Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 3

з дисципліни «Методи наукових досліджень»

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

ВИКОНАВ:

студент 2 курсу

групи ІВ-91

Щоткін М. А.

Залікова – 9131

ПЕРЕВІРИВ:

ас. Регіда П. Г.

Київ – 2021

Лабораторна робота № 3

Тема: ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ.

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

Завдання на лабораторну роботу

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.

130	10	50	20	60	20	25

Програмний код

```
from random import randint
from functools import reduce
from numpy.linalg import det


def naturalize(matrix_of_plan, min_max_arr):
    result = []
    for i in matrix_of_plan:
        result.append(min_max_arr[1]) if i == 1 else result.append(min_max_arr[0])
    return result


def main():
    x1 = [10, 50]
    x2 = [20, 60]
    x3 = [20, 25]

    print("x1: ", x1)
    print("x2: ", x2)
    print("x3: ", x3)
```

```
    x0_plan_array = [1, 1, 1, 1]
    x1_plan_array = [-1, -1, 1, 1]
    x2_plan_array = [-1, 1, -1, 1]
    x3_plan_array = [-1 * (x1_plan_array[i] * x2_plan_array[i]) for i in
range(len(x1_plan_array))]

    print("\nx0:", x0_plan_array)
    print("x1:", x1_plan_array)
    print("x2:", x2_plan_array)
    print("x3:", x3_plan_array)

    x1_plan_naturalized = naturalize(x1_plan_array, x1)
    x2_plan_naturalized = naturalize(x2_plan_array, x2)
    x3_plan_naturalized = naturalize(x3_plan_array, x3)

    print('\nx1:', x1_plan_naturalized)
    print('x2:', x2_plan_naturalized)
    print('x3:', x3_plan_naturalized)

    x_avg_max = (max(x1_plan_naturalized) + max(x2_plan_naturalized) +
max(x3_plan_naturalized)) / 3
    x_avg_min = (min(x1_plan_naturalized) + min(x2_plan_naturalized) +
min(x3_plan_naturalized)) / 3

    print("\nx_avg_max = ", x_avg_max)
    print("x_avg_min = ", x_avg_min)

    y_min = int(200 + x_avg_min)
    y_max = int(200 + x_avg_max)

    print("\ny_max = ", y_max)
    print("y_min = ", y_min)

    y1 = [randint(y_min, y_max) for i in range(4)]
    y2 = [randint(y_min, y_max) for i in range(4)]
    y3 = [randint(y_min, y_max) for i in range(4)]

    print("\ny1:", y1)
    print("y2:", y2)
    print("y3:", y3)

    y_avg_array = [(y1[i] + y2[i] + y3[i]) / 3 for i in range(4)]
    print("\nAverage y: ", y_avg_array)

    mx1 = reduce(lambda a, b: a + b, x1_plan_naturalized) / 4
    mx2 = reduce(lambda a, b: a + b, x2_plan_naturalized) / 4
    mx3 = reduce(lambda a, b: a + b, x3_plan_naturalized) / 4
    my = reduce(lambda a, b: a + b, y_avg_array) / 4

    print("\nmx1 = ", mx1)
    print("mx2 = ", mx2)
    print("mx3 = ", mx3)
    print("my = ", my)

    a1 = sum([x1_plan_naturalized[i] * y_avg_array[i] for i in range(4)]) / 4
    a2 = sum([x2_plan_naturalized[i] * y_avg_array[i] for i in range(4)]) / 4
    a3 = sum([x3_plan_naturalized[i] * y_avg_array[i] for i in range(4)]) / 4

    a11 = sum([i * i for i in x1_plan_naturalized]) / 4
    a22 = sum([i * i for i in x2_plan_naturalized]) / 4
    a33 = sum([i * i for i in x3_plan_naturalized]) / 4
```

```
    a12 = sum([x1_plan_naturalized[i] * x2_plan_naturalized[i] for i in range(4)])
/ 4
    a13 = sum([x1_plan_naturalized[i] * x3_plan_naturalized[i] for i in range(4)])
/ 4
    a23 = sum([x2_plan_naturalized[i] * x3_plan_naturalized[i] for i in range(4)])
/ 4

    a21 = a12
    a31 = a13
    a32 = a23

    print("\na1 = ", a1)
    print("a2 = ", a2)
    print("a3 = ", a3)

    print("\na11 = ", a11)
    print("a22 = ", a22)
    print("a33 = ", a33)

    print("\na12 = ", a12)
    print("a13 = ", a13)
    print("a23 = ", a23)

    print("\na21 = ", a21)
    print("a31 = ", a31)
    print("a32 = ", a32)

    b0 = det([[my, mx1, mx2, mx3],
              [a1, a11, a12, a13],
              [a2, a21, a22, a23],
              [a3, a31, a32, a33]]) / det([[1, mx1, mx2, mx3],
                                           [mx1, a11, a12, a13],
                                           [mx2, a21, a22, a23],
                                           [mx3, a31, a32, a33]])
    b1 = det([[1, my, mx2, mx3],
              [mx1, a1, a12, a13],
              [mx2, a2, a22, a23],
              [mx3, a3, a32, a33]]) / det([[1, mx1, mx2, mx3],
                                           [mx1, a11, a12, a13],
                                           [mx2, a21, a22, a23],
                                           [mx3, a31, a32, a33]])
    b2 = det([[1, mx1, my, mx3],
              [mx1, a11, a1, a13],
              [mx2, a21, a2, a23],
              [mx3, a31, a3, a33]]) / det([[1, mx1, mx2, mx3],
                                           [mx1, a11, a12, a13],
                                           [mx2, a21, a22, a23],
                                           [mx3, a31, a32, a33]])
    b3 = det([[1, mx1, mx2, my],
              [mx1, a11, a12, a1],
              [mx2, a21, a22, a2],
              [mx3, a31, a32, a3]]) / det([[1, mx1, mx2, mx3],
                                           [mx1, a11, a12, a13],
                                           [mx2, a21, a22, a23],
                                           [mx3, a31, a32, a33]])

    print("\ny = b0 + b1*x1 + b2*x2 + b3*x3")
    print(f"y = {b0} + {b1}*x1 + {b2}*x2 + b3*x3")

    for i in range(4):
        y = b0 + b1 * x1_plan_naturalized[i] + b2 * x2_plan_naturalized[i] + b3 *
```

```
x3_plan_naturalized[i]

    dispersion = [((y1[i] - y_avg_array[i]) ** 2 + (y2[i] - y_avg_array[i]) ** 2 +
(y3[i] - y_avg_array[i]) ** 2) / 3
                  for i in
                  range(4)]

    print("\ndispersion: ", dispersion)

    gp = max(dispersion) / sum(dispersion)

    print("\nКоефіцієнт Gp = ", gp)

    m = 3
    # f1=m-1=2, f2=N=4, q=0.05 => Gт=0.7679 за таблицею
    if gp > 0.7679:
        print("\nДисперсія неоднорідна!")
        exit()

    print("\nGp < 0.7679 => Дисперсія однорідна")

    #КРИТЕРІЙ СТЬЮДЕНТА
    s2b = sum(dispersion) / 4
    s2bs_avg = s2b / (4 * m)
    sb = s2bs_avg ** (1 / 2)

    beta0 = sum([y_avg_array[i] * x0_plan_array[i] for i in range(4)]) / 4
    beta1 = sum([y_avg_array[i] * x1_plan_array[i] for i in range(4)]) / 4
    beta2 = sum([y_avg_array[i] * x2_plan_array[i] for i in range(4)]) / 4
    beta3 = sum([y_avg_array[i] * x3_plan_array[i] for i in range(4)]) / 4

    beta_array = [beta0, beta1, beta2, beta3]

    print("\nbeta: ", beta_array)

    t_array = [abs(beta_array[i]) / sb for i in range(4)]

    print("\nt: ", t_array)
    print()


    d = 0
    indexes = []
    for i, v in enumerate(t_array):
        if t_array[i] > 2.306:
            indexes.append(i)
            d += 1
        else:
            print(f"Коефіцієнт b{i} = {v} є статистично незначущим і його слід
виключити з рівняння регресії.")

    b_array = [b0, b1, b2, b3]

    b_result = [b_array[indexes[0]] for i in range(4)]

    #КРИТЕРІЙ ФІШЕРА
    s2_ad = m * sum([(y_avg_array[i] - b_result[i]) ** 2 for i in range(4)]) / (4 -
d)
    fp = s2_ad / s2b

    print("\nFp = ", fp)
```

```
    if fp < 4.1:
        print("Fp < Fт.")
    else:
        print("Fp > Fт. ")


if __name__ == '__main__':
    main()
```

Результат програми

```
C:\Users\maksy\PycharmProjects\MND\venv\Scripts\python.exe C:/Users/maksy/PycharmProjects/MND/MND3.py
x1:  [10, 50]
x2:  [20, 60]
x3:  [20, 25]

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

x1: [10, 10, 50, 50]
x2: [20, 60, 20, 60]
x3: [20, 25, 25, 20]

x_avg_max =  45.0
x_avg_min =  16.666666666666668

y_max =  245
y_min =  216

y1: [219, 219, 236, 222]
y2: [238, 241, 226, 244]
y3: [224, 229, 232, 245]

Average y:  [227.0, 229.66666666666666, 231.33333333333334, 237.0]

mx1 =  30.0
mx2 =  40.0
mx3 =  22.5
my =  231.25
```

```
a1 =  6995.833333333334
a2 =  9291.666666666668
a3 =  5201.25

a11 =  1300.0
a22 =  2000.0
a33 =  512.5

a12 =  1200.0
a13 =  675.0
a23 =  900.0

a21 =  1200.0
a31 =  675.0
a32 =  900.0

y = b0 + b1*x1 + b2*x2 + b3*x3
y = 229.45833333333414 + 0.14583333333333495*x1 + 0.1041666666666705*x2 + b3*x3

dispersion:  [64.66666666666667, 80.88888888888889, 16.88888888888889, 112.66666666666667]

Коефіцієнт Gp =  0.40953150242326336

Gp < 0.7679 => Дисперсія однорідна

beta:  [231.25, 2.9166666666666785, 2.0833333333333286, -0.75]

t:  [96.5935871701168, 1.218297495839316, 0.8702124970280773, 0.31327649893010856]
```

```
Коефіцієнт b1 = 1.218297495839316 є статистично незначущим і його слід виключити з рівняння регресії.
Коефіцієнт b2 = 0.8702124970280773 є статистично незначущим і його слід виключити з рівняння регресії.
Коефіцієнт b3 = 0.31327649893010856 є статистично незначущим і його слід виключити з рівняння регресії.

Fp =  0.9665791599352125
Fp < Fт. Отримана математична модель з прийнятим рівнем статистичної значимості q адекватна оригіналу

Process finished with exit code 0
```