Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Методи наукових досліджень

Лабораторна робота №3

«Проведення трьохфакторного експерименту з використанням лінійного рівняння регресії»

Виконав:
студент 2 курсу, групи ІВ-91
Коренюк Андрій Олександрович
Залікова книжка № ІВ-9115
Варіант: 14

Перевірив: ас. Регіда П.Г.

Київ – 2021

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

Завдання

№варіанта	x_1		x_2		x_3	
	min	**max**	**min**	**max**	**min**	**max**
114	-25	75	25	65	25	40

Лістинг програми

criterions.py

```
"""Таблиця для критерія Кохрена"""
base_kohren = [
   [9985, 9750, 9392, 9057, 8772, 8534, 8332, 8159, 8010, 7880, 7341, 6602, 5813, 5000],
   [9669, 8709, 7977, 7457, 7071, 6771, 6530, 6333, 6167, 6025, 5466, 4748, 4031, 3333],
   [9065, 7679, 6841, 6287, 5892, 5598, 5365, 5175, 5017, 4884, 4366, 3720, 3093, 2500],
   [8412, 6838, 5981, 5440, 5063, 4783, 4564, 4387, 4241, 4118, 3645, 3066, 2513, 2000],
   [7808, 6161, 5321, 4803, 4447, 4184, 3980, 3817, 3682, 3568, 3135, 2612, 2119, 1667],
   [7271, 5612, 4800, 4307, 3974, 3726, 3535, 3384, 3259, 3154, 2756, 2278, 1833, 1429],
   [6798, 5157, 4377, 3910, 3595, 3362, 3185, 3043, 2926, 2829, 2462, 2022, 1616, 1250],
   [6385, 4775, 4027, 3584, 3286, 3067, 2901, 2768, 2659, 2568, 2226, 1820, 1446, 1111],
   [6020, 4450, 3733, 3311, 3029, 2823, 2666, 2541, 2439, 2353, 2032, 1655, 1308, 1000],
   [5410, 3924, 3264, 2880, 2624, 2439, 2299, 2187, 2098, 2020, 1737, 1403, 1000, 833],
   [4709, 3346, 2758, 2419, 2159, 2034, 1911, 1815, 1736, 1671, 1429, 1144, 889, 667],
   [3894, 2705, 2205, 1921, 1735, 1602, 1501, 1422, 1357, 1303, 1108, 879, 675, 500],
   [3434, 2354, 1907, 1656, 1493, 1374, 1286, 1216, 1160, 1113, 942, 743, 567, 417],
   [2929, 1980, 1593, 1377, 1237, 1137, 1061, 1002, 958, 921, 771, 604, 457, 333],
   [2370, 1576, 1259, 1082, 968, 887, 827, 780, 745, 713, 595, 462, 347, 250],
   [1737, 1131, 895, 766, 682, 623, 583, 552, 520, 497, 411, 316, 234, 167],
   [998, 632, 495, 419, 371, 337, 312, 292, 279, 266, 218, 165, 120, 83],
]
column_kohren_f1 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (6,): 5, (7,): 6, (8,): 7, (9,): 8,
                    (range(10, 14)): 9, (range(14, 26)): 10, (range(26, 91)): 11, (range(91, 145)): 12}
COLUMN_KOHREN_F1_ELSE = 13

row_kohren_f2 = {(2,): 0, (3,): 1, (4,): 2, (5,): 3, (6,): 4, (7,): 5, (8,): 6, (9,): 7,
                 (range(10, 12)): 8, (range(12, 14)): 9, (range(14, 18)): 10, (range(18, 23)): 11,
                 (range(23, 28)): 12, (range(28, 36)): 13, (range(36, 51)): 14, (range(51, 81)): 15,
                 (range(81, 121)): 16}

ROW_KOHREN_F2_ELSE = 16

"""Таблиця для t-критерія Стьюдента"""
base_student_f3 = {(1,): 12.71, (2,): 4.303, (3,): 3.182, (4,): 2.776, (5,): 2.571, (6,): 2.447, (7,): 2.365, (8,): 2.306,
                   (9,): 2.262, (10,): 2.228, (11,): 2.201, (12,): 2.179, (13,): 2.160, (14,): 2.145, (15,): 2.131,
                   (16,): 2.120, (17,): 2.110, (18,): 2.101, (19,): 2.093, (20,): 2.086, (21,): 2.080, (22,): 2.074,
                   (23,): 2.069, (24,): 2.064, (25,): 2.060, (26,): 2.056, (27,): 2.052, (28,): 2.048, (29,): 2.045,
                   (30,): 2.042}
T_STUDENT_ELSE = 1.960

"""Таблиця для F-критерія Фішера"""
base_phisher = [
```

```
    [164.4, 199.5, 215.7, 224.6, 230.2, 234.0, 244.9, 249.0, 254.3],
    [18.5,  19.2,  19.2,  19.3,  19.3,  19.3,  19.4,  19.4,  19.5],
    [10.1,  9.6,   9.3,   9.1,   9.0,   8.9,   8.7,   8.6,   8.5],
    [7.7,   6.9,   6.6,   6.4,   6.3,   6.2,   5.9,   5.8,   5.6],
    [6.6,   5.8,   5.4,   5.2,   5.1,   5.0,   4.7,   4.5,   4.4],
    [6.0,   5.1,   4.8,   4.5,   4.4,   4.3,   4.0,   3.8,   3.7],
    [5.5,   4.7,   4.4,   4.1,   4.0,   3.9,   3.6,   3.4,   3.2],
    [5.3,   4.5,   4.1,   3.8,   3.7,   3.6,   3.3,   3.1,   2.9],
    [5.1,   4.3,   3.9,   3.6,   3.5,   3.4,   3.1,   2.9,   2.7],
    [5.0,   4.1,   3.7,   3.5,   3.3,   3.2,   2.9,   2.7,   2.5],
    [4.8,   4.0,   3.6,   3.4,   3.2,   3.1,   2.8,   2.6,   2.4],
    [4.8,   3.9,   3.5,   3.3,   3.1,   3.0,   2.7,   2.5,   2.3],
    [4.7,   3.8,   3.4,   3.2,   3.0,   2.9,   2.6,   2.4,   2.2],
    [4.6,   3.7,   3.3,   3.1,   3.0,   2.9,   2.5,   2.3,   2.1],
    [4.5,   3.7,   3.3,   3.1,   2.9,   2.8,   2.5,   2.3,   2.1],
    [4.5,   3.6,   3.2,   3.0,   2.9,   2.7,   2.4,   2.2,   2.0],
    [4.5,   3.6,   3.2,   3.0,   2.8,   2.7,   2.4,   2.2,   2.0],
    [4.4,   3.6,   3.2,   2.9,   2.8,   2.7,   2.3,   2.1,   1.9],
    [4.4,   3.5,   3.1,   2.9,   2.7,   2.6,   2.3,   2.1,   1.9],
    [4.4,   3.5,   3.1,   2.9,   2.7,   2.6,   2.3,   2.1,   1.9],
    [4.3,   3.4,   3.1,   2.8,   2.7,   2.6,   2.2,   2.0,   1.8],
    [4.3,   3.4,   3.0,   2.8,   2.6,   2.5,   2.2,   2.0,   1.7],
    [4.2,   3.4,   3.0,   2.7,   2.6,   2.5,   2.2,   2.0,   1.7],
    [4.2,   3.3,   3.0,   2.7,   2.6,   2.4,   2.1,   1.9,   1.7],
    [4.2,   3.3,   2.9,   2.7,   2.5,   2.4,   2.1,   1.9,   1.6],
    [4.1,   3.2,   2.9,   2.6,   2.5,   2.3,   2.0,   1.8,   1.5],
    [4.0,   3.2,   2.8,   2.5,   2.4,   2.3,   1.9,   1.7,   1.4],
    [3.9,   3.1,   2.7,   2.5,   2.3,   2.2,   1.8,   1.6,   1.3],
    [3.8,   3.0,   2.6,   2.4,   2.2,   2.1,   1.8,   1.5,   1.0]
]

row_phisher_f3 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (6,): 5, (7,): 6, (8,): 7, (9,): 8, (10,): 9,
                  (11,): 10, (12,): 11, (13,): 12, (14,): 13, (15,): 14, (16,): 15, (17,): 16, (18,): 17, (19,): 18,
                  (20, 21): 19, (22, 23): 20, (24, 25): 21, (26, 27): 22, (28, 29): 23, (range(30, 36)): 24,
                  (range(36, 51)): 25, (range(51, 91)): 26, (range(91, 121)): 27}
ROW_PHISHER_F3_ELSE = 28

column_phisher_f4 = {(1,): 0, (2,): 1, (3,): 2, (4,): 3, (5,): 4, (range(6, 10)): 5, (range(10, 19)): 6,
                     (range(19, 25)): 7}
COLUMN_PHISHER_F4_ELSE = 8


def check_kohren(f1, f2, Gp):
    """True, якщо дисперсія однорідна"""
    row = -1
    column = -1
    for key in row_kohren_f2.keys():
        if f2 in key:
            row = row_kohren_f2[key]
            break
    if row == -1:
        row = ROW_KOHREN_F2_ELSE

    for key in column_kohren_f1.keys():
        if f1 in key:
            column = column_kohren_f1[key]
            break
    if column == -1:
        column = COLUMN_KOHREN_F1_ELSE
    return Gp < (base_kohren[row][column]/1000)
```

```
def check_student(f3, t_exp):
    """True, якщо коефіцієнт Bs є значущим."""
    t_teo = -1
    for key in base_student_f3.keys():
        if f3 in key:
            t_teo = base_student_f3[key]
            break

    if t_teo == -1:
        t_teo = T_STUDENT_ELSE

    return t_exp > t_teo


def check_phisher(f3, f4, Fp):
    """True, якщо отримана математична модель адекватна експериментальним даним."""
    row = -1
    column = -1
    for key in row_phisher_f3.keys():
        if f3 in key:
            row = row_phisher_f3[key]
            break
    if row == -1:
        row = ROW_PHISHER_F3_ELSE

    for key in column_phisher_f4.keys():
        if f4 in key:
            column = column_phisher_f4[key]
            break
    if column == -1:
        column = COLUMN_PHISHER_F4_ELSE
    return Fp <= base_phisher[row][column]
```

lab_3.py

```
from random import randint
from copy import deepcopy
from math import ceil, floor, sqrt
from numpy.linalg import det
import criterions as cr
from beautifultable import BeautifulTable

"""Довірча ймовірність p = 0.95 (критерій значимості 0.05)"""
variant = {"n": 114, "x1min": -25, "x1max": 75, "x2min": 25, "x2max": 65, "x3min": 25, "x3max": 40}
N = 4
K = 4

x_min_average = (variant["x1min"] + variant["x2min"] + variant["x3min"]) / 3
x_max_average = (variant["x1max"] + variant["x2max"] + variant["x3max"]) / 3
y_min = ceil(200 + x_min_average)
y_max = floor(200 + x_max_average)

x0 = [1, 1, 1, 1]
x1 = [-1, -1, 1, 1]
x2 = [-1, 1, -1, 1]
x3 = [-1, 1, 1, -1]

nx0 = [1, 1, 1, 1]
nx1 = [variant["x1min"] if x1[i] == -1 else variant["x1max"] for i in range(N)]
nx2 = [variant["x2min"] if x2[i] == -1 else variant["x2max"] for i in range(N)]
nx3 = [variant["x3min"] if x3[i] == -1 else variant["x3max"] for i in range(N)]

m = 3
```

```
y_1 = [randint(y_min, y_max) for _ in range(m)]
y_2 = [randint(y_min, y_max) for _ in range(m)]
y_3 = [randint(y_min, y_max) for _ in range(m)]
y_4 = [randint(y_min, y_max) for _ in range(m)]

ext_data = []


def check_model_suitability():
    global ext_data

    y_average = [sum(y_1)/m, sum(y_2)/m, sum(y_3)/m, sum(y_4)/m]

    #Перевірка критерія Кохрена
    S2_dis = [0, 0, 0, 0]
    for i in range(m):
        S2_dis[0] += (y_1[i] - y_average[0]) ** 2
        S2_dis[1] += (y_2[i] - y_average[1]) ** 2
        S2_dis[2] += (y_3[i] - y_average[2]) ** 2
        S2_dis[3] += (y_4[i] - y_average[3]) ** 2
    for i in range(N):
        S2_dis[i] /= m

    Gp = max(S2_dis) / sum(S2_dis)
    f1 = m - 1
    f2 = N
    if not cr.check_kohren(f1, f2, Gp):
        return False

    # Пошук коефіцієнтів
    mx1, mx2, mx3, my = sum(nx1) / N, sum(nx2) / N, sum(nx3) / N, sum(y_average) / N
    a11, a22, a33 = 0, 0, 0
    a12, a13, a23 = 0, 0, 0
    a1, a2, a3 = 0, 0, 0
    for i in range(N):
        a11 += nx1[i] ** 2
        a22 += nx2[i] ** 2
        a33 += nx3[i] ** 2
        a12 += nx1[i] * nx2[i]
        a13 += nx1[i] * nx3[i]
        a23 += nx2[i] * nx3[i]
        a1 += y_average[i] * nx1[i]
        a2 += y_average[i] * nx2[i]
        a3 += y_average[i] * nx3[i]
    a11, a22, a33 = a11 / N, a22 / N, a33 / N
    a12, a13, a23 = a12 / N, a13 / N, a23 / N
    a1, a2, a3 = a1 / N, a2 / N, a3 / N
    a21 = a12
    a31 = a13
    a32 = a23

    main_det = det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a21, a22, a23], [mx3, a31, a32, a33]])
    A0 = det([[my, mx1, mx2, mx3], [a1, a11, a12, a13], [a2, a21, a22, a23], [a3, a31, a32, a33]]) / main_det
    A1 = det([[1, my, mx2, mx3], [mx1, a1, a12, a13], [mx2, a2, a22, a23], [mx3, a3, a32, a33]]) / main_det
    A2 = det([[1, mx1, my, mx3], [mx1, a11, a1, a13], [mx2, a21, a2, a23], [mx3, a31, a3, a33]]) / main_det
    A3 = det([[1, mx1, mx2, my], [mx1, a11, a12, a1], [mx2, a21, a22, a2], [mx3, a31, a32, a3]]) / main_det
    A = [A0, A1, A2, A3]

    B = [0, 0, 0, 0]
    for i in range(N):
        B[0] += y_average[i] * x0[i]
        B[1] += y_average[i] * x1[i]
```

```
            B[2] += y_average[i] * x2[i]
            B[3] += y_average[i] * x3[i]
        for i in range(K):
            B[i] /= N

        Ac = deepcopy(A)
        Bc = deepcopy(B)

        # Перевірка критерія Стьюдента
        S2B = sum(S2_dis) / N
        S2_B = S2B / (N * m)
        S_B = sqrt(S2_B)
        t = [0, 0, 0, 0]
        for i in range(K):
            t[i] = abs(B[i]) / S_B
        f3 = f1 * f2
        d = K
        for i in range(K):
            if not cr.check_student(f3, t[i]):
                B[i] = 0
                A[i] = 0
                d -= 1

        # Перевірка критерія Фішера
        y_for_phisher = [0 for _ in range(N)]
        for i in range(N):
            y_for_phisher[i] = nx0[i]*A[0] + nx1[i]*A[1] + nx2[i]*A[2] + nx3[i]*A[3]
        S2ad = 0
        for i in range(N):
            S2ad += (y_for_phisher[i] - y_average[i]) ** 2
        S2ad = m * S2ad / (N - d)
        f4 = N - d
        Fp = S2ad / S2B
        if not cr.check_phisher(f3, f4, Fp):
            return False

        ext_data = [Ac, Bc, A, B, y_average, [f1, f2, f3, f4], Gp, t, Fp]
        return True


    while not check_model_suitability():
        y_1.append(randint(y_min, y_max))
        y_2.append(randint(y_min, y_max))
        y_3.append(randint(y_min, y_max))
        y_4.append(randint(y_min, y_max))
        m += 1

    Ac = ext_data[0]
    Bc = ext_data[1]
    A = ext_data[2]
    B = ext_data[3]
    y_average = ext_data[4]
    f = ext_data[5]
    Gp = ext_data[6]
    t = ext_data[7]
    Fp = ext_data[8]
    y = {0: y_1, 1: y_2, 2: y_3, 3: y_4}

    natural_plan = BeautifulTable()
    rationed_plan = BeautifulTable()
    natural_checking_m = BeautifulTable()
    rationed_checking_m = BeautifulTable()
```

```
new_natural_checking_m = BeautifulTable()
new_rationed_checking_m = BeautifulTable()

y_headers = [f"Y{i+1}" for i in range(m)]
natural_plan.column_headers = ["№", "N-red X1", "N-red X2", "N-red X3", *y_headers]
rationed_plan.column_headers = ["№", "R-ned X0", "R-ned X1", "R-ned X2", "R-ned X3", *y_headers]
natural_checking_m.column_headers = ["№", "N-red X1", "N-red X2", "N-red X3", "Average Y[j]", "Exp-tal
Y[j]"]
rationed_checking_m.column_headers = ["№", "R-ned X0", "R-ned X1", "R-ned X2", "R-ned X3", "Average
Y[j]", "Exp-tal Y[j]"]
new_natural_checking_m.column_headers = ["№", "N-red X1", "N-red X2", "N-red X3", "Average Y[j]", "Exp-tal
Y[j]"]
new_rationed_checking_m.column_headers = ["№", "R-ned X0", "R-ned X1", "R-ned X2", "R-ned X3", "Average
Y[j]", "Exp-tal Y[j]"]

for i in range(N):
    natural_plan.append_row([i+1, nx1[i], nx2[i], nx3[i], *y[i]])
    rationed_plan.append_row([i+1, x0[i], x1[i], x2[i], x3[i], *y[i]])
    natural_checking_m.append_row([i+1, nx1[i], nx2[i], nx3[i], y_average[i], nx0[i] * Ac[0] + Ac[1] * nx1[i] + Ac[2]
* nx2[i] + Ac[3] * nx3[i]])
    rationed_checking_m.append_row([i+1, x0[i], x1[i], x2[i], x3[i], y_average[i], x0[i] * Bc[0] + Bc[1] * x1[i] + Bc[2]
* x2[i] + Bc[3] * x3[i]])
    new_natural_checking_m.append_row([i+1, nx1[i], nx2[i], nx3[i], y_average[i], nx0[i] * A[0] + A[1] * nx1[i] +
A[2] * nx2[i] + A[3] * nx3[i]])
    new_rationed_checking_m.append_row([i+1, x0[i], x1[i], x2[i], x3[i], y_average[i], x0[i] * B[0] + B[1] * x1[i] +
B[2] * x2[i] + B[3] * x3[i]])

print("Матриця планування експерименту (натуральні значення коефіцієнтів)")
print(natural_plan, "\n")

print("Рівняння регресії до перевірки значущості коефіцієнтів (натуральних)")
print(f"{Ac[0]} + {Ac[1]} * x1 + {Ac[2]} * x2 + {Ac[3]} * x3", "\n")

print("Зробимо перевірку при натуральних значеннях коефіцієнтів")
print(natural_checking_m, "\n")

print("Матриця планування експерименту (кодовані значення коефіцієнтів)")
print(rationed_plan, "\n")

print("Рівняння регресії до перевірки значущості коефіцієнтів (кодованих)")
print(f"{Bc[0]} + {Bc[1]} * x1 + {Bc[2]} * x2 + {Bc[3]} * x3", "\n")

print("Зробимо перевірку при кодованих значеннях коефіцієнтів")
print(rationed_checking_m, "\n")

print(f"Ступені свободи:")
print(f"* f1 = {f[0]}")
print(f"* f2 = {f[1]}")
print(f"* f3 = {f[2]}")
print(f"* f4 = {f[3]}")
print("Експериментальне значення критеріїв:")
print(f"* Кохрена - Gp = {Gp}")
print(f"* Стьюдента - t = {t}")
print(f"* Фішера - Fp = {Fp}", "\n")

print("Нове рівняння регресії (натуральні значення коефіцієнтів)")
print(f"{A[0]} + {A[1]} * x1 + {A[2]} * x2 + {A[3]} * x3", "\n")

print("Зробимо перевірку при нових натуральних значеннях коефіцієнтів")
print(new_natural_checking_m, "\n")

print("Нове рівняння регресії (кодовані значення коефіцієнтів)")
```

```
print(f"{B[0]} + {B[1]} * x1 + {B[2]} * x2 + {B[3]} * x3", "\n")

print("Зробимо перевірку при нових кодованих значеннях коефіцієнтів")
print(new_rationed_checking_m)
```

Результат виконання роботи

```
Матриця планування експерименту (натуральні значення коефіцієнтів)
+---+----------+----------+----------+-----+-----+-----+
| № | N-red X1 | N-red X2 | N-red X3 | Y1  | Y2  | Y3  |
+---+----------+----------+----------+-----+-----+-----+
| 1 |   -25    |    25    |    25    | 222 | 219 | 258 |
+---+----------+----------+----------+-----+-----+-----+
| 2 |   -25    |    65    |    40    | 237 | 211 | 258 |
+---+----------+----------+----------+-----+-----+-----+
| 3 |    75    |    25    |    40    | 246 | 258 | 216 |
+---+----------+----------+----------+-----+-----+-----+
| 4 |    75    |    65    |    25    | 220 | 212 | 225 |
+---+----------+----------+----------+-----+-----+-----+

Рівняння регресії до перевірки значущості коефіцієнтів (натуральних)
218.22222222222186 + -0.04666666666666718 * x1 + -0.23333333333333076 * x2 + 0.7777777777777912 * x3

Зробимо перевірку при натуральних значеннях коефіцієнтів
+---+----------+----------+----------+--------------+--------------+
| № | N-red X1 | N-red X2 | N-red X3 | Average Y[j] | Exp-tal Y[j] |
+---+----------+----------+----------+--------------+--------------+
| 1 |   -25    |    25    |    25    |    233.0     |    233.0     |
+---+----------+----------+----------+--------------+--------------+
| 2 |   -25    |    65    |    40    |   235.333    |   235.333    |
+---+----------+----------+----------+--------------+--------------+
| 3 |    75    |    25    |    40    |    240.0     |    240.0     |
+---+----------+----------+----------+--------------+--------------+
| 4 |    75    |    65    |    25    |    219.0     |    219.0     |
+---+----------+----------+----------+--------------+--------------+

Матриця планування експерименту (кодовані значення коефіцієнтів)
+---+----------+----------+----------+----------+-----+-----+-----+
| № | R-ned X0 | R-ned X1 | R-ned X2 | R-ned X3 | Y1  | Y2  | Y3  |
+---+----------+----------+----------+----------+-----+-----+-----+
| 1 |    1     |    -1    |    -1    |    -1    | 222 | 219 | 258 |
+---+----------+----------+----------+----------+-----+-----+-----+
| 2 |    1     |    -1    |    1     |    1     | 237 | 211 | 258 |
+---+----------+----------+----------+----------+-----+-----+-----+
| 3 |    1     |    1     |    -1    |    1     | 246 | 258 | 216 |
+---+----------+----------+----------+----------+-----+-----+-----+
| 4 |    1     |    1     |    1     |    -1    | 220 | 212 | 225 |
+---+----------+----------+----------+----------+-----+-----+-----+

Рівняння регресії до перевірки значущості коефіцієнтів (кодованих)
231.83333333333334 + -2.333333333333343 * x1 + -4.666666666666664 * x2 + 5.833333333333336 * x3
```

Зробимо перевірку при кодованих значеннях коефіцієнтів

№	R-ned X0	R-ned X1	R-ned X2	R-ned X3	Average Y[j]	Exp-tal Y[j]
1	1	-1	-1	-1	233.0	233.0
2	1	-1	1	1	235.333	235.333
3	1	1	-1	1	240.0	240.0
4	1	1	1	-1	219.0	219.0

Ступені свободи:
* f1 = 2
* f2 = 4
* f3 = 8
* f4 = 3

Експериментальне значення критеріїв:
* Кохрена - Gp = 0.3608157951833369
* Стьюдента - t = [50.18794358268526, 0.5051266787617515, 1.0102533575234987, 1.2628166969043744]
* Фішера - Fp = 3.8509196981752245

Нове рівняння регресії (натуральні значення коефіцієнтів)
218.22222222222186 + 0 * x1 + 0 * x2 + 0 * x3

Зробимо перевірку при нових натуральних значеннях коефіцієнтів

№	N-red X1	N-red X2	N-red X3	Average Y[j]	Exp-tal Y[j]
1	-25	25	25	233.0	218.222
2	-25	65	40	235.333	218.222
3	75	25	40	240.0	218.222
4	75	65	25	219.0	218.222

Нове рівняння регресії (кодовані значення коефіцієнтів)
231.83333333333334 + 0 * x1 + 0 * x2 + 0 * x3

Зробимо перевірку при нових кодованих значеннях коефіцієнтів

№	R-ned X0	R-ned X1	R-ned X2	R-ned X3	Average Y[j]	Exp-tal Y[j]
1	1	-1	-1	-1	233.0	231.833
2	1	-1	1	1	235.333	231.833
3	1	1	-1	1	240.0	231.833
4	1	1	1	-1	219.0	231.833

Process finished with exit code 0

Відповіді на контрольні запитання

1. ***Що називається дробовим факторним експериментом?***

Відповідь: дробовий факторний експеримент – це використання регулярних дробових реплік від ПФЕ, що містять відповідну кількість дослідів та зберігають основні властивості матриці планування.

2. ***Для чого потрібно розрахункове значення Кохрена?***

Відповідь: розрахункове значення Кохрена необхідно для визначення однорідності дисперсії.

3. ***Для чого перевіряється критерій Стьюдента?***

Відповідь: критерій Стьюдента перевіряється для підтвердження нуль-гіпотези – знайдений коефіцієнт β_s є статистично незначущим.

4. ***Чим визначається критерій Фішера і як його застосовувати?***

Відповідь: критерій Фішера визначається як відношення дисперсії адекватності до дисперсії відтворюваності.

Для його застосування необхідно:

- обчислити розрахункове значення $F_p = \frac{S^2_{\text{ад}}}{S^2_{\text{в}}}$;
- вибрати рівень статистичної значимості q та ступені свободи *f3* і *f4*. Знайти в спеціальній таблиці $F_{\text{табл}}$;
- порівняти F_p та $F_{\text{табл}}$. Якщо $F_p < F_{\text{табл}}$, то отримана математична модель є адекватною експериментальним даним.