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

Лабораторна робота №2

з дисципліни “Методи оптимізації та планування експерименту”

на тему: «ПРОВЕДЕННЯ ДВОФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

Виконала:
студентка групи ІВ-81
Дьяченко Т. С.
Перевірив:
Регіда П. Г.

Київ
2020 р.

Варіант:

112	-40	20	-35	15

Код програми:

```
        #ymin = -920    ymax = -820
import random, math
m = 6
cr_sygma = math.sqrt((4 * m - 4) / (m * m - 4 * m))
js = 3
je = m + 3
table = [["N", "x1", "x2"],
          [1, -1, -1],
          [2, -1, 1],
          [3, 1, -1],
          [4, 1, 1]]
x1min = -40
x1max = 20
x2min = -35
x2max = 15
R_table = [["p\m", 2, 6, 8, 10, 12, 15, 20],
            [0.99, 1.73, 2.16, 2.43, 2.62, 2.75, 2.9, 3.08],
            [0.98, 1.72, 2.13, 2.37, 2.54, 2.66, 2.8, 2.96],
            [0.95, 1.71, 2.1, 2.27, 2.41, 2.52, 2.64, 2.78],
            [0.9, 1.69, 2, 2.17, 2.29, 2.39, 2.49, 2.62]]
mn = 2
pn = 4
def test(s, e, m, cr_s):
    for i in range(5):
        for j in range(s, e):
            if i == 0:
                table[i].append("Yi{}".format(j-2))
            else:
                table[i].append(random.uniform(-920, -820))
    sigma = []
    for i in range(1, 5):
        my = 0
        for j in range(3, m+3):
            my += table[i][j]
        my = my / m
        sy = 0
        for j in range(3, m+3):
            sy += pow(table[i][j] - my, 2)
        sigma.append(sy / m)
    f12 = max(sigma[0], sigma[1]) / min(sigma[0], sigma[1])
    f13 = max(sigma[0], sigma[2]) / min(sigma[0], sigma[2])
    f23 = max(sigma[2], sigma[1]) / min(sigma[2], sigma[1])
    teta12 = (m - 2) * f12 / m
    teta13 = (m - 2) * f13 / m
    teta23 = (m - 2) * f23 / m
    r12 = math.fabs(teta12 - 1) / cr_s
    r13 = math.fabs(teta13 - 1) / cr_s
    r23 = math.fabs(teta23 - 1) / cr_s
    return [r12, r13, r23]
def print_table():
    for i in range(len(table)):
        print("|", end="")
        for j in range(len(table[i])):
            if i > 0 and j > 2:
                print("{:.2f}".format(float(table[i][j])), end="    |")
            else:
                print(table[i][j], " "*(9-len(str(table[i][j]))), end="|")
```

```
        print("\n", "-" * (m+3)*11, "\n")
def coef():
    mx1 = (table[1][1] + table[2][1] + table[3][1])/3
    mx2 = (table[1][2] + table[2][2] + table[3][2]) /3
    my_l = []
    for i in range(1, 4):
        myi = 0
        for j in range(3, m + 3):
            myi += table[i][j]
        my_l.append(myi / m)
    my = sum(my_l)/3
    a1 = (pow(table[1][1], 2)+pow(table[2][1], 2)+pow(table[3][1], 2))/3
    a2 = (table[1][1]*table[1][2]+table[2][1]*table[2][2]+table[3][1]*table[3][2])/3
    a3 = (pow(table[1][2], 2) + pow(table[2][2], 2) + pow(table[3][2], 2)) / 3
    a11 = (table[1][1]*my_l[0]+table[2][1]*my_l[1]+table[3][1]*my_l[2])/3
    a22 = (table[1][2] * my_l[0] + table[2][2] * my_l[1] + table[3][2] * my_l[2]) / 3
    det = a1*a3+mx2*mx1*a2*2-a1*mx2*mx2-a3*mx1*mx1-a2*a2
    c_b0 = (my*a1*a3+(a22*mx1+mx2*a11)*a2-a1*mx2*a22-my*a2*a2-mx1*a11*a3)/det
    c_b1 = (a11 * a3 + (a22 * mx1 + a2 * my) * mx2 - a11 * mx2 * mx2 - my * a3 * mx1
- a22 * a2) / det
    c_b2 = (a1 * a22 + (a11 * mx2 + a2 * my) * mx1 - a1 * mx2 * my - mx1 * mx1 * a22
-  a11 * a2) / det
    print("Normalized equation:\ny = {:.2f} {:+.2f}*x1 {:+.2f}*x2".format(c_b0 ,
c_b1, c_b2))
    print("\nCheck:")
    for i in range(1, 4):
        print("{:.2f} {:+.2f} {:+.2f} = {:.2f}\ny = {:.2f}\n".format(c_b0,
c_b1*table[i][1], c_b2*table[i][2], c_b0+c_b1*table[i][1]+c_b2*table[i][2], my_l[i-
1]))
    x1 = abs(x1max-x1min)/2
    x2 = abs(x2max-x2min)/2
    x10 = (x1max+x1min)/2
    x20 = (x2max+x2min)/2
    c_a0 = c_b0 - c_b1*x10/x1 - c_b2*x20/x2
    c_a1 = c_b1/x1
    c_a2 = c_b2/x2
    print("\n\nNaturalized equation:\ny = {:.2f} {:+.2f}*x1 {:+.2f}*x2".format(c_a0,
c_a1, c_a2))
    print("\nCheck:")
    print("{:.2f} {:+.2f} {:+.2f} = {:.2f}\ny = {:.2f}\n".format(c_a0, c_a1 * x1min,
c_a2 * x2min, c_a0 + c_a1 * x1min + c_a2 * x2min, my_l[0]))
    print("{:.2f} {:+.2f} {:+.2f} = {:.2f}\ny = {:.2f}\n".format(c_a0, c_a1 * x1min,
c_a2 * x2max, c_a0 + c_a1 * x1min + c_a2 * x2max, my_l[1]))
    print("{:.2f} {:+.2f} {:+.2f} = {:.2f}\ny = {:.2f}\n".format(c_a0, c_a1 * x1max,
c_a2 * x2min, c_a0 + c_a1 * x1max + c_a2 * x2min, my_l[2]))
random.seed()
found = False
while not found:
    Rs = test(js, je, m, cr_sygma)
    while pn > 0:
        if Rs[0] < R_table[pn][mn] and Rs[1] < R_table[pn][mn] and Rs[2] <
R_table[pn][mn]:
            found = True
            break
        else:
            pn -= 1
    if found:
        print_table()
        print("m = {}\nR12 = {}\tR13 = {}\tR23 = {} < {}\n".format(m, Rs[0], Rs[1],
Rs[2], R_table[pn][mn]))
        coef()
        print("p = {}".format(R_table[pn][0]))
    elif mn < 8:
        mn += 1
```

```
            m = R_table[0][mn]
            js = je
            je = m + 3
            cr_sygma = math.sqrt((4 * m - 4) / (m * m - 4 * m))
            Rs = test(js, je, m, cr_sygma)
        else:
            print("insufficient data!")
            break
```

Результат виконання:

```
main (1)
/home/miya/PycharmProjects/TPE_2/venv/bin/python /home/miya/PycharmProjects/TPE_2/main.py
|N         |x1        |x2        |Yi1       |Yi2       |Yi3       |Yi4       |Yi5       |Yi6       |
 ---------------------------------------------------------------------------------------------------

|1         |-1        |-1        |-877.91   |-845.92   |-882.42   |-865.13   |-832.35   |-849.76   |
 ---------------------------------------------------------------------------------------------------

|2         |-1        |1         |-878.09   |-899.32   |-892.33   |-899.47   |-855.35   |-902.18   |
 ---------------------------------------------------------------------------------------------------

|3         |1         |-1        |-837.20   |-878.45   |-823.26   |-820.15   |-899.97   |-907.10   |
 ---------------------------------------------------------------------------------------------------

|4         |1         |1         |-835.56   |-897.88   |-878.78   |-887.35   |-838.97   |-866.83   |
 ---------------------------------------------------------------------------------------------------

m = 6
R12 = 0.1743971696597467    R13 = 1.2820470636583807    R23 = 1.6158019504295473 < 2

Normalized equation:
y = -874.41 -1.05*x1 -14.44*x2

Check:
-874.41 +1.05 +14.44 = -858.92
y = -858.92

-874.41 +1.05 -14.44 = -887.79
y = -887.79

-874.41 -1.05 +14.44 = -861.02
y = -861.02



Naturalized equation:
y = -880.53 -0.04*x1 -0.58*x2

Check:
-880.53 +1.40 +20.21 = -858.92
y = -858.92

-880.53 +1.40 -8.66 = -887.79
y = -887.79

-880.53 -0.70 +20.21 = -861.02
y = -861.02

p = 0.9

Process finished with exit code 0
```