

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

Методи наукових досліджень  
Лабораторна робота №5

«Проведення трьохфакторного експерименту при використанні  
рівняння регресії з урахуванням квадратичних членів (центральный  
ортogonalный композиційний план)»

Виконала:  
студентка групи ІВ-93  
Стефанович І. В.  
Варіант: 23  
Перевірив:  
Регіда П.Г

Київ - 2021 р.

**Мета:** провести трьохфакторний експеримент з урахуванням квадратичних членів ,використовуючи центральний ортогональний композиційний план. Знайти рівняння регресії, яке буде адекватним для опису об'єкту.

**Індивідуальне завдання:**

323 | -6 | 2 | -9 | 5 | -3 | 9

**Лістинг коду програми:**

```
from sklearn import linear_model
from scipy.stats import f, t
import numpy as np
import random

class Lab5:
    def __init__(self):
        self.m = 3

        self.x1min = -6
        self.x1max = 2
        self.x2min = -9
        self.x2max = 5
        self.x3min = -3
        self.x3max = 9

        self.y_max = 200 + (self.x1max + self.x2max + self.x3max) / 3
        self.y_min = 200 + (self.x1min + self.x2min + self.x3min) / 3

        self.xn = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                    [-1, -1, -1, -1, 1, 1, 1, 1, -1.215, 1.215, 0, 0, 0, 0],
                    [-1, -1, 1, 1, -1, -1, 1, 1, 0, 0, -1.215, 1.215, 0, 0],
                    [-1, 1, -1, 1, -1, 1, -1, 1, 0, 0, 0, 0, -1.215, 1.215, 0]]

        self.x1x2_norm = [0] * 15
        self.x1x3_norm = [0] * 15
```

```

self.x2x3_norm = [0] * 15
self.x1x2x3_norm = [0] * 15
self.x1kv_norm = [0] * 15
self.x2kv_norm = [0] * 15
self.x3kv_norm = [0] * 15

```

```

for i in range(15):

```

```

    self.x1x2_norm[i] = self.xn[1][i] * self.xn[2][i]
    self.x1x3_norm[i] = self.xn[1][i] * self.xn[3][i]
    self.x2x3_norm[i] = self.xn[2][i] * self.xn[3][i]
    self.x1x2x3_norm[i] = self.xn[1][i] * self.xn[2][i] * self.xn[3][i]
    self.x1kv_norm[i] = round(self.xn[1][i] ** 2, 3)
    self.x2kv_norm[i] = round(self.xn[2][i] ** 2, 3)
    self.x3kv_norm[i] = round(self.xn[3][i] ** 2, 3)

```

```

    self.Y = [[random.randint(int(self.y_min), int(self.y_max)) for i in range(self.m)] for j
in range(15)]

```

```

self.calc()

```

```

def calc(self):

```

```

    print("Матриця планування Y:")
    for i in range(15):
        print(self.Y[i])

```

```

    x01 = (self.x1max + self.x1min) / 2
    x02 = (self.x2max + self.x2min) / 2
    x03 = (self.x3max + self.x3min) / 2

```

```

    delta_x1 = self.x1max - x01
    delta_x2 = self.x2max - x02
    delta_x3 = self.x3max - x03

```

```

    x1 = [-5, -5, -5, -5, 4, 4, 4, 4, -1.215 * delta_x1 + x01, 1.215 * delta_x1 + x01, x01,
x01, x01, x01, x01]
    x2 = [-2, -2, 7, 7, -2, -2, 7, 7, x02, x02, -1.215 * delta_x2 + x02, 1.215 * delta_x2 +
x02, x02, x02, x02]

```

```
x3 = [-1, 2, -1, 2, -1, 2, -1, 2, x03, x03, x03, x03, -1.215 * delta_x3 + x03, 1.215 *  
delta_x3 + x03, x03]
```

```
x1x2 = [0] * 15
```

```
x1x3 = [0] * 15
```

```
x2x3 = [0] * 15
```

```
x1x2x3 = [0] * 15
```

```
x1kv = [0] * 15
```

```
x2kv = [0] * 15
```

```
x3kv = [0] * 15
```

```
for i in range(15):
```

```
    x1x2[i] = round(x1[i] * x2[i], 3)
```

```
    x1x3[i] = round(x1[i] * x3[i], 3)
```

```
    x2x3[i] = round(x2[i] * x3[i], 3)
```

```
    x1x2x3[i] = round(x1[i] * x2[i] * x3[i], 3)
```

```
    x1kv[i] = round(x1[i] ** 2, 3)
```

```
    x2kv[i] = round(x2[i] ** 2, 3)
```

```
    x3kv[i] = round(x3[i] ** 2, 3)
```

```
y_average = []
```

```
for i in range(len(self.Y)):
```

```
    y_average.append(np.mean(self.Y[i], axis=0))
```

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

```
list_for_b = list(
```

```
    zip(self.xn[0], self.xn[1], self.xn[2], self.xn[3], self.x1x2_norm, self.x1x3_norm,  
self.x2x3_norm, self.x1x2x3_norm, self.x1kv_norm, self.x2kv_norm,  
    self.x3kv_norm))
```

```
print("\nМатриця планування з нормованими коефіцієнтами X:")
```

```
for i in range(15):
```

```
    print(list_for_b[i])
```

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

```
skm.fit(list_for_b, y_average)
```

```
b = skm.coef_
```

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

```
print("\nРівняння регресії зі знайденими коефіцієнтами: \n y = {} + {}*x1 + {}*x2 + {}*x3 + " +  
      " {}*x1x2 + {}*x1x3 + {}*x2x3 + {}*x1x2x3 {}*x1^2 + {}*x2^2 + {}*x3^2").format(b[0], b[1], b[2], b[3], b[4], b[5], b[6], b[7], b[8], b[9], b[10]))
```

```
print("\nПеревірка за критерієм Кохрена:")
```

```
print("Середні значення відгуку за рядками:", "\n", +y_average[0], y_average[1],  
y_average[2], y_average[3],  
      y_average[4], y_average[5], y_average[6], y_average[7], y_average[8],  
y_average[9], y_average[10],
```

```
y_average[11], y_average[12], y_average[13], y_average[14])
```

```
dispersions = []
```

```
for i in range(len(self.Y)):
```

```
    a = 0
```

```
    for k in self.Y[i]:
```

```
        a += (k - np.mean(self.Y[i], axis=0)) ** 2
```

```
    dispersions.append(a / len(self.Y[i]))
```

```
gp = max(dispersions) / sum(dispersions)
```

```
gt = 0.3346
```

```
if gp < gt:
```

```
    print("Дисперсія однорідна")
```

```
else:
```

```
    print("Дисперсія неоднорідна")
```

```
print("\nПеревірка значущості коефіцієнтів за критерієм Стьюдента:")
```

```
sb = sum(dispersions) / len(dispersions)
```

```
sbs = (sb / (15 * self.m)) ** 0.5
```

```
t_list = [abs(b[i]) / sbs for i in range(0, 11)]
```

```
d = 0
```

```
res = [0] * 11
```

```

coefs1 = []
coefs2 = []
n = 15
F3 = (self.m - 1) * n
for i in range(11):
    if t_list[i] < t.ppf(q=0.975, df=F3):
        coefs2.append(b[i])
        res[i] = 0
    else:
        coefs1.append(b[i])
        res[i] = b[i]
        d += 1
print("Значущі коефіцієнти регресії:", coefs1)
print("Незначущі коефіцієнти регресії:", coefs2)

y_st = []
for i in range(15):
    y_st.append(
        res[0] + res[1] * self.xn[1][i] + res[2] * self.xn[2][i] + res[3] * self.xn[3][i] +
res[4] * self.x1x2_norm[i] + res[5] *
        self.x1x3_norm[i] + res[6] * self.x2x3_norm[i] + res[7] * self.x1x2x3_norm[i] +
res[8] * self.x1kv_norm[i] + res[9] *
        self.x2kv_norm[i] + res[10] * self.x3kv_norm[i])
print("Значення з отриманими коефіцієнтами:", "\n", y_st)

print("\nПеревірка адекватності за критерієм Фішера:")
sad = self.m * sum([(y_st[i] - y_average[i]) ** 2 for i in range(15)]) / (n - d)
fp = sad / sb
f4 = n - d

print("Fp =", fp)

if fp < f.ppf(q=0.95, dfn=f4, dfd=F3):
    print("Рівняння регресії адекватне при рівні значимості 0.05")
else:
    print("Рівняння регресії неадекватне при рівні значимості 0.05")

```

Lab5()

## Результати роботи програми:

Матриця планування Y:

```
[198, 194, 205]
[199, 202, 195]
[195, 199, 195]
[203, 204, 198]
[204, 195, 201]
[198, 194, 196]
[203, 195, 196]
[203, 204, 197]
[197, 203, 202]
[203, 198, 196]
[205, 197, 202]
[204, 196, 200]
[200, 196, 198]
[200, 204, 199]
[202, 195, 199]
```

Матриця планування з нормованими коефіцієнтами X:

```
(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, 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, -1.215, 0, 0, -0.0, -0.0, 0, -0.0, 1.476, 0, 0)
(1, 1.215, 0, 0, 0.0, 0.0, 0, 0.0, 1.476, 0, 0)
(1, 0, -1.215, 0, -0.0, 0, -0.0, -0.0, 0, 1.476, 0)
(1, 0, 1.215, 0, 0.0, 0, 0.0, 0.0, 0, 1.476, 0)
(1, 0, 0, -1.215, 0, -0.0, -0.0, -0.0, 0, 0, 1.476)
(1, 0, 0, 1.215, 0, 0.0, 0.0, 0.0, 0, 0, 1.476)
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

Рівняння регресії зі знайденими коефіцієнтами:

$y = 200.041 + -0.215 \cdot x_1 + 0.187 \cdot x_2 + 0.729 \cdot x_3 + 0.375 \cdot x_1 x_2 + -0.708 \cdot x_1 x_3 + 1.625 \cdot x_2 x_3 + 0.208 \cdot x_1 x_2 x_3 - 0.446 \cdot x_1^2 + 0.118 \cdot x_2^2 + -0.672 \cdot x_3^2$

Перевірка за критерієм Кохрена:

Середні значення відгуку за рядками:

199.0 198.667 196.333 201.667 200.0 196.0 198.0 201.333 200.667 199.0 201.333 200.0 198.0 201.0 198.667

Дисперсія однорідна

Перевірка значущості коефіцієнтів за критерієм Стюдента:

Значущі коефіцієнти регресії: [200.041, 1.625]

Незначущі коефіцієнти регресії: [-0.215, 0.187, 0.729, 0.375, -0.708, 0.208, -0.446, 0.118, -0.672]

Значення з отриманими коефіцієнтами:

[201.666, 198.416, 198.416, 201.666, 201.666, 198.416, 198.416, 201.666, 200.041, 200.041, 200.041, 200.041, 200.041, 200.041]

Перевірка адекватності за критерієм Фішера:

$F_p = 0.8072956980540663$

Рівняння регресії адекватне при рівні значимості 0.05

**Висновки:** в даній лабораторній роботі проведено трьохфакторний експеримент з урахуванням квадратичних членів, використовуючи центральний ортогональний композиційний план. Знайдено рівняння регресії, яке буде адекватним для опису об'єкту.