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

Методи планування експерименту

Лабораторна робота №3 «Проведення трьохфакторного експерименту з використанням лінійного рівняння регресії»

Виконав:
студент II курсу ФІОТ
групи ІВ-92
Салун Кирило
номер у списку групи – 20

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 *y*. 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.

Варіанти обираються по номеру в списку в журналі викладача.

Варіант завдання:

220	-30	20	-30	45	-30	-15

Лістинг програми:

```
import random
import scipy.stats
```

```python
import numpy as np


# Блок даних, заданих за варіантом 220
x1_min, x1_max = -30, 20
x2_min, x2_max = -30, 45
x3_min, x3_max = -30, -15
experiments_count = 3  # Кількість експериментів
y_min = round(200 + (x1_min + x2_min + x3_min) / 3)
y_max = round(200 + (x1_max + x2_max + x3_max) / 3)
def find_coefficient(a, b=None):
    return sum([a[i] ** 2 for i in range(len(a))]) / len(a) if b is None \
        else sum(a[i] * b[i] for i in range(len(a))) / len(a)


def solve_cramer(arr, ins, pos):
    return np.linalg.det(np.insert(np.delete(arr, pos, 1), pos, ins, 1)) / np.linalg.det(arr)


def get_dispersion(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))]

def cochrane(dispersion, m):
    gp = max(dispersion) / sum(dispersion)
    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

def student(dispersion_reproduction, m, y_mean, xn):
    dispersion_statistic_mark = (dispersion_reproduction / (4 * m)) ** 0.5

    beta = [1 / 4 * sum(y_mean[j] for j in range(4))]
    for i in range(3):
        b = 0
        for j in range(4):
            b += y_mean[j] * xn[j][i]
        beta.append(1 / 4 * b)

    t = []
    for i in beta:
        t.append(abs(i) / dispersion_statistic_mark)

    check_st = scipy.stats.t.ppf((1 + 0.95)/2, (m-1) * 4)

    return t[0] > check_st, t[1] > check_st, t[2] > check_st, t[3] > check_st

def fisher(y_r, y_st, b_det, dispersion, 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(dispersion) / 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'Рівняння регресії адекватно оригіналу Fap < Ft: {round(fap, 2)} < {ft}'
    else:
        return f'Рівняння регресії неадекватно оригіналу Fap > Ft: {round(fap, 2)} > {ft}'


def simulate_experiment(m, min_x1, max_x1, min_x2, max_x2, min_x3, max_x3):
    x_appropriate = 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]
    ])
    x_normalized = np.array([
        [1, -1, -1, -1],
        [1, -1, 1, 1],
        [1, 1, -1, 1],
        [1, 1, 1, -1]
    ])
    # Пошук значення функції відгуку Y
    y = [[random.randint(y_min, y_max) for _ in range(m)] for _ in range(len(x_appropriate))]
    y_r = [round(sum(y[i]) / len(y[i]), 2) for i in range(len(y))]
    N = len(y_r)
    # Оцінка дисперції за критерієм Кохрена
    dispersion = get_dispersion(y, y_r)
    cochrane_criterion = cochrane(dispersion, m)
    if cochrane_criterion is None:
        raise ValueError('Потрібно провести більше експериментів.')
    else:
        pass
    # Розрахунок коефіцієнтів
    mx = [sum(i) / len(i) for i in x_appropriate.T]
    my = sum(y_r) / N
    x_T = x_appropriate.T
    a1 = find_coefficient(x_T[0], y_r)
    a2 = find_coefficient(x_T[1], y_r)
    a3 = find_coefficient(x_T[2], y_r)
    a11 = find_coefficient(x_T[0])
    a22 = find_coefficient(x_T[1])
    a33 = find_coefficient(x_T[2])
    a12 = a21 = find_coefficient(x_T[0], x_T[1])
    a13 = a31 = find_coefficient(x_T[0], x_T[2])
    a23 = a32 = find_coefficient(x_T[1], x_T[2])

    # Вирішення системи алгебраїчних рівнянь методом Крамера
    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 = [solve_cramer(b_delta, b_set, i) for i in range(N)]
    disp = sum(dispersion) / 4
    b_det = student(disp, m, y_r, x_normalized)
    b_cut = b.copy()
    if b_det is None:
```

```
        raise ValueError('Потрібно провести більше експериментів.')
    else:
        for i in range(N):
            if not b_det[i]:
                b_cut[i] = 0
        y_st = [round(b_cut[0] + x_appropriate[i][0] * b_cut[1] +
                      x_appropriate[i][1] * b_cut[2] + x_appropriate[i][2] * b_cut[3], 2) for
i in range(N)]
    # Оцінка адекватності моделі за критерієм Фішера
    fisher_criterion = fisher(y_r, y_st, b_det, dispersion, m)
    # Блок роздруківки результатів
    print(f'Матриця планування для m = {m}:')
    for i in range(m):
        print(f'Y{i + 1} - {np.array(y).T[i]}')

    print(f'Середні значення функції відгуку y_r = {y_r}')
    print(f'Коефіцієнти рівняння регресії:')
    for i in range(len(b)):
        print(f"b{i} = {round(b[i], 3)}")
    print(f'Дисперсії в рядках S²(y) = {dispersion}')
    print(f'За критерієм Кохрена дисперсія однорідна gp < gt: {cochrane_criterion[0]} <
{cochrane_criterion[1]}')
    print(f'За критерієм Стьюдента коефіцієнти '
          f'{[f"b{i}" for i in range(len(b_det)) if not b_det[i]]} приймаємо незначними')
    print(f'Отримані функції відгуку зі спрощеними коефіцієнтами y_st = {y_st}')
    print(fisher_criterion)
def try_start_experiment(m, min_x1, max_x1, min_x2, max_x2, min_x3, max_x3):
    try:
        simulate_experiment(m, min_x1, max_x1, min_x2, max_x2, min_x3, max_x3)
    except ValueError:
        m += 1
        try_start_experiment(m, min_x1, max_x1, min_x2, max_x2, min_x3, max_x3)
# Точка входу програми
if __name__ == '__main__':
    try_start_experiment(experiments_count, x1_min, x1_max, x2_min, x2_max, x3_min, x3_max)
```

Результати виконання роботи:

```
Матриця планування для m = 3:
Y1 - [213 173 180 214]
Y2 - [188 194 196 186]
Y3 - [188 205 203 179]
Середні значення функції відгуку y_r = [196.33, 190.67, 193.0, 193.0]
Коефіцієнти рівняння регресії:
b0 = 189.238
b1 = -0.01
b2 = -0.038
b3 = -0.189
Дисперсії в рядках S²(y) = [138.889, 176.222, 92.667, 228.667]
За критерієм Кохрена дисперсія однорідна gp < gt: 0.3593 < 0.7679
За критерієм Стьюдента коефіцієнти ['b2', 'b3'] приймаємо незначними
Отримані функції відгуку зі спрощеними коефіцієнтами y_st = [189.54, 189.54, 189.04, 189.04]
Рівняння регресії адекватно оригіналу Fap < Ft: 0.74 < 4.5
```

Висновок:

У ході виконання лабораторної роботи проведено трьохфакторний дробовий експеримент. В ході дослідження було розроблено відповідну

програму мовою програмування Python, яка моделює проведення трьохфакторного експерименту. Складено матрицю планування, знайдено коефіцієнти рівняння регресії, перевірено однорідність дисперсії за критерієм Кохрена, нуль-гіпотезу за критерієм Стьюдента та адекватність моделі за критерієм Фішера. Результати роботи, наведені у протоколі, підтверджують правильність виконання – кінцеву мету роботи було досягнуто.