

Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №6
З дисципліни «Методи наукових досліджень»
**Проведення трьохфакторного експерименту при використанні рівняння
регресії з квадратичними членами**

ВИКОНАВ:
Студент II курсу ФІОТ
Групи ІВ-91
Онищук Ю. І.
Номер заліковки: 9122
Номер у списку: 20

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ 2021 р.

Мета роботи: Провести трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи **рототабельний** композиційний план.

Завдання до лабораторної роботи:

1. Ознайомитися з теоретичними відомостями.
2. Вибрати з таблиці варіантів і записати в протокол інтервали значень x_1 , x_2 , x_3 . Обчислити і записати значення, відповідні кодованим значенням факторів +1; -1; + ; - ; 0 для 1, 2, 3.
3. Значення функції відгуку знайти за допомогою підстановки в формулу: $y_i = f(x_1, x_2, x_3) + \text{random}(10) - 5$, де $f(x_1, x_2, x_3)$ вибирається по номеру в списку в журналі викладача.
4. Провести експерименти і аналізуючи значення статистичних перевірок, отримати адекватну модель рівняння регресії. При розрахунках використовувати натуральні значення факторів.
5. Зробити висновки по виконаній роботі.

Алгоритм отримання адекватної моделі рівняння регресії

- 1) Вибір рівняння регресії (лінійна форма, рівняння з урахуванням ефекту взаємодії і з урахуванням квадратичних членів);
- 2) Вибір кількості повторів кожної комбінації ($m = 2$);
- 3) Складення матриці планування експерименту і вибір кількості рівнів (N)
- 4) Проведення експериментів;
- 5) Перевірка однорідності дисперсії. Якщо не однорідна – повертаємося на п. 2 і збільшуємо m на 1);
- 6) Розрахунок коефіцієнтів рівняння регресії. При розрахунку використовувати **натуральні** значення x_1 , x_2 и x_3 .
- 7) Перевірка нуль-гіпотези. Визначення значимих коефіцієнтів;
- 8) Перевірка адекватності моделі рівняння оригіналу. При неадекватності – повертаємося на п.1, змінивши при цьому рівняння регресії;

Виконання роботи

№ варіанту	x_1		x_2		x_3		$f(x_1, x_2, x_3)$
	min	max	min	max	min	max	
120	-30	20	-70	-10	-70	-40	$2,1+1,7*x_1+6,8*x_2+6,6*x_3+9,5*x_1*x_1+1,0*x_2*x_2+3,9*x_3*x_3+3,0*x_1*x_2+0,1*x_1*x_3+4,5*x_2*x_3+1,8*x_1*x_2*x_3$

Лістинг програми

```
from math import fabs
from random import randrange
import numpy as np
from numpy.linalg import solve
from scipy.stats import f, t

m = 3
n = 15

x1_min = -30
x1_max = 20
x2_min = -70
x2_max = -10
x3_min = -70
x3_max = -40
```

```

coef_1 = [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

while True:
    def function(x_1, x_2, x_3, c):
        y = 2.1*c[0] + 1.7 * x_1*c[1] + 6.8 * x_2*c[2] + 6.6 * x_3*c[3] + 9.5 *
x_1 * x_1*c[4] + 1.0 * x_2 * x_2*c[5] + \
        3.9 * x_3 * x_3*c[6] + 3.0 * x_1 * x_2*c[7] + 0.1 * x_1 * x_3*c[8] +
4.5 * x_2 * x_3*c[9] + 1.8 * x_1 * x_2 \
        * x_3*c[10] + randrange(0, 10) - 5
        return y

    x01 = (x1_max + x1_min) / 2
    x02 = (x2_max + x2_min) / 2
    x03 = (x3_max + x3_min) / 2
    x1_delt = x1_max - x01
    x2_delt = x2_max - x02
    x3_delt = x3_max - x03

    xn = [[-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],
[-1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],
[+1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],
[0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],
[0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],
[0, 0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929],
[0, 0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

    x1 = [x1_min, x1_min, x1_min, x1_min, x1_max, x1_max, x1_max, x1_max, -1.73
* x1_delt + x01, 1.73 * x1_delt + x01,
x01, x01, x01, x01, x01]
    x2 = [x2_min, x2_min, x2_max, x2_max, x2_min, x2_min, x2_max, x2_max, x02,
x02, -1.73 * x2_delt + x02, 1.73 *
x2_delt + x02, x02, x02, x02]
    x3 = [x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x03,
x03, x03, -1.73 * x3_delt + x03,
1.73 * x3_delt + x03, x03]

    x1x2, x1x3, x2x3, x1x2x3 = [0] * n, [0] * n, [0] * n, [0] * n
    x1kv, x2kv, x3kv = [0] * n, [0] * n, [0] * n

    for i in range(15):
        x1x2[i] = x1[i] * x2[i]
        x1x3[i] = x1[i] * x3[i]
        x2x3[i] = x2[i] * x3[i]
        x1x2x3[i] = x1[i] * x2[i] * x3[i]
        x1kv[i] = x1[i] ** 2
        x2kv[i] = x2[i] ** 2
        x3kv[i] = x3[i] ** 2

    list_for_a = list(zip(x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3, x1kv, x2kv,
x3kv))
    print("")

    print("Матриця планування з натуралізованими коефіцієнтами X")
    print("      x1      x2      x3      x1x2      x1x3      x2x3
x1x2x3      x1x1"
"      x2x2      x3x3")

```

```

for i in range(n):
    for j in range(len(list_for_a[0])):
        print("{:^12.3f}".format(list_for_a[i][j]), end='')
    print("")
# вивід матриці планування
Y = [[function(list_for_a[j][0], list_for_a[j][1], list_for_a[j][2], coef_1)
for i in range(m)] for j in range(15)]
print("")

print("Матриця планування Y")
print("      Y1      Y2      Y3")
for i in range(n):
    print(end='')
    for j in range(len(Y[0])):
        print("{:^12.3f}".format(Y[i][j]), end='')
    print("")
# середні y
aver_y = []
for i in range(len(Y)):
    aver_y.append(np.mean(Y[i], axis=0))
print("")

print("Середні значення відгуку за рядками")
for i in range(n):
    print("{:.3f}".format(aver_y[i]))
# розрахунок дисперсій
dispersions = []
for i in range(len(Y)):
    a = 0
    for k in Y[i]:
        a += (k - np.mean(Y[i], axis=0)) ** 2
    dispersions.append(a / len(Y[i]))

def find_kn(num):
    aa = 0
    for o in range(n):
        aa += aver_y[o] * list_for_a[o][num - 1] / n
    return aa

def a(first, second):
    aaa = 0
    for p in range(n):
        aaa += list_for_a[p][first - 1] * list_for_a[p][second - 1] / n
    return aaa

my = sum(aver_y) / n
mx = []
for i in range(10):
    number_lst = []
    for j in range(n):
        number_lst.append(list_for_a[j][i])
    mx.append(sum(number_lst) / len(number_lst))

det_1 = [
    1, mx[0], mx[1], mx[2], mx[3], mx[4], mx[5], mx[6], mx[7], mx[8],
mx[9]],
    [mx[0], a(1, 1), a(1, 2), a(1, 3), a(1, 4), a(1, 5), a(1, 6), a(1, 7),
a(1, 8), a(1, 9), a(1, 10)],
    [mx[1], a(2, 1), a(2, 2), a(2, 3), a(2, 4), a(2, 5), a(2, 6), a(2, 7),
a(2, 8), a(2, 9), a(2, 10)],
    [mx[2], a(3, 1), a(3, 2), a(3, 3), a(3, 4), a(3, 5), a(3, 6), a(3, 7),

```

```

a(3, 8), a(3, 9), a(3, 10)],
    [mx[3], a(4, 1), a(4, 2), a(4, 3), a(4, 4), a(4, 5), a(4, 6), a(4, 7),
a(4, 8), a(4, 9), a(4, 10)],
    [mx[4], a(5, 1), a(5, 2), a(5, 3), a(5, 4), a(5, 5), a(5, 6), a(5, 7),
a(5, 8), a(5, 9), a(5, 10)],
    [mx[5], a(6, 1), a(6, 2), a(6, 3), a(6, 4), a(6, 5), a(6, 6), a(6, 7),
a(6, 8), a(6, 9), a(6, 10)],
    [mx[6], a(7, 1), a(7, 2), a(7, 3), a(7, 4), a(7, 5), a(7, 6), a(7, 7),
a(7, 8), a(7, 9), a(7, 10)],
    [mx[7], a(8, 1), a(8, 2), a(8, 3), a(8, 4), a(8, 5), a(8, 6), a(8, 7),
a(8, 8), a(8, 9), a(8, 10)],
    [mx[8], a(9, 1), a(9, 2), a(9, 3), a(9, 4), a(9, 5), a(9, 6), a(9, 7),
a(9, 8), a(9, 9), a(9, 10)],
    [mx[9], a(10, 1), a(10, 2), a(10, 3), a(10, 4), a(10, 5), a(10, 6),
a(10, 7), a(10, 8), a(10, 9), a(10, 10)]]

det_2 = [my, find_kn(1), find_kn(2), find_kn(3), find_kn(4), find_kn(5),
find_kn(6), find_kn(7), find_kn(8),
    find_kn(9), find_kn(10)]

beta = solve(det_1, det_2)
print("")

print("Отримане рівняння регресії")
print("{:.3f} + {:.3f} * X1 + {:.3f} * X2 + {:.3f} * X3 + {:.3f} * X1X2 +
 {:.3f} * X1X3 +\n + {:.3f} * X2X3 + "
    "{:.3f} * X1X2X3 + {:.3f} * X11^2 + {:.3f} * X22^2 + {:.3f} * X33^2 =
y".format(beta[0], beta[1], beta[2],
beta[3], beta[4], beta[5],
beta[6], beta[7], beta[8],
beta[9], beta[10]))
y_i = [0] * n
print("")
print("Експериментальні значення")
for k in range(n):
    y_i[k] = beta[0] + beta[1] * list_for_a[k][0] + beta[2] *
list_for_a[k][1] + beta[3] * list_for_a[k][2] + \
        beta[4] * list_for_a[k][3] + beta[5] * list_for_a[k][4] +
beta[6] * list_for_a[k][5] + beta[7] * \
        list_for_a[k][6] + beta[8] * list_for_a[k][7] + beta[9] *
list_for_a[k][8] + beta[10] * \
        list_for_a[k][9]
    for i in range(n):
        print("{:.3f}".format(y_i[i]))
    print("")

print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка за критерієм
Кохрена"))
Gp = max(dispersions) / sum(dispersions)
GT = 0.3346
print("Gp =", Gp)
if Gp < GT:
    print("\033[1m\033[30m\033[42mGp={} < GT={} - Дисперсія
однорідна\033[0m".format(round(Gp, 3), GT))
else:
    print("\033[1m\033[30m\033[41m{}\033[0m".format("Дисперсія
неоднорідна"))
    print("")
    print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка значущості
коефіцієнтів за критерієм Стьюдента"))
    sb = sum(dispersions) / len(dispersions)

```

```

sbs = (sb / (n * m)) ** 0.5

F3 = (m - 1) * n
coef_1 = []
coef_2 = []
d = 11
res = [0] * 11
for j in range(11):
    pract_t = 0
    for i in range(15):
        if j == 0:
            pract_t += aver_y[i] / 15
        else:
            pract_t += aver_y[i] * xn[i][j - 1]
        res[j] = beta[j]
    if fabs(pract_t / sbs) < t.ppf(q=0.975, df=F3):
        coef_2.append(beta[j])
        res[j] = 0
        d -= 1
    else:
        coef_1.append(beta[j])
print("Значущі коефіцієнти регресії", [round(i, 3) for i in coef_1])
print("Незначущі коефіцієнти регресії", [round(i, 3) for i in coef_2])
st_y = []
for i in range(n):
    st_y.append(res[0] + res[1] * x1[i] + res[2] * x2[i] + res[3] * x3[i] +
res[4] * x1x2[i] + res[5] *
        x1x3[i] + res[6] * x2x3[i] + res[7] * x1x2x3[i] + res[8] *
x1kv[i] + res[9] *
        x2kv[i] + res[10] * x3kv[i])

print("")

print("Значення з отриманими коефіцієнтами")
for i in range(n):
    print("{:.3f}".format(st_y[i]))
print("")

print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка адекватності за
критерієм Фішера"))
Sad = m * sum([(st_y[i] - aver_y[i]) ** 2 for i in range(n)]) / (n - d)
Fp = Sad / sb
F4 = n - d
print("Fp =", Fp)
if Fp < f.ppf(q=0.95, dfn=F4, dfd=F3):
    print("\033[1m\033[30m\033[42m{}\033[0m".format("При рівні значимості
0.05 рівняння регресії адекватне"))
    break
else:
    print("\033[1m\033[30m\033[41m{}\033[0m".format("При рівні значимості
0.05 рівняння регресії неадекватне"))

```

Результат роботи програми

Lab6 X

C:\Users\~Admin~\AppData\Local\Programs\Python\Python39\python.exe D:/SEM4/MND/Lab6/Lab6.py

Матриця планування з натуралізованими коефіцієнтами X

X1	X2	X3	X1X2	X1X3	X2X3	X1X2X3	X1X1	X2X2	X3X3
-30.000	-70.000	-70.000	2100.000	2100.000	4900.000	-147000.000	900.000	4900.000	4900.000
-30.000	-70.000	-40.000	2100.000	1200.000	2800.000	-84000.000	900.000	4900.000	1600.000
-30.000	-10.000	-70.000	300.000	2100.000	700.000	-21000.000	900.000	100.000	4900.000
-30.000	-10.000	-40.000	300.000	1200.000	400.000	-12000.000	900.000	100.000	1600.000
20.000	-70.000	-70.000	-1400.000	-1400.000	4900.000	98000.000	400.000	4900.000	4900.000
20.000	-70.000	-40.000	-1400.000	-800.000	2800.000	56000.000	400.000	4900.000	1600.000
20.000	-10.000	-70.000	-200.000	-1400.000	700.000	14000.000	400.000	100.000	4900.000
20.000	-10.000	-40.000	-200.000	-800.000	400.000	8000.000	400.000	100.000	1600.000
-48.250	-40.000	-55.000	1930.000	2653.750	2200.000	-106150.000	2328.062	1600.000	3025.000
38.250	-40.000	-55.000	-1530.000	-2103.750	2200.000	84150.000	1463.062	1600.000	3025.000
-5.000	-91.900	-55.000	459.500	275.000	5054.500	-25272.500	25.000	8445.610	3025.000
-5.000	11.900	-55.000	-59.500	275.000	-654.500	3272.500	25.000	141.610	3025.000
-5.000	-40.000	-80.950	200.000	404.750	3238.000	-16190.000	25.000	1600.000	6552.903
-5.000	-40.000	-29.050	200.000	145.250	1162.000	-5810.000	25.000	1600.000	843.903
-5.000	-40.000	-55.000	200.000	275.000	2200.000	-11000.000	25.000	1600.000	3025.000

Lab6 X

Матриця планування Y

Y1	Y2	Y3
-204466.900	-204464.900	-204470.900
-113274.900	-113274.900	-113274.900
-6355.900	-6361.900	-6355.900
-4272.900	-4275.900	-4269.900
221018.100	221020.100	221014.100
123357.100	123360.100	123360.100
50130.100	50121.100	50129.100
25367.100	25366.100	25359.100
-140317.456	-140313.456	-140320.456
183301.344	183296.344	183294.344
-1852.960	-1855.960	-1849.960
14685.380	14679.380	14685.380
12649.625	12653.625	12646.625
47.115	46.115	43.115
3722.100	3725.100	3717.100

```
Lab6 x
↑ Середні значення відгуку за рядками
-204467.567
↓ -113274.900
-6357.900
-4272.900
↕ 221017.433
↕ 123359.100
↕ 50126.767
↕ 25364.100
↕ -140317.123
↕ 183297.344
↕ -1852.960
↕ 14683.380
↕ 12649.958
↕ 45.448
↕ 3721.433

Отримане рівняння регресії
0.003 + 1.823 * X1 + 6.638 * X2 + 6.594 * X3 + 3.002 * X1X2 + 0.102 * X1X3 +
+ 4.497 * X2X3 + 1.800 * X1X2X3 + 9.499 * X11^2 + 1.000 * X22^2 + 3.901 * X33^2 = y
```

```
Lab6 x
↑ Експериментальні значення
-204468.413
↓ -113275.750
-6357.857
-4272.861
↕ 221016.681
↕ 123358.343
↕ 50126.904
↕ 25364.233
↕ -140316.538
↕ 183297.712
↕ -1851.455
↕ 14682.829
↕ 12650.430
↕ 45.930
↕ 3721.427

Перевірка за критерієм Кохрена
Gr = 0.14717741935483872
Gr=0.147 < GT=0.3346 - Дисперсія однорідна

Перевірка значущості коефіцієнтів за критерієм Стюдента
Значущі коефіцієнти регресії [0.003, 1.823, 6.638, 6.594, 3.002, 0.102, 4.497, 1.8, 9.499, 1.0, 3.901]
Незначущі коефіцієнти регресії []
```


Перевірка значущості коефіцієнтів за критерієм Стюдента

Значущі коефіцієнти регресії [0.003, 1.823, 6.638, 6.594, 3.002, 0.102, 4.497, 1.8, 9.499, 1.0, 3.901]

Незначущі коефіцієнти регресії []

Значення з отриманими коефіцієнтами

-204468.413

-113275.750

-6357.857

-4272.861

221016.681

123358.343

50126.904

25364.233

-140316.538

183297.712

-1851.455

14682.829

12650.430

45.930

3721.427

Перевірка адекватності за критерієм Фішера

$F_p = 0.6245029581352286$

При рівні значимості 0.05 рівняння регресії адекватне

Process finished with exit code 0