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

Лабораторна робота №3

з дисципліни

«Методи оптимізації та планування експерименту»

Тема: «ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ.»

Виконав:

студент групи ІО-93
Комаровський Роман Сергійович
Номер у списку: 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:

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.

Варіант завдання:

314	X_1		X_2		X_3	
	-25	75	5	40	15	25

Лістинг програми:

```
from random import randint
import numpy as np

# Find coefficients
def findA(a, b=None):
    if b is None:
        return sum([a[i] ** 2 for i in range(len(a))])/len(a)
    else:
        return sum(a[i] * b[i] for i in range(len(a)))/len(a)

# Cramer's rule
def cramer(arr, ins, pos):
    matrix = np.insert(np.delete(arr, pos, 1), pos, ins, 1)

    return np.linalg.det(matrix)/np.linalg.det(arr)

# Method to get dispersion
def getDispersion(y, y_r):
    return [round(sum([(y[i][j] - y_r[i]) ** 2 for j in range(len(y[i]))]) /
3, 3) for i in range(len(y_r))]

# Cochran criteria
def cochran(disp, m):
    Gp = max(disp) / sum(disp)
    Gt = [.9065, .7679, .6841, .6287, .5892, .5598, .5365, .5175, .5017,
.4884]
```

```
    if Gp < Gt[m - 2]:
        return [round(Gp, 4), Gt[m - 2]]
    else:
        return

# Student criteria
def student(disp, m, y_r, x_n):
    table = {
        8: 2.306,
        12: 2.179,
        16: 2.120,
        20: 2.086,
        24: 2.064,
        28: 2.048,
        'inf': 1.960
    }

    x_nT = x_n.T
    N = len(y_r)

    Sb = sum(disp)/len(y_r)
    Sbeta = (Sb / (m * N))**(1/2)

    beta = [sum([y_r[j] * x_nT[i][j] for j in range(N)])/N for i in range(N)]
    t = [abs(beta[i])/Sbeta for i in range(len(beta))]

    f3 = N*(m - 1)

    if f3 > 30:
        t_t = table['inf']
    elif f3 > 0:
        t_t = table[f3]
    else:
        return

    result = []
    for i in t:
        if i < t_t:
            result.append(False)
        else:
            result.append(True)

    return result

# Fisher criteria
def fisher(y_r, y_st, b_det, disp, m):
    table = {
        8: [5.3, 4.5, 4.1, 3.8, 3.7, 3.6],
        12: [4.8, 3.9, 3.5, 3.3, 3.1, 3.0],
        16: [4.5, 3.6, 3.2, 3.0, 2.9, 2.7],
        20: [4.5, 3.5, 3.1, 2.9, 2.7, 2.6],
        24: [4.3, 3.4, 3.0, 2.8, 2.6, 2.5]
    }

    N = len(y r)
    Sb = sum(disp)/N
    d = 0
    for b in b_det:
        if b:
            d += 1

    f4 = N - d
    f3 = N * (m - 1)
```

```
    Sad = (m / f4) * sum([(y_st[i] - y_r[i])**2 for i in range(N)])
    Fap = Sad / Sb
    Ft = table[f3][f4 - 1]

    if Fap < Ft:
        return f"\nРівняння регресії адекватно оригіналу:\nFap < Ft:
{round(Fap, 2)} < {Ft}"
    else:
        return f"\nРівняння регресії неадекватно оригіналу: \nFap > Ft:
{round(Fap, 2)} > {Ft}"

# Main function
def experiment(m, min_x1, max_x1, min_x2, max_x2, min_x3, max_x3):
    y_min = round((min_x1 + min_x2 + min_x3) / 3) + 200
    y_max = round((max_x1 + max_x2 + max_x3) / 3) + 200

    x_norm = np.array([
        [1, -1, -1, -1],
        [1, -1, 1, 1],
        [1, 1, -1, 1],
        [1, 1, 1, -1]
    ])

    x = np.array([
        [min_x1, min_x2, min_x3],
        [min_x1, max_x2, max_x3],
        [max_x1, min_x2, max_x3],
        [max_x1, max_x2, min_x3]
    ])

    # Generate Y and calculate their Response
    y = [[randint(y_min, y_max) for _ in range(m)] for _ in range(len(x))]
    y_r = [round(sum(y[i])/len(y[i]), 2) for i in range(len(y))]
    N = len(y_r)

    # Calculate Cochran's C test
    disp = getDispersion(y, y_r)
    cochran_cr = cochran(disp, m)

    if cochran_cr is None:
        return False
    else:
        pass

    # Coefficients
    mx = [sum(i) / len(i) for i in x.T]
    my = sum(y_r)/N

    x_T = x.T

    a1 = findA(x_T[0], y_r)
    a2 = findA(x_T[1], y_r)
    a3 = findA(x_T[2], y_r)

    a11 = findA(x_T[0])
    a22 = findA(x_T[1])
    a33 = findA(x_T[2])

    a12 = a21 = findA(x_T[0], x_T[1])
    a13 = a31 = findA(x_T[0], x_T[2])
    a23 = a32 = findA(x_T[1], x_T[2])

    # Solve SoLE by Cramer's rule
    b_delta = np.array([
```

```
            [1, mx[0], mx[1], mx[2]],
            [mx[0], a11, a12, a13],
            [mx[1], a21, a22, a23],
            [mx[2], a31, a32, a33]
        ])

        b_set = np.array([my, a1, a2, a3])
        b = [cramer(b_delta, b_set, i) for i in range(N)]

        b_det = student(disp, m, y_r, x_norm)
        b_cut = b.copy()

        # Simplified equations
        if b_det is None:
            return
        else:
            for i in range(N):
                if not b_det[i]:
                    b_cut[i] = 0

            y_st = [round(b_cut[0] + x[i][0]*b_cut[1] + x[i][1]*b_cut[2] +
x[i][2]*b_cut[3], 2) for i in range(N)]

        # Calculate F-test
        fisher_cr = fisher(y_r, y_st, b_det, disp, m)

        # Print out results
        print(f"\nМатриця планування для m = {m}:")
        for i in range(m):
            print(f"Y{i + 1} - {np.array(y).T[i]}")

        print(f"\nСередні значення функції відгуку за рядками:\nY_R: {y_r}")
        print(f"\nКоефіцієнти рівняння регресії:")
        for i in range(len(b)):
            print(f"b{i} = {round(b[i], 3)}")

        print("\nДисперсії по рядках:\nS²{y} = ", disp, sep="")
        print(f"\nЗа критерієм Кохрена дисперсія однорідна:\nGp < Gt -
{cochran_cr[0]} < {cochran_cr[1]}")

        print(f"\nЗа критерієм Стьюдента коефіцієнти ", end="")
        for i in range(len(b_det)):
            if not b_det[i]:
                print(f"b{i} ", end="")
        print("приймаємо незначними")

        print(f"\nОтримані функції відгуку зі спрощеними коефіцієнтами:\nY_St -
{y_st}")
        print(fisher_cr)

        return True

if __name__ == '__main__':
    Min_x1, Max_x1 = -25, 75
    Min_x2, Max_x2 = 5, 40
    Min_x3, Max_x3 = 15, 25

    M = 3

    success = False
    while not success:
        success = experiment(M, Min_x1, Max_x1, Min_x2, Max_x2, Min_x3,
Max_x3)
        M += 1
```

Результат виконання:

```
Матриця планування для m = 3:
Y1 - [239 223 215 234]
Y2 - [231 204 211 244]
Y3 - [209 215 199 217]

Середні значення функції відгуку за рядками:
Y_R: [226.33, 214.0, 208.33, 231.67]

Коефіцієнти рівняння регресії:
b0 = 252.255
b1 = -0.002
b2 = 0.157
b3 = -1.784

Дисперсії по рядках:
S²{y} = [160.889, 60.667, 46.222, 124.222]

За критерієм Кохрена дисперсія однорідна:
Gp < Gt - 0.4104 < 0.7679

За критерієм Стьюдента коефіцієнти b1 b2 приймаємо незначними

Отримані функції відгуку зі спрощеними коефіцієнтами:
Y_St - [225.5, 207.67, 207.67, 225.5]

Рівняння регресії адекватно оригіналу:
Fap < Ft: 1.21 < 4.5
```

Відповіді на контрольні запитання:

1. **Що називається дробовим факторним експериментом?**
   У деяких випадках немає необхідності проводити повний факторний експеримент (ПФЕ). Якщо буде використовуватися лінійна регресія, то можливо зменшити кількість рядків матриці ПФЕ до кількості коефіцієнтів регресійної моделі. Кількість дослідів слід скоротити, використовуючи для планування так звані регулярні дробові репліки від повного факторного експерименту, що містять відповідну кількість дослідів і зберігають основні властивості матриці планування – це означає дробовий факторний експеримент (ДФЕ).

2. **Для чого потрібне розрахункове значення Кохрена?**
   Статистична перевірка за критерієм Кохрена використовується для перевірки гіпотези про однорідність дисперсії з довірчою ймовірністю p. Якщо експериментальне значення $G < G_{кр}$, яке обирається з таблиці, то гіпотеза підтверджується, якщо ні, то відповідно не підтверджується.

3. **Для чого перевіряється критерій Стьюдента?**
   Критерій Стьюдента використовується для перевірки значимості коефіцієнта рівняння регресії. Якщо з'ясувалось, що будь-який коефіцієнт рівняння регресії не значимий, то відповідний $b_i = 0$ і відповідний член рівняння регресії треба викреслити. Іноді ця статистична перевірка має назву «нуль-гіпотеза». Якщо експериментальне значення $t > t_{кр}$, тонуль-гіпотеза не підтверджується і даний коефіцієнт значимий, інакше нуль-гіпотеза підтверджується і даний коефіцієнт рівняння регресії не значимий.

4. **Чим визначається критерій Фішера та як його застосовувати?**
   Критерій Фішера застосовується для перевірки адекватності моделі (рівняння регресії) оригіналу (експериментальним даним). Обчислюється експериментальне значення F, яке порівнюється з $F_{кр}$, взятим з таблиці залежно від кількості значимих коефіцієнтів та ступенів вільності. Якщо $F < F_{кр}$, то модель адекватна оригіналу, інакше – ні.

Висновок

Під час виконання лабораторної роботи було змодельовано трьохфакторний експеримент з використанням лінійного рівняння регресії, скаладено матрицю планування експерименту, було визначено коефіцієнти рівняння регресії(натуралізовані та нормовані), виконано перевірку правильності розрахунку коефіцієнтів рівняння регресії. Також було проведено 3 статистичні перевірки(використання критеріїв Кохрена, Стьюдента та Фішера).