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

ВИКОНАВ:
студент ІІ курсу ФІОТ
групи ІВ-91
Мусійчук Я. С.
Залікова - 9121

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

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

Завдання:

1. Ознайомитися з теоретичними відомостями.
2. Вибрати з таблиці варіантів і записати в протокол інтервали значень x1, x2, x3. Обчислити і записати значення, відповідні кодованим значенням факторів +1; -1;+ ; - ; 0 для x1, x2, x3.
3. Значення функції відгуку знайти за допомогою підстановки в формулу:
yi = f(x1, x2, x3) + random(10)-5,
де f(x1, x2, x3) 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.

Варіант:

119	15	45	-35	15	-35	-5	6,4+6,1*x1+7,8*x2+1,5*x3+6,6*x1*x1+0,2*x2*x2+8,8*x3*x3+0,9*x1*x2+0,5*x1*x3+5,7*x2*x3+2,7*x1*x2*x3

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
from prettytable import PrettyTable

m = 3
n = 15

x1min = 15
x1max = 45
x2min = -35
x2max = 15
x3min = -35
x3max = -5

x01 = (x1max + x1min) / 2
x02 = (x2max + x2min) / 2
x03 = (x3max + x3min) / 2
deltax1 = x1max - x01
deltax2 = x2max - x02
deltax3 = x3max - 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 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max, -1.73 * deltax1 + x01,
1.73 * deltax1 + x01, x01, x01,
      x01, x01, x01]
x2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max, x02, x02, -1.73 *
deltax2 + x02, 1.73 * deltax2 + x02,
      x02, x02, x02]
x3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max, x03, x03, x03, x03, -
1.73 * deltax3 + x03,
      1.73 * deltax3 + 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 = 6.4 + 6.1 * X1 + 7.8 * X2 + 1.5 * X3 + 6.6 * X1 * X1 + 0.2 * X2 * X2 + 8.8 *
X3 * X3 + 0.9 * X1 * X2 + \
        0.5 * X1 * X3 + 5.7 * X2 * X3 + 2.7 * 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):
    a = 0
    for j in range(15):
        a += Y_average[j] * list_for_a[j][num - 1] / 15
    return a


def a(first, second):
    a = 0
    for j in range(15):
        a += list_for_a[j][first - 1] * list_for_a[j][second - 1] / 15
    return a
```

```
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 роботи:

```
Матриця планування з натуралізованими коефіцієнтами X:
+-------+--------+--------+---------+---------+--------+----------+----------+----------+----------+
|   X1  |   X2   |   X3   |   X1X2  |   X1X3  |  X2X3  |  X1X2X3  |   X1X1   |   X2X2   |   X3X3   |
+-------+--------+--------+---------+---------+--------+----------+----------+----------+----------+
|   15  |  -35   |  -35   |   -525  |   -525  |  1225  |  18375   |   225    |   1225   |   1225   |
|   15  |  -35   |   -5   |   -525  |   -75   |  175   |   2625   |   225    |   1225   |    25    |
|   15  |   15   |  -35   |   225   |   -525  |  -525  |  -7875   |   225    |   225    |   1225   |
|   15  |   15   |   -5   |   225   |   -75   |  -75   |  -1125   |   225    |   225    |    25    |
|   45  |  -35   |  -35   |  -1575  |  -1575  |  1225  |  55125   |   2025   |   1225   |   1225   |
|   45  |  -35   |   -5   |  -1575  |   -225  |  175   |   7875   |   2025   |   1225   |    25    |
|   45  |   15   |  -35   |   675   |  -1575  |  -525  |  -23625  |   2025   |   225    |   1225   |
|   45  |   15   |   -5   |   675   |   -225  |  -75   |  -3375   |   2025   |   225    |    25    |
|  4.05 | -10.0  | -20.0  |  -40.5  |  -81.0  | 200.0  |  810.0   |  16.403  |  100.0   |  400.0   |
| 55.95 | -10.0  | -20.0  |  -559.5 | -1119.0 | 200.0  | 11190.0  | 3130.403 |  100.0   |  400.0   |
|  30.0 | -53.25 | -20.0  | -1597.5 |  -600.0 | 1065.0 | 31950.0  |  900.0   | 2835.562 |  400.0   |
|  30.0 | 33.25  | -20.0  |  997.5  |  -600.0 | -665.0 | -19950.0 |  900.0   | 1105.562 |  400.0   |
|  30.0 | -10.0  | -45.95 |  -300.0 | -1378.5 | 459.5  | 13785.0  |  900.0   |  100.0   | 2111.403 |
|  30.0 | -10.0  |  5.95  |  -300.0 |  178.5  | -59.5  | -1785.0  |  900.0   |  100.0   |  35.402  |
|  30.0 | -10.0  | -20.0  |  -300.0 |  -600.0 | 200.0  |  6000.0  |  900.0   |  100.0   |  400.0   |
+-------+--------+--------+---------+---------+--------+----------+----------+----------+----------+
Матриця планування Y:
+---------------------+---------------------+---------------------+
|          Y1         |          Y2         |          Y3         |
+---------------------+---------------------+---------------------+
|       68142.4       |       68137.4       |       68137.4       |
|        9343.4       |        9345.4       |        9346.4       |
|       -11847.6      |       -11845.6      |       -11845.6      |
|       -1342.6       |       -1339.6       |       -1342.6       |
|  177959.40000000002 |  177958.40000000002 |  177956.40000000002 |
|       34556.4       |       34562.4       |       34562.4       |
| -42424.600000000006 | -42425.600000000006 | -42421.600000000006 |
|  4973.399999999998  |  4979.399999999998  |  4975.399999999998  |
|      6822.4115      |      6820.4115      |      6818.4115      |
|  54725.30150000001  |  54728.30150000001  |  54727.30150000001  |
|     100372.9125     |     100364.9125     |     100367.9125     |
|     -46954.8875     |     -46960.8875     |     -46958.8875     |
|  63460.217000000004 |  63463.217000000004 |  63464.217000000004 |
|  1052.4670000000006 |  1055.4670000000006 |  1048.4670000000006 |
|       26326.4       |       26327.4       |       26331.4       |
+---------------------+---------------------+---------------------+
Середні значення відгуку за рядками:
68139.067 9345.067 -11846.267 -1341.600 177958.067 34560.400 -42423.933 4976.067 6820.411 54726.968 100368.579 -46958.221 63462.550 1052.134 26328.400
Отримане рівняння регресії:
10.284 + 5.925 * X1 + 7.767 * X2 + 1.878 * X3 + 0.901 * X1X2 + 0.495 * X1X3 + 5.699 * X2X3+ 2.700 * X1X2X3 + 6.601 * X11^2 + 0.202 * X22^2 + 8.805 * X33^2 = ŷ
Експериментальні значення:
68140.223 9344.670 -11845.288 -1342.175 177958.396 34559.176 -42423.782 4974.664 6819.620 54728.088 100368.537 -46957.850 63460.918 1054.094 26328.398
------------------------------ Перевірка за критерієм Кохрена ------------------------------
Gp = 0.16554054054054054
Дисперсія однорідна
------------------ Перевірка значущості коефіцієнтів за критерієм Стьюдента ------------------
Значущі коефіцієнти регресії: [10.284, 5.925, 7.767, 1.878, 0.901, 0.495, 5.699, 2.7, 6.601, 0.202, 8.805]
Незначущі коефіцієнти регресії: []
Значення з отриманими коефіцієнтами:
68140.223 9344.670 -11845.288 -1342.175 177958.396 34559.176 -42423.782 4974.664 6819.617 54728.085 100368.537 -46957.850 63460.914 1054.099 26328.398
------------------------- Перевірка адекватності за критерієм Фішера -------------------------
Fp = 2.554713435252207
Рівняння регресії адекватне при рівні значимості 0.05
```