Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 3

з дисципліни «МНД» на тему
«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ
ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ.»

ВИКОНАВ:
студент ІІ курсу ФІОТ
групи ІВ-91
Мусійчук Я. С.
Залікова - 9121

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

Мета: провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

Завдання:

1. Скласти матрицю планування для дробового трьохфакторного експерименту. Провести експеримент в усіх точках факторного простору, повторивши N експериментів, де N – кількість експериментів (рядків матриці планування) в усіх точках факторного простору – знайти значення функції відгуку У. Значення функції відгуку знайти у відповідності з варіантом діапазону, зазначеного далі (випадковим чином).

$y_{max} = 200 + x_{cp\ max}$;

$y_{min} = 200 + x_{cp\ min}$

де $x_{cp\,max} = \frac{x_{1max} + x_{2max} + x_{3max}}{3}$, $x_{cp\,min} = \frac{x_{1min} + x_{2min} + x_{3min}}{3}$

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.

3. Провести 3 статистичні перевірки.

4. Написати комп'ютерну програму, яка усе це виконує.

Варіант: 119

№ варіанту	X1		X2		X3	
	min	max	min	max	min	Max
119	15	45	-35	15	-35	-5

Лістинг програми:

```
from random import *
from scipy import linalg
from scipy.stats import t as critT
from scipy.stats import f as critF
from math import sqrt
import prettytable
###----------Проміжки--------
x1 = [15, 45]
x2 = [-35, 15]
x3 = [-35, -5]
listX = [x1, x2, x3]

m, N, d = 3, 4, 4
###----------Матриця планування експерименту--------
planMatrix = [[1, -1, -1, -1],
          [1, -1, 1, 1],
          [1, 1, -1, 1],
          [1, 1, 1, -1]]

###----------Відповідна матриця X-------
Xmatrix = [[listX[0][0], listX[1][0], listX[2][0]],
      [listX[0][0], listX[1][1], listX[2][1]],
      [listX[0][1], listX[1][0], listX[2][1]],
      [listX[0][1], listX[1][1], listX[2][0]]]

###----------Транспоновані матриці--------
transposePlanMatrix = [list(i) for i in zip(*planMatrix)]
transposeXmatrix = [list(i) for i in zip(*Xmatrix)]

###----------Середні занчення--------
minMaxAvgX = [sum(listX[i][k] for i in range(3)) / 3 for k in range(2)]
minMaxY = [int(200 + minMaxAvgX[i]) for i in range(2)]

###----------Матрицця Y--------
Ymatrix = [[randint(minMaxY[0], minMaxY[1]) for _ in range(m)] for _ in range(N)]

def AvgY():
    return [sum(Ymatrix[k1]) / m for k1 in range(N)]

def find_cf():
    tran = transposeXmatrix
    mx = [sum(Xmatrix[i][k] for i in range(N)) / N for k in range(m)]
    my = sum(AvgY()) / N
    ai = [sum(transposeXmatrix[k][i] * AvgY()[i] for i in range(N)) / N for k in
range(m)]
    aii = [sum(transposeXmatrix[k][i] ** 2 for i in range(N)) / N for k in range(m)]

    a12 = a21 = (tran[0][0] * tran[1][0] + tran[0][1] * tran[1][1] + tran[0][2] *
tran[1][2] + tran[0][3] * tran[1][
        3]) / N
    a13 = a31 = (tran[0][0] * tran[2][0] + tran[0][1] * tran[2][1] + tran[0][2] *
tran[2][2] + tran[0][3] * tran[2][
        3]) / N
    a23 = a32 = (tran[1][0] * tran[2][0] + tran[1][1] * tran[2][1] + tran[1][2] *
tran[2][2] + tran[1][3] * tran[2][
        3]) / N

    determnt = linalg.det(
        [[1, mx[0], mx[1], mx[2]],
         [mx[0], aii[0], a12, a13],
```

```
        [mx[1], a12, aii[1], a32],
        [mx[2], a13, a23, aii[2]]])

    b0 = linalg.det([[my, mx[0], mx[1], mx[2]],
                     [ai[0], aii[0], a12, a13],
                     [ai[1], a12, aii[1], a32],
                     [ai[2], a13, a23, aii[2]]]) / determnt

    b1 = linalg.det([[1, my, mx[1], mx[2]],
                     [mx[0], ai[0], a12, a13],
                     [mx[1], ai[1], aii[1], a32],
                     [mx[2], ai[2], a23, aii[2]]]) / determnt

    b2 = linalg.det([[1, mx[0], my, mx[2]],
                     [mx[0], aii[0], ai[0], a13],
                     [mx[1], a12, ai[1], a32],
                     [mx[2], a13, ai[2], aii[2]]]) / determnt

    b3 = linalg.det([[1, mx[0], mx[1], my],
                     [mx[0], aii[0], a12, ai[0]],
                     [mx[1], a12, aii[1], ai[1]],
                     [mx[2], a13, a23, ai[2]]]) / determnt

    check = [b0 + b1 * tran[0][i] + b2 * tran[1][i] + b3 * tran[2][i] for i in
range(4)]
    b_list = [b0, b1, b2, b3]
    return check, b_list

f1 = m-1
f2 = N
f3 = f1*f2
f4 = N - d

Ydisperssion = [sum([((k1 - AvgY()[j]) ** 2) for k1 in Ymatrix[j]]) / m for j in
range(N)]
check, b_list = find_cf()
print('Матриця планування:')

table = prettytable.PrettyTable()
table.field_names = ["X1", "X2", "X3", "Y1", "Y2", "Y3"]
for i in range(0, 4):
    table.add_row([*[x for x in Xmatrix[i]], *[y for y in Ymatrix[i]]])
print(table)
print(f"\nРівняння регресії:\ny = {b_list[0]} + {b_list[1]}*x1 + {b_list[2]}*x2 +
{b_list[3]}*x3")

print('\nПеревірка однорідності дисперсії за критерієм Кохрена:')

if max(Ydisperssion) / sum(Ydisperssion) < 0.7679:
    print('Дисперсія однорідна: ', max(Ydisperssion) / sum(Ydisperssion))
else:
    print('Дисперсія неоднорідна: ', max(Ydisperssion) / sum(Ydisperssion))

print('\nПеревірка нуль-гіпотези за критерієм Стьюдента:')

S2b = sum(Ydisperssion) / N
S2bs = S2b / (m * N)
Sbs = sqrt(S2bs)
bb = [sum(AvgY()[k] * transposePlanMatrix[i][k] for k in range(N)) / N for i in
range(N)]
t = [round((abs(bb[i]) / Sbs),3) for i in range(N)]
table1 = prettytable.PrettyTable()
```

```
table1.field_names = ["t0", "t1", "t2", "t3"]
table1.add_row([*t])
print(table1)
for i in range(N):
    if t[i] < critT.ppf(q=0.975, df=f3):
        b_list[i] = 0
        d -= 1

y_reg = [b_list[0] + b_list[1] * Xmatrix[i][0] + b_list[2] * Xmatrix[i][1] +
b_list[3] * Xmatrix[i][2]
         for i in range(N)]
print('Значення y:')
[print(
    f"{b_list[0]} + {b_list[1]}*x1 + {b_list[2]}*x2 + {b_list[3]}*x3 = {b_list[0] +
b_list[1] * Xmatrix[i][0] + b_list[2] * Xmatrix[i][1] + b_list[3] * Xmatrix[i][2]}")
    for i in range(N)]

print('\nПеревірка адекватності моделі за критерієм Фішера:')
Sad = (m / (N - d)) * int(sum(y_reg[i] - AvgY()[i] for i in range(N)) ** 2)
Fp = Sad / S2b
q = 0.05
F_table = critF.ppf(q=1-q, dfn=f4, dfd=f3)
print('Fp  =', Fp)
if Fp > F_table:
    print('Неадекватно при 0.05')
else:
    print('Адекватно при 0.05')
```

Контрольні запитання:

1.Що називається дробовим факторним експериментом?

Дробовим факторним експериментом називається експеримент з використанням частини повного факторного експерименту

2.Для чого потрібно розрахункове значення Кохрена?

Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.

3. Для чого перевіряється критерій Стьюдента?

За допомогою критерію Стьюдента перевіряється значущість коефіцієнтів рівняння

4.Чим визначається критерій Фішера і як його застосовувати?

Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваного об”єкта.

Результат виконання роботи:

```
Матриця планування:
+----+-----+-----+-----+-----+-----+
| X1 |  X2 |  X3 |  Y1 |  Y2 |  Y3 |
+----+-----+-----+-----+-----+-----+
| 15 | -35 | -35 | 184 | 190 | 199 |
| 15 |  15 |  -5 | 197 | 186 | 191 |
| 45 | -35 |  -5 | 212 | 189 | 218 |
| 45 |  15 | -35 | 198 | 187 | 181 |
+----+-----+-----+-----+-----+-----+

Рівняння регресії:
y = 192.26666666666688 + 0.21111111111110947*x1 + -0.17333333333333373*x2 + 0.30000000000000054*x3

Перевірка однорідності дисперсії за критерієм Кохрена:
Дисперсія однорідна:  0.5917508417508418

Перевірка нуль-гіпотези за критерієм Стьюдента:
+--------+------+-------+-------+
|   t0   |  t1  |   t2  |   t3  |
+--------+------+-------+-------+
| 82.864 | 1.35 | 1.848 | 1.919 |
+--------+------+-------+-------+
Значення y:
192.26666666666688 + 0*x1 + 0*x2 + 0*x3 = 192.26666666666688
192.26666666666688 + 0*x1 + 0*x2 + 0*x3 = 192.26666666666688
192.26666666666688 + 0*x1 + 0*x2 + 0*x3 = 192.26666666666688
192.26666666666688 + 0*x1 + 0*x2 + 0*x3 = 192.26666666666688

Перевірка адекватності моделі за критерієм Фішера:
Fp  = 1.0303030303030305
Адекватно при 0.05
```