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

Методи наукових досліджень
Лабораторна робота № 6

«Проведення трьохфакторного експерименту при використанні рівняння регресії з квадратичними членами»

Виконав:
студент групи ІВ-93
Манчук М.В.

Київ
2021 р.

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

Завдання на лабораторну роботу

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;+ l; - l; 0 для $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$.
3. Значення функції відгуку знайти за допомогою підстановки в формулу:

$$y_i = f(x_1, x_2, x_3) + random(10) - 5,$$

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.

315	10	50	25	65	50	65	7,9+2,1*x1+5,3*x2+3,0*x3+8,1*x1*x1+1,0*x2*x2+8,4*x3*x3+7,2*x1*x2+0,8*x1*x3+2,3*x2*x3+6,4*x1*x2*x3

Код програми:

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

m = 3
n = 15

x1_min = 10
x1_max = 50
x2_min = 25
x2_max = 65
x3_min = 50
x3_max = 65

x01 = (x1_max + x1_min) / 2
x02 = (x2_max + x2_min) / 2
x03 = (x3_max + x3_min) / 2
delta_x1 = x1_max - x01
delta_x2 = x2_max - x02
delta_x3 = 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 * delta_x1 + x01,
      1.73 * delta_x1 + 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 *
delta_x2 + x02,
      1.73 * delta_x2 + x02, x02, x02, x02]
x3 = [x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x03, x03, x03, x03, -
1.73 * delta_x3 + x03,
      1.73 * 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] = 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))

for i in range(len(list_for_a)):
    list_for_a[i] = list(list_for_a[i])
    for j in range(len(list_for_a[i])):
        list_for_a[i][j] = round(list_for_a[i][j], 3)

planning_matrix_x = PrettyTable()
planning_matrix_x.field_names = ['X1', 'X2', 'X3', 'X1X2', 'X1X3', 'X2X3', 'X1X2X3', 'X1X1',
'X2X2', 'X3X3']
print("Матриця планування з натуралізованими коефіцієнтами X:")
planning_matrix_x.add_rows(list_for_a)
print(planning_matrix_x)


def function(X1, X2, X3):
    y = 7.9 + 2.1*X1 + 5.3*X2 + 3.0*X3 + 8.1*X1*X1 + 1.0*X2*X2 + 8.4*X3*X3 + 7.2*X1*X2 +
0.8*X1*X3 + 2.3*X2*X3 + \
        6.4*X1*X2*X3 + randrange(0, 10) - 5
    return y


Y = [[function(list_for_a[j][0], list_for_a[j][1], list_for_a[j][2]) for i in range(m)] for j
in range(15)]

planing_matrix_y = PrettyTable()
planing_matrix_y.field_names = ['Y1', 'Y2', 'Y3']
print("Матриця планування Y:")
planing_matrix_y.add_rows(Y)
print(planing_matrix_y)

Y_average = []
for i in range(len(Y)):
    Y_average.append(np.mean(Y[i], axis=0))
print("Середні значення відгуку за рядками:")
for i in range(15):
    print("{:.3f}".format(Y_average[i]), end=" ")

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_known(num):
    b = 0
    for kf in range(15):
        b += Y_average[kf] * list_for_a[kf][num - 1] / 15
    return b


def a(first, second):
    c = 0
    for kf in range(15):
```

```
        c += list_for_a[kf][first - 1] * list_for_a[kf][second - 1] / 15
    return c


my = sum(Y_average) / 15
mx = []
for i in range(10):
    number_lst = []
    for j in range(15):
        number_lst.append(list_for_a[j][i])
    mx.append(sum(number_lst) / len(number_lst))

det1 = [
    [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)]]

det2 = [my, find_known(1), find_known(2), find_known(3), find_known(4), find_known(5),
find_known(6), find_known(7),
        find_known(8), find_known(9), find_known(10)]

beta = solve(det1, det2)
print("\nОтримане рівняння регресії:")
print("{:.3f} + {:.3f} * X1 + {:.3f} * X2 + {:.3f} * X3 + {:.3f} * X1X2 + {:.3f} * X1X3 +
{:.3f} * X2X3"
      "+ {:.3f} * X1X2X3 + {:.3f} * X11^2 + {:.3f} * X22^2 + {:.3f} * X33^2 = ŷ"
      .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] * 15
print("Експериментальні значення:")
for k in range(15):
    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(15):
    print("{:.3f}".format(y_i[i]), end=" ")
print("\n-------------------Перевірка за критерієм Кохрена-------------------")
Gp = max(dispersions) / sum(dispersions)
Gt = 0.3346
print("Gp =", Gp)
if Gp < Gt:
    print("Дисперсія однорідна")
else:
    print("Дисперсія неоднорідна")

print("----------------Перевірка значущості "
      "коефіцієнтів за критерієм Стьюдента----------------")
sb = sum(dispersions) / len(dispersions)
```

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

F3 = (m - 1) * n
coefs1 = []
coefs2 = []
d = 11
res = [0] * 11
for j in range(11):
    t_pract = 0
    for i in range(15):
        if j == 0:
            t_pract += Y_average[i] / 15
        else:
            t_pract += Y_average[i] * xn[i][j - 1]
        res[j] = beta[j]
    if fabs(t_pract / sbs) < t.ppf(q=0.975, df=F3):
        coefs2.append(beta[j])
        res[j] = 0
        d -= 1
    else:
        coefs1.append(beta[j])
print("Значущі коефіцієнти регресії:", [round(i, 3) for i in coefs1])
print("Незначущі коефіцієнти регресії:", [round(i, 3) for i in coefs2])
y_st = []
for i in range(15):
    y_st.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("Значення з отриманими коефіцієнтами:")
for i in range(15):
    print("{:.3f}".format(y_st[i]), end=" ")

print("\n------------------------ Перевірка адекватності "
      "за критерієм Фішера ------------------------")
Sad = 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")
```

Результат виконання програми:

```
C:\Anaconda3\python.exe C:/PythonProjects/MND/Lab6.py
Матриця планування з натуралізованими коефіцієнтами X:
+------+------+--------+--------+---------+----------+----------+---------+---------+----------+
|  X1  |  X2  |   X3   |  X1X2  |   X1X3  |   X2X3   |  X1X2X3  |   X1X1  |   X2X2  |   X3X3   |
+------+------+--------+--------+---------+----------+----------+---------+---------+----------+
|  10  |  25  |   50   |  250   |   500   |   1250   |  12500   |   100   |   625   |   2500   |
|  10  |  25  |   65   |  250   |   650   |   1625   |  16250   |   100   |   625   |   4225   |
|  10  |  65  |   50   |  650   |   500   |   3250   |  32500   |   100   |   4225  |   2500   |
|  10  |  65  |   65   |  650   |   650   |   4225   |  42250   |   100   |   4225  |   4225   |
|  50  |  25  |   50   |  1250  |   2500  |   1250   |  62500   |   2500  |   625   |   2500   |
|  50  |  25  |   65   |  1250  |   3250  |   1625   |  81250   |   2500  |   625   |   4225   |
|  50  |  65  |   50   |  3250  |   2500  |   3250   |  162500  |   2500  |   4225  |   2500   |
|  50  |  65  |   65   |  3250  |   3250  |   4225   |  211250  |   2500  |   4225  |   4225   |
| -4.6 | 45.0 |  57.5  | -207.0 |  -264.5 |  2587.5  | -11902.5 |  21.16  |  2025.0 | 3306.25  |
| 64.6 | 45.0 |  57.5  | 2907.0 |  3714.5 |  2587.5  | 167152.5 | 4173.16 |  2025.0 | 3306.25  |
| 30.0 | 10.4 |  57.5  | 312.0  |  1725.0 |  598.0   |  17940.0 |  900.0  |  108.16 | 3306.25  |
| 30.0 | 79.6 |  57.5  | 2388.0 |  1725.0 |  4577.0  | 137310.0 |  900.0  | 6336.16 | 3306.25  |
| 30.0 | 45.0 | 44.525 | 1350.0 | 1335.75 | 2003.625 | 60108.75 |  900.0  |  2025.0 | 1982.476 |
| 30.0 | 45.0 | 70.475 | 1350.0 | 2114.25 | 3171.375 | 95141.25 |  900.0  |  2025.0 | 4966.726 |
| 30.0 | 45.0 |  57.5  | 1350.0 |  1725.0 |  2587.5  | 77625.0  |  900.0  |  2025.0 | 3306.25  |
+------+------+--------+--------+---------+----------+----------+---------+---------+----------+
Матриця планування Y:
+--------------------+--------------------+--------------------+
|         Y1         |         Y2         |         Y3         |
+--------------------+--------------------+--------------------+
|      107816.4      |      107816.4      |      107823.4      |
|      147334.9      |      147341.9      |      147338.9      |
|      247109.4      |      247113.4      |      247111.4      |
|      326413.9      |      326410.9      |      326410.9      |
|      456147.4      |      456148.4      |      456149.4      |
|      592142.9      |      592145.9      |      592143.9      |
|     1118955.4      |     1118958.4      |     1118955.4      |
|     1448330.9      |     1448336.9      |     1448334.9      |
|     -41552.614     |     -41547.614     |     -41553.614     |
|    1163786.906     |    1163785.906     |    1163778.906     |
|     155286.98      |     155285.98      |     155284.98      |
| 949951.6399999999  | 949951.6399999999  | 949944.6399999999  |
|    426505.70775    |    426506.70775    |    426500.70775    |
|    679164.88275    |    679169.88275    |    679167.88275    |
|     551417.65      |     551415.65      |     551424.65      |
+--------------------+--------------------+--------------------+
Середні значення відгуку за рядками:
107818.733 147338.567 247111.400 326411.900 456148.400 592144.233 1118956.400 1448334.233 -41551.281 1163783.906 155285.980 949949.307 426504.374 679167.549 551419.317
Отримане рівняння регресії:
41.150 + 2.718 * X1 + 5.254 * X2 + 1.513 * X3 + 7.194 * X1X2 + 0.792 * X1X3 + 2.301 * X2X3+ 6.400 * X1X2X3 + 8.100 * X11^2 + 1.001 * X22^2 + 8.415 * X33^2 = ŷ
Експериментальні значення:
107818.455 147338.108 247111.962 326412.281 456148.068 592143.721 1118956.908 1448334.561 -41551.375 1163783.935 155286.919 949948.302 426504.133 679167.725 551419.317
-------------------Перевірка за критерієм Кохрена-------------------
Gp = 0.16301703163017034
Дисперсія однорідна
----------------Перевірка значущості коефіцієнтів за критерієм Стьюдента----------------
Значущі коефіцієнти регресії: [41.15, 2.718, 5.254, 1.513, 7.194, 0.792, 2.301, 6.4, 8.1, 1.001, 8.415]
Незначущі коефіцієнти регресії: []
Значення з отриманими коефіцієнтами:
107818.455 147338.108 247111.962 326412.281 456148.068 592143.721 1118956.908 1448334.561 -41551.375 1163783.935 155286.919 949948.302 426504.130 679167.722 551419.317
------------------------- Перевірка адекватності за критерієм Фішера -------------------------
Fp = 0.42831319522106054
Рівняння регресії адекватне при рівні значимості 0.05

Process finished with exit code 0
```

Висновок:

В даній лабораторній роботі я провів трьохфакторний експеримент. Знайшов адекватну модель – рівняння регресії, використовуючи рототабельний композиційний план.