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 № 6

з дисципліни «МНД» на тему

«Проведення трьохфакторного експерименту при використанні рівняння регресії з квадратичними членами»

ВИКОНАВ:
студент ІІ курсу ФІОТ
групи ІО-93
Прокопчук Д.І.
Залікова - 9326

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

Мета: Провести трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи рототабельний композиційний план.

Завдання:

1. Ознайомитися з теоретичними відомостями.
2. Вибрати з таблиці варіантів і записати в протокол інтервали значень x_1, x_2, x_3. Обчислити і записати значення, відповідні кодованим значенням факторів +1; -1;+ l; - l; 0 для $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$.
3. Значення функції відгуку знайти за допомогою підстановки в формулу:

$$y_i = f(x_1, x_2, x_3) + \text{random}(10)-5,$$

де $f(x_1, x_2, x_3)$ вибирається по номеру в списку в журналі викладача.

4. Провести експерименти і аналізуючи значення статистичних перевірок, отримати адекватну модель рівняння регресії. При розрахунках використовувати натуральні значення факторів.
5. Зробити висновки по виконаній роботі.

Алгоритм отримання адекватної моделі рівняння регресії

1) *Вибір рівняння регресії (лінійна форма, рівняння з урахуванням ефекту взаємодії і з урахуванням квадратичних членів);*
2) *Вибір кількості повторів кожної комбінації (m = 2);*
3) *Складення матриці планування експерименту і вибір кількості рівнів (N)*
4) *Проведення експериментів;*
5) *Перевірка однорідності дисперсії. Якщо не однорідна – повертаємося на п. 2 і збільшуємо m на 1);*
6) *Розрахунок коефіцієнтів рівняння регресії. При розрахунку використовувати* ***натуральні*** *значення x_1, x_2 и x_3.*
7) *Перевірка нуль-гіпотези. Визначення значимих коефіцієнтів;*
8) *Перевірка адекватності моделі рівняння оригіналу. При неадекватності – повертаємося на п.1, змінивши при цьому рівняння регресії;*

Варіант: 323

323	-5	15	-25	10	15	45	5,0+4,7*x1+3,6*x2+6,4*x3+6,8*x1*x1+0,3*x2*x2+5,3*x3*x3+3,2*x1*x2+0,9*x1*x3+2,7*x2*x3+0,1*x1*x2*x3

Лістинг програми:

```
from math import fabs
from random import randrange
import numpy as np
from numpy.linalg import solve
from scipy.stats import f, t
from prettytable import PrettyTable


def round_matrix(matrix, n_to_round=3):
    for i in range(len(matrix)):
        matrix[i] = list(matrix[i])
        for j in range(len(matrix[i])):
            matrix[i][j] = round(matrix[i][j], n_to_round)
    return matrix


def function(X1, X2, X3):
    y = 5 + 4.7 * X1 + 3.6 * X2 + 6.4 * X3 + 6.8 * X1 * X1 + 0.3 * X2 * X2 +
5.3 * X3 * X3 + 3.2 * X1 * X2 + \
        0.9 * X1 * X3 + 2.7 * X2 * X3 + 0.1 * X1 * X2 * X3 + randrange(0, 10)
- 5
    return y


def main(n, m):
    global major_counter
    x1min = -5
    x1max = 15
    x2min = -25
    x2max = 10
    x3min = 15
    x3max = 45

    x01 = (x1max + x1min) / 2
    x02 = (x2max + x2min) / 2
    x03 = (x3max + x3min) / 2
    deltax1 = x1max - x01
    deltax2 = x2max - x02
    deltax3 = x3max - x03

    xn = [[-1, -1, -1, +1, +1, +1, -1, +1, +1, +1],
          [-1, -1, +1, +1, -1, -1, +1, +1, +1, +1],
          [-1, +1, -1, -1, +1, -1, +1, +1, +1, +1],
          [-1, +1, +1, -1, -1, +1, -1, +1, +1, +1],
          [+1, -1, -1, -1, -1, +1, +1, +1, +1, +1],
          [+1, -1, +1, -1, +1, -1, -1, +1, +1, +1],
          [+1, +1, -1, +1, -1, -1, -1, +1, +1, +1],
          [+1, +1, +1, +1, +1, +1, +1, +1, +1, +1],
          [-1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],
          [+1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],
          [0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],
          [0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],
          [0, 0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929],
          [0, 0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929],
          [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

    x1 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max, -1.73 *
deltax1 + x01, 1.73 * deltax1 + x01, x01, x01,
          x01, x01, x01]
    x2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max, x02, x02, -
1.73 * deltax2 + x02, 1.73 * deltax2 + x02,
```

```
          x02, x02, x02]
    x3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max, x03, x03,
x03, x03, -1.73 * deltax3 + x03,
          1.73 * deltax3 + x03, x03]

    x1x2 = [0] * 15
    x1x3 = [0] * 15
    x2x3 = [0] * 15
    x1x2x3 = [0] * 15
    x1kv = [0] * 15
    x2kv = [0] * 15
    x3kv = [0] * 15

    for i in range(15):
        x1x2[i] = x1[i] * x2[i]
        x1x3[i] = x1[i] * x3[i]
        x2x3[i] = x2[i] * x3[i]
        x1x2x3[i] = x1[i] * x2[i] * x3[i]
        x1kv[i] = x1[i] ** 2
        x2kv[i] = x2[i] ** 2
        x3kv[i] = x3[i] ** 2

    list_for_a = round_matrix(list(zip(x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3,
x1kv, x2kv, x3kv)))

    planning_matrix_with_naturalized_coeffs_x = PrettyTable()
    planning_matrix_with_naturalized_coeffs_x.title = 'Матриця планування з
натуралізованими коефіцієнтами X'
    planning_matrix_with_naturalized_coeffs_x.field_names = ['X1', 'X2',
'X3', 'X1X2', 'X1X3', 'X2X3', 'X1X2X3', 'X1X1', 'X2X2', 'X3X3']
    planning_matrix_with_naturalized_coeffs_x.add_rows(list_for_a)
    print(planning_matrix_with_naturalized_coeffs_x)

    Y = round_matrix([[function(list_for_a[j][0], list_for_a[j][1],
list_for_a[j][2]) for i in range(m)] for j in range(15)])

    planning_matrix_y = PrettyTable()
    planning_matrix_y.title = 'Матриця планування Y'
    planning_matrix_y.field_names = ['Y1', 'Y2', 'Y3']
    planning_matrix_y.add_rows(Y)
    print(planning_matrix_y)

    Y_average = []
    for i in range(len(Y)):
        Y_average.append(np.mean(Y[i], axis=0))
    print("Середні значення відгуку за рядками:")
    for i in range(15):
        print("{:.3f}".format(Y_average[i]), end=" ")

    dispersions = []
    for i in range(len(Y)):
        a = 0
        for k in Y[i]:
            a += (k - np.mean(Y[i], axis=0)) ** 2
        dispersions.append(a / len(Y[i]))

    def find_known(num):
        a = 0
        for j in range(15):
            a += Y_average[j] * list_for_a[j][num - 1] / 15
        return a

    def a(first, second):
        a = 0
```

```
        for j in range(15):
            a += list_for_a[j][first - 1] * list_for_a[j][second - 1] / 15
        return a

    my = sum(Y_average) / 15
    mx = []

    for i in range(10):
        number_lst = []
        for j in range(15):
            number_lst.append(list_for_a[j][i])
        mx.append(sum(number_lst) / len(number_lst))

    det1 = [
        [1, mx[0], mx[1], mx[2], mx[3], mx[4], mx[5], mx[6], mx[7], mx[8],
mx[9]],
        [mx[0], a(1, 1), a(1, 2), a(1, 3), a(1, 4), a(1, 5), a(1, 6), a(1,
7), a(1, 8), a(1, 9), a(1, 10)],
        [mx[1], a(2, 1), a(2, 2), a(2, 3), a(2, 4), a(2, 5), a(2, 6), a(2,
7), a(2, 8), a(2, 9), a(2, 10)],
        [mx[2], a(3, 1), a(3, 2), a(3, 3), a(3, 4), a(3, 5), a(3, 6), a(3,
7), a(3, 8), a(3, 9), a(3, 10)],
        [mx[3], a(4, 1), a(4, 2), a(4, 3), a(4, 4), a(4, 5), a(4, 6), a(4,
7), a(4, 8), a(4, 9), a(4, 10)],
        [mx[4], a(5, 1), a(5, 2), a(5, 3), a(5, 4), a(5, 5), a(5, 6), a(5,
7), a(5, 8), a(5, 9), a(5, 10)],
        [mx[5], a(6, 1), a(6, 2), a(6, 3), a(6, 4), a(6, 5), a(6, 6), a(6,
7), a(6, 8), a(6, 9), a(6, 10)],
        [mx[6], a(7, 1), a(7, 2), a(7, 3), a(7, 4), a(7, 5), a(7, 6), a(7,
7), a(7, 8), a(7, 9), a(7, 10)],
        [mx[7], a(8, 1), a(8, 2), a(8, 3), a(8, 4), a(8, 5), a(8, 6), a(8,
7), a(8, 8), a(8, 9), a(8, 10)],
        [mx[8], a(9, 1), a(9, 2), a(9, 3), a(9, 4), a(9, 5), a(9, 6), a(9,
7), a(9, 8), a(9, 9), a(9, 10)],
        [mx[9], a(10, 1), a(10, 2), a(10, 3), a(10, 4), a(10, 5), a(10, 6),
a(10, 7), a(10, 8), a(10, 9), a(10, 10)]]

    det2 = [my, find_known(1), find_known(2), find_known(3), find_known(4),
find_known(5), find_known(6), find_known(7),
            find_known(8), find_known(9), find_known(10)]

    beta = solve(det1, det2)
    print("\nОтримане рівняння регресії:")
    print("{:.3f} + {:.3f} * X1 + {:.3f} * X2 + {:.3f} * X3 + {:.3f} * X1X2 +
{:.3f} * X1X3 + {:.3f} * X2X3"
          "+ {:.3f} * X1X2X3 + {:.3f} * X11^2 + {:.3f} * X22^2 + {:.3f} *
X33^2 = ŷ"
          .format(beta[0], beta[1], beta[2], beta[3], beta[4], beta[5],
beta[6], beta[7], beta[8], beta[9], beta[10]))
    y_i = [0] * 15
    print("Експериментальні значення:")
    for k in range(15):
        y_i[k] = beta[0] + beta[1] * list_for_a[k][0] + beta[2] *
list_for_a[k][1] + beta[3] * list_for_a[k][2] + \
                 beta[4] * list_for_a[k][3] + beta[5] * list_for_a[k][4] +
beta[6] * list_for_a[k][5] + beta[7] * \
                 list_for_a[k][6] + beta[8] * list_for_a[k][7] + beta[9] *
list_for_a[k][8] + beta[10] * list_for_a[k][9]
    for i in range(15):
        print("{:.3f}".format(y_i[i]), end=" ")
    print("\n\nПеревірка за критерієм Кохрена")
    Gp = max(dispersions) / sum(dispersions)
    Gt = 0.3346
    print("Gp =", Gp)
```

```
    if Gp < Gt:
        print("Дисперсія однорідна")
    else:
        print("Дисперсія неоднорідна")

    print("\nПеревірка значущості коефіцієнтів за критерієм Стьюдента")
    sb = sum(dispersions) / len(dispersions)
    sbs = (sb / (15 * m)) ** 0.5

    F3 = (m - 1) * n
    coefs1 = []
    coefs2 = []
    d = 11
    res = [0] * 11
    for j in range(11):
        t_pract = 0
        for i in range(15):
            if j == 0:
                t_pract += Y_average[i] / 15
            else:
                t_pract += Y_average[i] * xn[i][j - 1]
            res[j] = beta[j]
        if fabs(t_pract / sbs) < t.ppf(q=0.975, df=F3):
            coefs2.append(beta[j])
            res[j] = 0
            d -= 1
        else:
            coefs1.append(beta[j])

    print("Значущі коефіцієнти регресії:", [round(i, 3) for i in coefs1])
    print("Незначущі коефіцієнти регресії:", [round(i, 3) for i in coefs2])
    y_st = []
    for i in range(15):
        y_st.append(res[0] + res[1] * x1[i] + res[2] * x2[i] + res[3] * x3[i]
+ res[4] * x1x2[i] + res[5] *
                    x1x3[i] + res[6] * x2x3[i] + res[7] * x1x2x3[i] + res[8]
* x1kv[i] + res[9] *
                    x2kv[i] + res[10] * x3kv[i])
    print("Значення з отриманими коефіцієнтами:")
    for i in range(15):
        print("{:.3f}".format(y_st[i]), end=" ")

    print("\n\nПеревірка адекватності за критерієм Фішера")
    Sad = m * sum([(y_st[i] - Y_average[i]) ** 2 for i in range(15)]) / (n -
d)
    Fp = Sad / sb
    F4 = n - d
    print("Fp =", Fp)
    if Fp < f.ppf(q=0.95, dfn=F4, dfd=F3):
        print("Рівняння регресії адекватне при рівні значимості 0.05")
    else:
        print("Рівняння регресії неадекватне при рівні значимості 0.05")


main(15, 3)
```

Результат виконання роботи:

Матриця планування з натуралізованими коефіцієнтами X

X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	X1X1	X2X2	X3X3
-5	-25	15	125	-75	-375	1875	25	625	225
-5	-25	45	125	-225	-1125	5625	25	625	2025
-5	10	15	-50	-75	150	-750	25	100	225
-5	10	45	-50	-225	450	-2250	25	100	2025
15	-25	15	-375	225	-375	-5625	225	625	225
15	-25	45	-375	675	-1125	-16875	225	625	2025
15	10	15	150	225	150	2250	225	100	225
15	10	45	150	675	450	6750	225	100	2025
-12.3	-7.5	30.0	92.25	-369.0	-225.0	2767.5	151.29	56.25	900.0
22.3	-7.5	30.0	-167.25	669.0	-225.0	-5017.5	497.29	56.25	900.0
5.0	-37.775	30.0	-188.875	150.0	-1133.25	-5666.25	25.0	1426.951	900.0
5.0	22.775	30.0	113.875	150.0	683.25	3416.25	25.0	518.701	900.0
5.0	-7.5	4.05	-37.5	20.25	-30.375	-151.875	25.0	56.25	16.403
5.0	-7.5	55.95	-37.5	279.75	-419.625	-2098.125	25.0	56.25	3130.403
5.0	-7.5	30.0	-37.5	150.0	-225.0	-1125.0	25.0	56.25	900.0

Матриця планування Y

Y1	Y2	Y3
1047.0	1041.0	1048.0
8993.0	8990.0	8996.0
1610.5	1612.5	1609.5
11866.5	11867.5	11866.5
416.0	416.0	423.0
7403.0	7409.0	7404.0
4269.5	4276.5	4269.5
15673.5	15671.5	15664.5
5555.187	5560.187	5556.187
7396.907	7396.907	7401.907
1356.795	1355.795	1356.795
8079.9	8079.9	8082.9
101.253	102.253	99.253
15930.563	15928.563	15929.563
4449.375	4444.375	4447.375

```
Середні значення відгуку за рядками:
1045.333 8993.000 1610.833 11866.833 418.333 7405.333 4271.833 15669.833 5557.187 7398.574 1356.462 8080.900 100.920 15929.563 4447.042
Отримане рівняння регресії:
5.742 + 4.706 * X1 + 3.589 * X2 + 6.419 * X3 + 3.195 * X1X2 + 0.902 * X1X3 + 2.699 * X2X3+ 0.100 * X1X2X3 + 6.789 * X11^2 + 0.297 * X22^2 + 5.300 * X33^2 = ŷ
Експериментальні значення:
1044.222 8992.889 1608.706 11865.706 418.100 7406.101 4270.584 15669.585 5559.111 7398.467 1356.196 8082.984 102.985 15929.315 4447.029

Перевірка за критерієм Кохрена
Gp = 0.18662952646239553
Дисперсія однорідна

Перевірка значущості коефіцієнтів за критерієм Стьюдента
Значущі коефіцієнти регресії: [5.742, 4.706, 3.589, 6.419, 3.195, 0.902, 2.699, 0.1, 6.789, 0.297, 5.3]
Незначущі коефіцієнти регресії: []
Значення з отриманими коефіцієнтами:
1044.222 8992.889 1608.706 11865.706 418.100 7406.101 4270.584 15669.585 5559.111 7398.467 1356.195 8082.984 102.983 15929.312 4447.029

Перевірка адекватності за критерієм Фішера
Fp = 3.0681874250557444
Рівняння регресії неадекватне при рівні значимості 0.05
```