Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Методи оптимізації та планування експерименту

ЛАБОРАТОРНА РОБОТА № 4

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ ЕФЕКТУ ВЗАЄМОДІЇ »

ВИКОНАЛА:

студентка ІІ курсу ФІОТ

групи ІО-92

Варіант №211

Карнаухова Анастасія

ПЕРЕВІРИВ:

Доц. Порєв В.М.

Київ – 2021

ЛАБОРАТОРНА РОБОТА № 4

Тема: ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ ЕФЕКТУ ВЗАЄМОДІЇ

Мета: Провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об'єкту.

Завдання на лабораторну роботу

1. Скласти матрицю планування для повного трьохфакторного експерименту.
2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

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

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

3. Знайти коефіцієнти рівняння регресії і записати його.
4. Провести 3 статистичні перевірки – за критеріями Кохрена, Стьюдента, Фішера.
5. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
6. Написати комп'ютерну програму, яка усе це моделює.

Варіант завдання:

211	-25	-5	-70	-10	-25	-5

Код програми:

```
import math
import numpy as np
from scipy.stats import t, f
import random as r
from functools import partial
import prettytable as p
from numpy.linalg import solve
from prettytable import PrettyTable

table0 = PrettyTable()
table0.field_names = (["Студент", "Группа"])
name = "Карнаухова Анастасія"
group = "ІО-92"
table0.add_row([name, group])
print(table0)
```

```python
m = 3
prob = 0.95
x1_min = -25
x1_max = -5
x2_min = -70
x2_max = -10
x3_min = -25
x3_max = -5
k = 3
counter = 0
x_ranges = [[x1_min, x1_max], [x2_min, x2_max], [x3_min, x3_max]]
x0_norm = [1, 1, 1, 1]
x1_norm = [-1, -1, 1, 1]
x2_norm = [-1, 1, -1, 1]
x3_norm = [-1, 1, 1, -1]
x1x2_norm = [a * b for a, b in zip(x1_norm, x2_norm)]
x1x3_norm = [a * b for a, b in zip(x1_norm, x3_norm)]
x2x3_norm = [a * b for a, b in zip(x2_norm, x3_norm)]
x1x2x3_norm = [a * b * c for a, b, c in zip(x1_norm, x2_norm, x3_norm)]
N = len(x1_norm)
xcp_max = (x1_max + x2_max + x3_max) / 3
xcp_min = (x1_min + x2_min + x3_min) / 3
x_norm = [x1_norm, x2_norm, x3_norm]
Y_min = 200 + xcp_min
Y_max = 200 + xcp_max

x_abs = []

for i in range(k):
    temp = []
    for j in x_norm[i]:
        if j == 1:
            temp.append(x_ranges[i][1])
        else:
            temp.append(x_ranges[i][0])
    x_abs.append(temp)
print("Абсолютні значення: " + str(x_abs))

Y_exp = []
for i in range(N):
    temp = []
    for _ in range(m):
        temp.append(r.randint(math.floor(Y_min), math.floor(Y_max)))
    Y_exp.append(temp)


def y_perevirka_norm(x1, x2, x3):
    return b0 + x1 * b1 + x2 * b2 + x3 * b3


def y_perevirka_abs(x1, x2, x3):
    return a0 + a1 * x1 + a2 * x2 + a3 * x3


def get_cohren_critical(prob, f1, f2):
    f_crit = f.isf((1 - prob) / f2, f1, (f2 - 1) * f1)
    return f_crit / (f_crit + f2 - 1)
```

```
fisher_teor = partial(f.ppf, q=1 - 0.05)
student_teor = partial(t.ppf, q=1 - 0.025)

odnorid = False
adekvat = False
while not adekvat:
    while not odnorid:
        table1 = p.PrettyTable()
        table1.add_column("X0", x0_norm)
        for i in range(k):
            table1.add_column("X{0}".format(i + 1), x_norm[i])
        for i in range(m):
            table1.add_column("Y{0}".format(i + 1), [j[i] for j in Y_exp])
        print("Нормалізована матриця:\n", table1)

        mx_norm_list = [np.mean(i) for i in x_norm]
        y_aver = [np.mean(i) for i in Y_exp]
        my = np.mean(y_aver)
        a1 = np.mean([x_norm[0][i] * y_aver[i] for i in range(N)])
        a2 = np.mean([x_norm[1][i] * y_aver[i] for i in range(N)])
        a3 = np.mean([x_norm[2][i] * y_aver[i] for i in range(N)])
        a11 = np.mean([x_norm[0][i] ** 2 for i in range(N)])
        a22 = np.mean([x_norm[1][i] ** 2 for i in range(N)])
        a33 = np.mean([x_norm[2][i] ** 2 for i in range(N)])
        a12 = np.mean([x_norm[0][i] * x_norm[1][i] for i in range(N)])
        a13 = np.mean([x_norm[0][i] * x_norm[2][i] for i in range(N)])
        a23 = np.mean([x_norm[1][i] * x_norm[2][i] for i in range(N)])
        a21 = a12
        a31 = a13
        a32 = a23

        znam = np.array([[1, mx_norm_list[0], mx_norm_list[1],
mx_norm_list[2]],
                         [mx_norm_list[0], a11, a12, a13],
                         [mx_norm_list[1], a12, a22, a32],
                         [mx_norm_list[2], a13, a23, a33]])

        b0_matr = np.array([[my, mx_norm_list[0], mx_norm_list[1],
mx_norm_list[2]],
                            [a1, a11, a12, a13],
                            [a2, a12, a22, a32],
                            [a3, a13, a23, a33]])

        b1_matr = np.array([[1, my, mx_norm_list[1], mx_norm_list[2]],
                            [mx_norm_list[0], a1, a12, a13],
                            [mx_norm_list[1], a2, a22, a32],
                            [mx_norm_list[2], a3, a23, a33]])

        b2_matr = np.array([[1, mx_norm_list[0], my, mx_norm_list[2]],
                            [mx_norm_list[0], a11, a1, a13],
                            [mx_norm_list[1], a12, a2, a32],
                            [mx_norm_list[2], a13, a3, a33]])

        b3_matr = np.array([[1, mx_norm_list[0], mx_norm_list[1], my],
                            [mx_norm_list[0], a11, a12, a1],
                            [mx_norm_list[1], a12, a22, a2],
                            [mx_norm_list[2], a13, a23, a3]])

        znam_value = np.linalg.det(znam)
        b0 = np.linalg.det(b0_matr) / znam_value
        b1 = np.linalg.det(b1_matr) / znam_value
```

```
        b2 = np.linalg.det(b2_matr) / znam_value
        b3 = np.linalg.det(b3_matr) / znam_value
        print("Рівняння регресії для нормованих значень:\ny = {0} + {1}*x1
+ {2}*x2 + {3}*x3".format(b0, b1, b2, b3))

        delt_x1 = (x1_max - x1_min) / 2
        delt_x2 = (x2_max - x2_min) / 2
        delt_x3 = (x3_max - x3_min) / 2
        x10 = (x1_max + x1_min) / 2
        x20 = (x2_max + x2_min) / 2
        x30 = (x3_max + x3_min) / 2
        a0 = b0 - b1 * (x10 / delt_x1) - b2 * (x20 / delt_x2) - b3 * (x30 /
delt_x3)
        a1 = b1 / delt_x1
        a2 = b2 / delt_x2
        a3 = b3 / delt_x3

        print("Рівняння регресії для абсолютних значень:\ny = {0} + {1}*x1
+ {2}*x2 + {3}*x3".format(a0, a1, a2, a3))
        # Кохрен
        y_var = [np.var(Y_exp[i]) for i in range(N)]
        flag = False
        f1 = m - 1
        f2 = N
        f3 = f2 * f1
        Gp = max(y_var) / sum(y_var)
        Gkr = get_cohren_critical(prob, f1, f2)
        print('-' * 100)
        if (Gkr > Gp):
            print("Gkr = {0} > Gp = {1} ---> Дисперсії
однорідні".format(Gkr, Gp))
            odnorid = True
        else:
            print("Gkr = {0} < Gp = {1} ---> Дисперсії неоднорідні,
збільшимо m і проведемо розрахунки".format(Gkr, Gp))
            Y_exp[0].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
            Y_exp[1].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
            Y_exp[2].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
            Y_exp[3].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
            m += 1

    # Стьюдент
    m = 3
    S2B = sum(y_var) / N
    S2b = S2B / (N * m)
    Sb = math.sqrt(S2b)
    beta0 = sum([y_aver[i] * x0_norm[i] for i in range(N)]) / N
    beta1 = sum([y_aver[i] * x1_norm[i] for i in range(N)]) / N
    beta2 = sum([y_aver[i] * x2_norm[i] for i in range(N)]) / N
    beta3 = sum([y_aver[i] * x3_norm[i] for i in range(N)]) / N
    t0 = abs(beta0) / Sb
    t1 = abs(beta1) / Sb
    t2 = abs(beta2) / Sb
    t3 = abs(beta3) / Sb
    tkr = student_teor(df=f3)

    d = sum([1 if tkr < i else 0 for i in [t0, t1, t2, t3]])
```

```
    a0 = a0 if tkr < t0 else 0
    a1 = a1 if tkr < t1 else 0
    a2 = a2 if tkr < t2 else 0
    a3 = a3 if tkr < t3 else 0

    y_new = [y_perevirka_abs(x_abs[0][i], x_abs[1][i], x_abs[2][i]) for i
in range(N)]

    print("-" * 100)
    f4 = N - d
    S2ad = (m / (N - d)) * sum([(y_new[i] - y_aver[i]) ** 2 for i in
range(N)])
    Fp = S2ad / S2b
    Fkr = fisher_teor(dfn=f4, dfd=f3)
    if (Fkr > Fp):
        print("Fkr = {0} > Fp = {1} ---> Р-ня адекватне
оригіналу".format(Fkr, Fp))
        adekvat = True
    else:
        N = 8
        print("Fkr = {0} < Fp = {1} ---> Р-ня неадекватне
оригіналу".format(Fkr, Fp))
        x0_factor = [1, 1, 1, 1, 1, 1, 1, 1]
        x1_factor = [-1, -1, -1, -1, 1, 1, 1, 1]
        x2_factor = [-1, -1, 1, 1, -1, -1, 1, 1]
        x3_factor = [-1, 1, -1, 1, -1, 1, -1, 1]
        x1x2_factor = [a * b for a, b in zip(x1_factor, x2_factor)]
        x1x3_factor = [a * b for a, b in zip(x1_factor, x3_factor)]
        x2x3_factor = [a * b for a, b in zip(x2_factor, x3_factor)]
        x1x2x3_factor = [a * b * c for a, b, c in zip(x1_factor, x2_factor,
x3_factor)]
        x0 = [1, 1, 1, 1, 1, 1, 1, 1]
        x1 = [-25, -25, -25, -25, -5, -5, -5, -5]
        x2 = [-70, -70, -10, -10, -70, -70, -10, -10]
        x3 = [-25, -5, -25, -5, -25, -5, -25, -5]
        x1x2 = [a * b for a, b in zip(x1, x2)]
        x1x3 = [a * b for a, b in zip(x1, x3)]
        x2x3 = [a * b for a, b in zip(x2, x3)]
        x1x2x3 = [a * b * c for a, b, c in zip(x1, x2, x3)]
        m = 3
        odnorid2 = False
        while not odnorid2:
            y1, y2, y3 = [], [], []
            for i in range(N):
                y1.append(r.randint(math.floor(Y_min), math.floor(Y_max)))
                y2.append(r.randint(math.floor(Y_min), math.floor(Y_max)))
                y3.append(r.randint(math.floor(Y_min), math.floor(Y_max)))
            Y_row_arr = [
                [y1[0], y2[0], y3[0]],
                [y1[1], y2[1], y3[1]],
                [y1[2], y2[2], y3[2]],
                [y1[3], y2[3], y3[3]],
                [y1[4], y2[4], y3[4]],
                [y1[5], y2[5], y3[5]],
                [y1[6], y2[6], y3[6]],
                [y1[7], y2[7], y3[7]]
            ]
            Y_row_av_arr = list(map(lambda x: np.average(x), Y_row_arr))
            Y_row_av_arr = list(map(lambda x: round(x, 3), Y_row_av_arr))
```

```
            column_names1 = ["X0", "X1", "X2", "X3", "X1X2", "X1X3",
"X2X3", "X1X2X3", "Y1", "Y2", "Y3"]
            pt2 = p.PrettyTable()  # Table
            pt2.add_column(column_names1[0], x0_factor)
            pt2.add_column(column_names1[1], x1_factor)
            pt2.add_column(column_names1[2], x2_factor)
            pt2.add_column(column_names1[3], x3_factor)
            pt2.add_column(column_names1[4], x1x2_factor)
            pt2.add_column(column_names1[5], x1x3_factor)
            pt2.add_column(column_names1[6], x2x3_factor)
            pt2.add_column(column_names1[7], x1x2x3_factor)
            pt2.add_column(column_names1[8], y1)
            pt2.add_column(column_names1[9], y2)
            pt2.add_column(column_names1[10], y3)
            print(pt2)

            list_for_solve_b = [x0_factor, x1_factor, x2_factor, x3_factor,
x1x2_factor, x1x3_factor, x2x3_factor,
                                x1x2x3_factor]
            list_for_solve_a = list(zip(x0, x1, x2, x3, x1x2, x1x3, x2x3,
x1x2x3))

            list_bi = []
            for k in range(N):
                S = 0
                for i in range(N):
                    S += (list_for_solve_b[k][i] * Y_row_av_arr[i]) / N
                list_bi.append(round(S, 5))

            print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 +
{}*x2x3 + {}*x1x2x3 \n".format(list_bi[0],

list_bi[1],

list_bi[2],

list_bi[3],

list_bi[4],

list_bi[5],

list_bi[6],

list_bi[7]))

            pt2 = p.PrettyTable()  # Table
            pt2.add_column(column_names1[0], x0)
            pt2.add_column(column_names1[1], x1)
            pt2.add_column(column_names1[2], x2)
            pt2.add_column(column_names1[3], x3)
            pt2.add_column(column_names1[4], x1x2)
            pt2.add_column(column_names1[5], x1x3)
            pt2.add_column(column_names1[6], x2x3)
            pt2.add_column(column_names1[7], x1x2x3)
            pt2.add_column(column_names1[8], y1)
            pt2.add_column(column_names1[9], y2)
            pt2.add_column(column_names1[10], y3)
            print(pt2)

            list_ai = [round(i, 5) for i in solve(list_for_solve_a,
```

```
Y_row_av_arr)]
            print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 +
{}*x2x3 + {}*x1x2x3".format(list_ai[0],

list_ai[1],

list_ai[2],

list_ai[3],

list_ai[4],

list_ai[5],

list_ai[6],

list_ai[7]))

            list_for_solve_b = [x0_factor, x1_factor, x2_factor, x3_factor,
x1x2_factor, x1x3_factor, x2x3_factor,
                                x1x2x3_factor]
            disp = []
            for k in range(N):
                disp.append(np.var(Y_row_arr[k]))
            # Кохрен
            y_var = [np.var(Y_row_arr[i]) for i in range(N)]
            f1 = m - 1
            f2 = N
            f3 = f2 * f1
            Gp = max(y_var) / sum(y_var)
            Gkr = get_cohren_critical(prob, f1, f2)
            print('-' * 100)
            if (Gkr > Gp):
                print("Gkr = {0} > Gp = {1} ---> Дисперсії
однорідні".format(Gkr, Gp))
                odnorid2 = True

            else:
                print("Gkr = {0} < Gp = {1} ---> Дисперсії неоднорідні,
збільшимо m і проведемо розрахунки".format(Gkr,

Gp))
                Y_exp[0].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
                Y_exp[1].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
                Y_exp[2].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
                Y_exp[3].append(r.randint(math.floor(Y_min),
math.floor(Y_max)))
                m += 1
    # Стьюдент
    beta = []
    Dispersion_B = sum(y_var) / N
    Dispersion_beta = Dispersion_B / (m * N)
    S_beta = math.sqrt(abs(Dispersion_beta))
    x_factor_for_solve = [x0_factor, x1_factor, x2_factor, x3_factor,
x1x2_factor, x1x3_factor, x2x3_factor,
                          x1x2x3_factor]
    for j in range(N):
        beta.append(sum([Y_row_av_arr[i] * x_factor_for_solve[j][i] for i
```

```
in range(N)]) / N)

    t_list = []
    for i in range(N):
        t_list.append(abs(beta[i]) / S_beta)

    f3 = f1 * f2
    d = 0
    T = student_teor(df=f3)


    def student(t_teor, t_pr):
        return t_pr < t_teor


    print("t табличне = ", T)
    for i in range(1, len(t_list)):
        if student(t_list[i], T):
            list_ai[i] = 0
            print("Гіпотеза підтверджена, beta{} = 0".format(i))
        else:
            print("Гіпотеза не підтверджена.\nbeta{} = {}".format(i,
list_ai[i]))
            d += 1
    Y_counted_for_Student = [0, 0, 0, 0, 0, 0, 0, 0]
    x_arr_for_solve = [x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3]
    for i in range(len(x_arr_for_solve) + 1):
        Y_counted_for_Student[i] += list_ai[0]
        for j in range(len(list_ai) - 1):
            Y_counted_for_Student[i] += list_ai[j + 1] *
x_arr_for_solve[j][i]

    f4 = N - d
    try:
        S2ad = (m / (N - d)) * sum([(Y_counted_for_Student[i] -
Y_row_av_arr[i]) ** 2 for i in range(N)])
    except:
        S2ad = 0

    Fp = S2ad / Dispersion_beta
    Ft = fisher_teor(dfn=f4, dfd=f3)
    print(Fp, Ft)
    if Ft > Fp:
        print("Рівняння регресії адекватне")
        adekvat = True
    else:
        print("Рівняння регресії неадекватне")
        print("Новий експеремент")
        print("#" * 100)
        N = 4
        k = 3
        odnorid2 = False
        odnorid = False
        counter += 1
```

Результат роботи програми:

```
+----------------------+--------+
|       Студент        | Группа |
+----------------------+--------+
| Карнаухова Анастасія | IO-92  |
+----------------------+--------+
Абсолютні значення: [[-25, -25, -5, -5], [-70, -10, -70, -10], [-25, -5, -5, -25]]
Нормалізована матриця:
 +----+----+----+----+-----+-----+-----+
| X0 | X1 | X2 | X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+-----+-----+-----+
| 1  | -1 | -1 | -1 | 169 | 161 | 169 |
| 1  | -1 | 1  | 1  | 169 | 186 | 166 |
| 1  | 1  | -1 | 1  | 188 | 173 | 176 |
| 1  | 1  | 1  | -1 | 180 | 163 | 166 |
+----+----+----+----+-----+-----+-----+
Рівняння регресії для нормованих значень:
y = 172.1666666666666 + 2.1666666666666643*x1 + -0.5000000000000071*x2 + 4.166666666666664*x3
Рівняння регресії для абсолютних значень:
y = 180.99999999999991 + 0.21666666666666642*x1 + -0.01666666666666902*x2 + 0.4166666666666664*x3
----------------------------------------------------------------------------------------------------
Gkr = 0.7679205583193613 > Gp = 0.41107184923439344 ---> Дисперсії однорідні
----------------------------------------------------------------------------------------------------
Fkr = 4.06618055135116 < Fp = 102.10600706713629 ---> Р-ня неадекватне оригіналу
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| X0 | X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| 1  | -1 | -1 | -1 |  1   |  1   |  1   |   -1   | 189 | 174 | 164 |
| 1  | -1 | -1 | 1  |  1   |  -1  |  -1  |   1    | 169 | 172 | 165 |
| 1  | -1 | 1  | -1 |  -1  |  1   |  -1  |   1    | 177 | 182 | 169 |
| 1  | -1 | 1  | 1  |  -1  |  -1  |  1   |   -1   | 172 | 178 | 181 |
| 1  | 1  | -1 | -1 |  -1  |  -1  |  1   |   1    | 168 | 181 | 164 |
| 1  | 1  | -1 | 1  |  -1  |  1   |  -1  |   -1   | 187 | 192 | 163 |
| 1  | 1  | 1  | -1 |  1   |  -1  |  -1  |   -1   | 161 | 179 | 170 |
| 1  | 1  | 1  | 1  |  1   |  1   |  1   |   1    | 183 | 184 | 190 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
y = 175.5835 + 1.25*x1 + 1.58325*x2 + 2.41675*x3 + -0.58325*x1x2 + 3.91675*x1x3 + 1.75*x2x3 + -0.25*x1x2x3

+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| X0 |  X1 |  X2 |  X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| 1  | -25 | -70 | -25 | 1750 | 625  | 1750 | -43750 | 189 | 174 | 164 |
| 1  | -25 | -70 |  -5 | 1750 | 125  | 350  | -8750  | 169 | 172 | 165 |
| 1  | -25 | -10 | -25 | 250  | 625  | 250  | -6250  | 177 | 182 | 169 |
| 1  | -25 | -10 |  -5 | 250  | 125  |  50  | -1250  | 172 | 178 | 181 |
| 1  |  -5 | -70 | -25 | 350  | 125  | 1750 | -8750  | 168 | 181 | 164 |
| 1  |  -5 | -70 |  -5 | 350  |  25  | 350  | -1750  | 187 | 192 | 163 |
| 1  |  -5 | -10 | -25 |  50  | 125  | 250  | -1250  | 161 | 179 | 170 |
| 1  |  -5 | -10 |  -5 |  50  |  25  |  50  |  -250  | 183 | 184 | 190 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
y = 193.59081 + 0.58475*x1 + 0.09236*x2 + 1.01252*x3 + -0.00319*x1x2 + 0.03583*x1x3 + 0.00458*x2x3 + -8e-05*x1x2x3
----------------------------------------------------------------------------------------------------
Gkr = 0.5156874570365015 > Gp = 0.370123203285421 ---> Дисперсії однорідні
t табличне =  2.119905299221011
Гіпотеза не підтверджена.
beta1 = 0.58475
Гіпотеза не підтверджена.
beta2 = 0.09236
Гіпотеза не підтверджена.
beta3 = 1.01252
Гіпотеза не підтверджена.
beta4 = -0.00319
Гіпотеза підтверджена, beta5 = 0
Гіпотеза не підтверджена.
beta6 = 0.00458
Гіпотеза не підтверджена.
beta7 = -8e-05
```

```
beta7 = -8e-05
727.2070835796387 3.63372346759163
Рівняння регресії неадекватне
Новий експеремент
########################################################################################################
Нормалізована матриця:
 +----+----+----+----+-----+-----+-----+
| X0 | X1 | X2 | X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+-----+-----+-----+
| 1  | -1 | -1 | -1 | 169 | 161 | 169 |
| 1  | -1 | 1  | 1  | 169 | 186 | 166 |
| 1  | 1  | -1 | 1  | 188 | 173 | 176 |
| 1  | 1  | 1  | -1 | 180 | 163 | 166 |
+----+----+----+----+-----+-----+-----+
Рівняння регресії для нормованих значень:
y = 172.1666666666666 + 2.1666666666666643*x1 + -0.5000000000000071*x2 + 4.166666666666664*x3
Рівняння регресії для абсолютних значень:
y = 180.99999999999991 + 0.2166666666666642*x1 + -0.01666666666666902*x2 + 0.4166666666666664*x3
--------------------------------------------------------------------------------------------------------
Gkr = 0.7679205583193613 > Gp = 0.41107184923439344 ---> Дисперсії однорідні
--------------------------------------------------------------------------------------------------------
Fkr = 4.06618055135116 < Fp = 102.10600706713629 ---> Р-ня неадекватне оригіналу
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| X0 | X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| 1  | -1 | -1 | -1 |  1   |  1   |  1   |   -1   | 164 | 190 | 186 |
| 1  | -1 | -1 | 1  |  1   |  -1  |  -1  |   1    | 162 | 174 | 178 |
| 1  | -1 | 1  | -1 |  -1  |  1   |  -1  |   1    | 190 | 178 | 180 |
| 1  | -1 | 1  | 1  |  -1  |  -1  |  1   |   -1   | 163 | 179 | 160 |
| 1  | 1  | -1 | -1 |  -1  |  -1  |  1   |   1    | 162 | 181 | 186 |
| 1  | 1  | -1 | 1  |  -1  |  1   |  -1  |   -1   | 161 | 182 | 165 |
| 1  | 1  | 1  | -1 |  1   |  -1  |  -1  |   -1   | 174 | 169 | 162 |
| 1  | 1  | 1  | 1  |  1   |  1   |  1   |   1    | 173 | 168 | 168 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
y = 173.12488 + -2.20838*x1 + -1.12487*x2 + -3.70838*x3 + -0.79163*x1x2 + 2.29188*x1x3 + 0.20838*x2x3 + 1.87513*x1x2x3

+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| X0 |  X1 |  X2 |  X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| 1  | -25 | -70 | -25 | 1750 | 625  | 1750 | -43750 | 164 | 190 | 186 |
| 1  | -25 | -70 |  -5 | 1750 | 125  | 350  | -8750  | 162 | 174 | 178 |
| 1  | -25 | -10 | -25 | 250  | 625  | 250  | -6250  | 190 | 178 | 180 |
| 1  | -25 | -10 |  -5 | 250  | 125  |  50  | -1250  | 163 | 179 | 160 |
| 1  |  -5 | -70 | -25 | 350  | 125  | 1750 | -8750  | 162 | 181 | 186 |
| 1  |  -5 | -70 |  -5 | 350  |  25  | 350  | -1750  | 161 | 182 | 165 |
| 1  |  -5 | -10 | -25 |  50  | 125  | 250  | -1250  | 174 | 169 | 162 |
| 1  |  -5 | -10 |  -5 |  50  |  25  |  50  |  -250  | 173 | 168 | 168 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
y = 172.36551 + 0.39242*x1 + 0.07398*x2 + 0.37575*x3 + 0.00674*x1x2 + 0.04792*x1x3 + 0.01007*x2x3 + 0.00063*x1x2x3
--------------------------------------------------------------------------------------------------------
Gkr = 0.5156874570365015 > Gp = 0.2647456100855471 ---> Дисперсії однорідні
t табличне =  2.119905299221011
Гіпотеза не підтверджена.
beta1 = 0.39242
Гіпотеза не підтверджена.
beta2 = 0.07398
Гіпотеза підтверджена, beta3 = 0
Гіпотеза не підтверджена.
beta4 = 0.00674
Гіпотеза не підтверджена.
beta5 = 0.04792
Гіпотеза не підтверджена.
beta6 = 0.01007
Гіпотеза не підтверджена.
beta7 = 0.00063
210.9161133065463 3.63372346759163
Рівняння регресії неадекватне
Новий експеремент
```

```
########################################################################################################################
Нормалізована матриця:
 +----+----+----+----+-----+-----+-----+
| X0 | X1 | X2 | X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+-----+-----+-----+
| 1  | -1 | -1 | -1 | 169 | 161 | 169 |
| 1  | -1 | 1  | 1  | 169 | 186 | 166 |
| 1  | 1  | -1 | 1  | 188 | 173 | 176 |
| 1  | 1  | 1  | -1 | 180 | 163 | 166 |
+----+----+----+----+-----+-----+-----+
Рівняння регресії для нормованих значень:
y = 172.1666666666666 + 2.1666666666666643*x1 + -0.500000000000071*x2 + 4.166666666666664*x3
Рівняння регресії для абсолютних значень:
y = 180.99999999999991 + 0.21666666666666642*x1 + -0.016666666666666902*x2 + 0.4166666666666664*x3
------------------------------------------------------------------------------------------------------------------------
Gkr = 0.7679205583193613 > Gp = 0.41107184923439344 ---> Дисперсії однорідні
------------------------------------------------------------------------------------------------------------------------
Fkr = 4.06618055135116 < Fp = 102.10600706713629 ---> Р-ня неадекватне оригіналу
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| X0 | X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
| 1  | -1 | -1 | -1 |  1   |  1   |  1   |   -1   | 162 | 174 | 162 |
| 1  | -1 | -1 | 1  |  1   |  -1  |  -1  |   1    | 191 | 181 | 165 |
| 1  | -1 | 1  | -1 |  -1  |  1   |  -1  |   1    | 179 | 175 | 166 |
| 1  | -1 | 1  | 1  |  -1  |  -1  |  1   |   -1   | 190 | 169 | 162 |
| 1  | 1  | -1 | -1 |  -1  |  -1  |  1   |   1    | 191 | 189 | 162 |
| 1  | 1  | -1 | 1  |  -1  |  1   |  -1  |   -1   | 167 | 186 | 185 |
| 1  | 1  | 1  | -1 |  1   |  -1  |  -1  |   -1   | 167 | 175 | 163 |
| 1  | 1  | 1  | 1  |  1   |  1   |  1   |   1    | 193 | 193 | 171 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+
y = 175.75 + 2.75*x1 + -0.5*x2 + 3.66675*x3 + -1.0*x1x2 + 0.33325*x1x3 + 0.75025*x2x3 + 3.91675*x1x2x3

 +----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| X0 |  X1 |  X2 |  X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
| 1  | -25 | -70 | -25 | 1750 | 625  | 1750 | -43750 | 162 | 174 | 162 |
| 1  | -25 | -70 |  -5 | 1750 | 125  | 350  | -8750  | 191 | 181 | 165 |
| 1  | -25 | -10 | -25 | 250  | 625  | 250  | -6250  | 179 | 175 | 166 |
| 1  | -25 | -10 |  -5 | 250  | 125  |  50  | -1250  | 190 | 169 | 162 |
| 1  |  -5 | -70 | -25 | 350  | 125  | 1750 | -8750  | 191 | 189 | 162 |
| 1  |  -5 | -70 |  -5 | 350  |  25  | 350  | -1750  | 167 | 186 | 185 |
| 1  |  -5 | -10 | -25 |  50  | 125  | 250  | -1250  | 167 | 175 | 163 |
| 1  |  -5 | -10 |  -5 |  50  |  25  |  50  |  -250  | 193 | 193 | 171 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+
y = 196.70902 + 0.975*x1 + 0.2646*x2 + 1.30005*x3 + 0.01625*x1x2 + 0.05556*x1x3 + 0.02208*x2x3 + 0.00131*x1x2x3
------------------------------------------------------------------------------------------------------------------------
Gkr = 0.5156874570365015 > Gp = 0.24936628643852982 ---> Дисперсії однорідні
t табличне =  2.119905299221011
Гіпотеза не підтверджена.
beta1 = 0.975
Гіпотеза не підтверджена.
beta2 = 0.2646
Гіпотеза не підтверджена.
beta3 = 1.30005
Гіпотеза не підтверджена.
beta4 = 0.01625
Гіпотеза не підтверджена.
beta5 = 0.05556
Гіпотеза не підтверджена.
beta6 = 0.02208
Гіпотеза не підтверджена.
beta7 = 0.00131
0.03637435335056443 4.493998477666352
Рівняння регресії адекватне
```

Висновок:

Проробивши лабораторну роботу, було проведено повний трьохфакторний експеримент. Та знайдено рівняння регресії адекватне об'єкту. У ході виконання лабораторної роботи проблем не виникло. Результати виконання лабораторної висвітлені на роздруківках.