

**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

|  |
| --- |
| **РТУ МИРЭА** |
|  |
| **Институт кибербезопасности и цифровых технологий (ИКБ)** |
|  |
| КБ-2 «Информационно-аналитические системы кибербезопасности» |

**ОТЧЕТ О ВЫПОЛНЕНИИ КОНТРОЛЬНОГО ДОМАШНЕГО ЗАДАНИЯ №4**

**ПО ДИСЦИПЛИНЫ «ФОРМАЛИЗОВАННЫЕ МОДЕЛИ И МЕТОДЫ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКОГО МОНИТОРИНГА»**

Выполнил:

Студент 3-ого курса

Учебной группы БИСО-02-22

Зубарев В.С.

Оглавление

[Цели работы 3](#_Toc197337062)

[Метод последовательных уступок 4](#_Toc197337063)

[Практическая реализация 5](#_Toc197337064)

[Приложение 1 9](#_Toc197337065)

# **Цели работы**

Дана многокритериальная аналитическая задача:

При ограничениях:

Решить поставленную задачу методом последовательных уступок.

Условия задачи заданы в таблице.

|  |  |  |
| --- | --- | --- |
|  |  |  |

# **Метод последовательных уступок**

Алгоритм нахождение оптимальной точки по n-критериям в зависимости от степени важности каждого из них.

1. Задаем лексикографический порядок, т.е. располагаем критерии по степени важности:
2. Решаем последовательность задач:
3. Назначается уступка и решается задача
4. Назначается уступка и решается задача
5. На шаге M уступка и решается задача
6. Полученное решение X\*m – считается оптимальным

# **Практическая реализация**

Дана многокритериальная аналитическая задача:

При ограничениях:

Лексикографическим порядком:

И уступками:

Построим D и определим ограничения

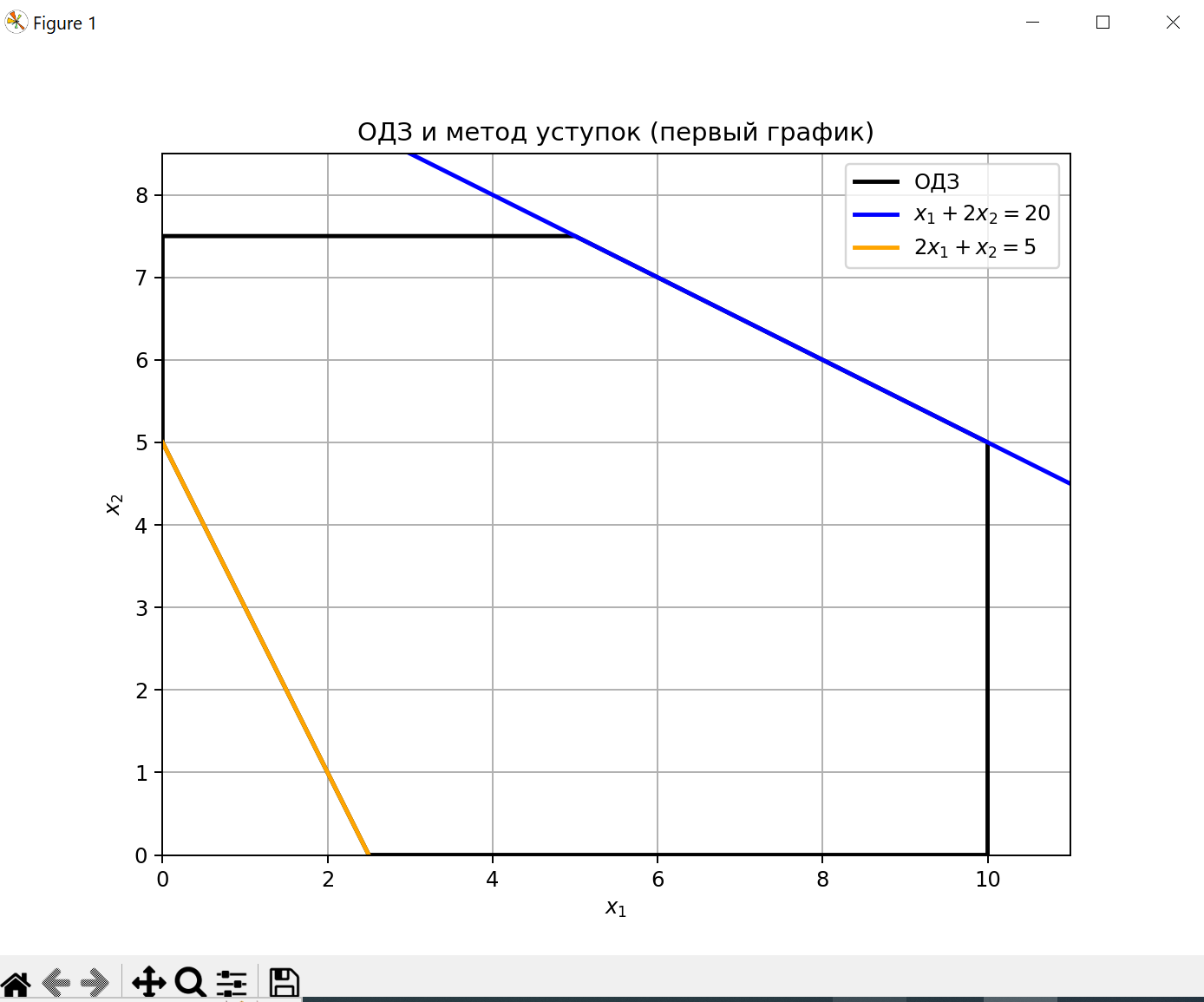


Рисунок 1 - Начальные условия

Тогда решаем задачу:

Построим область D2

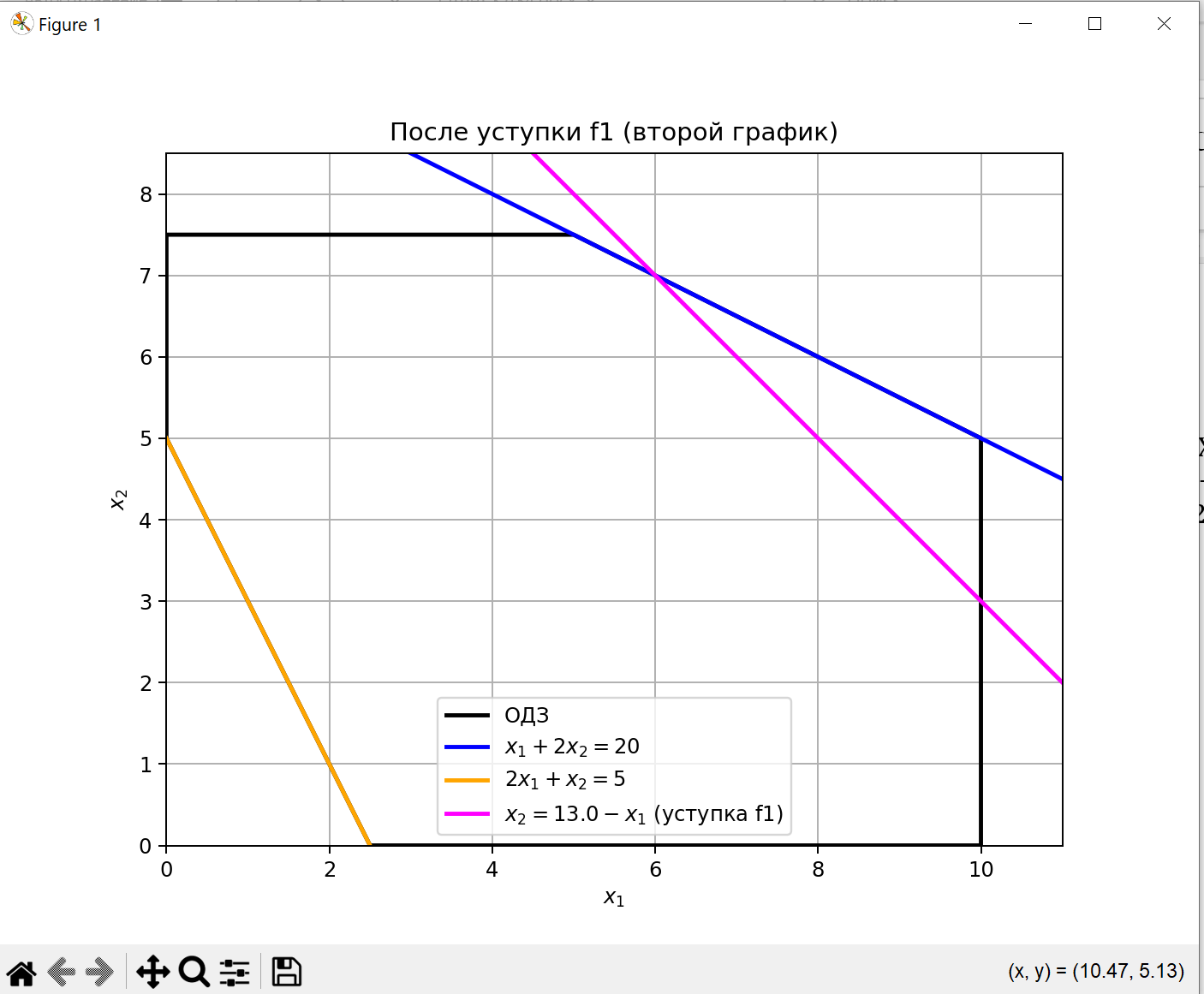


Рисунок 2 - ОДЗ поле первой уступки

Построим окончательную область и решим задачу максимизации для всех критериев.

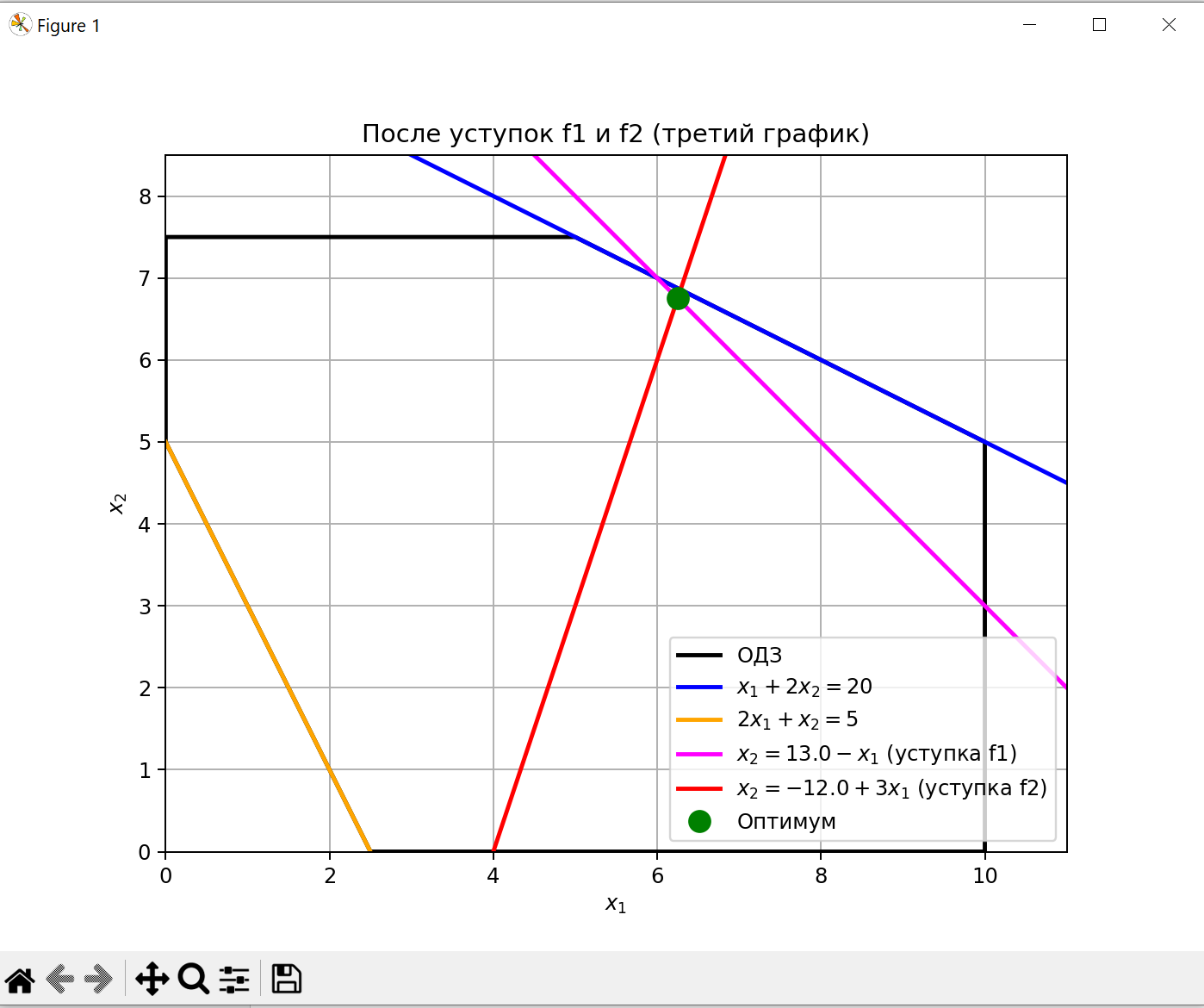


Рисунок 3 - ОДЗ поле второй уступки

Получаем решение *.*

# **Приложение 1**

В приложении 1 представлен листинг кода программы

import numpy as np  
import matplotlib.pyplot as plt  
from shapely.geometry import Polygon  
from scipy.optimize import linprog  
from itertools import combinations  
from numpy.linalg import solve  
  
# Параметры задачи  
n = 5  
delta\_f1 = 2  
delta\_f2 = 1  
  
# Матрица ограничений (левая часть Ax ≤ b)  
A = np.array([  
 [1, 2], # x1 + 2x2 ≤ 20  
 [-2, -1], # -2x1 - x2 ≤ -5  
 [-1, 0], # -x1 ≤ 0  
 [1, 0], # x1 ≤ 10  
 [0, -1], # -x2 ≤ 0  
 [0, 1] # x2 ≤ 7.5  
])  
  
b = np.array([  
 4 \* n, # 20  
 -n, # -5  
 0, # 0  
 2 \* n, # 10  
 0, # 0  
 1.5 \* n # 7.5  
])  
  
#Поиск вершин многоугольника ОДЗ  
vertices = []  
for c1, c2 in combinations(range(len(A)), 2):  
 A\_eq = np.array([A[c1], A[c2]])  
 b\_eq = np.array([b[c1], b[c2]])  
 if np.linalg.matrix\_rank(A\_eq) == 2:  
 try:  
 point = solve(A\_eq, b\_eq)  
 if all(np.dot(A, point) <= b + 1e-5):  
 vertices.append(tuple(point))  
 except np.linalg.LinAlgError:  
 continue  
  
# Удаляем дубликаты и сортируем по полярному углу  
vertices = list(set(vertices))  
center = np.mean(vertices, axis=0)  
vertices.sort(key=lambda p: np.arctan2(p[1]-center[1], p[0]-center[0]))  
  
# Параметры для графиков  
x1\_vals = np.linspace(0, 2 \* n + 1, 100)  
  
# 1-й график: ОДЗ и ограничения  
plt.figure(figsize=(8, 6))  
poly = Polygon(vertices)  
x, y = poly.exterior.xy  
plt.plot(x, y, 'k-', linewidth=2, label='ОДЗ')  
  
# x1 + 2x2 = 20  
x2\_vals1 = (4 \* n - x1\_vals) / 2  
plt.plot(x1\_vals, x2\_vals1, 'blue', linewidth=2, label=r'$x\_1 + 2x\_2 = 20$')  
  
# 2x1 + x2 = 5  
x2\_vals2 = n - 2 \* x1\_vals  
plt.plot(x1\_vals, x2\_vals2, 'orange', linewidth=2, label=r'$2x\_1 + x\_2 = 5$')  
  
# Метод последовательных уступок  
  
# 1. Максимизируем f1 = x1 + x2  
c1 = [-1, -1]  
res1 = linprog(c1, A\_ub=A, b\_ub=b)  
opt1 = res1.x  
f1\_max = -res1.fun  
f1\_limit = f1\_max - delta\_f1  
print(f"f1 max = {f1\_max:.3f} в точке ({opt1[0]:.3f}, {opt1[1]:.3f})")  
  
# Уступка по f1: -x1 - x2 ≤ -(f1\_max - delta\_f1)  
A2 = np.vstack([A, [-1, -1]])  
b2 = np.append(b, -f1\_limit)  
  
# 2. Максимизируем f2 = -3x1 + x2  
c2 = [3, -1]  
res2 = linprog(c2, A\_ub=A2, b\_ub=b2)  
opt2 = res2.x  
f2\_max = -res2.fun  
f2\_limit = f2\_max - delta\_f2  
print(f"f2 max = {f2\_max:.3f} в точке ({opt2[0]:.3f}, {opt2[1]:.3f})")  
  
# Уступка по f2: 3x1 - x2 ≤ -(f2\_max - delta\_f2)  
A3 = np.vstack([A2, [3, -1]])  
b3 = np.append(b2, -f2\_limit)  
  
# 3. Максимизируем f3 = x1 - 3x2  
c3 = [-1, 3]  
res3 = linprog(c3, A\_ub=A3, b\_ub=b3)  
opt3 = res3.x  
print(f"f3 max = {-res3.fun:.3f} в точке ({opt3[0]:.3f}, {opt3[1]:.3f})")  
  
plt.xlabel(r'$x\_1$')  
plt.ylabel(r'$x\_2$')  
plt.title('ОДЗ и метод уступок (первый график)')  
plt.grid(True)  
plt.legend()  
plt.gca().set\_aspect('equal', adjustable='box')  
plt.xlim(0, 2 \* n + 1)  
plt.ylim(0, 1.5 \* n + 1)  
plt.show()  
  
# 2-й график: с ограничением после уступки f1  
plt.figure(figsize=(8, 6))  
plt.plot(x, y, 'k-', linewidth=2, label='ОДЗ')  
plt.plot(x1\_vals, x2\_vals1, 'blue', linewidth=2, label=r'$x\_1 + 2x\_2 = 20$')  
plt.plot(x1\_vals, x2\_vals2, 'orange', linewidth=2, label=r'$2x\_1 + x\_2 = 5$')  
  
# Прямая после уступки f1  
x2\_vals\_f1 = f1\_limit - x1\_vals  
plt.plot(x1\_vals, x2\_vals\_f1, 'magenta', linewidth=2, label=fr'$x\_2 = {f1\_limit:.1f}- x\_1$ (уступка f1)')  
  
plt.xlabel(r'$x\_1$')  
plt.ylabel(r'$x\_2$')  
plt.title('После уступки f1 (второй график)')  
plt.grid(True)  
plt.legend()  
plt.gca().set\_aspect('equal', adjustable='box')  
plt.xlim(0, 2 \* n + 1)  
plt.ylim(0, 1.5 \* n + 1)  
plt.show()  
  
# 3-й график: добавляем уступку после f2  
plt.figure(figsize=(8, 6))  
plt.plot(x, y, 'k-', linewidth=2, label='ОДЗ')  
plt.plot(x1\_vals, x2\_vals1, 'blue', linewidth=2, label=r'$x\_1 + 2x\_2 = 20$')  
plt.plot(x1\_vals, x2\_vals2, 'orange', linewidth=2, label=r'$2x\_1 + x\_2 = 5$')  
  
# Прямая после уступки f1  
plt.plot(x1\_vals, x2\_vals\_f1, 'magenta', linewidth=2, label=fr'$x\_2 = {f1\_limit:.1f}- x\_1$ (уступка f1)')  
  
# Прямая после уступки f2: -3x1 - x2 = f2\_limit  
x2\_vals\_f2 = 3 \* x1\_vals + f2\_limit  
plt.plot(x1\_vals, x2\_vals\_f2, 'red', linewidth=2, label=fr'$x\_2 = {f2\_limit:.1f}+3x\_1$ (уступка f2)')  
  
plt.plot(opt3[0], opt3[1], 'go', markersize=10, label='Оптимум')  
  
plt.xlabel(r'$x\_1$')  
plt.ylabel(r'$x\_2$')  
plt.title('После уступок f1 и f2 (третий график)')  
plt.grid(True)  
plt.legend()  
plt.gca().set\_aspect('equal', adjustable='box')  
plt.xlim(0, 2 \* n + 1)  
plt.ylim(0, 1.5 \* n + 1)  
plt.show()  
  
# Итоговое решение  
print(f"\nИтоговое оптимальное решение:")  
print(f"x1 = {opt3[0]:.3f}, x2 = {opt3[1]:.3f}")