Міністерство освіти і науки України

Національний технічний університет України «КПІ» імені Ігоря Сікорського

Кафедра обчислювальної техніки ФІОТ

ЗВІТ

з лабораторної роботи №3

з навчальної дисципліни «Методи наукових досліджень»

Тема:

ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ

Виконала:

Студентка 2 курсу кафедри ОТ ФІОТ,

Навчальної групи ІВ-92

Орлова Ю.Д.

Номер у списку групи: 15

Перевірив:

Регіда П.Г.

Київ 2020

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		x 2		x 3	
215	min	max	min	max	min	max
	10	50	-20	60	10	15

Код програми:

```
import random
import numpy as np

x1_min = 10; x1_max = 50
x2_min = -20; x2_max = 60
x3_min = 10; x3_max = 15

x_min = (x1_min + x2_min + x3_min) / 3
x_max = (x1_max + x2_max + x3_max) / 3

y_min = 200 + x_min
y_max = 200 + x_max

n = 4

x0_n = (1, 1, 1, 1)
x1_n = (-1, -1, 1, 1)
x2_n = (-1, 1, -1, 1)
x3_n = (-1, 1, 1, -1)

x1 = (x1_min, x1_min, x1_max, x1_max)
x2 = (x2_min, x2_max, x2_min, x2_max)
x3 = (x3_min, x3_max, x3_max, x3_min)

def experiment(m):
```

```
    global t_tabular, f_tabular
    y = [[round(random.uniform(y_min, y_max), 3) for i in range(m)] for j in
range(n)]
    print('Матриця планування експерименту:\n{0}\n{1}\n{2}\n{3}\n'.format(y[0], y[1],
y[2], y[3]))

    # the average value of the response functions in the rows
    y_response = (sum(y[0][i] for i in range(m)) / m,
                  sum(y[1][i] for i in range(m)) / m,
                  sum(y[2][i] for i in range(m)) / m,
                  sum(y[3][i] for i in range(m)) / m)

    print('Середні значення функції відгуку:\n{0}  {1}  {2}  {3}\n'
          .format(round(y_response[0], 3), round(y_response[1], 3),
round(y_response[2], 3), round(y_response[3], 3)))

    # calculation of normalized coefficients of the regression equation
    mx1 = (x1[0] + x1[1] + x1[2] + x1[3]) / n
    mx2 = (x2[0] + x2[1] + x2[2] + x2[3]) / n
    mx3 = (x3[0] + x3[1] + x3[2] + x3[3]) / n
    my = (y_response[0] + y_response[1] + y_response[2] + y_response[3]) / n

    a1 = (x1[0] * y_response[0] + x1[1] * y_response[1] + x1[2] * y_response[2] +
x1[3] * y_response[3]) / 4
    a2 = (x2[0] * y_response[0] + x2[1] * y_response[1] + x2[2] * y_response[2] +
x2[3] * y_response[3]) / 4
    a3 = (x3[0] * y_response[0] + x3[1] * y_response[1] + x3[2] * y_response[2] +
x3[3] * y_response[3]) / 4

    a11 = (x1[0] ** 2 + x1[1] ** 2 + x1[2] ** 2 + x1[3] ** 2) / 4
    a22 = (x2[0] ** 2 + x2[1] ** 2 + x2[2] ** 2 + x2[3] ** 2) / 4
    a33 = (x3[0] ** 2 + x3[1] ** 2 + x3[2] ** 2 + x3[3] ** 2) / 4

    a12 = (x1[0] * x2[0] + x1[1] * x2[1] + x1[2] * x2[2] + x1[3] * x2[3]) / 4
    a13 = (x1[0] * x3[0] + x1[1] * x3[1] + x1[2] + x3[2] + x1[3] + x3[3]) / 4
    a23 = (x2[0] * x3[0] + x2[1] * x3[1] + x2[2] * x3[2] + x2[3] * x3[3]) / 4

    b = [np.linalg.det([[my, mx1, mx2, mx3], [a1, a11, a12, a13], [a2, a12, a22,
a23], [a3, a13, a23, a33]]) /
         np.linalg.det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22,
a23], [mx3, a13, a23, a33]]),

         np.linalg.det([[1, my, mx2, mx3], [mx1, a1, a12, a13], [mx2, a2, a22, a23],
[mx3, a3, a23, a33]]) /
         np.linalg.det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22,
a23], [mx3, a13, a23, a33]]),

         np.linalg.det([[1, mx1, my, mx3], [mx1, a11, a1, a13], [mx2, a12, a2, a23],
[mx3, a13, a3, a33]]) /
         np.linalg.det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22,
a23], [mx3, a13, a23, a33]]),

         np.linalg.det([[1, mx1, mx2, my], [mx1, a11, a12, a1], [mx2, a12, a22, a2],
[mx3, a13, a23, a3]]) /
         np.linalg.det([[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22,
a23], [mx3, a13, a23, a33]])]

    print('Отримане рівняння регресії:\ny = {0} + {1} * x1 + {2} * x2 + {3} * x3\n'
          '\nПеревірка:\nb0 + b1 * x1_min + b2 * x2_min + b3 * x3_min = {4}\n'
          'b0 + b1 * x1_min + b2 * x2_max + b3 * x3_max = {5}\n'
          'b0 + b1 * x1_max + b2 * x2_min + b3 * x3_max = {6}\n'
          'b0 + b1 * x1_max + b2 * x2_max + b3 * x3_min = {7}\n'
```

```
        .format(round(b[0], 3), round(b[1], 3), round(b[2], 3), round(b[3], 3),
                round(b[0] + b[1] * x1[0] + b[2] * x2[0] + b[3] * x3[0], 3),
                round(b[0] + b[1] * x1[1] + b[2] * x2[1] + b[3] * x3[1], 3),
                round(b[0] + b[1] * x1[2] + b[2] * x2[2] + b[3] * x3[2], 3),
                round(b[0] + b[1] * x1[3] + b[2] * x2[3] + b[3] * x3[3], 3)))

    # checking the homogeneity of the variance according to the Cochren's criterion
    # variance by lines and the main deviation
    dispersions = [sum([(y[0][i] - y_response[0]) ** 2 for i in range(m)]) / m,
                   sum([(y[1][i] - y_response[1]) ** 2 for i in range(m)]) / m,
                   sum([(y[2][i] - y_response[2]) ** 2 for i in range(m)]) / m,
                   sum([(y[3][i] - y_response[3]) ** 2 for i in range(m)]) / m]

    gp = max(dispersions) ** 2 / sum([dispersions[i] ** 2 for i in range(n)])

    f1 = m - 1; f2 = n; q = 0.05

    if f1 == 1: gt = 0.9065
    elif f1 == 2: gt = 0.7679
    elif f1 == 3: gt = 0.6841
    elif f1 == 4: gt = 0.6287
    elif f1 == 5: gt = 0.5892
    elif f1 == 6: gt = 0.5598
    elif f1 == 7: gt = 0.5365
    elif f1 == 8: gt = 0.5175
    elif f1 == 9: gt = 0.5017
    elif f1 == 10: gt = 0.4884
    elif 11 <= f1 <= 16: gt = 0.4366
    elif 17 <= f1 <= 136: gt = 0.3720
    else: gt = 0.2500

    if gp > gt:
        i = input(
            'Дисперсія неоднорідна. Якщо ви хочете повторити експериметн при m = m +
1 = {}, введіть 1: \n'.format(
                m + 1))
        if i == '1':
            experiment(m + 1)
            m += 1
    else:
        print('Дисперсія однорідна.')

        # assessment of the significance of regression coefficients according to
Student's criterion
        s_b = sum(dispersions) / n
        s = np.sqrt(s_b / (n * m))

        beta = [sum([dispersions[i] * x0_n[i] for i in range(n)]) / n,
                sum([dispersions[i] * x1_n[i] for i in range(n)]) / n,
                sum([dispersions[i] * x2_n[i] for i in range(n)]) / n,
                sum([dispersions[i] * x3_n[i] for i in range(n)]) / n]

        t = [abs(beta[i]) / s for i in range(n)]

        f3 = f1 * f2

        # t_tabular fot f2 = n = 4
        if f3 == 4: t_tabular = 2.776
        elif f3 == 8: t_tabular = 2.306
        elif f3 == 12: t_tabular = 2.179
        elif f3 == 16: t_tabular = 2.12
        elif f3 == 20: t_tabular = 2.086
```

```
        elif f3 == 24: t_tabular = 2.064
        elif f3 == 28: t_tabular = 2.048
        elif f3 > 28: t_tabular = 1.96

        d = 4
        for i in range(n):
            if t[i] < t_tabular:
                print('Коефіцієнт рівняння регресії b{0} приймаємо незначним при 
рівні значимості 0.05'.format(i))
                b[i] = 0
                d -= 1


        # Fisher's criterion
        f4 = n - d
        s_ad = (m * sum([(b[0] + b[1] * x1[i] + b[2] * x2[i] + b[3] * x3[i] - 
y_response[i]) ** 2 for i in range(n)]) / f4)
        f_p = s_ad / s_b

        # f_tabular fot f2 = n = 4
        if f3 == 4:
            if f4 == 1: f_tabular = 7.7
            elif f4 == 2: f_tabular = 6.9
            elif f4 == 3: f_tabular = 6.6
            elif f4 == 4: f_tabular = 6.4
        elif f3 == 8:
            if f4 == 1: f_tabular = 5.3
            elif f4 == 2: f_tabular = 4.5
            elif f4 == 3: f_tabular = 4.1
            elif f4 == 4: f_tabular = 3.8
        elif f3 == 12:
            if f4 == 1: f_tabular = 4.8
            elif f4 == 2: f_tabular = 3.9
            elif f4 == 3: f_tabular = 3.5
            elif f4 == 4: f_tabular = 3.3
        elif f3 == 16:
            if f4 == 1: f_tabular = 4.5
            elif f4 == 2: f_tabular = 3.6
            elif f4 == 3: f_tabular = 3.2
            elif f4 == 4: f_tabular = 3
        elif f3 == 20:
            if f4 == 1: f_tabular = 4.4
            elif f4 == 2: f_tabular = 3.5
            elif f4 == 3: f_tabular = 3.1
            elif f4 == 4: f_tabular = 2.9
        elif f3 == 24:
            if f4 == 1: f_tabular = 4.3
            elif f4 == 2: f_tabular = 3.4
            elif f4 == 3: f_tabular = 3
            elif f4 == 4: f_tabular = 2.8
        elif f3 == 28:
            if f4 == 1: f_tabular = 4.2
            elif f4 == 2: f_tabular = 3.3
            elif f4 == 3: f_tabular = 3
            elif f4 == 4: f_tabular = 2.7
        elif f3 > 28:
            if f4 == 1: f_tabular = 3.8
            elif f4 == 2: f_tabular = 3
            elif f4 == 3: f_tabular = 2.6
            elif f4 == 4: f_tabular = 2.4

        if f_p > f_tabular: print('Рівняння регресії неадекватно оригіналу при рівні 
```

```
значимості 0.05')
        else: print('Рівняння регресії адекватно оригіналу при рівні значимості
0.05')

try:
    m = int(input(("Введіть значення m: ")))
    experiment(m)
except:
    breakpoint()
    print("Ви ввели не ціле число. Спробуйте знову.")
```

Результат виконання програми:

```
Введіть значення m: 3
Матриця планування експерименту:
[240.682, 232.783, 236.088]
[226.218, 223.021, 206.569]
[232.495, 203.956, 225.883]
[201.635, 228.205, 237.643]

Середні значення функції відгуку:
236.518  218.603  220.778  222.494

Отримане рівняння регресії:
y = 221.607 + 0.05 * x1 + -0.101 * x2 + 0.282 * x3

Перевірка:
b0 + b1 * x1_min + b2 * x2_min + b3 * x3_min = 226.946
b0 + b1 * x1_min + b2 * x2_max + b3 * x3_max = 220.255
b0 + b1 * x1_max + b2 * x2_min + b3 * x3_max = 230.349
b0 + b1 * x1_max + b2 * x2_max + b3 * x3_min = 220.842

Дисперсія однорідна.
Коефіцієнт рівняння регресії b3 приймаємо незначним при рівні значимості 0.05
Рівняння регресії неадекватно оригіналу при рівні значимості 0.05
```

Висновки: під час виконання програми ми провели дробовий трьохфакторний експеримент. Було складено матрицю планування, знайдено коефіцієнти рівняння регресії та проведено 3 статичні перевірки. Було написано програму з використанням можливостей алгоритмічної мови високого рівня Python, яка це все виконує. Результати роботи програми підтвердили правильність її виконання.

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 потрібно розрахункове значення Кохрена?
   Критерій Кохрена використовують для порівняння трьох і більше виборок однакового обсягу n.
3) Для чого перевіряється критерій Стьюдента?
   Якщо теоретичний коефіцієнт bi= 0, це означає, що в апроксимуючому поліномі відповідний доданок (фактор) відсутній. Чим менше значення bi, тим менше вплив відповідного фактора. За критерієм Стьюдента перевіряється значущість коефіцієнтів.
4) 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 y вихідної величини, отриманої в точках факторного простору, і значення y, отриманого з рівняння регресії в тих самих точках факторного простору. Для цього використовують дисперсію адекватності. Адекватність моделі перевіряють за F-критерієм Фішера, який дорівнює відношенню дисперсії адекватності до дисперсії відтворюваності.