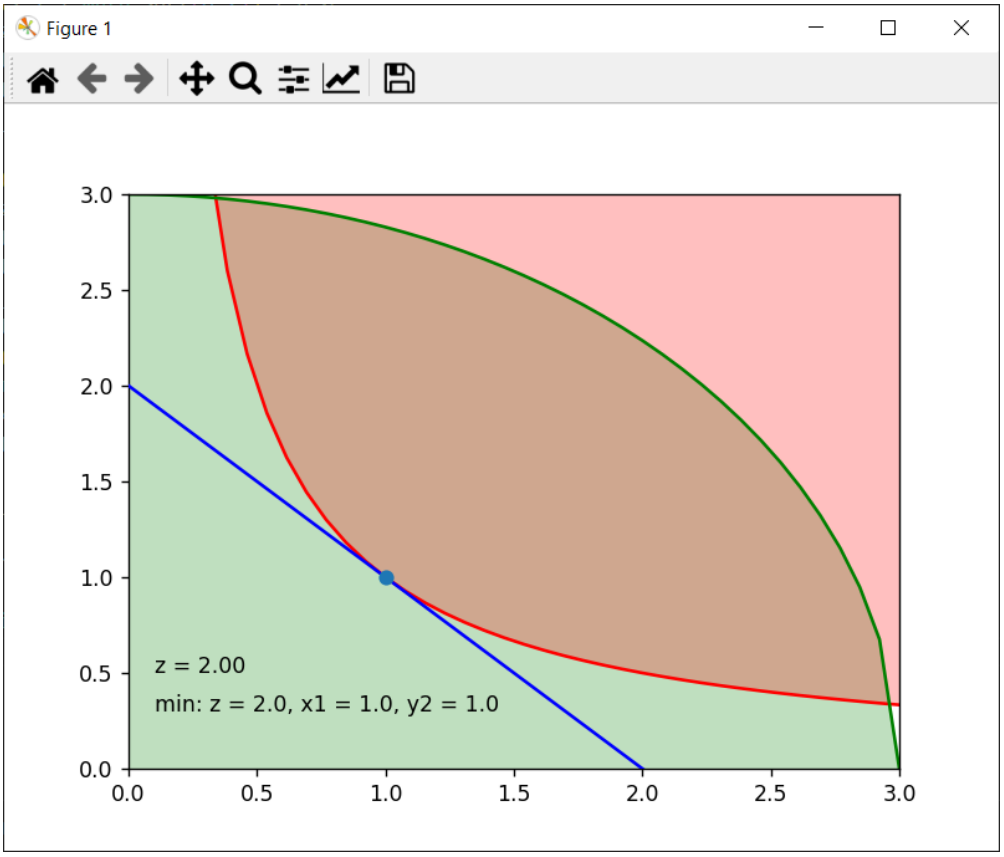

Рис. 1.2. Решение

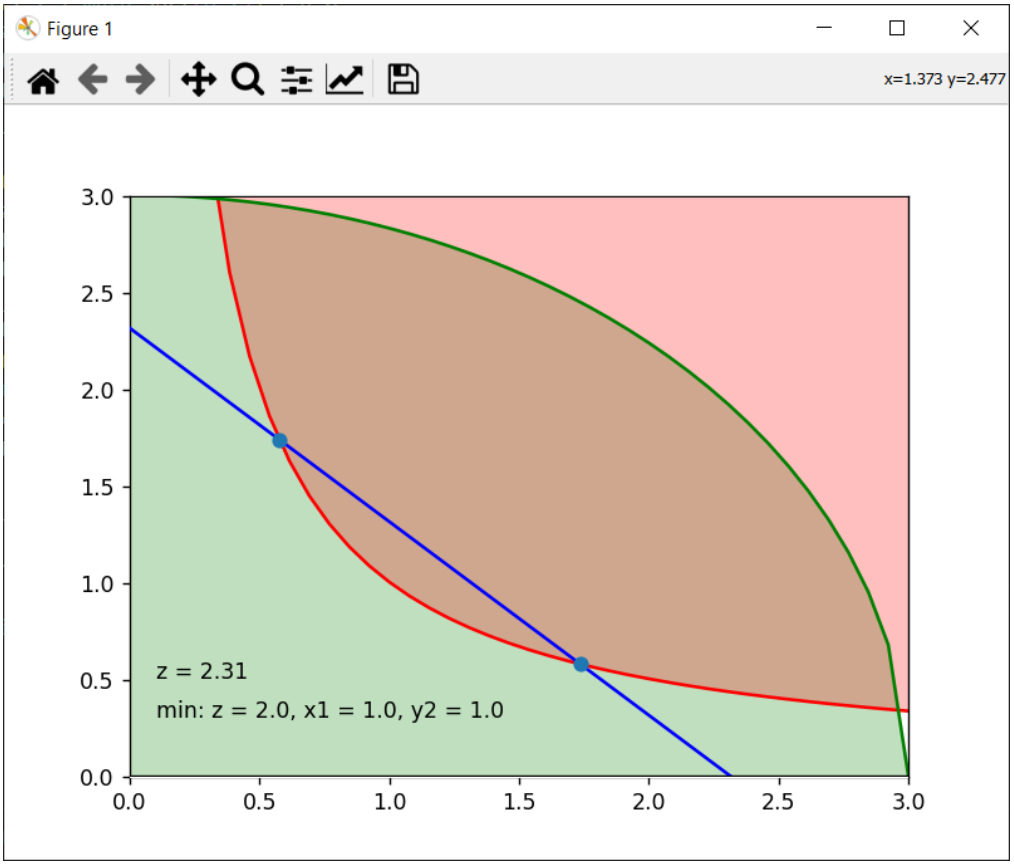

Рис. 1.3. Решение

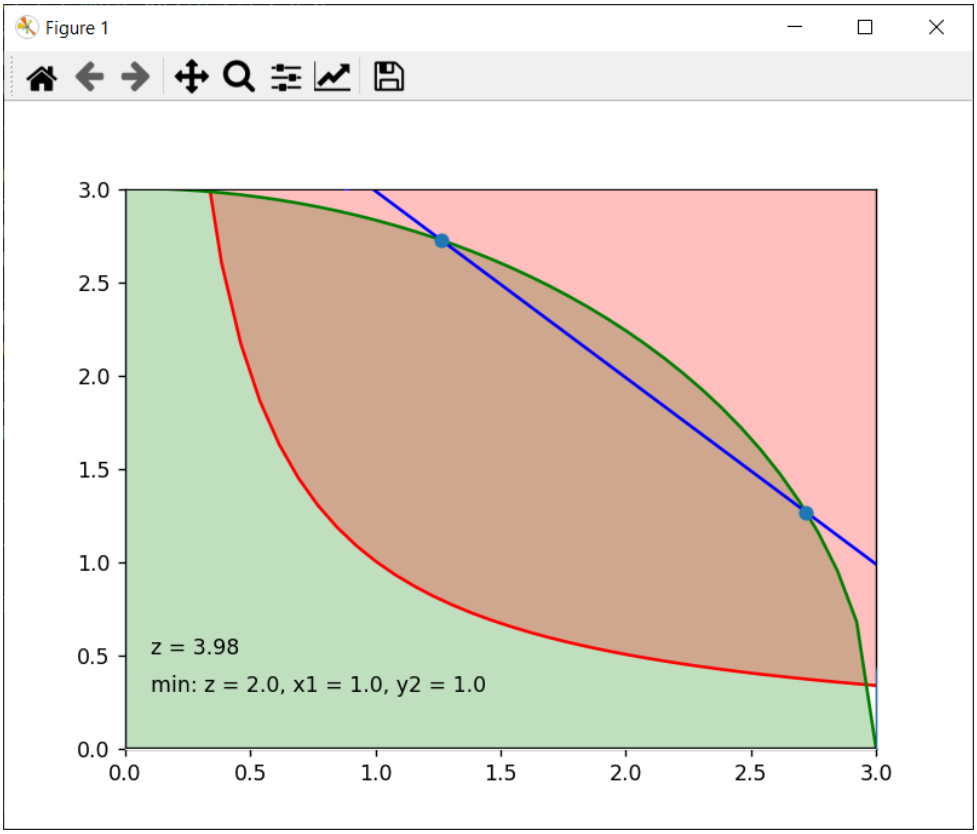

Рис. 1.4. Решение

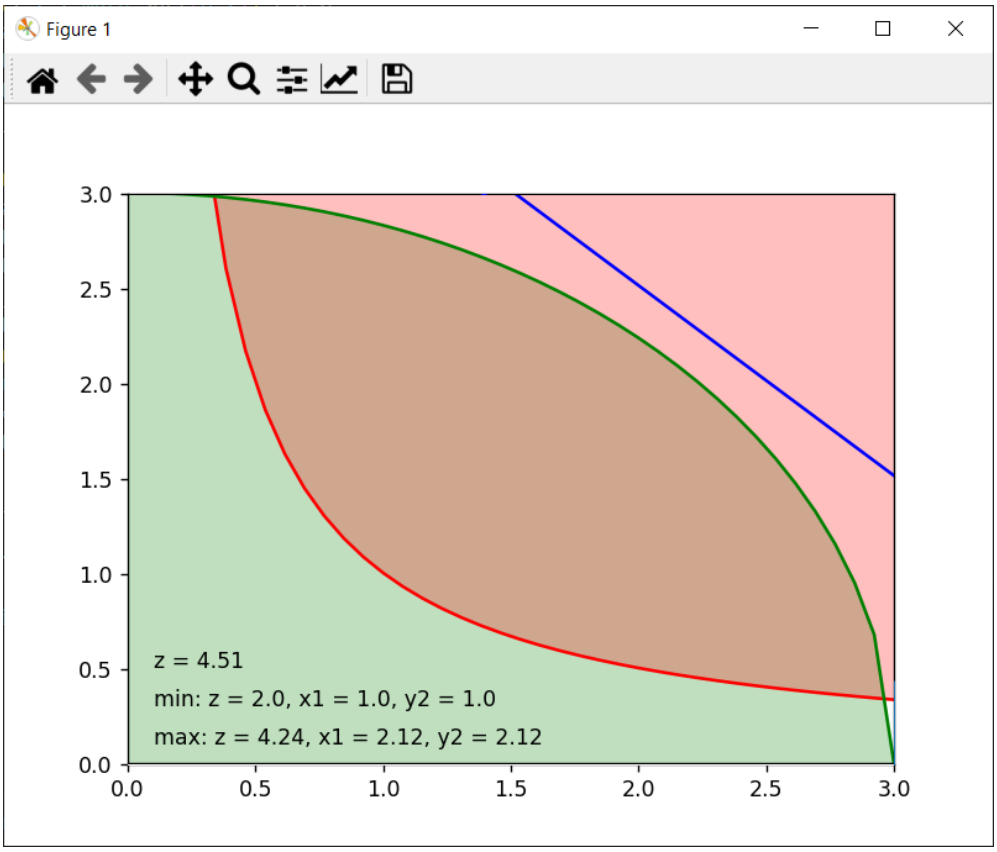

Рис. 1.5. Решение

Вывод: в ходе выполнения лабораторной работы был изучен математический аппарат математического программирования на примере задач небольшой размерности, допускающих графическое решение.

ПРИЛОЖЕНИЯ

Листинг:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
import math

X = np.linspace(0, 3, 40)
Y_1 = 1 / X
Y_2 = np.sqrt(9 - X**2)

red_fill_x = [X[-1]]
for x in X:
    red_fill_x.append(x)
red_fill_x.append(X[-1])

red_fill_y = [Y_1[1]]
for y in Y_1:
    red_fill_y.append(y)
red_fill_y.append(Y_1[1])

green_fill_x = [X[0]]
for x in X:
    green_fill_x.append(x)
green_fill_x.append(X[0])

green_fill_y = [Y_2[-1]]
for y in Y_2:
    green_fill_y.append(y)
green_fill_y.append(Y_2[-1])


fig, ax = plt.subplots()
ln, = ax.plot([], [], 'b')
dots, = ax.plot([], [], 'o')
min_text = ax.text(0.1, 0.3, '', fontsize=10)
max_text = ax.text(0.1, 0.1, '', fontsize=10)
z_text = ax.text(0.1, 0.5, '', fontsize=10)

ax.fill(green_fill_x, green_fill_y, 'green', alpha=0.25)
ax.fill(red_fill_x, red_fill_y, 'red', alpha=0.25)

ax.plot(X, Y_1, 'r')
ax.plot(X, Y_2, 'g')

z = 0
last_x_points = None

def init():
    ax.set_xlim(0, 3)
```

```
    ax.set_ylim(0, 3)
    return ln, dots, min_text, max_text, z_text,

def update(frame):
    global z
    global last_x_points

    ydata = frame - X
    ln.set_data(X, ydata)

    z = frame
    z_text.set_text(f'z = {z:.2f}')

    x_dots = []
    y_dots = []

    D_1 = z**2 - 4
    if D_1 == 0:
        x = z / 2.0
        y = z - x
        x_dots.append(x)
        y_dots.append(y)

        min_text.set_text(f'min: z = {z}, x1 = {x}, y2 = {y}')
    elif D_1 > 0:
        x = (z + math.sqrt(D_1)) / 2.0
        y = z - x
        x_dots.append(x)
        y_dots.append(y)

        x = (z - math.sqrt(D_1)) / 2.0
        y = z - x
        x_dots.append(x)
        y_dots.append(y)

    D_2 = 72 - 4*z**2
    if D_2 == 0:
        x = 2*z / 4.0
        y = z - x
        x_dots.append(x)
        y_dots.append(y)

        max_text.set_text(f'max: z = {z}, x1 = {x}, y2 = {y}')
    elif D_2 > 0:
        last_x_points = []

        x = (2*z + math.sqrt(D_2)) / 4.0
        y = z - x
        x_dots.append(x)
        y_dots.append(y)
        last_x_points.append(x)
```

```
            x = (2*z - math.sqrt(D_2)) / 4.0
            y = z - x
            x_dots.append(x)
            y_dots.append(y)
            last_x_points.append(x)
        elif last_x_points != None:
            x = (last_x_points[0] + last_x_points[1]) / 2.0
            y = math.sqrt(9 - x**2)
            z = x + y
            max_text.set_text(f'max: z = {z:.2f}, x1 = {x:.2f}, y2 = {y:.2f}')

        dots.set_data(x_dots, y_dots)

        return ln, dots, min_text, max_text, z_text

ani = FuncAnimation(fig, update, frames=np.linspace(1, 5, 501),
                    init_func=init, blit=True, interval=10)
plt.show()
```