

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 №4**  
«Проведення трьохфакторного експерименту  
при використанні рівняння регресії з урахуванням ефекту взаємодії»

**Виконав:**

студент II курсу ФІОТ

групи ІО-93

Сукач Артем

**Перевірив:**

Регіда П.Г.

Київ – 2021

**Мета роботи:** Провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об'єкту.

**Завдання на лабораторну роботу:**

1. Скласти матрицю планування для повного трьохфакторного експерименту.
2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

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

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

$$\text{де } x_{\text{cp max}} = \frac{x_{1\max} + x_{2\max} + x_{3\max}}{3}, \quad x_{\text{cp min}} = \frac{x_{1\min} + x_{2\min} + x_{3\min}}{3}$$

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.
3. Провести 3 статистичні перевірки.
4. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
5. Написати комп'ютерну програму, яка усе це виконує.

**Варіант завдання:**

Варіант	X <sub>1</sub>		X <sub>2</sub>		X <sub>3</sub>	
	min	max	min	max	min	max
104	-30	0	15	50	-30	35

**Роздруківка тексту програми:**

```
import random
```

```
import numpy as np
```

```
import sklearn.linear_model as lm
```

```
from scipy.stats import f, t
```

```
from numpy.linalg import solve
```

```
def regression(x, b):
```

```
    y = sum([x[i] * b[i] for i in range(len(x))])
```

```
    return y
```

```
def dispersion(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(round(s, 3))
```

```
    return res
```

```
def planing_matrix_interaction_effect(n, m):
```

```
    x_normalized = [[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]]
```

```
    y = np.zeros(shape=(n, m), dtype=np.int64)
```

```
    for i in range(n):
```

```
        for j in range(m):
```

```
y[i][j] = random.randint(y_min, y_max)
```

```
for x in x_normalized:
```

```
    x.append(x[1] * x[2])
```

```
    x.append(x[1] * x[3])
```

```
    x.append(x[2] * x[3])
```

```
    x.append(x[1] * x[2] * x[3])
```

```
x_normalized = np.array(x_normalized[:len(y)])
```

```
x = np.ones(shape=(len(x_normalized), len(x_normalized[0])), dtype=np.int64)
```

```
for i in range(len(x_normalized)):
```

```
    for j in range(1, 4):
```

```
        if x_normalized[i][j] == -1:
```

```
            x[i][j] = x_range[j - 1][0]
```

```
        else:
```

```
            x[i][j] = x_range[j - 1][1]
```

```
for i in range(len(x)):
```

```
    x[i][4] = x[i][1] * x[i][2]
```

```
    x[i][5] = x[i][1] * x[i][3]
```

```
    x[i][6] = x[i][2] * x[i][3]
```

```
    x[i][7] = x[i][1] * x[i][3] * x[i][2]
```

```
print(f'\nМатриця планування для n = {n}, m = {m}:')
```

```
print('\n3 кодованими значеннями факторів:')
```

```
print('\n  X0  X1  X2  X3  X1X2  X1X3  X2X3  X1X2X3  Y1  Y2  Y3')
```

```
print(np.concatenate((x, y), axis=1))
```

```
print('\nНормовані значення факторів:\n')
```

```
print(x_normalized)
```

```
return x, y, x_normalized
```

```
def find_coef(X, Y, norm=False):
```

```
    skm = lm.LinearRegression(fit_intercept=False)
```

```
    skm.fit(X, Y)
```

```
    B = skm.coef_
```

```
    if norm == 1:
```

```
        print('\nКоефіцієнти рівняння регресії з нормованими X:')
```

```
    else:
```

```
        print('\nКоефіцієнти рівняння регресії:')
```

```
    B = [round(i, 3) for i in B]
```

```
    print(B)
```

```
    return B
```

```
def bs(x, y, y_aver, n):
```

```
    res = [sum(1 * y for y in y_aver) / n]
```

```
    for i in range(7):
```

```
        b = sum(j[0] * j[1] for j in zip(x[:, i], y_aver)) / n
```

```
        res.append(b)
```

```
    return res
```

```
def kriteriy_students2(x, y, y_aver, n, m):
```

```
    S_kv = dispersion(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 = [round(abs(B) / s_Bs, 3) for B in Bs]

return ts

```

```

def kriteriy_studentsa(x, y_average, n, m, dispersion):

    dispersion_average = sum(dispersion) / n

    s_beta_s = (dispersion_average / n / m) ** 0.5

    beta = [sum(1 * y for y in y_average) / n]

    for i in range(3):

        b = sum(j[0] * j[1] for j in zip(x[:,i], y_average)) / n

        beta.append(b)

    t = [round(abs(b) / s_beta_s, 3) for b in beta]

    return t

```

```

def kriteriy_fishera(y, y_average, y_new, n, m, d, dispersion):

    S_ad = m / (n - d) * sum([(y_new[i] - y_average[i])**2 for i in range(len(y))])

    dispersion_average = sum(dispersion) / n

    return S_ad / dispersion_average

```

```

def check(X, Y, B, n, m, norm=False):

    f1 = m - 1

```

```

f2 = n

f3 = f1 * f2

q = 0.05

y_aver = [round(sum(i) / len(i), 3) for i in Y]

print('\nСереднє значення y:', y_aver)

dispersion_arr = dispersion(Y, y_aver, n, m)

qq = (1 + 0.95) / 2

student_cr_table = t.ppf(df=f3, q=qq)

ts = kriteriy_studenta2(X[:, 1:], Y, y_aver, n, m)

temp_cohren = f.ppf(q=(1 - q / f1), dfn=f2, dfd=(f1 - 1) * f2)

cohren_cr_table = temp_cohren / (temp_cohren + f1 - 1)

Gp = max(dispersion_arr) / sum(dispersion_arr)

print('Дисперсія y:', dispersion_arr)

print(f'Gp = {Gp}')

if Gp < cohren_cr_table:

    print(f'З ймовірністю {1-q} дисперсії однорідні.')

else:

    print("Необхідно збільшити кількість дослідів")

    m += 1

    with_interaction_effect(n, m)

print('\nКритерій Стюдента:\n', ts)

res = [t for t in ts if t > student_cr_table]

```

```

final_k = [B[i] for i in range(len(ts)) if ts[i] in res]

print('\nКоефіцієнти {} статистично незначущі, тому ми виключаємо їх з рівняння.'.format(
    [round(i, 3) for i in B if i not in final_k]))

y_new = []

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

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

d = len(res)

if d >= n:
    print('\nF4 <= 0')
    print("")
    return

f4 = n - d

Fp = kriteriy_fishera(Y, y_aver, y_new, n, m, d, dispersion_arr)

Ft = f.ppf(dfn=f4, dfd=f3, q=1 - 0.05)

print('\nПеревірка адекватності за критерієм Фішера')
print('Fp =', Fp)
print('Ft =', Ft)

if Fp < Ft:
    print('Математична модель адекватна експериментальним даним')
    return True

else:
    print('Математична модель не адекватна експериментальним даним')

```



```
return False
```

```
def with_interaction_effect(n, m):
```

```
    X, Y, X_norm = planing_matrix_interaction_effect(n, m)
```

```
    y_aver = [round(sum(i) / len(i), 3) for i in Y]
```

```
    B_norm = find_coef(X_norm, y_aver, norm=True)
```

```
    return check(X_norm, Y, B_norm, n, m, norm=True)
```

```
def planning_matrix_linear(n, m, x_range):
```

```
    x_normalized = 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]])
```

```
    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_normalized = x_normalized[:len(y)]
```

```
    x = np.ones(shape=(len(x_normalized), len(x_normalized[0])))
```

```

for i in range(len(x_normalized)):
    for j in range(1, len(x_normalized[i])):
        if x_normalized[i][j] == -1:
            x[i][j] = x_range[j-1][0]
        else:
            x[i][j] = x_range[j-1][1]

```

```

print('\nМатриця планування:')
print('\n  X0 X1 X2 X3 Y1 Y2 Y3 ')
print(np.concatenate((x, y), axis=1))

```

```

return x, y, x_normalized

```

```

def regression_equation(x, y, n):
    y_average = [round(sum(i) / len(i), 2) for i in y]

    mx1 = sum(x[:, 1]) / n
    mx2 = sum(x[:, 2]) / n
    mx3 = sum(x[:, 3]) / n

    my = sum(y_average) / n

    a1 = sum([y_average[i] * x[i][1] for i in range(len(x))]) / n
    a2 = sum([y_average[i] * x[i][2] for i in range(len(x))]) / n
    a3 = sum([y_average[i] * x[i][3] for i in range(len(x))]) / 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
```

```
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'y = {B[0]} + {B[1]}*x1 + {B[2]}*x2 + {B[3]}*x3')
```

```
return y_average, B
```

```
def linear(n, m):
```

```
    f1 = m - 1
```

```
    f2 = n
```

```
    f3 = f1 * f2
```

```
    q = 0.05
```

```
x, y, x_norm = planning_matrix_linear(n, m, x_range)
```

```
y_average, B = regression_equation(x, y, n)
```

```
dispersion_arr = dispersion(y, y_average, n, m)
```

```
temp_cohren = f.ppf(q=(1 - q / f1), dfn=f2, dfd=(f1 - 1) * f2)
```

```
cohren_cr_table = temp_cohren / (temp_cohren + f1 - 1)
```

```
Gp = max(dispersion_arr) / sum(dispersion_arr)
```

```

print('\nПеревірка за критерієм Кохрена:\n')

print(f'Розрахункове значення: Gp = {Gp}')

    f'\nТабличне значення: Gt = {cohren_cr_table}')

if Gp < cohren_cr_table:

    print(f'З ймовірністю {1-q} дисперсії однорідні.')

else:

    print("Необхідно збільшити ксть дослідів")

    m += 1

    linear(n, m)

qq = (1 + 0.95) / 2

student_cr_table = t.ppf(df=f3, q=qq)

student_t = kriteriy_studenta(x_norm[:,1:], y_average, n, m, dispersion_arr)

print('\nТабличне значення критерій Стьюдента:\n', student_cr_table)

print('Розрахункове значення критерій Стьюдента:\n', student_t)

res_student_t = [temp for temp in student_t if temp > student_cr_table]

final_coefficients = [B[student_t.index(i)] for i in student_t if i in res_student_t]

print('Коефіцієнти {} статистично незначущі.'.

    format([i for i in B if i not in final_coefficients]))

y_new = []

for j in range(n):

    y_new.append(regression([x[j][student_t.index(i)] for i in student_t if i in res_student_t],

final_coefficients))

print(f'\nОтримаємо значення рівня регресії для {m} дослідів: ')

print(y_new)

```

```

d = len(res_student_t)

f4 = n - d

Fp = kriteriy_fishera(y, y_average, y_new, n, m, d, dispersion_arr)

Ft = f.ppf(dfn=f4, dfd=f3, q=1 - 0.05)

print('\nПеревірка адекватності за критерієм Фішера:\n')

print('Розрахункове значення критерія Фішера: Fp =', Fp)

print('Табличне значення критерія Фішера: Ft =', Ft)

if Fp < Ft:

    print('Математична модель адекватна експериментальним даним')

    return True

else:

    print('Математична модель не адекватна експериментальним даним')

    return False

```

```

def main(n, m):

    main_1 = linear(n, m)

    if not main_1:

        interaction_effect = with_interaction_effect(n, m)

        if not interaction_effect:

            main(n, m)

```

```

if __name__ == '__main__':

    x_range = ((-30, 0), (15, 50), (-30, -35))

    y_max = 200 + int(sum([x[1] for x in x_range]) / 3)

    y_min = 200 + int(sum([x[0] for x in x_range]) / 3)

```

main(8, 3)

## Результати роботи програми:

Матриця планування:

	X0	X1	X2	X3	Y1	Y2	Y3
[	1.	-30.	15.	-30.	200.	186.	193.]
[	1.	-30.	50.	-35.	199.	185.	203.]
[	1.	0.	15.	-35.	192.	186.	193.]
[	1.	0.	50.	-30.	185.	195.	188.]
[	1.	-30.	15.	-35.	196.	202.	193.]
[	1.	-30.	50.	-30.	193.	201.	203.]
[	1.	0.	15.	-30.	186.	200.	185.]
[	1.	0.	50.	-35.	185.	190.	204.]]

Рівняння регресії:

$y = 182.22 + -0.18 \cdot x_1 + 0.05 \cdot x_2 + -0.22 \cdot x_3$

Перевірка за критерієм Кохрена:

Розрахункове значення:  $G_p = 0.24536155229588935$

Табличне значення:  $G_t = 0.815948432359917$

З ймовірністю 0.95 дисперсії однорідні.

Табличне значення критерій Стюдента:

2.1199052992210112

Розрахункове значення критерій Стюдента:

[165.119, 2.313, 0.676, 0.463]

Коефіцієнти [0.05, -0.22] статистично незначущі.

Отримаємо значення рівня регресії для 3 дослідів:

[187.62, 187.62, 182.22, 182.22, 187.62, 187.62, 182.22, 182.22]

Перевірка адекватності за критерієм Фішера:

Розрахункове значення критерія Фішера:  $F_p = 9.250948937235828$

Табличне значення критерія Фішера:  $F_t = 2.741310828338778$

Математична модель не адекватна експериментальним даним

Матриця планування для  $n = 8$ ,  $m = 3$ :

3 кодованими значеннями факторів:

	X0	X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	Y1	Y2	Y3
[	1	-30	15	-30	-450	900	-450	13500	187	203	192]
[	1	-30	50	-35	-1500	1050	-1750	52500	203	190	187]
[	1	0	15	-35	0	0	-525	0	188	196	205]
[	1	0	50	-30	0	0	-1500	0	199	199	198]
[	1	-30	15	-35	-450	1050	-525	15750	185	188	190]
[	1	-30	50	-30	-1500	900	-1500	45000	185	185	191]
[	1	0	15	-30	0	0	-450	0	192	190	197]
[	1	0	50	-35	0	0	-1750	0	201	203	186]]

Нормовані значення факторів:

[	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]
[	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]]

Коефіцієнти рівняння регресії з нормованими X:

[193.333, 2.833, 0.583, 0.167, 0.917, 0.167, 0.917, -2.25]

Середнє значення  $y$ : [194.0, 193.333, 196.333, 198.667, 187.667, 187.0, 193.0, 196.667]

Дисперсія  $y$ : [44.667, 48.222, 48.222, 0.222, 4.222, 8.0, 8.667, 57.556]

$\sigma_p = 0.2618824450126946$

З ймовірністю 0.95 дисперсії однорідні.

Критерій Стюдента:

[180.703, 2.648, 0.545, 0.156, 0.857, 0.156, 0.857, 2.103]

Коефіцієнти [0.583, 0.167, 0.917, 0.167, 0.917, -2.25] статистично незначущі, тому ми виключаємо їх з рівняння.

Значення "y" з коефіцієнтами [193.333, 2.833]

[190.5, 190.5, 196.166, 196.166, 190.5, 190.5, 196.166, 196.166]

Перевірка адекватності за критерієм Фішера

$F_p = 1.0393983929237682$

$F_t = 2.741310828338778$

Математична модель адекватна експериментальним даним

Process finished with exit code 0

**Висновок:**

У ході лабораторної роботи я змодельював трьохфакторний експеримент з використанням лінійного рівняння регресії, склав матрицю планування експерименту, визначив коефіцієнти рівняння регресії, натуралізовані та нормовані, виконав перевірку правильності розрахунку коефіцієнтів рівняння регресії. Також я реалізував 3 статистичні перевірки за критерієм Кохрена, Стюдента та Фішера. При виявленні неадекватності лінійного рівняння регресії оригіналу було застосовано ефект взаємодії факторів.