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 ЕКСПЕРИМЕТНУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАВ:

студент 2 курсу

групи ІВ-91

Степанюк Р. В.

Залікова – 9127

ПЕРЕВІРИВ:

ас. Регіда П. Г.

Київ – 2021

**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).

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.

3. Провести 3 статистичні перевірки.

4. Написати комп'ютерну програму, яка усе це виконує.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № варіанту | x1 | | x2 | | x3 | |
| min | max | min | max | min | max |
| 125 | -25 | -5 | 25 | 45 | 25 | 30 |

**Програмний код**

# Методи наукових досліджень

#

# Степанюк Роман Вікторович ІВ-91 ФІОТ

#

# Варіант 225

import random, math, numpy as np

*def* average(*list*):

    average = 0

    for element in *list*:

        average += element / len(*list*)

    return average

*def* dispersion(*list*):

    list\_average = average(*list*)

    dispersion = 0

    for element in *list*:

        dispersion += (element - list\_average)\*\*2 / len(*list*)

    return dispersion

*def* cochrane\_criteria():

    global m, N

    # 7.1. Знайдемо дисперсії по рядках

    dispersion\_list = [dispersion(y1), dispersion(y2), dispersion(y3), dispersion(y4)]

    print(f"Дисперсії по рядках:\n{dispersion\_list}")

    gp\_denominator = 0

    for disp in dispersion\_list:

        gp\_denominator += disp

    gp = max(dispersion\_list) / gp\_denominator

    # print(f"Gp = {round(gp, 3)}")

    # 7.2. Рівень значимості приймемо за 0.05

    f1 = m - 1

    f2 = N

    gt = 0.7679

    if gp < gt: return True

    else: return False

*def* students\_criteria():

    global m, N

    sb = average(dispersion\_list)

    # print(f"\nSᵦ\u00b2 = {sb}")

    s\_beta\_2 = sb / (N \* m)

    # print("S\u00b2{β} = ", s\_beta\_2)

    s\_beta = math.sqrt(s\_beta\_2)

    # print("S {β} = ", s\_beta)

    beta0 = 0

    for i in range(N):

        beta0 += average\_list[i] \* x0[i] / N

    beta1 = 0

    for i in range(N):

        beta1 += average\_list[i] \* x1[i] / N

    beta2 = 0

    for i in range(N):

        beta2 += average\_list[i] \* x2[i] / N

    beta3 = 0

    for i in range(N):

        beta3 += average\_list[i] \* x3[i] / N

    # print(f"β\u2080 = {beta0}\nβ\u2081 = {beta1}\nβ\u2082 = {beta2}\nβ\u2083 = {beta3}")

    t0 = abs(beta0) / s\_beta

    t1 = abs(beta1) / s\_beta

    t2 = abs(beta2) / s\_beta

    t3 = abs(beta3) / s\_beta

    # print(f"\nt\u2080 = {t0}\nt\u2081 = {t1}\nt\u2082 = {t2}\nt\u2083 = {t3}")

    f3 = (m - 1) \* N

    tt = 2.306

    student\_check = {}

    if (t0 > tt): student\_check[0] = b[0]

    else: student\_check[0] = 0

    if (t1 > tt): student\_check[1] = b[1]

    else: student\_check[1] = 0

    if (t2 > tt): student\_check[2] = b[2]

    else: student\_check[2] = 0

    if (t3 > tt): student\_check[3] = b[3]

    else: student\_check[3] = 0

    return student\_check

*def* fisher\_criteria():

    global m, N

    d = 0

    for key in std\_ch:

        if std\_ch[key] != 0: d += 1

    f1 = m - 1

    f2 = N

    f3 = (m - 1) \* N

    f4 = N - d

    s2\_ad = 0

    for i in range(N):

        s2\_ad += (y\_std[i] - average\_list[i])\*\*2

    s2\_ad \*=  m / f4

    # print(s2\_ad)

    s2\_b = average(dispersion\_list)

    fp = s2\_ad / s2\_b

    # print(f"Fp = {fp}")

    ft = 4.5

    if (fp > ft): print("Fp > Fт\nОтже, рівняння регресії неадекватно оригіналу при рівні значимості 0.05")

    else: print("Fp < Fт\nРівняння регресії адекватно оригіналу")

x\_min = [-25, 25, 25]

x\_max = [-5, 45, 30]

y\_min = 200 + average(x\_min)

y\_max = 200 + average(x\_max)

x0 = [1, 1, 1, 1]

x1 = [-1, -1, 1, 1]

x2 = [-1, 1, -1, 1]

x3 = [-1, 1, 1, -1]

print(f"Мінімальне значення ф-ції відгуку: {int(y\_min)}")

print(f"Максимальне значення ф-ції відгуку: {int(y\_max)}")

# 1. Матриця планування експерименту

# -1 -1 -1

# -1 +1 +1

# +1 -1 +1

# +1 +1 -1

# 2. Заповнимо матрицю планування для  m = 3

plan\_matrix = []

plan\_matrix.append([x\_max[0], x\_min[1], x\_min[2]])

plan\_matrix.append([x\_min[0], x\_max[1], x\_max[2]])

plan\_matrix.append([x\_max[0], x\_min[1], x\_max[2]])

plan\_matrix.append([x\_max[0], x\_max[1], x\_min[2]])

print(f"\nМатриця експерименту:")

for line in plan\_matrix:

    print(line)

m = 3

N = 4

y1 = [random.randint(*int*(y\_min), *int*(y\_max)) for \_ in range(m)]

y2 = [random.randint(*int*(y\_min), *int*(y\_max)) for \_ in range(m)]

y3 = [random.randint(*int*(y\_min), *int*(y\_max)) for \_ in range(m)]

y4 = [random.randint(*int*(y\_min), *int*(y\_max)) for \_ in range(m)]

print(f"\nФункції відгуку\n{y1}\n{y2}\n{y3}\n{y4}")

dispersion\_list = [dispersion(y1), dispersion(y2), dispersion(y3), dispersion(y4)]

# Рівняння регресії

# y = b0 + b1\*x1 + b2\*x2 + b3\*x3

# 3. Знайдемо середні значення функцій відгуку за рядками

y1\_average = average(y1)

y2\_average = average(y2)

y3\_average = average(y3)

y4\_average = average(y4)

print(f"\nСередні значення ф-цій відгуку\ny\u2081: {y1\_average}\ny\u2082: {y2\_average}\ny\u2083: {y3\_average}\ny\u2084: {y4\_average}")

average\_list = [y1\_average, y2\_average, y3\_average, y4\_average]

# 4. Знайдемо коефіцієнти рівняння регресії

mx1 = (plan\_matrix[0][0] + plan\_matrix[1][0] + plan\_matrix[2][0] + plan\_matrix[3][0]) / 4

mx2 = (plan\_matrix[0][1] + plan\_matrix[1][1] + plan\_matrix[2][1] + plan\_matrix[3][1]) / 4

mx3 = (plan\_matrix[0][2] + plan\_matrix[1][2] + plan\_matrix[2][2] + plan\_matrix[3][2]) / 4

my = average(average\_list)

a1 = (plan\_matrix[0][0] \* y1\_average + plan\_matrix[1][0] \* y2\_average + plan\_matrix[2][0] \* y3\_average + plan\_matrix[3][0] \* y4\_average) / 4

a2 = (plan\_matrix[0][1] \* y1\_average + plan\_matrix[1][1] \* y2\_average + plan\_matrix[2][1] \* y3\_average + plan\_matrix[3][1] \* y4\_average) / 4

a3 = (plan\_matrix[0][2] \* y1\_average + plan\_matrix[1][2] \* y2\_average + plan\_matrix[2][2] \* y3\_average + plan\_matrix[3][2] \* y4\_average) / 4

a11 = (plan\_matrix[0][0]\*\*2 + plan\_matrix[1][0]\*\*2 + plan\_matrix[2][0]\*\*2 + plan\_matrix[3][0]\*\*2) / 4

a22 = (plan\_matrix[0][1]\*\*2 + plan\_matrix[1][1]\*\*2 + plan\_matrix[2][1]\*\*2 + plan\_matrix[3][1]\*\*2) / 4

a33 = (plan\_matrix[0][2]\*\*2 + plan\_matrix[1][2]\*\*2 + plan\_matrix[2][2]\*\*2 + plan\_matrix[3][2]\*\*2) / 4

a12 = (plan\_matrix[0][0]\*plan\_matrix[0][1] + plan\_matrix[1][0]\*plan\_matrix[1][1] + plan\_matrix[2][0]\*plan\_matrix[2][1] + plan\_matrix[3][0]\*plan\_matrix[3][1]) / 4

a13 = (plan\_matrix[0][0]\*plan\_matrix[0][2] + plan\_matrix[1][0]\*plan\_matrix[1][2] + plan\_matrix[2][0]\*plan\_matrix[2][2] + plan\_matrix[3][0]\*plan\_matrix[3][2]) / 4

a23 = (plan\_matrix[0][1]\*plan\_matrix[0][2] + plan\_matrix[1][1]\*plan\_matrix[1][2] + plan\_matrix[2][1]\*plan\_matrix[2][2] + plan\_matrix[3][1]\*plan\_matrix[3][2]) / 4

a21 = a12

a31 = a13

a32 = a23

b0\_numerator = np.array([[my, mx1, mx2, mx3],

                        [a1, a11, a12, a13],

                        [a2, a12, a22, a32],

                        [a3, a13, a23, a33]])

b0\_denominator = np.array([[1, mx1, mx2, mx3],

                          [mx1, a11, a12, a13],

                          [mx2, a12, a22, a32],

                          [mx3, a13, a23, a33]])

b0 = np.linalg.det(b0\_numerator) / np.linalg.det(b0\_denominator)

b1\_numerator = np.array([[1, my, mx2, mx3],

                        [mx1, a1, a12, a13],

                        [mx2, a2, a22, a32],

                        [mx3, a3, a23, a33]])

b1\_denominator = np.array([[1, mx1, mx2, mx3],

                          [mx1, a11, a12, a13],

                          [mx2, a12, a22, a32],

                          [mx3, a13, a23, a33]])

b1 = np.linalg.det(b1\_numerator) / np.linalg.det(b1\_denominator)

b2\_numerator = np.array([[1, mx1, my, mx3],

                        [mx1, a11, a1, a13],

                        [mx2, a12, a2, a32],

                        [mx3, a13, a3, a33]])

b2\_denominator = np.array([[1, mx1, mx2, mx3],

                          [mx1, a11, a12, a13],

                          [mx2, a12, a22, a32],

                          [mx3, a13, a23, a33]])

b2 = np.linalg.det(b2\_numerator) / np.linalg.det(b2\_denominator)

b3\_numerator = np.array([[1, mx1, mx2, my],

                        [mx1, a11, a12, a1],

                        [mx2, a12, a22, a2],

                        [mx3, a13, a23, a3]])

b3\_denominator = np.array([[1, mx1, mx2, mx3],

                          [mx1, a11, a12, a13],

                          [mx2, a12, a22, a32],

                          [mx3, a13, a23, a33]])

b3 = np.linalg.det(b3\_numerator) / np.linalg.det(b3\_denominator)

b = [b0, b1, b2, b3]

# 5. Запишемо отримане рівняння регресії

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

# 6. Зробимо перевірку (підставимо значення факторів з матриці планування

#   і порівняємо результат з середнім значенням фунції відгуку за рядками)

y1\_reg = b0 + b1 \* plan\_matrix[0][0] + b2 \* plan\_matrix[0][1] + b3 \* plan\_matrix[0][2]

y2\_reg = b0 + b1 \* plan\_matrix[1][0] + b2 \* plan\_matrix[1][1] + b3 \* plan\_matrix[1][2]

y3\_reg = b0 + b1 \* plan\_matrix[2][0] + b2 \* plan\_matrix[2][1] + b3 \* plan\_matrix[2][2]

y4\_reg = b0 + b1 \* plan\_matrix[3][0] + b2 \* plan\_matrix[3][1] + b3 \* plan\_matrix[3][2]

if (round(y1\_reg, 0) == round(y1\_average, 0)

    and round(y2\_reg, 0) == round(y2\_average, 0)

    and round(y3\_reg, 0) == round(y3\_average, 0)

    and round(y4\_reg, 0) == round(y4\_average, 0)):

    print(f"\nПеревірку закінчено. Коефіцієнти розраховані ПРАВИЛЬНО")

else: print(f"\nПеревірку закінчено. Коефіцієнти розраховані НЕПРАВИЛЬНО")

# 7. Перевірка однорідності дисперсії за критерієм Кохрена

print("\nПеревірка за критерієм Кохрена.....\n")

if cochrane\_criteria(): print(f"\nЗа критерієм Кохрена дисперсія ОДНОРІДНА\n")

else: print(f"\nЗа критерієм Кохрена дисперсія НЕОДНОРІДНА\n")

# 8. Далі оцінимо значимість коефіцієнтів регресії згідно критерію Стьюдента

print("-"\*42 + "\nПеревірка значущості коефіцієнтів за критерієм Стьюдента.....\n")

std\_ch = students\_criteria()

print(f"\n{std\_ch}")

print(f"\nРівняння регресії\ny = {std\_ch[0]} + {std\_ch[1]}\*x1 + {std\_ch[2]}\*x2 + {std\_ch[3]}\*x3")

y1\_std = std\_ch[0] + std\_ch[1] \* plan\_matrix[0][0] + std\_ch[2] \* plan\_matrix[0][1] + std\_ch[3] \* plan\_matrix[0][2]

y2\_std = std\_ch[0] + std\_ch[1] \* plan\_matrix[1][0] + std\_ch[2] \* plan\_matrix[1][1] + std\_ch[3] \* plan\_matrix[1][2]

y3\_std = std\_ch[0] + std\_ch[1] \* plan\_matrix[2][0] + std\_ch[2] \* plan\_matrix[2][1] + std\_ch[3] \* plan\_matrix[2][2]

y4\_std = std\_ch[0] + std\_ch[1] \* plan\_matrix[3][0] + std\_ch[1] \* plan\_matrix[3][1] + std\_ch[3] \* plan\_matrix[3][2]

y\_std = [y1\_std, y2\_std, y3\_std, y4\_std]

print(f"\ny\u2081 = {y1\_std}\ny\u2082 = {y2\_std}\ny\u2083 = {y3\_std}\ny\u2084 = {y4\_std}")

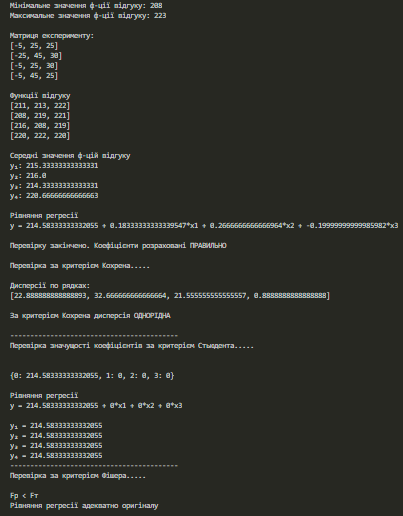
# 9. Критерій Фішера

print("-"\*42 + "\nПеревірка за критерієм Фішера.....\n")

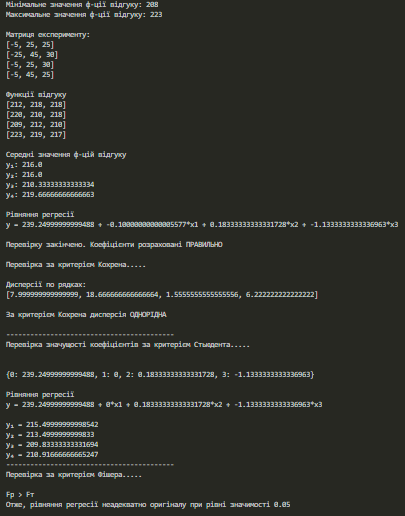
fisher\_criteria()

**Результат роботи програми**

Позитивний результат



Негативний результат



**Висновок:**

Під час виконання даної лабораторної роботи я провів трьохфакторний експеримент, перевірив однорідність дисперсії за критерієм Кохрена, отримав коефіцієнти рівняння регресії, оцінив значимість знайдених коефіцієнтів за критеріями Стьюдента та Фішера.  
Отже, мета лабораторної роботи була досягнута.

**Контрольні питання**

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

ДФЕ – це частина ПФЕ, який мінімізує число дослідів, за рахунок тієї інформації, яка не дуже істотна для побудови лінійної моделі.

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

Критерій Кохрена використовують для порівняння трьох і більше виборок однакового обсягу n

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

За допомогою критерія Стьюдента перевіряється значущість коефіцієнта рівняння регресії.

Тобто, якщо виконується нерівність ts< tтабл, то приймається нуль-гіпотеза, тобто вважається, що знайдений коефіцієнт βs є статистично незначущим і його слід виключити з рівняння регресії.

Якщо ts > tтабл то гіпотеза не підтверджується, тобто βs – значимий коефіцієнт і він залишається в рівнянні регресії.

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

Отримане рівняння регресії необхідно перевірити на адекватність досліджуваному об'єкту. Для цієї мети необхідно оцінити, наскільки відрізняються середні значення у вихідної величини, отриманої в точках факторного простору, і значення у, отриманого з рівняння регресії в тих самих точках факторного простору. Для цього використовують дисперсію адекватності.

Адекватність моделі перевіряють за F-критерієм Фішера.

Якщо Fпрак < Fтеор, то отримана математична модель з прийнятим рівнем статистичної значимості q адекватна експериментальним даним.