

Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра теоретической и прикладной информатики

Лабораторная работа № 3

по дисциплине «Математические методы оптимального планирования
эксперимента»

ПОСТРОЕНИЕ ДИСКРЕТНЫХ ОПТИМАЛЬНЫХ ПЛАНОВ ЭКСПЕРИМЕНТА

Факультет:	ПМИ
Группа:	ПМИМ-21
Вариант:	3
Студенты:	Демидович Е. Стародубцев С. Цыганков А.
Преподаватель:	Попов А.А.

Новосибирск

2022

1. Цель работы

Изучить алгоритмы, используемые при построении дискретных оптимальных планов.

2. Содержание работы

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 отчёт, включающий в себя постановку задачи, результаты проведённых в п. 3 исследований, текст программы.
5. Защитить лабораторную работу.

3. Постановка задачи

Двухфакторная квадратичная модель на квадрате со сторонами $[-1, +1]$. Дискретное множество X – сетки 10×10 и 20×20 . Строить D-оптимальные планы. Алгоритм Митчелла. Повторные наблюдения допускаются.

Пусть область действия факторов представляет собой дискретное множество точек \tilde{X} . Задача построения Ψ -оптимального плана ε_N^* с N наблюдениями имеет вид:

$$\varepsilon_N^* = \operatorname{Arg} \max_{\varepsilon_N} \Psi[M(\varepsilon_N)] \text{ с } p_i = 1/N, i = \overline{1, N}$$

Алгоритм Митчелла

1. Выбирается невырожденный начальный план $\varepsilon_N^0, s = 0$.
2. Выбирается точка x^s , не принадлежащая плану ε_N^s , по правилу:
$$x^s = \arg \max_x d(x, \varepsilon_N^s), \quad \text{где } d(x, \varepsilon) = f^T(x)M^{-1}(\varepsilon)f(x).$$
3. Точка x^s добавляется в план ε_N^s . В результате формируется план ε_{N+1}^s , состоящий из $N+1$ точек.
4. Выбирается точка x_j^s , принадлежащая плану ε_{N+1}^s , по правилу:

$$x_j^s = \arg \min_{x \in \varepsilon_{N+1}^s} d(x, \varepsilon_{N+1}^s)$$

5. Точка x_j^s исключается из плана ε_{N+1}^s и формируется план ε_N^{s+1} .
6. Если точка x^s , выбранная на шаге 2, совпадает с точкой x_j^s , выбранной на шаге 4, то вычисления прекращаются, в противном случае s заменяется на $s+1$ и осуществляется переход на шаг 2.

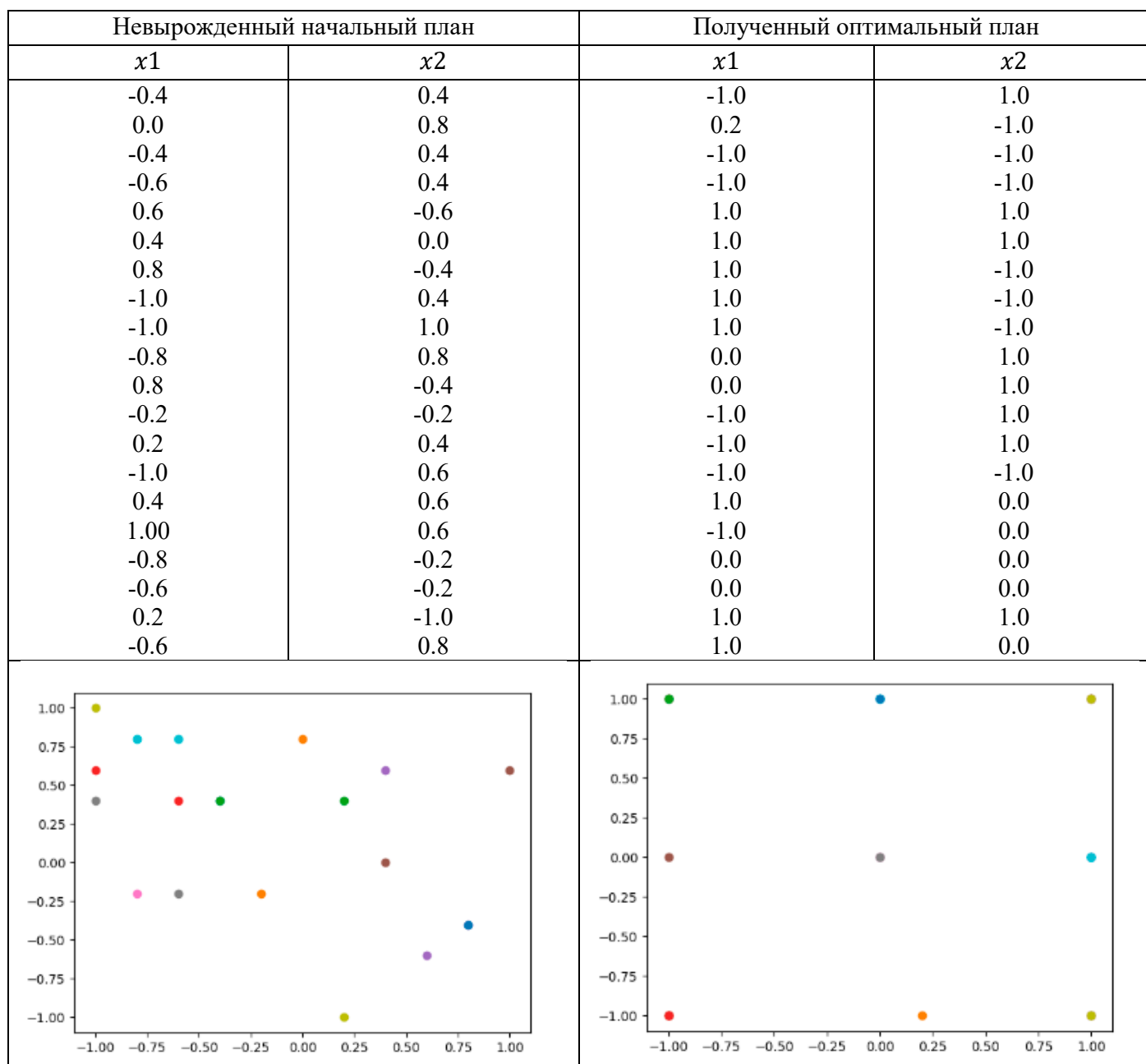
4. Ход работы

1. $f(x) = [1, x_1, x_2, x_1 * x_2, x_1^2, x_2^2]$
2. Моделирование начальных планов на сетках 10x10 и 20x20 с количеством экспериментов $N = 20, 25, 30, 35, 40$:
 - а. 10x10: $[-1;+1]$, $\text{delta} = +0.2$;
 - б. 20x20: $[-1;+1]$, $\text{delta} = +0.1$.
3. Получение оптимального плана с помощью алгоритма Митчелла.
4. Нахождение максимального функционала для каждой сетки.

5. Исследования

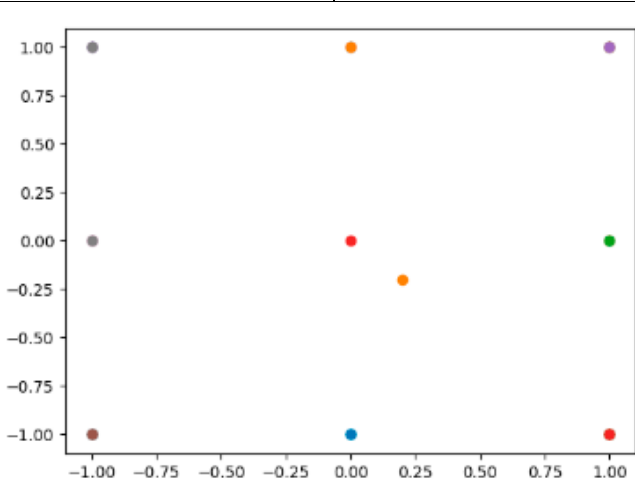
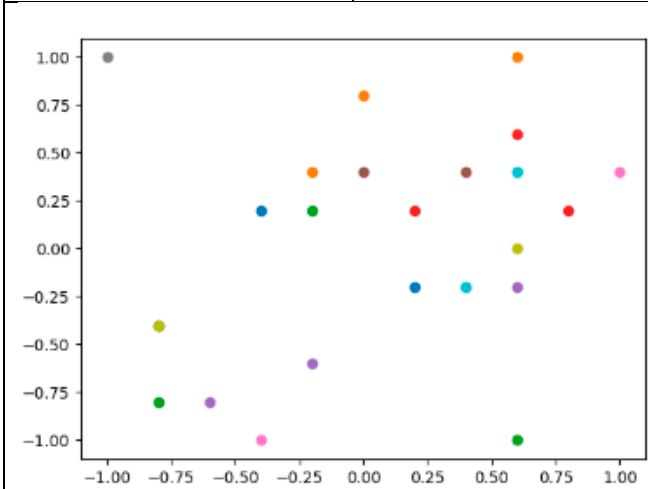
Сетка 10 x 10:

$N = 20, p = 0.05$



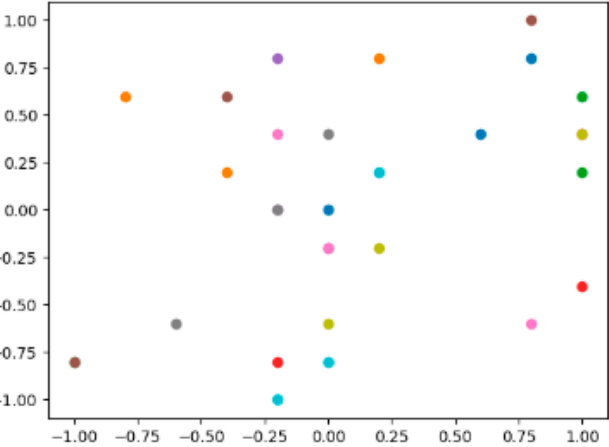
$N = 25, p = 0.04$

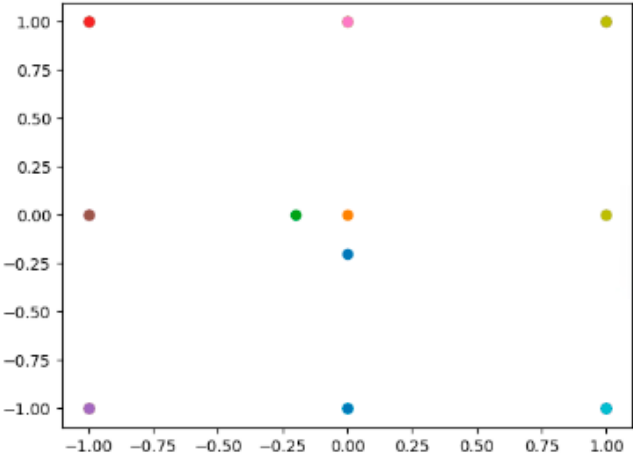
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
-0.8	-0.4	-1.0	1.0
-0.2	0.4	0.2	-0.2
-0.2	0.2	1.0	-1.0
0.6	0.6	1.0	-1.0
-0.6	-0.8	1.0	1.0
0.0	0.4	1.0	1.0
1.0	0.4	-1.0	1.0
-0.8	-0.4	-1.0	1.0
0.6	0.0	-1.0	-1.0
0.4	-0.2	-1.0	-1.0
-0.4	0.2	0.0	1.0
0.0	0.8	0.0	1.0
-0.8	-0.8	1.0	-1.0
0.8	0.2	1.0	-1.0
-0.2	-0.6	-1.0	-1.0
0.4	0.4	-1.0	-1.0
-0.4	-1.0	-1.0	0.0
-1.0	1.0	-1.0	0.0
-0.8	-0.4	1.0	1.0
0.6	0.4	0.0	-1.0
0.2	-0.2	0.0	-1.0
0.6	1.0	1.0	0.0
0.6	-1.0	1.0	0.0
0.2	0.2	0.0	0.0
0.6	-0.2	1.0	1.0



$N = 30, p = 0.0(3)$

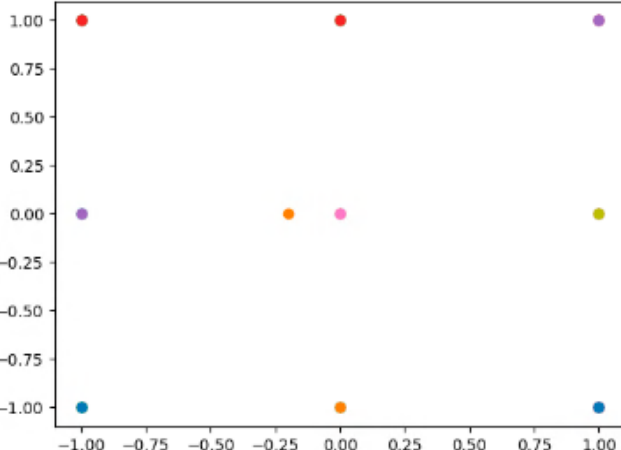
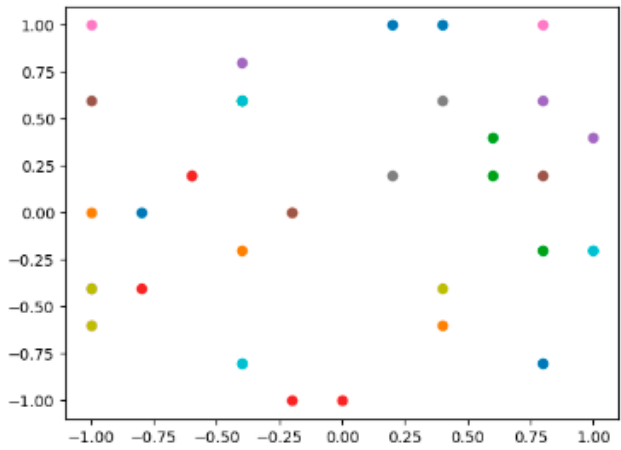
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
0.6	0.4	0.0	-0.2
-0.8	0.6	0.0	0.0
-1.0	-0.8	-0.2	0.0
-0.2	-0.8	-1.0	1.0
0.0	-0.2	-1.0	1.0
-1.0	-0.8	1.0	-1.0
0.8	-0.6	1.0	-1.0
-0.6	-0.6	1.0	-1.0
0.0	-0.6	-1.0	-1.0
0.0	-0.8	-1.0	-1.0
0.0	0.0	1.0	1.0
-0.4	0.2	1.0	1.0
1.0	0.2	-1.0	1.0
1.0	-0.4	-1.0	1.0
-0.2	0.8	-1.0	0.0
0.8	1.0	-1.0	0.0
-0.2	0.4	1.0	-1.0
0.0	0.4	1.0	1.0
0.2	-0.2	1.0	1.0
0.2	0.2	0.0	-1.0
0.8	0.8	0.0	-1.0
0.2	0.8	0.0	1.0
1.0	0.6	0.0	1.0
1.0	0.4	-1.0	-1.0
-0.2	0.0	-1.0	-1.0
-0.4	0.6	-1.0	0.0
0.0	-0.2	0.0	1.0
-0.2	0.0	1.0	0.0
1.0	0.4	1.0	0.0
-0.2	-1.0	1.0	-1.0





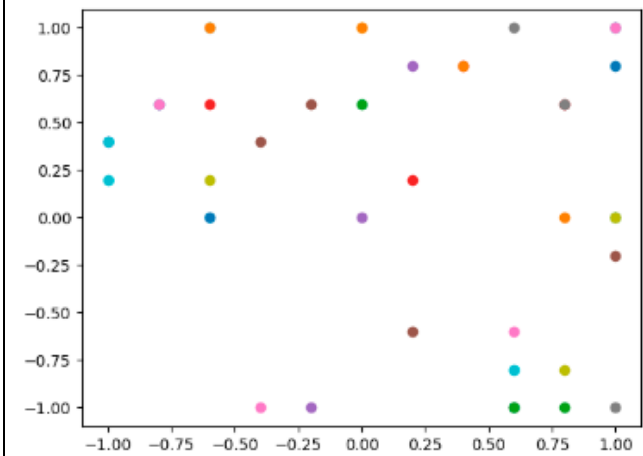
$N = 30, p = 0.02857142857142857$

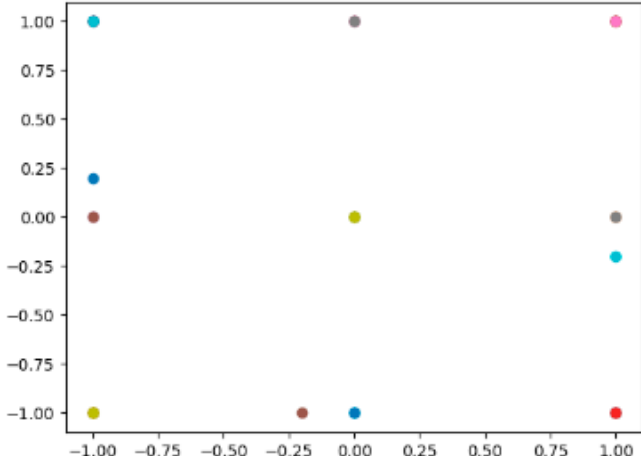
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
0.4	1.0	-1.0	1.0
0.4	-0.6	-0.2	0.0
-0.4	0.6	0.0	-1.0
-0.2	-1.0	-1.0	0.0
-1.0	-0.4	1.0	-1.0
0.8	0.2	1.0	-1.0
-1.0	1.0	1.0	1.0
0.2	0.2	1.0	1.0
0.4	-0.4	-1.0	-1.0
-0.4	-0.8	-1.0	-1.0
0.2	1.0	-1.0	1.0
-0.2	0.0	-1.0	1.0
0.6	0.2	-1.0	1.0
0.0	-1.0	1.0	-1.0
-0.4	0.8	1.0	-1.0
-0.2	0.0	1.0	1.0
-1.0	-0.6	1.0	1.0
-0.6	0.2	-1.0	-1.0
-1.0	-0.6	-1.0	-1.0
1.0	-0.2	0.0	-1.0
-0.8	0.0	1.0	-1.0
-0.4	-0.2	1.0	0.0
0.8	-0.2	0.0	1.0
-0.6	0.2	-1.0	1.0
0.8	0.6	1.0	1.0
-1.0	0.6	0.0	0.0
0.8	1.0	0.0	0.0
0.4	0.6	1.0	0.0
-1.0	-0.4	1.0	0.0
-0.4	0.6	-1.0	0.0
0.8	-0.8	-1.0	-1.0
-1.0	0.0	0.0	-1.0
0.6	0.4	0.0	1.0
-0.8	-0.4	0.0	1.0
1.0	0.4	-1.0	0.0



$N = 40, p = 0.025$

Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
1.0	1.0	1.0	1.0
0.8	0.0	1.0	0.0
1.0	0.0	1.0	-1.0
0.8	0.6	0.0	1.0
-1.0	0.4	0.0	1.0
-0.2	0.6	-0.2	-1.0
-0.4	-1.0	1.0	1.0
1.0	-1.0	1.0	0.0
0.0	1.0	0.0	0.0
-0.6	1.0	1.0	-0.2
-0.6	0.0	-1.0	0.2
0.0	1.0	-1.0	-1.0
0.8	-1.0	-1.0	-1.0
0.4	0.8	-1.0	-1.0
-0.2	-1.0	-1.0	1.0
0.2	-0.6	-1.0	1.0
1.0	1.0	-1.0	1.0
0.6	1.0	-1.0	-1.0
1.0	0.0	-1.0	-1.0
0.6	-0.8	1.0	-1.0
1.0	0.8	1.0	-1.0
-0.6	1.0	1.0	1.0
0.0	0.6	1.0	1.0
0.2	0.2	1.0	1.0
0.0	0.0	-1.0	1.0
-0.4	0.4	-1.0	1.0
0.6	-0.6	1.0	-1.0
0.6	-1.0	1.0	-1.0
-0.6	0.2	-1.0	-1.0
-1.0	0.4	0.0	-1.0

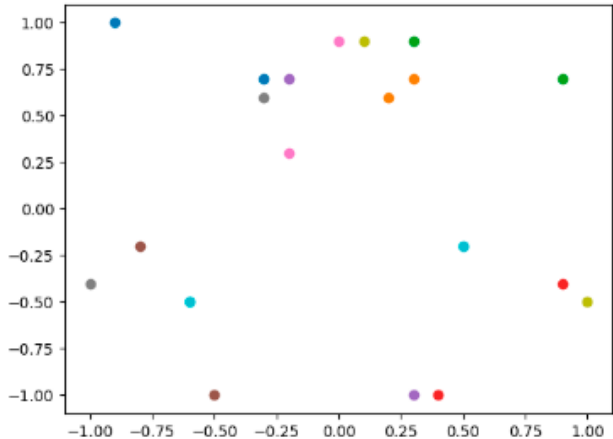


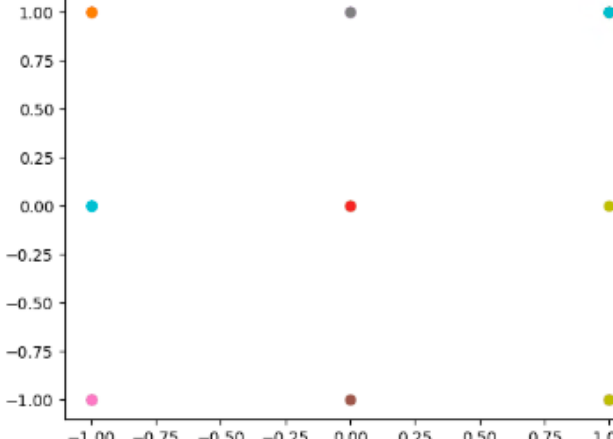


Сетка 20 x 20:

$N = 20, p = 0.05$

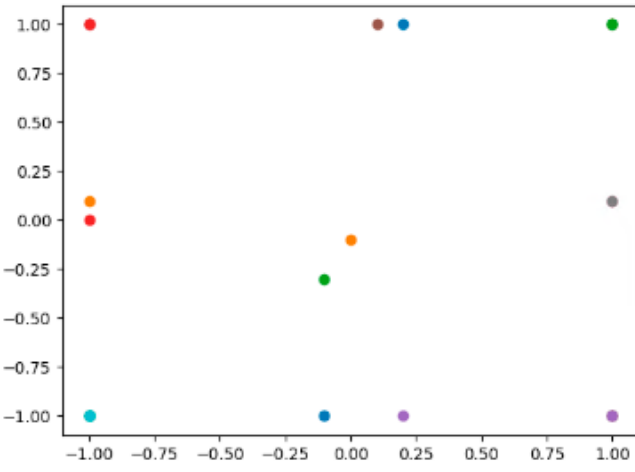
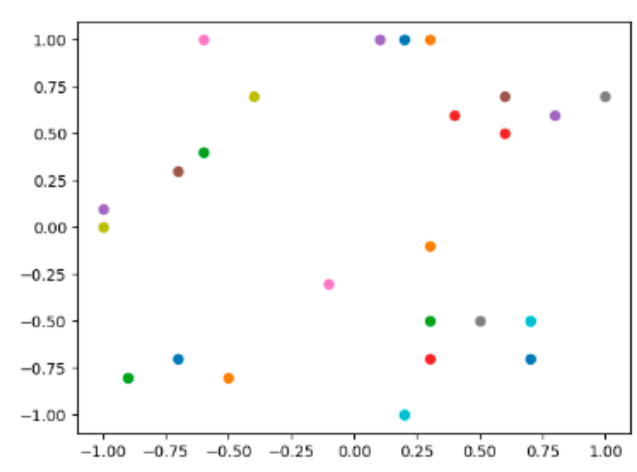
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
-0.9	1.0	1.0	1.0
0.3	0.7	1.0	1.0
0.9	0.7	-1.0	1.0
0.9	-0.4	-1.0	1.0
0.3	-1.0	-1.0	-1.0
-0.5	-1.0	-1.0	-1.0
0.0	0.9	1.0	-1.0
-1.0	-0.4	1.0	-1.0
1.0	-0.5	1.0	-1.0
-0.6	-0.5	1.0	1.0
-0.3	0.7	-1.0	0.0
0.2	0.6	-1.0	1.0
0.3	0.9	0.0	0.0
0.4	-1.0	0.0	0.0
-0.2	0.7	0.0	1.0
-0.8	-0.2	0.0	-1.0
-0.2	0.3	-1.0	-1.0
-0.3	0.6	0.0	1.0





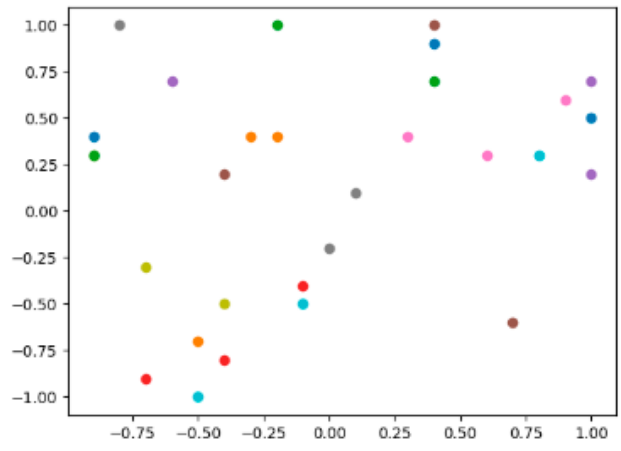
$N = 25, p = 0.04$

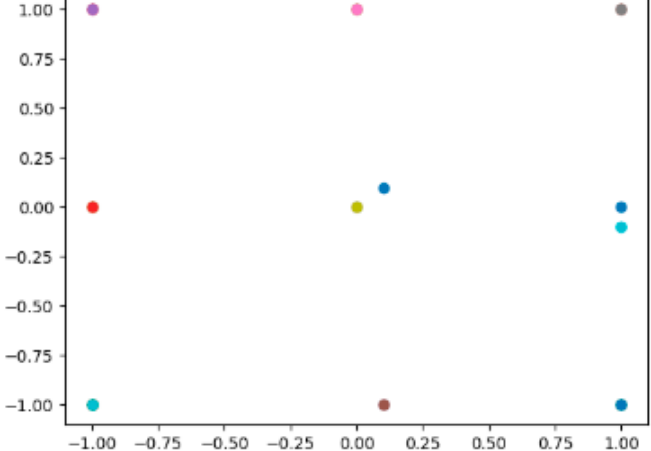
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
-0.7	-0.7	0.2	1.0
0.3	-0.1	-1.0	0.1
-0.9	-0.8	-1.0	-0.3
0.3	-0.7	-1.0	0.0
0.8	0.6	0.2	-1.0
0.6	0.7	0.1	1.0
-0.6	1.0	1.0	-1.0
1.0	0.7	1.0	-1.0
-0.4	0.7	-1.0	1.0
0.7	-0.5	-1.0	-1.0
0.2	1.0	-1.0	-1.0
-0.5	-0.8	1.0	1.0
0.3	-0.5	1.0	1.0
0.6	0.5	-1.0	1.0
-1.0	0.1	-1.0	1.0
-0.7	0.3	1.0	-1.0
-0.1	-0.3	1.0	0.1
0.5	-0.5	1.0	0.1
-1.0	0.0	-1.0	-1.0
0.2	-1.0	-1.0	-1.0
0.7	-0.7	-0.1	-1.0
0.3	1.0	0.0	-0.1
-0.6	0.4	-1.0	1.0
0.4	0.6	-1.0	1.0
0.1	1.0	1.0	-1.0



$N = 30, p = 0.0(3)$

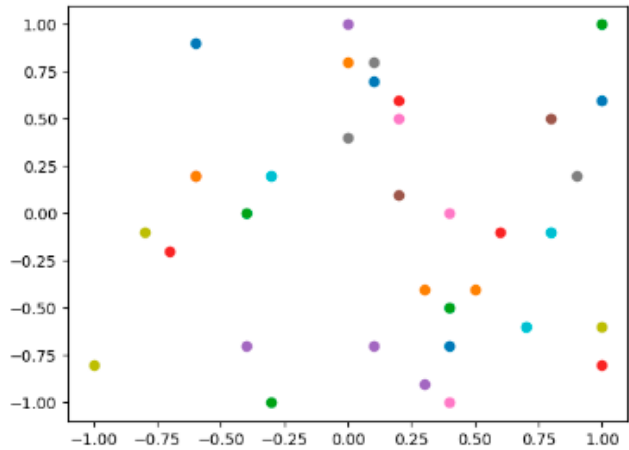
Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
1.0	0.5	1.0	0.1
-0.3	0.4	1.0	-1.0
-0.2	1.0	1.0	-1.0
-0.4	-0.8	-1.0	-1.0
1.0	0.2	-1.0	-1.0
-0.4	0.2	-1.0	1.0
0.9	0.6	-1.0	1.0
-0.8	1.0	1.0	1.0
-0.9	0.3	1.0	1.0
0.8	0.3	1.0	-1.0
0.4	0.9	1.0	-1.0
-0.5	-0.7	-1.0	-1.0
-0.9	0.3	-1.0	-1.0
-0.1	-0.4	-1.0	1.0
1.0	0.7	1.0	-1.0
0.7	-0.6	1.0	-1.0
0.6	0.3	1.0	1.0
0.0	-0.2	1.0	1.0
-0.4	-0.5	0.0	1.0
-0.1	-0.5	1.0	-0.1
-0.9	0.4	1.0	0.0
-0.2	0.4	-1.0	0.0
0.4	0.7	-1.0	0.0
-0.7	-0.9	-1.0	0.0
-0.6	0.7	-1.0	1.0
0.4	1.0	0.0	1.0
0.3	0.4	0.0	1.0
0.1	0.1	0.0	0.0
-0.7	-0.3	0.0	0.0
-0.5	-1.0	-1.0	-1.0

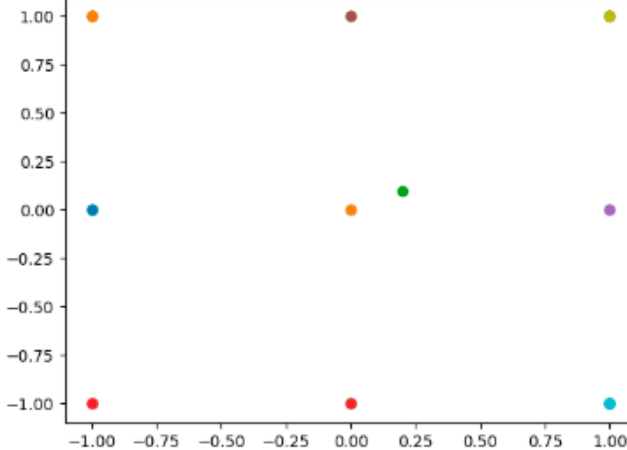




$N = 35, p = 0.028571428571428570$

Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
1.0	0.6	1.0	1.0
0.3	-0.4	0.0	1.0
1.0	1.0	0.2	0.1
0.6	-0.1	-1.0	1.0
0.3	-0.9	-1.0	1.0
0.8	0.5	-1.0	-1.0
0.4	-1.0	-1.0	-1.0
0.1	0.8	1.0	1.0
-1.0	-0.8	1.0	1.0
0.8	-0.1	1.0	-1.0
0.4	-0.7	1.0	-1.0
0.5	-0.4	-1.0	1.0
0.4	-0.5	-1.0	1.0
0.2	0.6	-1.0	-1.0
-0.4	-0.7	-1.0	-1.0
-0.6	0.2	1.0	1.0
0.4	0.0	1.0	-1.0
0.0	0.4	1.0	-1.0
1.0	-0.6	-1.0	0.0
-0.3	0.2	-1.0	0.0
-0.6	0.9	-1.0	0.0
0.0	0.8	-1.0	1.0
-0.3	-1.0	0.0	-1.0
1.0	-0.8	-1.0	-1.0
0.0	1.0	0.0	1.0
0.2	0.1	0.0	1.0
0.2	0.5	1.0	0.0
0.9	0.2	1.0	0.0
-0.8	-0.1	1.0	1.0
0.7	-0.6	1.0	-1.0
0.1	0.7	0.0	0.0





$N = 40, p = 0.025$

Невырожденный начальный план		Полученный оптимальный план	
x_1	x_2	x_1	x_2
-0.8	-0.3	-1.0	1.0
-0.3	0.5	-1.0	1.0
0.3	0.5	1.0	-1.0
-0.9	0.4	1.0	1.0
0.6	0.1	1.0	1.0
-0.4	-1.0	-1.0	-1.0
-0.6	0.1	-1.0	-1.0
0.9	-0.6	1.0	-1.0
-0.9	0.2	1.0	-1.0
-0.1	-0.1	-1.0	1.0
-0.1	0.7	-1.0	1.0
0.5	0.3	1.0	1.0
0.9	0.1	1.0	1.0
1.0	-0.2	-1.0	-1.0
0.8	-0.6	-1.0	-1.0
1.0	-0.2	1.0	-1.0
0.9	-0.2	1.0	-1.0
0.7	0.2	-1.0	1.0
-0.9	-0.8	-1.0	1.0
-0.4	0.7	0.0	-1.0
0.0	-0.3	0.0	-1.0
0.2	0.6	-1.0	-1.0
-0.6	-0.8	-1.0	-1.0
1.0	0.6	-1.0	1.0
0.2	-0.9	-1.0	0.0
-0.7	0.1	-1.0	0.0
0.4	-0.4	0.0	1.0
0.8	-0.1	0.0	1.0
0.0	0.9	0.0	0.0
-0.7	-0.2	0.0	0.0
0.3	-0.6	1.0	0.0
0.6	0.7	1.0	0.0
0.3	0.5	1.0	0.0
0.8	-0.6	0.0	-1.0
-0.1	0.7	-1.0	0.0
-1.0	-0.4	0.0	1.0
1.0	0.8	0.0	0.0
-0.2	-0.6	0.0	0.0
-0.9	-0.2	1.0	1.0
0.0	0.9	1.0	-1.0

Сравнение определителей матрицы M и функционалов D-плана (max):

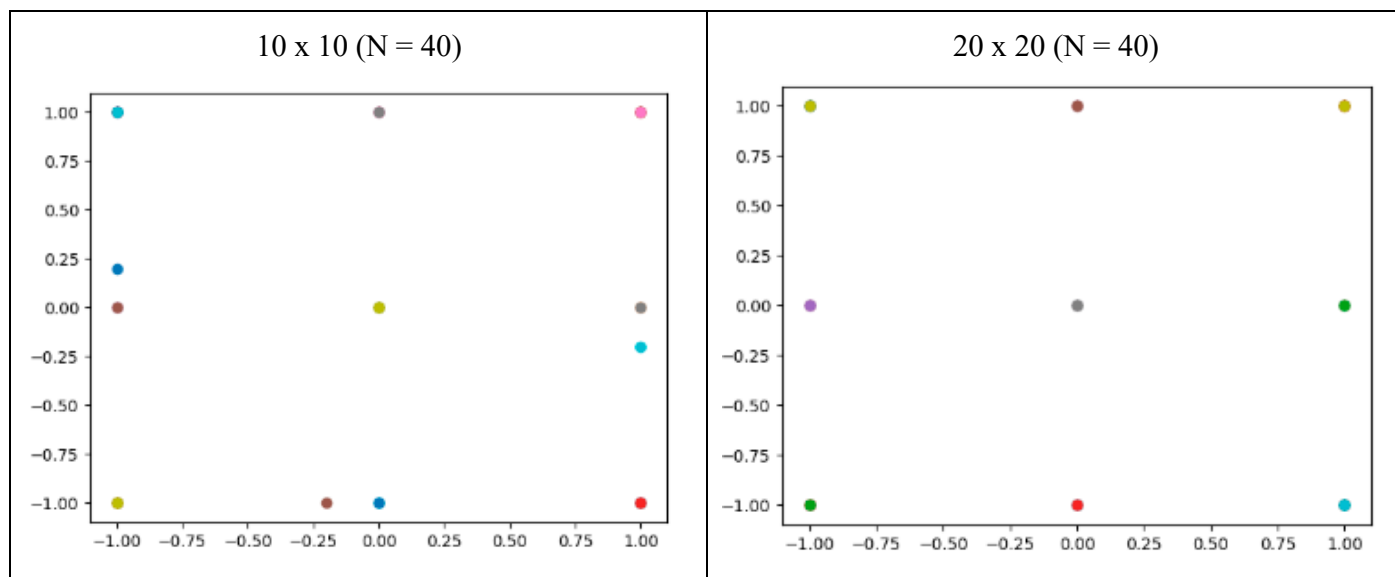
10 x 10

N	s	M(e)	ln M(e)
20	23	0.010088104000000004	-4.596398371094304
25	24	0.01089671790592002	-4.519293644638244
30	28	0.010872987459312046	-4.521473780346014
35	39	0.01104201914454014	-4.506047361328983
40	34	0.01117884664976641	-4.493731978493877

20 x 20

N	s	M(e)	ln M(e)
20	31	0.011021999999999995	-4.507862003521094
25	22	0.010051071990671515	-4.600075984426979
30	30	0.01097825020484668	-4.511839217636052
35	37	0.01129118723350984	-4.483732748376751
40	48	0.011390625000000027	-4.474964630333148

Максимального функционала мы достигли при N = 40 на обеих сетках.



6. Исходный код

```
import sys

import numpy as np
import matplotlib.pyplot as plt

# определение f(x)
def f(a, b):
    f = np.array([[1], [a], [b], [a * b], [a ** 2], [b ** 2]])
    return f

# построение информационной матрицы M
def calc_M(c, d, pl, m):
    M1 = np.zeros((m, m))
    n = len(c)
    for q in range(0, n - 1):
        M1 += pl[q] * f(c[q], d[q]) @ np.transpose(f(c[q], d[q]))
    return M1

# построение дисперсионной матрицы D
def calc_D(M1):
    D1 = np.linalg.inv(M1)
    return D1

# поиск d(x, e)
def calc_d(x1, x2, D1):
    d1 = np.transpose(f(x1, x2)) @ D1 @ f(x1, x2)
    return d1

# отрисовка графика
def draw_graph(a, b):
    for i1 in range(0, len(a)):
        plt.scatter(a[i1], b[i1])
    plt.plot()
    plt.show()

# параметры генератора сетки
plan = []
grid = 20 # 10 20
grid_step = {
    10: 0.2,
    20: 0.1
}

N = int(sys.argv[1])
m = 6
a = -1
plan.append(a)
for i in range(0, grid):
    a += grid_step[grid]
    plan.append(a)

# выбор невырожденного плана
s = 0
en_x1 = np.random.choice(plan, N, replace=True, p=None)
en_x2 = np.random.choice(plan, N, replace=True, p=None)
print("en_x1 = ", en_x1)
print("en_x2 = ", en_x2)
draw_graph(en_x1, en_x2)

while True:
    print("s = ", s)
    print("plan = ", plan)
    print("en_x1 = ", en_x1)
    print("en_x2 = ", en_x2)
```

```

pi = 1 / len(en_x1)
p = []
for i in range(0, len(en_x1)):
    p.append(pi)
print("p = ", p)
print("sum(p) = ", np.sum(p))
# выбор точки x_s

M = calc_M(en_x1, en_x2, p, m)
D = calc_D(M)
delta = grid_step[grid]
maxd = -100
k = 0
x1_max = -2
x2_max = -2
xs_1 = -1
while (xs_1 <= 1):
    xs_2 = -1
    while (xs_2 <= 1):
        for i in range(0, len(en_x1)):
            d = calc_d(xs_1, xs_2, D)
            if maxd <= d:
                maxd = d
                x1_max = xs_1
                x2_max = xs_2
            xs_2 += delta
        xs_1 += delta

print("maxd = ", maxd)
print("x1_max = ", x1_max)
print("x2_max = ", x2_max)

# точка x_s добавляется в план
en1_x1 = np.append(en_x1, x1_max)
en1_x2 = np.append(en_x2, x2_max)

# выбор точки x_j из плана
new_pi = 1 / len(en1_x1)
new_p = []
for i in range(0, len(en1_x1)):
    new_p.append(pi)

M = calc_M(en1_x1, en1_x2, new_p, m)
D = calc_D(M)
mind = 100
x1_min = 2
x2_min = 2
i_min = 0
for i in range(0, len(en1_x1)):
    d = calc_d(en1_x1[i], en1_x2[i], D)
    print("d = ", d)
    print("en1_x1[i] = ", en1_x1[i])
    print("en1_x2[i] = ", en1_x2[i])
    if mind >= d:
        mind = d
        x1_min = en1_x1[i]
        x2_min = en1_x2[i]
        i_min = i

print("mind = ", mind)
print("x1_min = ", x1_min)
print("x2_min = ", x2_min)

# точка x_j исключается из плана en1
es1_x1 = np.delete(en1_x1, i_min)
es1_x2 = np.delete(en1_x2, i_min)

```

```

print("xs = ", x1_max, ", ", x2_max)
print("xj = ", x1_min, ", ", x2_min)

if (x1_max == x1_min) and (x2_max == x2_min):
    print("alg opt")
    print("|M(e)| = ", np.linalg.det(M), "ln|M(e)| = ", np.log(np.linalg.det(M)))
    print("es1_x1 = ", es1_x1)
    print("es1_x2 = ", es1_x2)
    draw_graph(es1_x1, es1_x2)
    break
else:
    print("alg not opt")
    print("|M(e)| = ", np.linalg.det(M), "ln|M(e)| = ", np.log(np.linalg.det(M)))
    s += 1
    en_x1 = es1_x1
    en_x2 = es1_x2

```