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 № 1

по дисциплине «Математические методы оптимального планирования эксперимента»

КРИТЕРИИ ОПТИМАЛЬНОСТИ 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 понятие оптимального плана эксперимента, критерии оптимальности планов, свойства информационной матрицы.

2. Содержание работы

1. Изучить понятия непрерывного плана эксперимента и информационной матрицы, а также критерии оптимальности, связанные с точностью оценивания параметров модели и точностью оценивания математического ожидания функции отклика.
2. Разработать программное приложение по обработке различных планов эксперимента для регрессионных моделей. Программа должна иметь возможность обрабатывать несколько различных планов для одной и той же модели. Обработка заключается в вычислении различных функционалов от информационной матрицы, связанных с тем или иным критерием оптимальности.
3. Для каждого из заданных планов вычислить значения функционалов от информационной (дисперсионной) матриц, связанных с такими критериями как: D- , A-, E-, Ф2-, Λ-, MV-, G-оптимальности. Проранжировать планы, указанные в варианте, с позиций различных критериев. Выбрать план, наиболее предпочтительный по совокупности критериев. Список планов приведен в табл. 1.
4. В качестве спектра плана выбрать один из приведенных в табл. 1 для соответствующей модели. Веса точек выразить в виде зависимости от одного параметра как в примере аналитического построения оптимального плана. Для этого параметра определить допустимые интервалы значений, руководствуясь тем, что веса точек должны быть неотрицательные, а число таких точек с ненулевыми весами должно быть не меньше числа параметров в модели. Построить графики изменения критерия оптимальности плана, указанного в варианте, в зависимости от этого скалярного параметра; определить по графику оптимальные значения параметра и критерия. Сравнить полученный результат с результатами из п. 3.
5. Оформить отчет, включающий в себя постановку задачи, полученные результаты и текст программы.
6. Защитить лабораторную работу.

3. Постановка задачи

Исследуемая модель имеет вид:

$y = f^T(x)\theta + e = \sum_{l=1}^{2} f_l(x)\,\theta_l + e$, где y - значение зависимой переменной,

$f^T(x) = \big(f_1(x), f_2(x)\big) = (1,\ x^2)$ - заданная вектор функция от независимой переменной x, которая может изменяться на отрезке [-1, 1],

$\theta = (\theta_1, \theta_2, \theta_3)^T$ - вектор неизвестных параметров, которые необходимо определить по результатам экспериментов (измерений),

e - ошибка, распределённая по нормальному закону.

<u>Критерии оптимальности:</u>

1. D: $\quad \varepsilon^* = \arg\max_{\varepsilon} |M(\varepsilon)|$
2. A: $\quad \varepsilon^* = \arg\min_{\varepsilon} tr(D(\varepsilon))$

3. E: $\varepsilon^* = \arg\min_{\varepsilon} \max_{i} \lambda_i(D(\varepsilon))$

4. Φ_p: $\varepsilon^* = Arg\min_{\varepsilon} \Phi_p(\varepsilon) = Arg\min_{\varepsilon} (m^{-1} trD^p(\varepsilon))^{\frac{1}{p}}$,

5. Λ: $\varepsilon^* = \arg\min_{\varepsilon} \sum_{i=1}^{m} \left[\lambda_i(D(\varepsilon)) - \overline{\lambda}(D(\varepsilon)) \right]^2$

6. MV: $\varepsilon^* = \arg\min_{\varepsilon} \max_{i} D_{ii}(\varepsilon))$

7. G: $\varepsilon^* = Arg\min_{\varepsilon} \max_{x \in X} d(x,\varepsilon)$, где $d(x,\varepsilon) = f^T(x) M^{-1} f(x)$

4. Ход решения

Исследуемые планы:

№	x_1/p_1	x_2/p_2	x_3/p_3
1	−1 0.2	0 0.6	1 0.2
2	−1 0.25	0 0.5	1 0.25
3	−1 0.1884	0 0.6233	1 0.1884
4	−1 0.333	0 0.333	1 0.333

Информационные и дисперсионные матрицы:

1 план:

```
Информационная матрица Фишера:
[[1.  0.4]
 [0.4 0.4]]

Дисперсионная матрица:
[[ 1.66666667 -1.66666667]
 [-1.66666667  4.16666667]]
```

2 план:

```
Информационная матрица Фишера:
[[1.  0.5]
 [0.5 0.5]]

Дисперсионная матрица:
[[ 2. -2.]
 [-2.  4.]]
```

3 план:

```
Информационная матрица Фишера:
[[1.0001 0.3768]
 [0.3768 0.3768]]

Дисперсионная матрица:
[[ 1.60436387 -1.60436387]
 [-1.60436387  4.25829168]]
```

4 план:

```
Информационная матрица Фишера:
[[0.999 0.666]
 [0.666 0.666]]

Дисперсионная матрица:
[[ 3.003003  -3.003003 ]
 [-3.003003   4.5045045]]
```

Критерии оптимальности для исследуемых планов:

№ плана	D	A	E	Фр	Λ	MV	G
1	4.166667	5.833333	5.000000	10.069444	8.680556	4.166667	2.500000
2	4.000000	6.000000	5.236068	10.000000	10.000000	4.000000	2.000000
3	4.257866	5.862656	5.013350	10.353516	8.669633	4.258292	2.653928
4	4.509014	7.507508	6.849178	14.654294	19.163307	4.504505	3.003003

Наиболее оптимальный план - №2

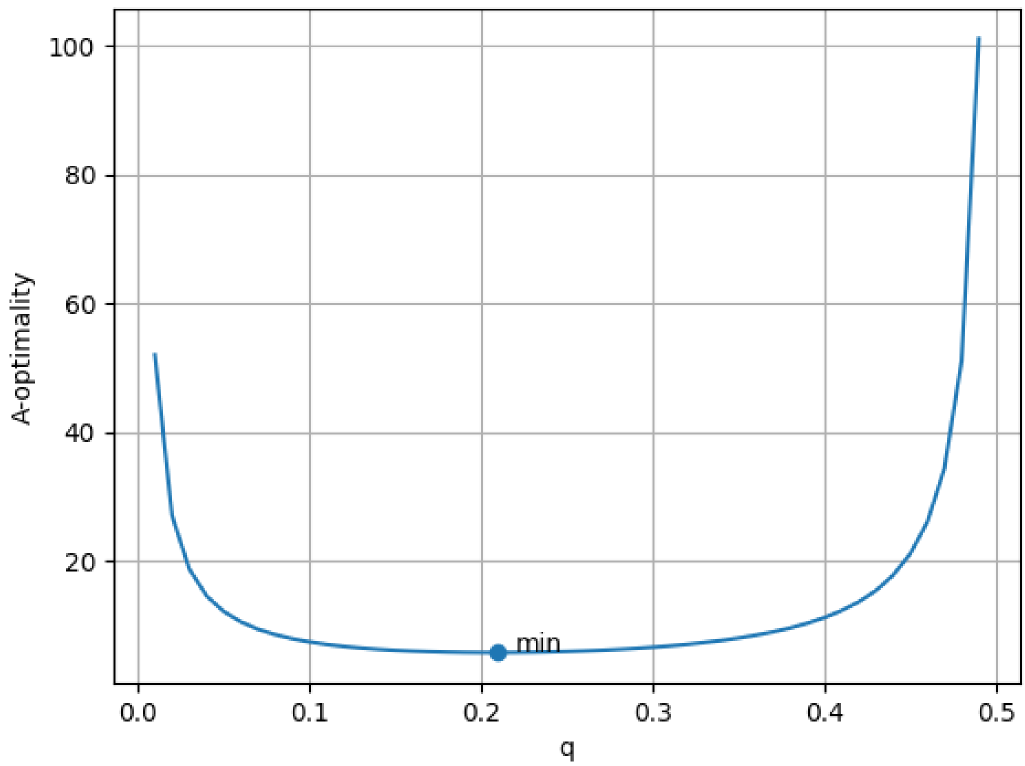
Аналитическое построение оптимального плана:

Спектр плана: $x = (-1, -0, 1)$

Предположим, что оптимальный план принадлежит классу планов следующего вида:

$$q = \begin{Bmatrix} -1 & 0 & 1 \\ q & 1-2q & q \end{Bmatrix}, \text{где } q \in (0; 0.5)$$

График изменения критерия A-оптимальности:



Оптимальные значения, найденные по графику:

- q = 0.210000
- A(ε) = 5.829228

5. Текст программы

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

class PlanElement(object):
    def __init__(self, x, p):
        self.x = x
        self.p = p

class Model(object):
    def __init__(self, q):
        self.plan = np.array([PlanElement(-1, q), PlanElement(0, 1-2*q), PlanElement(1,
q)])
        self.a_optimality(self.plan)

    def a_optimality(self, plan):
        M = np.zeros((2, 2))
        for k in range(len(plan)):
            f = np.array([1, self.plan[k].x ** 2])
            for i in range(2):
                M[i] += self.plan[k].p * f[i] * f
        self.A = np.linalg.inv(M).trace()

# Перемножение вектор функции
def func(x):
    st = np.array([1, x**2]).reshape(1, 2)
    column = np.array([1, x**2]).reshape(2, 1)
    return column * st

# D-оптимальность
def d_optimality(d):
    return np.linalg.det(d)

# A-оптимальность
def a_optimality(d):
    return np.trace(d)

# E-оптимальность
def e_optimality(d):
    return np.max(np.linalg.eig(d)[0])

# Ф-оптимальность
def f_optimality(d):
    p = 2
    return 1.0/p * (D**p).trace()

# Λ-оптимальность
def l_optimality(d):
```

```
    return np.sum((np.linalg.eig(d)[0] - np.average(np.linalg.eig(d)[0]))**2)

# MV-оптимальность
def mv_optimality(d):
    return np.max(np.diag(d))

# G-оптимальность
def g_optimality(D, x):
    d = np.zeros(3)
    for i in range(3):
        st = np.array([1, x[i]*x[i]]).reshape(1, 2)
        column = np.array([1, x[i]*x[i]]).reshape(2, 1)
        d[i] = st @ D @ column
    return np.max(d)

# Чтение планов
def read_plans():
    with open("data.json", "r") as f:
        plans = json.load(f)["plans"]
        return plans

plan_num = int(input("Выберите план (от 1 до 4): "))
plans = read_plans()
x = plans[plan_num-1]["x"]
p = plans[plan_num-1]["p"]

# Информационная и дисперсионная матрицы
M = 0
for i in range(3):
    M += p[i] * func(x[i])
D = np.linalg.inv(M)
print("Информационная матрица Фишера:")
print(M)
print("\nДисперсионная матрица:")
print(D)

# Критерии оптимальности
print("\nD-оптимальность:", '%.6f' % d_optimality(D))
print("A-оптимальность:", '%.6f' % a_optimality(D))
print("E-оптимальность:", '%.6f' % e_optimality(D))
print("Ф-оптимальность:", '%.6f' % f_optimality(D))
print("Λ-оптимальность:", '%.6f' % l_optimality(D))
print("MV-оптимальность:", '%.6f' % mv_optimality(D))
print("G-оптимальность:", '%.6f' % g_optimality(D, x))

# Поиск оптимальных значений параметра и критерия A-оптимальности
xPlot, yPlot = [], []
for q in np.arange(0.01, 0.5, 0.01):
    if q not in (-0.5, 0.5, 0):
        xPlot.append(q)
        yPlot.append(Model(q).A)
print("\nОптимальные значения, найденные по графику:")
print("q:", '%.6f' % xPlot[yPlot.index(min(yPlot))])
print("A-optimality:", '%.6f' % min(yPlot))
```

```python
# Построение графика изменения критерия А-оптимальности
fig = plt.figure()
plt.plot(xPlot, yPlot)
plt.scatter((yPlot.index(min(yPlot)) + 1) * 0.01, min(yPlot))
plt.ylabel('A-optimality')
plt.xlabel('q')
plt.grid(True)
plt.text((yPlot.index(min(yPlot)) + 1) * 0.01 + 0.01, min(yPlot), 'min')
plt.show()
```