|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра теоретической и прикладной информатики | | |
| Практическое задание №3 | | |
| по дисциплине «Планирование и анализ экспериментов» | | |
| **Построение дискретных оптимальных планов эксперимента** | | |
|  | | |
|  |  |  |
| Группа ПМ-12 | Вострецова екатерина |
| Вариант 4 | зиянуров артём |
|  | хамитова екатерина |
|  |  |
|  |  |
| Преподаватели | Попов александр александрович |
|  |  |
| Новосибирск,2025 | | |

# **Задание**

* 1. Изучить алгоритмы построения дискретных оптимальных планов.
  2. Разработать программу построения дискретных оптимальных планов эксперимента, реализующую заданный алгоритм.
  3. Для числа наблюдений 20, 25, 30, 35, 40 построить оптимальные планы на каждой из сеток, указанных в варианте задания. Выбрать лучшие дискретные планы для заданного числа наблюдений.
  4. Оформить отчет, включающий в себя постановку задачи, результаты проведенных в п. 3 исследований, текст программы.
  5. Защитить лабораторную работу.

**Вариант 4**

Двухфакторная квадратичная модель на квадрате со сторонами [ 1, +1]. Дискретное множество X – сетки 20 × 20 и 30 × 30 . Строить D- оптимальные планы. Последовательный алгоритм достраивания. Повторные наблюдения допускаются.

**Постановка задачи**

У – значение зависимой переменной,

е – ошибка,

– заданная вектор функция, от независимой переменной х,

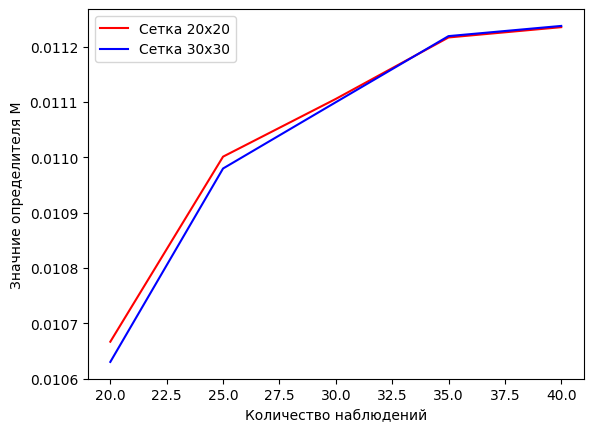
*–* вектор неизвестных параметров

1. **Ход работы**

Значение определителя для каждого плана:

|  |  |  |
| --- | --- | --- |
| **Число набл.\ размер сетки** | **20х20** | **30х30** |
| **20** | 1.0667E-02 | 1.0630E-02 |
| **25** | 1.1001E-02 | 1.0980E-02 |
| **30** | 1.1105E-02 | 1.1099E-02 |
| **35** | 1.1216E-02 | 1.1219E-02 |
| **40** | 1.1235E-02 | 1.1237E-02 |

**Зависимость определителя от сетки:**

****

**Графики планов:**

|  |  |
| --- | --- |
| Сетка 20х20 | Сетка 30х30 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Наилучшими планами стали планы, где количество наблюдение было наибольшим, также сетка 20х20 оказалась более оптимальной и имеет большее значение определителя чем сетка 30х30.

**Протокол решения**

**(для сетки 20х20 и N = 20)**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер  итерации | Добавляемая Точка | | Значение p  в добавляемой  точке |
| 1 | -1.0 | -1.0 | 1.0 |
| 2 | 1.0 | -1.0 | 0.5 |
| 3 | -1.0 | 1.0 | 0.333 |
| 4 | 1.0 | 1.0 | 0.25 |
| 5 | 0.053 | -1.0 | 0.2 |
| 6 | 1.0 | -0.053 | 0.167 |
| 7 | -0.053 | 0.053 | 0.143 |
| 8 | -1.0 | -0.053 | 0.125 |
| 9 | 0.053 | 1.0 | 0.111 |
| 10 | -1.0 | 1.0 | 0.2 |
| 11 | 1.0 | 1.0 | 0.182 |
| 12 | -1.0 | -1.0 | 0.167 |
| 13 | 1.0 | -1.0 | 0.154 |
| 14 | 0.053 | -0.053 | 0.071 |
| 15 | -0.053 | -1.0 | 0.067 |
| 16 | -1.0 | 1.0 | 0.188 |
| 17 | 1.0 | 1.0 | 0.176 |
| 18 | -1.0 | -1.0 | 0.167 |
| 19 | 1.0 | -1.0 | 0.158 |
| 20 | 1.0 | -0.053 | 0.1 |

1. **Текст программы**

#импорт библиотек

import numpy as np

import matplotlib.pyplot as plt

from numpy.linalg import inv

#изначальные параметры

m = 6#количество неизвестных параметров

# для сетки 30х30 изначальное количество точек будет равно 900

N\_list = [20, 25, 30, 35, 40]#для такой размерности нужно делать

grid\_size = [20, 30]#30

n = grid\_size[0]\*grid\_size[0]

s = n#для итераций

p=[0]\*n#наши изначальные пэшки

# наша функция

def func(x):

    return np.array([1, x[0], x[1], x[0]\*x[1], x[0]\*\*2, x[1]\*\*2])

# Информационная матрица

def M\_calc(x, p, m):

    n\_rg = np.count\_nonzero(p)

    # Если n\_rg < m, то матрица вырожденная, мы её регуляризируем

    M = np.zeros((m, m))

    for i in range(n):

        M += p[i] \* np.outer(func(x[i]), func(x[i]))

    # Регуляризация

    if n\_rg < m:

        gamma = 0.1

        M += gamma \* np.eye(m)

    return M

#Дисперсионная матрица

def D\_calc(M):

    return np.linalg.inv(M)

def X\_calc(grid\_size):

    #создаем по 20(30) точек в этом отрезке

    x1 = np.linspace(-1,1,grid\_size)

    x2 = np.linspace(-1,1,grid\_size)

    #для создания в целом квадрата точек

    X1, X2 = np.meshgrid(x1, x2)#матрицы, где каждая строка копия x1(x2)

    X = np.column\_stack([X1.ravel(), X2.ravel()])#преобразе матрицы в вектора, и выводим координаты

    return X

def d\_calc(x, D):

    return func(x) @ D @ func(x).T

#изначальные матрицы и х

x = X\_calc(grid\_size[0])

M = M\_calc(x, p, m)

D = D\_calc(M)

len(x)

determinats = [[], []]

p = {20 : [],

     30 : []}

for grid in range(2):

    x = X\_calc(grid\_size[grid])

    n = grid\_size[grid]\*grid\_size[grid]

    for j in range(len(N\_list)):

        #тут начинается алгоритм

        CurP = np.zeros(n)

        print(f"Протокол решения к плану с сеткой {grid\_size[grid]}x{grid\_size[grid]} и c количеством наблюдений {N\_list[j]}")

        print("i (индекс) -> x[i] (добавленной) -> p[i] (изменёный у x) -> max (f M f.T)")

        for s in range(N\_list[j]):

            M = M\_calc(x, CurP, m)

            D = D\_calc(M)

            maxD = 0

            i = -1

            for i in range(len(x)):

                if (s < m and CurP[i] == 0) or (s >= m):

                    cur\_d = d\_calc(x[i], D)

                    if maxD < cur\_d:

                        maxD = cur\_d

                        ind = i

            # перераспределение весов

            for k in range(n):

                if CurP[k] != 0:

                    if k == ind:

                        CurP[k] = (CurP[k] \* s + 1) / (s + 1)

                    else:

                        CurP[k] = CurP[k] \* s / (s + 1)

                else:

                    if k == ind:

                        CurP[k] += 1 / (s + 1)

            print(ind, '\t', \*[round(i, 3) for i in x[ind]],'\t', round(CurP[ind], 3), "\t", round(maxD, 3))

        print('\n\n\n\n ')

        p[grid\_size[grid]].append(CurP)

        determinats[grid].append(np.linalg.det(M))

def printGrid(p, N, j, N\_list):

    x = np.arange(-1, 1, 2/N)

    y = np.arange(-1, 1, 2/N)

    X, Y = np.meshgrid(x, y)

    sizes = []

    for i in range(len(p)):

        if p[i] > 0:

            sizes.append(100)

        else:

            sizes.append(0)

    plt.scatter(X, Y, s=sizes, c='blue')

    plt.axhline(0, color='black', linewidth=0.5)

    plt.axvline(0, color='black', linewidth=0.5)

    plt.xticks([-1, -0.5, 0, 0.5, 1])

    plt.yticks([-1, -0.5, 0, 0.5, 1])

    plt.grid(True)

    titl = f'Полученный план сетки {N}x{N} для {N\_list[j]} наблюдений'

    plt.title(titl)

    plt.show()

plt.plot(N\_list, determinats[0], color='r', label='Сетка 20x20')

plt.plot(N\_list, determinats[1], color='b', label='Сетка 30x30')

plt.xlabel('Количество наблюдений')

plt.ylabel('Значние определителя M')

plt.legend()

plt.show()