

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

«Методи оптимізації та планування експерименту»
Лабораторна робота №3

**«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З
ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»**

Виконав:
студент групи ІО-91

Герейханов Тимур

Варіант: 105

Перевірив Регіда П. Г.

Київ 2021 р.

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 Y. Значення функції відгуку знайти у відповідності з варіантом діапазону, зазначеного далі (випадковим чином).

№ _{варіанта}	X ₁		X ₂		X ₃	
	min	max	min	max	min	max
105	-30	20	15	50	20	35

$$y_{\max} = 200 + x_{\text{cp max}}$$

$$y_{\min} = 200 + x_{\text{cp min}}$$

$$x_{\text{cp max}} = (x_{1\max} + x_{2\max} + x_{3\max}) / 3$$

$$x_{\text{cp min}} = (x_{1\min} + x_{2\min} + x_{3\min}) / 3$$

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

3. Провести 3 статистичні перевірки.

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

Роздруківка тексту програми:

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

x_range = [(-5, 15), (-15, 35), (15, 30)]
x_aver_max = (15 + 35 + 30) / 3
x_aver_min = (-5 - 15 + 15) / 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('\nPівняння регресії')
print(f'{B[0]} + {B[1]}*x1 + {B[2]}*x2 + {B[3]}*x3')
```

```
return B
```

```
# квадратна дисперсія
```

```
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
```

```
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
```

```
# оцінки коефіцієнтів
```

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

```
def kriteriy_students(x, y, y_aver, n, m):
```

```
    S_kv = s_kv(y, y_aver, n, m)
    s_kv_aver = sum(S_kv) / n
```

```
    # статистична оцінка дисперсії
```

```
    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 main(n, m):
    f1 = m - 1
    f2 = n
    f3 = f1 * f2
    q = 0.05

    # табличні значення
    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
        main(n, m)

    ts = kriteriy_students(x_norm[:, 1:], y, y_aver, n, m)
    print(f'\nКритерій Ст'юдента:\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 незначні, 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) # табличне знач

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')

if __name__ == '__main__':
    main(4, 4)

```

Результат виконання програми:

Матриця планування

```
[ [ 1.  -5. -15.  15. 203. 199. 225. 199. ]  
 [ 1.  -5.  35.  30. 218. 211. 216. 212. ]  
 [ 1.  15. -15.  30. 211. 200. 215. 199. ]  
 [ 1.  15.  35.  15. 220. 213. 204. 216. ]]
```

Рівняння регресії

$208.18 + -0.03 \cdot x_1 + 0.15 \cdot x_2 + 0.02 \cdot x_3$

Перевірка за критерієм Кохрена

$G_p = 0.5631594814591498$

З ймовірністю 0.95 дисперсії однорідні.

Критерій Стюдента:

[116.71475759189453, 0.17363099909535037, 2.0488457893251346, 0.10417859945721022]

Коефіцієнти [-0.03, 0.15, 0.02] статистично незначні, тому ми виключаємо їх з рівняння.

Значення "y" з коефіцієнтами [208.18]

[208.18, 208.18, 208.18, 208.18]

Перевірка адекватності за критерієм Фішера

$F_p = 1.7775959803034844$

$F_t = 3.490294819497605$

Математична модель адекватна експериментальним даним

Process finished with exit code 0

Відповіді на контрольні питання:

1. Що називається дробовим факторним експериментом?

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?

Розрахункове значення Кохрена потрібне для перевірки однорідності дисперсії.

3. Для чого перевіряється критерій Стьюдента?

Критерій Стьюдента перевіряється для перевірки значущості коефіцієнтів регресії.

4. Чим визначається критерій Фішера і як його застосовувати?

За F-критерієм Фішера перевіряється адекватність моделі, він дорівнює відношенню дисперсії адекватності до дисперсії відтворюваності. Знайдене шляхом розрахунку F_p порівнюють з табличним значенням F_t , що визначається при рівні значимості q та кількості ступенів свободи. Якщо $F_p < F_t$ то отримана математична модель з прийнятим рівнем статистичної значимості q адекватна експериментальним даним.