

Министерство науки и высшего образования Российской Федерации  
Новосибирский государственный технический университет  
Кафедра теоретической и прикладной информатики

Планирование и анализ эксперимента

Лабораторная работа №1

Факультет:	ФПМИ
Группа:	ПМ-63
Студенты:	Кожекин М.В. Майер В.А. Назарова Т.А. Утюганов Д.С.
Вариант:	9(1)

Новосибирск

2020

# 1. Цель работы

Изучить понятие оптимального плана эксперимента и критерии оптимальности планов.

## 2. Задание

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 программу по обработке различных планов эксперимента для регрессионных моделей. Обработка заключается в вычислении различных характеристик плана, связанных с тем или иным критерием оптимальности.

3. Для каждого из планов вычислить значения функционалов от информационной (дисперсионной) матриц, связанных с такими критериями, как:  $D-$ ,  $A-$ ,  $E-$ ,  $\Phi_2-$ ,  $\Lambda-$ ,  $MV-$ ,  $G-$  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. Анализ

В рамках работы исследуется общая линейная модель:

$$y = f^T(x)\Theta + \varepsilon = \sum_{l=1}^m f_l(x)\Theta_l + \varepsilon$$

$$y = \Theta_0 + \Theta_1 \cdot x + \Theta_2 \cdot x^2$$

Исходя из варианта задания (1) эксперимент проводится в трёх точках -1, 0, 1

**Непрерывным нормированным планом** эксперимента называется совокупность величин вида

$$\varepsilon^* = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ p_1 & p_2 & \cdots & p_n \end{bmatrix} \text{ где } \sum_{i=1}^n p_i = 1, p_i \geq 0$$

**Информационной матрицей** дискретного непрерывного плана называется величина

$$M = \sum_{j=1}^n p_j f(x_j) f^T(x_j) = \frac{X^T X}{N}$$

Матрица, обратная к информационной, называется **дисперсионной**

$$D(\varepsilon) = M^{-1}(\varepsilon) \quad D(\varepsilon_N) = M^{-1}(\varepsilon_N)$$

Критерии оптимальности плана эксперимента:

$$D - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} |D(\varepsilon)|$$

$$A - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \text{tr} D(\varepsilon)$$

$$E - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \max_i \lambda_i[D(\varepsilon)]$$

$$\Phi_2 - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \sqrt{m^{-1} \text{tr} D^2(\varepsilon)}$$

$$\Lambda - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \sum_{i=1}^n [\lambda_j(D(\varepsilon)) - \bar{\lambda}(D(\varepsilon))]^2$$

$$MV - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \max_i D_{ii}(\varepsilon)$$

$$G - \text{оптимальности } \varepsilon^* = \text{Arg min}_{\varepsilon} \max_{x \in \hat{X}} d(x, \varepsilon)$$

$$d(x, \varepsilon) = f^T(x) M^{-1} f(x)$$

## 4. Выбор оптимального плана

В каждом из 4 вариантов эксперимент проводится в трёх точках: -1, 0, 1

$$y = \Theta_0 + \Theta_1 \cdot x + \Theta_2 \cdot x^2$$

№	$p_1$	$p_2$	$p_3$
1	0.2	0.6	0.2
2	0.25	0.5	0.25
3	0.1884	0.6233	0.1884
4	0.333	0.333	0.333

Найдём информационные и дисперсионные матрицы для каждого из вариантов:

**Матрицы М:**

variant 1			variant 2			variant 3			variant 4		
1.0	0.0	0.4	1.0	0.0	0.5	1.00	0.00	0.38	1.00	0.00	0.67
0.0	0.4	0.0	0.0	0.5	0.0	0.00	0.38	0.00	0.00	0.67	0.00
0.4	0.0	0.4	0.5	0.0	0.5	0.38	0.00	0.38	0.67	0.00	0.67

## Матрицы D:

variant 1			variant 2			variant 3			variant 4		
1.67	0.0	-1.67	2.0	0.0	-2.0	1.6	0.00	-1.60	3.0	0.0	-3.0
0.00	2.5	0.00	0.0	2.0	0.0	0.0	2.65	0.00	0.0	1.5	0.0
-1.67	0.0	4.17	-2.0	0.0	4.0	-1.6	0.00	4.26	-3.0	0.0	4.5

D		A		E		Phi_2		Lambda		MV		G	
i	value	i	value	i	value	i	value	i	value	i	value	i	value
4	6.77	2	8.00	1	5.00	2	2.83	3	8.72	2	4.00	2	4.00
2	8.00	1	8.33	3	5.01	1	2.97	1	8.80	1	4.17	1	4.17
1	10.42	3	8.52	2	5.24	3	3.04	2	10.67	3	4.26	3	4.26
3	11.30	4	9.01	4	6.85	4	3.24	4	22.55	4	4.50	4	4.50

Исходя из совокупности критериев оптимальным является план из варианта 2.

$$\varepsilon^* = \begin{bmatrix} -1 & 0 & 1 \\ 0.25 & 0.5 & 0.25 \end{bmatrix}$$

## 5. Выбор оптимального параметра

Под планом понимается модель из варианта 4 вида:

$$\varepsilon^* = \begin{bmatrix} -1 & 0 & 1 \\ q & 1 - 2 \cdot q & q \end{bmatrix} q \in (0, 0.5)$$

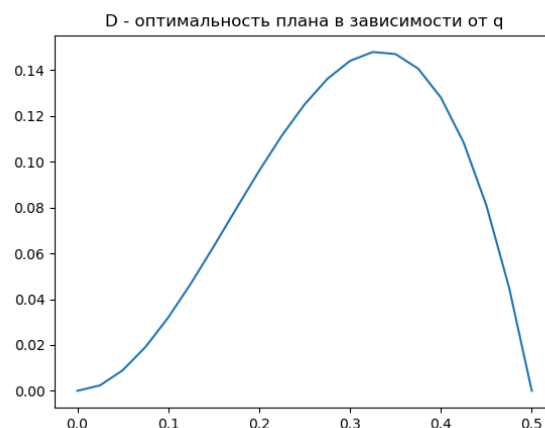


Рис. 1: Оптимальное значение параметра  $q = 0.333$

Проверим решение аналитически:

$$M = q \cdot \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} \cdot [1 \quad -1 \quad 1] + (1 - 2 \cdot q) \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot [1 \quad 0 \quad 0] + q \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \cdot [1 \quad 1 \quad 1]$$

$$M = q \cdot \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{bmatrix} + (1 - 2 \cdot q) \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} + q \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\det(M) = \begin{vmatrix} 1 & 0 & 2q \\ 0 & 2q & 0 \\ 2q & 0 & 2q \end{vmatrix} = 4q^2 + 0 + 0 - 8q^3 - 0 - 0 = 4q^2 - 8q^3$$

$$\frac{d(|M|)}{dq} = 4 \cdot 2q - 8 \cdot 3q^2 = 0$$

$$\frac{d(|M|)}{dq} = 3q^2 - q = 0$$

Аналитическое значение параметра  $q = \frac{1}{3}$

## 6. Исходный код программы

lab1.py

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import pandas as pd
4 import re
5 from numpy.linalg import inv, det, eigvals, norm
6
7
8 pd.set_option('precision', 2)
9 n = 3
10 m = 3
11 points_count = 51
12
13 def f(theta, x):
14     return theta[0] + theta[1]*x + theta[2]*x**2
15
16 def f_vector(x):
17     return np.array([ 1], [x], [x**2] ])
18
19 def f_vector_T(x):
20     return np.array([ 1, x, x**2 ])
21
22
23 #
24 #
25 #
26 class Lab1():
27     '''
28     Класс для 1 лабораторной работы
29     Для стандартизации все критерии ищут минимальное значение
30     '''
31     def __init__(self):
32         ''' Выделение памяти под массивы '''
33         self.x = np.ndarray(n)

```

```

34     self.p = np.ndarray(n)
35     self.M = np.ndarray((m, m))
36
37     def read_plan_from_file(self, filename):
38         ''' Считываем данные из файла '''
39         data = np.loadtxt(filename)
40         self.x = data[0]
41         self.p = data[1]
42
43     def create_plan(self, q):
44         ''' Создание плана '''
45         self.p = np.array([q, 1-2*q, q])
46
47     def build_matrix_M(self):
48         ''' Построение информационной матрицы M '''
49         self.M = np.zeros((m,m))
50         for i in range(n):
51             self.M += self.p[i] * f_vector(self.x[i]) * f_vector_T(self.x[i])
52
53     def build_matrix_D(self):
54         ''' Построение дисперсионной матрицы D '''
55         self.D = inv(self.M)
56
57     def find_optimal_q(self):
58         ''' Поиск оптимального значения q '''
59         self.x_axis = np.linspace(0, 0.5, points_count)
60         self.D_from_E = np.ndarray(points_count)
61         i = 0
62         for q in self.x_axis:
63             self.create_plan(q)
64             self.build_matrix_M()
65             # self.build_matrix_D() # тк.. матрица сингулярная
66             self.D_from_E[i] = self.calc_M()
67             i+=1
68
69         i = np.where(self.D_from_E == max(self.D_from_E))
70         print('Оптимальное значение параметра q: ' + str(self.x_axis[i][0]))
71
72     def compare_plans(self, filename):
73         ''' Сравнение планов различными критериями '''
74         plan_files = [
75             '1.txt',
76             '2.txt',
77             '3.txt',
78             '4.txt'
79         ]
80         with open('report/matricies_M.tex', 'w') as f:
81             f.write('\begin{tabular}{|c|c|c|c|}\n')
82             f.write('\hline\n')
83             f.write('\tvariant 1 & variant 2 & variant 3 & variant 4 \\\ \n')
84             f.write('\hline\n')
85             s = ''
86             for plan_file in plan_files:
87                 s += '&'
88                 self.read_plan_from_file('plans/' + plan_file)
89                 self.build_matrix_M()
90                 self.build_matrix_D()
91                 df = pd.DataFrame(data = self.M)
92                 s += df.to_latex(index=False, header = False, column_format='ccc

```

```

93         s = re.sub(r'\\[a-z]+rule\\n', '', s)# удаляем \toprule и \bottomrule
94         s = re.sub(r'\\n&', ' &\\n', s)      # переносим &
95         f.write(s[1:-1] + ' \\\\ \\n')
96         f.write('\\hline\\n\\end{tabular}')
97
98     with open('report/matrices_D.tex', 'w') as f:
99         f.write('\\begin{tabular}{|c|c|c|c|}\\n')
100        f.write('\\hline\\n')
101        f.write('\\tvariant 1 & variant 2 & variant 3 & variant 4 \\\\ \\n')
102        f.write('\\hline\\n')
103        s = ''
104        for plan_file in plan_files:
105            s += '&'
106            self.read_plan_from_file('plans/' + plan_file)
107            self.build_matrix_M()
108            self.build_matrix_D()
109            df = pd.DataFrame(data = self.D)
110            s += df.to_latex(index=False, header = False, column_format='ccc
111        ')
112
113        s = re.sub(r'\\[a-z]+rule\\n', '', s)# удаляем \toprule и \bottomrule
114        s = re.sub(r'\\n&', ' &\\n', s)      # переносим &
115        f.write(s[1:-1] + ' \\\\ \\n')
116        f.write('\\hline\\n\\end{tabular}')
117
118    criterion_files = [
119        'D.txt',
120        'A.txt',
121        'E.txt',
122        'Phi_2.txt',
123        'Lambda.txt',
124        'MV.txt',
125        'G.txt'
126    ]
127    criterion_functions = {
128        'D.txt': self.calc_D,
129        'A.txt': self.calc_A,
130        'E.txt': self.calc_E,
131        'Phi_2.txt': self.calc_Phi_2,
132        'Lambda.txt': self.calc_Lambda,
133        'MV.txt': self.calc_MV,
134        'G.txt': self.calc_G
135    }
136
137    i_vec = np.ndarray(4)
138    cr_vec = np.ndarray(4)
139    table = pd.DataFrame()
140
141    # для каждого критерия
142    for criterion_file in reversed(criterion_files):
143        criterion_name = criterion_file[:-4]
144        # рассмотрим 4 плана
145        for plan_file in plan_files:
146            self.read_plan_from_file('plans/' + plan_file)
147            self.build_matrix_M()
148            self.build_matrix_D()
149            result = criterion_functions[criterion_file]()
150            i = int(plan_file[:-4]) - 1
151            i_vec[i] = i + 1

```

```

152         cr_vec[i] = result
153         i_vec = np.argsort(cr_vec) + 1
154         cr_vec = np.sort(cr_vec)
155         d = {(criterion_name, 'i'): i_vec,
156              (criterion_name, 'value'): cr_vec}
157         df = pd.DataFrame(data = d)
158         table = df.join(table, lsuffix='_caller', rsuffix='_other')
159         with open(filename, 'w') as f:
160             s = table.to_latex(index=False)
161             s = re.sub(r'D &', '\\\\multicolumn{2}{l}{D} &', s)
162             f.writelines(s)
163
164     def draw_plot(self):
165         ''' Отрисовка зависимости D(E) от q '''
166         plt.title('D — оптимальность плана в зависимости от q')
167         plt.plot(self.x_axis, self.D_from_E)
168         plt.savefig('report/opt_q.png')
169
170     def calc_M(self):
171         '''
172         Критерий D — оптимальности. (D — determinant)
173         Эллипсоид рассеивания имеет минимальный объём
174         '''
175         return det(self.M)
176
177     def calc_D(self):
178         '''
179         Критерий D — оптимальности. (D — determinant)
180         Эллипсоид рассеивания имеет минимальный объём
181         '''
182         return det(self.D)
183
184     def calc_A(self):
185         '''
186         Критерий A — оптимальности. (A — average variance)
187         Эллипсоид рассеивания с наименьшей суммой квадратов длин осей
188         '''
189         return np.trace(self.D)
190
191     def calc_E(self):
192         '''
193         Критерий E — оптимальности. (E — eigenvalue)
194         Минимизация максимального собственного значения дисперсионной матрицы
195         '''
196         return np.max(eigvals(self.D))
197
198     def calc_Phi_2(self):
199         '''
200         Критерий \Phi_2 — оптимальности. (\Phi_2 — функционал класса \Phi_p, p in
201         (0, inf))
202         Минимизация максимального собственного значения дисперсионной матрицы
203         '''
204         return np.sqrt(np.trace(self.D**2) / m)
205
206     def calc_Lambda(self):
207         '''
208         Критерий \Lambda — оптимальности. (Lambda — собственное значение)
209         Минимизация дисперсии собственных значений
210         '''
211         eig_vec = eigvals(self.D)

```



```

211     avg_eig = np.mean(eig_vec)
212     return np.sum((eig_vec[:] - avg_eig)**2)
213
214     def calc_MV(self):
215         '''
216         Критерий MV — оптимальности. (MV — maximum variation)
217         Минимизация максимального диагонального значения дисперсионной матрицы
218         '''
219         return max(np.diag(self.D))
220
221     def calc_G(self):
222         '''
223         Критерий G — оптимальности. (G — general variance)
224         Минимизация максимального значения общей дисперсии
225         '''
226         return np.max([f_vector_T(x) * self.D * f_vector(x) for x in self.x])
227
228
229
230 # -----
231 # -----
232 # -----
233
234 l1 = Lab1()
235 # выбор оптимального плана
236 l1.compare_plans('report/plans_comparison.tex')
237 # выбор оптимального значения q
238 l1.read_plan_from_file('plans/4.txt')
239 l1.find_optimal_q()
240 l1.draw_plot()

```