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»

Виконав:
студент 2 курсу ФІОТ
групи ІВ-92
Увін Д. І.

Перевірив:
Регіда П.Г.

Київ 2021

Мета: провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти

коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

Варіант:

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.

225	-25	-5	15	50	-25	-15

Роздруківка програми:

```
import numpy as np
import random
from numpy.linalg import solve
from scipy.stats import f, t
from functools import partial

x_range = [(-25, -5), (15, 50), (-25, -15)]
x_aver_max = (-5 + 50 + -15) / 3
x_aver_min = (-25 - 15 - -25) / 3

y_max = 200 + int(x_aver_max)
y_min = 200 + int(x_aver_min)


def regression(x, b):
    y = sum([x[i]*b[i] for i in range(len(x))])
    return y


def plan_matrix(n, m):
```

```
    y = np.zeros(shape=(n, m))
    for i in range(n):
        for j in range(m):
            y[i][j] = random.randint(y_min, y_max)
    x_norm = np.array([
        [1, -1, -1, -1],
        [1, -1, 1, 1],
        [1, 1, -1, 1],
        [1, 1, 1, -1],
        [1, -1, -1, 1],
        [1, -1, 1, -1],
        [1, 1, -1, -1],
        [1, 1, 1, 1]
    ])
    x_norm = x_norm[:len(y)]

    x = np.ones(shape=(len(x_norm), len(x_norm[0])))
    for i in range(len(x_norm)):
        for j in range(1, len(x_norm[i])):
            if x_norm[i][j] == -1:
                x[i][j] = x_range[j-1][0]
            else:
                x[i][j] = x_range[j-1][1]

    print('\nМатриця планування')
    print(np.concatenate((x, y), axis=1))

    return x, y, x_norm


def find_coefficient(x, y_aver, n):
    mx1 = sum(x[:, 1]) / n
    mx2 = sum(x[:, 2]) / n
    mx3 = sum(x[:, 3]) / n
    my = sum(y_aver) / n
    a12 = sum([x[i][1] * x[i][2] for i in range(len(x))]) / n
    a13 = sum([x[i][1] * x[i][3] for i in range(len(x))]) / n
    a23 = sum([x[i][2] * x[i][3] for i in range(len(x))]) / n
    a11 = sum([i ** 2 for i in x[:, 1]]) / n
    a22 = sum([i ** 2 for i in x[:, 2]]) / n
    a33 = sum([i ** 2 for i in x[:, 3]]) / n
    a1 = sum([y_aver[i] * x[i][1] for i in range(len(x))]) / n
    a2 = sum([y_aver[i] * x[i][2] for i in range(len(x))]) / n
    a3 = sum([y_aver[i] * x[i][3] for i in range(len(x))]) / n

    X = [[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22, a23], [mx3, a13,
a23, a33]]
    Y = [my, a1, a2, a3]
    B = [round(i, 2) for i in solve(X, Y)]
    print('\nРівняння регресії')
    print(f'{B[0]} + {B[1]}*x1 + {B[2]}*x2 + {B[3]}*x3')

    return B


def s_kv(y, y_aver, n, m):
    res = []
    for i in range(n):
        s = sum([(y_aver[i] - y[i][j])**2 for j in range(m)]) / m
        res.append(s)
    return res
```

```python
def kriteriy_cochrena(y, y_aver, n, m):
    S_kv = s_kv(y, y_aver, n, m)
    Gp = max(S_kv) / sum(S_kv)
    print('\nПеревірка за критерієм Кохрена')
    return Gp


def bs(x, y, y_aver, n):
    res = [sum(1 * y for y in y_aver) / n]
    for i in range(3):  # 4 - кість факторів
        b = sum(j[0] * j[1] for j in zip(x[:, i], y_aver)) / n
        res.append(b)
    return res


def kriteriy_studenta(x, y, y_aver, n, m):
    S_kv = s_kv(y, y_aver, n, m)
    s_kv_aver = sum(S_kv) / n

    s_Bs = (s_kv_aver / n / m) ** 0.5
    Bs = bs(x, y, y_aver, n)
    ts = [abs(B) / s_Bs for B in Bs]

    return ts


def kriteriy_fishera(y, y_aver, y_new, n, m, d):
    S_ad = m / (n - d) * sum([(y_new[i] - y_aver[i])**2 for i in range(len(y))])
    S_kv = s_kv(y, y_aver, n, m)
    S_kv_aver = sum(S_kv) / n

    return S_ad / S_kv_aver


def cohren(f1, f2, q=0.05):
    q1 = q / f1
    fisher_value = f.ppf(q=1 - q1, dfn=f2, dfd=(f1 - 1) * f2)
    return fisher_value / (fisher_value + f1 - 1)


def execute(n, m):
    f1 = m - 1
    f2 = n
    f3 = f1 * f2
    q = 0.05

    student = partial(t.ppf, q=1-0.025)
    t_student = student(df=f3)

    G_kr = cohren(f1, f2)

    x, y, x_norm = plan_matrix(n, m)
    y_aver = [round(sum(i) / len(i), 2) for i in y]

    B = find_coefficient(x, y_aver, n)

    Gp = kriteriy_cochrena(y, y_aver, n, m)
    print(f'Gp = {Gp}')
    if Gp < G_kr:
        print(f'З ймовірністю {1-q} дисперсії однорідні.')
    else:
```

```
        print("Необхідно збільшити ксть дослідів")
        m += 1
        execute(n, m)

    ts = kriteriy_studenta(x_norm[:, 1:], y, y_aver, n, m)
    print('\nКритерій Стьюдента:\n', ts)
    res = [t for t in ts if t > t_student]
    final_k = [B[ts.index(i)] for i in ts if i in res]
    print('Коефіцієнти {} статистично незначущі, тому ми виключаємо їх з
рівняння.'.format([i for i in B if i not in final_k]))

    y_new = []
    for j in range(n):
        y_new.append(regression([x[j][ts.index(i)] for i in ts if i in res],
final_k))

    print(f'\nЗначення "y" з коефіцієнтами {final_k}')
    print(y_new)

    d = len(res)
    f4 = n - d
    F_p = kriteriy_fishera(y, y_aver, y_new, n, m, d)

    fisher = partial(f.ppf, q=1 - 0.05)
    f_t = fisher(dfn=f4, dfd=f3)

    print('\nПеревірка адекватності за критерієм Фішера')
    print('Fp =', F_p)
    print('F_t =', f_t)
    if F_p < f_t:
        print('Математична модель адекватна експериментальним даним')
    else:
        print('Математична модель не адекватна експериментальним даним')


execute(4, 4)
```

Результати роботи програми:

```
C:\Users\dimau\AppData\Local\Continuum\anaconda3\python.exe C:/univer/mope/lab3/main.py

Матриця планування
[[  1. -25.  15. -25. 201. 206. 195. 203.]
 [  1. -25.  50. -15. 202. 200. 206. 204.]
 [  1.  -5.  15. -15. 210. 205. 200. 209.]
 [  1.  -5.  50. -25. 195. 197. 209. 197.]]

Рівняння регресії
213.36 + 0.03*x1 + -0.07*x2 + 0.41*x3

Перевірка за критерієм Кохрена
Gp = 0.45597775718257644
З ймовірністю 0.95 дисперсії однорідні.

Критерій Стьюдента:
 [197.21058498099168, 0.3044312827739915, 1.1568388745411677, 2.009246466308344]
Коефіцієнти [0.03, -0.07, 0.41] статистично незначущі, тому ми виключаємо їх з рівняння

Значення "y" з коефіцієнтами [213.36]
[213.36, 213.36, 213.36, 213.36]

Перевірка адекватності за критерієм Фішера
Fp = 39.56262910101955
F_t = 3.490294819497605
Математична модель не адекватна експериментальним даним

Process finished with exit code 0
```

Контрольні запитання

1. Що називається дробовим факторним експериментом?

Дробовим факторним експериментом називається експеримент з використанням частини повного факторного експерименту

2. Для чого потрібно розрахункове значення Кохрена?

Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.

3. Для чого перевіряється критерій Стьюдента?

За допомогою критерію Стьюдента перевіряється значущість коефіцієнтів рівняння

4. Чим визначається критерій Фішера і як його застосовувати?

Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваного об”єкта.