**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

по дисциплине «**Математическое моделирование сложных систем**»

на тему: «**Планирование эксперимента**»

Выполнил: студент гр. ИП-32

Суховенко Э.С.

Принял: доцент

Трохова Т.А.

Гомель 2022

**Цель:** получить навыки разработки планов полного факторного эксперимента в СКМ, научиться выполнять регрессионный и корреляционный анализ результатов эксперимента.

**Практическая часть**

**Часть1**

1. Ввести исходные данные для построения ПФЭ:

- число факторов - 3;

- область планирования;

- функцию отклика.

Индивидуальные задания приведены в приложении А.

1. Составить матрицу ПФЭ.
2. Найти среднее значение функции отклика для пяти повторов опытов.
3. Построить регрессионную модель без учета взаимодействия факторов и определить значения коэффициентов регрессии.
4. Получить расчетные значения функции отклика по регрессионной модели
5. Вычислить абсолютную и относительную ошибки моделирования
6. Построить регрессионную модель с учетом взаимодействия факторов и определить значения коэффициентов регрессии.
7. Получить расчетные значения функции отклика по регрессионной модели
8. Вычислить абсолютную и относительную ошибки моделирования
9. Сделать вывод об адекватности модели

Вариант 12

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1,983 | 1,951 | 1,969 | 1,981 | 1,935 |
| 3,004 | 3,024 | 2,984 | 2,983 | 3,007 |
| 2,435 | 2,415 | 2,428 | 2,394 | 2,438 |
| 3,767 | 3,794 | 3,784 | 3,783 | 3,803 |
| 2,788 | 2,823 | 2,815 | 2,777 | 2,773 |
| 4,491 | 4,467 | 4,492 | 4,473 | 4,460 |
| 3,485 | 3,510 | 3,515 | 3,524 | 3,475 |
| 5,883 | 5,879 | 5,863 | 5,870 | 5,877 |

**Часть 2**

***Приложение 2***

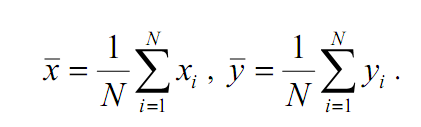
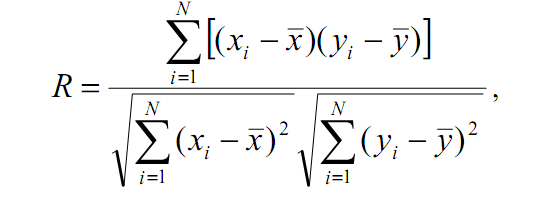
***Задание 2***

1. Создать с использованием программы «Блокнот» файлы значений двух наборов данных X и Y, приведенный в таблице 2 (данные лучше представить в виде строк).
2. Считать файлы в вектора X и Y .
3. Построить диаграмму рассеивания (простой двумерный график), не соединяя точки отрезками прямых.
4. Найти коэффициент парной корреляции по формуле.
5. Сделать вывод о наличии корреляционной связи.

**Описание задачи**

Пусть в результате эксперимента получены два набора данных х и y, характеризующие параметры технического объекта. Нужно определить, являются ли эти параметры взаимосвязанными. Простейшей, но информативной характеристикой связи двух величин X и Y является коэффициент корреляции.

Коэффициент корреляции вычисляется по формуле:



В зависимости от полученного значения R можно сделать вывод о взаимосвязи заданных параметров:

R=0,3 и менее – связь слабая;

R=0,3 – 0,5 – связь умеренная;

R=0,5 – 0,7 – связь существенная;

R=0,7 – 0,9 – связь сильная;

R=0,9 – 1 – связь очень сильная.

|  |  |  |
| --- | --- | --- |
| 1 | X | 0.1 2.89 8.77 5.25 12.34 2.61 10.65 4.55 1.37 2.2 14.82 1.78 0.13 7.97 9.02 2.49 6.76 0.85 11.74 7.79 |
| Y | 2.72 4.92 7.53 5.41 7.17 10.25 7.49 6.78 2.21 6.66 2.91 7.25 7.01 5.93 8.88 7.44 9.65 6.91 10.93 8.73 |

**Листинг задания 1:**

**// main.py**

from src.constants.constants import average\_ms, xs, count\_experiments, index\_arrays  
from src.table.table import to\_table  
import numpy as np  
  
  
def get\_b\_array(num, x\_arr, y\_arr):  
 def get\_multiple\_b(indexes):  
 return sum([np.prod([x\_arr[i][index] for index in indexes]) \* y\_arr[i] for i in  
 range(count\_experiments)]) / count\_experiments  
  
 return [get\_multiple\_b(index\_array) for index\_array in index\_arrays[:num]]  
  
  
def get\_y(b\_arr, x\_arr):  
 res = []  
 for i in range(count\_experiments):  
 cur\_y = 0  
 for j in range(len(b\_arr)):  
 cur\_y += b\_arr[j] \* np.prod([x\_arr[i][index] for index in index\_arrays[j]])  
 res.append(cur\_y)  
 return np.array(res)  
  
  
def calc\_error(y\_arr\_left, y\_arr\_right):  
 return np.abs(y\_arr\_left - y\_arr\_right)  
  
  
def main():  
 print('\nAverage expert: ', average\_ms, end='\n\n')  
 count\_person = len(average\_ms)  
 for b\_index in range(4, 9):  
 cur\_bs = []  
 cur\_ys = []  
 cur\_errors = []  
 for i in range(count\_person):  
 b\_array = get\_b\_array(b\_index, xs, average\_ms[i])  
 y\_array = get\_y(b\_array, xs)  
  
 cur\_bs.append(b\_array)  
 cur\_ys.append(y\_array)  
 cur\_errors.append(calc\_error(y\_array, average\_ms))  
 average\_errors = np.array(list(map(lambda row: np.average(cur\_errors), cur\_errors)))  
 average\_error = np.average(average\_errors)  
 print(f'Current model, step({b\_index - 4}), average error: ', average\_error)  
 print(to\_table(cur\_ys))  
 print('Coefficients: ')  
 print(to\_table(cur\_bs), end='\n\n')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

**// table.py**

from prettytable import PrettyTable  
  
  
def to\_table(a):  
 table = PrettyTable()  
 table.add\_rows([[f'{round(elem, 10):.5f}' for elem in row] for row in a])  
 return table

**// constants.py**

import numpy as np  
from src.full\_factors\_exp.full\_factors\_exp import get\_full\_factors\_exp  
  
count\_factors = 3  
ms\_t = np.array([  
 [1.983, 1.951, 1.969, 1.981, 1.935],  
 [3.004, 3.024, 2.984, 2.983, 3.007],  
 [2.435, 2.415, 2.428, 2.394, 2.438],  
 [3.767, 3.794, 3.784, 3.783, 3.803],  
 [2.788, 2.823, 2.815, 2.777, 2.773],  
 [4.491, 4.467, 4.492, 4.473, 4.460],  
 [3.485, 3.510, 3.515, 3.524, 3.475],  
 [5.883, 5.879, 5.863, 5.870, 5.877]  
])  
  
average\_ms\_t = np.array([np.average(measures\_of\_persons) for measures\_of\_persons in ms\_t])  
  
average\_ms = np.array([average\_ms\_t.transpose()])  
xs = np.array(get\_full\_factors\_exp(count\_factors))  
count\_experiments = 2 \*\* count\_factors  
  
index\_arrays = [  
 [],  
 [0],  
 [1],  
 [2],  
 [0, 1],  
 [0, 2],  
 [1, 2],  
 [0, 1, 2]  
]

**// full\_factors\_exp.py**

def to\_binary\_array(num):  
 res = []  
 while num:  
 res.append(num % 2)  
 num //= 2  
 while len(res) < 3:  
 res.append(0)  
 return res  
  
  
def to\_factors\_array(binary\_array):  
 return [(-1 if i == 0 else 1) for i in binary\_array]  
  
  
def get\_full\_factors\_exp(factor\_number):  
 res = []  
 max\_value = 2 \*\* factor\_number  
 for i in range(0, max\_value):  
 binary\_array = to\_binary\_array(i)  
 factors\_array = to\_factors\_array(binary\_array)  
 res.append(factors\_array)  
 return res

**Результат выполнения задания 1:**

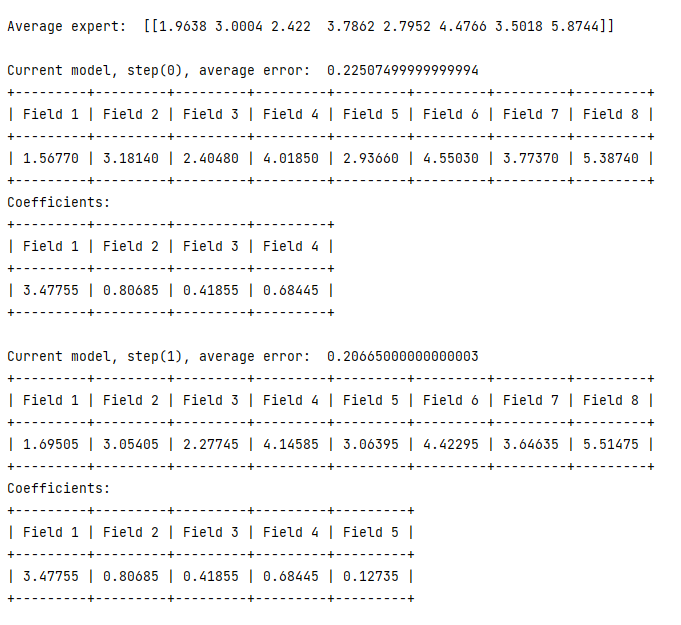


Рисунок 1 – Результат выполнения задания 1

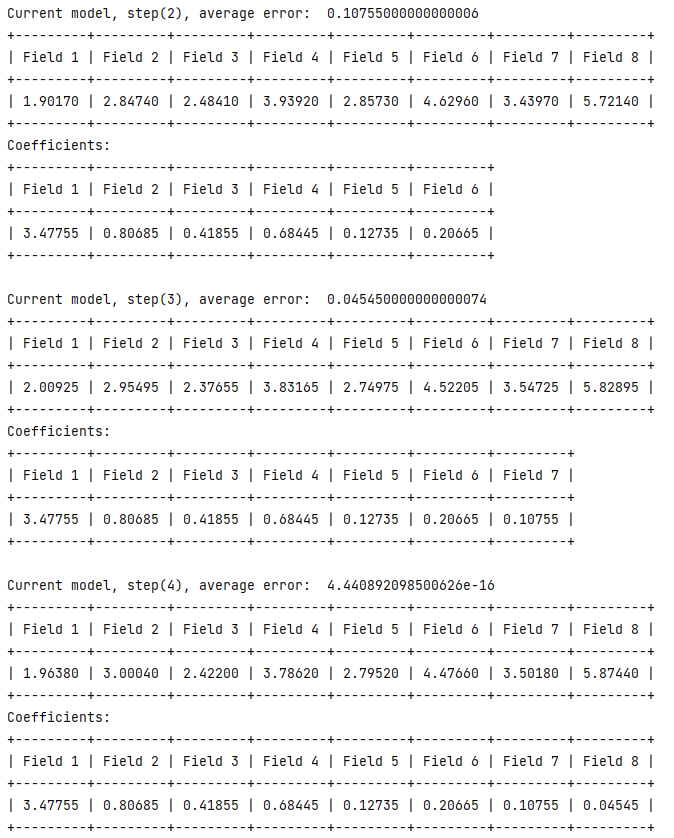


Рисунок 2 – Результат выполнения задания 1

**Листинг задания 2:**

**// main.py**

import numpy as np  
import matplotlib.pyplot as plt  
  
*# 1*xs = [0.1, 2.89, 8.77, 5.25, 12.34, 2.61, 10.65, 4.55, 1.37, 2.2, 14.82, 1.78, 0.13, 7.97, 9.02, 2.49, 6.76, 0.85,  
 11.74, 7.79]  
ys = [2.72, 4.92, 7.53, 5.41, 7.17, 10.25, 7.49, 6.78, 2.21, 6.66, 2.91, 7.25, 7.01, 5.93, 8.88, 7.44, 9.65, 6.91,  
 10.93, 8.73]  
  
*# 12  
# xs = [18.99, 1.92, 15.60, 19.01, 6.44, 14.181, 18.23, 19.18, 20.07, 9.68, 7.78, 9.40, 0.01, 17.01, 1.97, 15.04, 3.38, 1.06, 0.40]  
# ys = [9.71, 2.61, 4.12, 12.58, 5.56, 3.48, 3.474, 14.56, 0.34, 2.27, 0.13, 9.65, 6.21, 1.36, 10.48, 9.86, 8.03, 0.80, 3.89]*average\_x = np.sum(xs) / len(xs)  
average\_y = np.sum(ys) / len(ys)  
  
r = np.sum([(xs[i] - average\_x) \* (ys[i] - average\_y) for i in range(len(xs))]) \  
 / (np.sqrt(np.sum([(x - average\_x) \*\* 2 for x in xs])) \* np.sqrt(np.sum([(y - average\_y) \*\* 2 for y in ys])))  
  
print(f'The correlation coefficient: {r}')  
print('The connection is ', end='')  
if r < 0.3:  
 print('Weak', end='')  
elif r < 0.5:  
 print('Normal', end='')  
elif r < 0.7:  
 print('Good', end='')  
elif r < 0.9:  
 print('Powerful', end='')  
elif r <= 1:  
 print('Very strong', end='')  
print('.')  
  
plt.scatter(xs, ys)  
plt.grid()  
plt.show()

**Результат выполнения задания 2:**

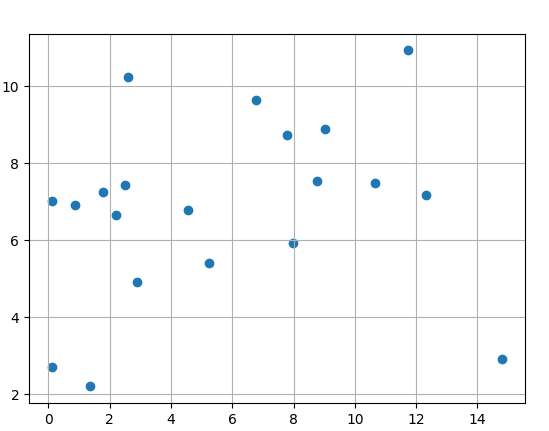


Рисунок 3 – Результат выполнения задания 2

**Вывод:** получил навыки разработки планов полного факторного эксперимента в СКМ, научился выполнять регрессионный и корреляционный анализ результатов эксперимента.