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»

ВИКОНАЛА:
студентка ІІ курсу ФІОТ
групи ІВ-91
Бузулук М.В.
Залікова - 9103

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

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

**Варіант завдання:**

| № варіанту | X1 | | X2 | | X3 | |
|---|---|---|---|---|---|---|
| | min | max | min | max | min | max |
| 103 | -20 | 30 | 30 | 80 | 30 | 45 |
| F(X1, X2, X3) | | | | | | |
| $0.7+5.4*X1+4.8*X2+5.3*X3+1.1*X1^2+0.3*X2^2+8.9*X3^2+8.1*X1*X2+0.2*X1*X3+3.5*X2*X3+1.9*X1*X2*X3$ | | | | | | |

**Лістинг програми:**

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

n = 15
m = 3

while True:
    x1_min = -20
    x1_max = 30
    x2_min = 30
    x2_max = 80
    x3_min = 30
    x3_max = 45
    x01 = (x1_max + x1_min) / 2
    x02 = (x2_max + x2_min) / 2
    x03 = (x3_max + x3_min) / 2

    dx1 = x1_max - x01
    dx2 = x2_max - x02
    dx3 = 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 * dx1 + x01, 1.73 * dx1 + x01, x01, x01, x01, x01, x01]
    x2 = [x2_min, x2_min, x2_max, x2_max, x2_min, x2_min, x2_max, x2_max, x02, x0
2, -1.73 * dx2 + x02, 1.73 * dx2 + x02, x02, x02, x02]
    x3 = [x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x3_min, x3_max, x03, x0
3, x03, x03, -1.73 * dx3 + x03, 1.73 * dx3 + 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)

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

    plan_table = PrettyTable()
    plan_table.field_names = ['X1', 'X2', 'X3', 'X1X2', 'X1X3', 'X2X3', 'X1X2X3',
 'X1^2', 'X2^2', 'X3^2']
    print("Матриця планування з натуралізованими X:")
    for i in range(len(tmp_list_a)):
        plan_table.add_row(tmp_list_a[i])
    print(plan_table)

    def f123(X1, X2, X3):
        return 0.7 + 5.4*X1 + 4.8*X2 + 5.3*X3 + 1.1*X1*X1 + 0.3*X2*X2 + 8.9*X3*X3
 + 8.1*X1*X2 + 0.2*X1*X3 + 3.5*X2*X3 + 1.9*X1*X2*X3 + random.randint(0, 10) - 5

    y = [[f123(tmp_list_a[j][0], tmp_list_a[j][1], tmp_list_a[j][2]) for _ in ran
ge(m)] for j in range(15)]

    plan_y = PrettyTable()
    plan_y.field_names = ['y1', 'y2', 'y3']
    print("Матриця планування значень Y:")
    for i in range(len(y)):
```

```
        plan_y.add_row(y[i])
    print(plan_y)

    aver_y = []
    for i in range(len(y)):
        aver_y.append(np.mean(y[i], axis=0))
    print("Середні значення Y:\n{}".format(np.array(list(map(lambda x:round(x, 5)
, aver_y)))))

    disp = []
    for i in range(len(y)):
        a = 0
        for k in y[i]:
            a += (k - np.mean(y[i], axis=0)) ** 2
        disp.append(a / len(y[i]))
    print("Дисперсія:\n{}".format(np.array(list(map(lambda x:round(x, 5), disp)))
))

    def finds_value(num):
        a = 0
        for j in range(15):
            a += aver_y[j] * tmp_list_a[j][num - 1] / 15
        return a

    def a(f, s):
        a = 0
        for j in range(15):
            a += tmp_list_a[j][f - 1] * tmp_list_a[j][s - 1] / 15
        return a

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

    determinant1 = [[1, mx[0], mx[1], mx[2], mx[3], mx[4], mx[5], mx[6], mx[7], m
x[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(1
0, 6), a(10, 7), a(10, 8), a(10, 9), a(10, 10)]]

    determinant2 = [my, finds_value(1), finds_value(2), finds_value(3), finds_val
ue(4), finds_value(5), finds_value(6), finds_value(7),
                    finds_value(8), finds_value(9), finds_value(10)]

    beta = list(map(lambda x:round(x, 5), solve(determinant1, determinant2)))
    print("\nРівняння регресії:")
    print("y = {} + {} * X1 + {} * X2 + {} * X3 + {} * X1X2 + \n  + {} * X1X3 + {
} * X2X3 + {} * X1X2X3 + {} * X11^2 + {} * X22^2 + {} * X33^2"
          .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


    for k in range(15):
        y_i[k] = beta[0] + beta[1] * tmp_list_a[k][0] + beta[2] * tmp_list_a[k][1
] + beta[3] * tmp_list_a[k][2] + \
                 beta[4] * tmp_list_a[k][3] + beta[5] * tmp_list_a[k][4] + beta[6
] * tmp_list_a[k][5] + beta[7] * \
                 tmp_list_a[k][6] + beta[8] * tmp_list_a[k][7] + beta[9] * tmp_li
st_a[k][8] + beta[10] * tmp_list_a[k][9]

    print("Експеременталні значення:\n{}".format(np.array(list(map(lambda x:roun
d(x, 5), y_i)))))

    gp = max(disp) / sum(disp)
    gt = 0.3346
    print("\nКритерій Кохрена\nGp = {}".format(gp))
    if gp < gt:
        print("Дисперсія однорідна")
    else:
        print("Дисперсія неоднорідна")
        m += 1
        continue

    sb = sum(disp) / len(disp)
    sbs = (sb / (15 * m)) ** 0.5

    f3 = (m - 1) * n
    sign_coef = []
    insign_coef = []
```

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

    for j in range(11):
        t_pract = 0
        for i in range(15):
            if j == 0:
                t_pract += aver_y[i] / 15
            else:
                t_pract += aver_y[i] * xn[i][j - 1]
            res[j] = beta[j]
        if math.fabs(t_pract / sbs) < t.ppf(q=0.975, df=f3):
            insign_coef.append(beta[j])
            res[j] = 0
            d-=1
        else:
            sign_coef.append(beta[j])
    print("\nКритерій Стьюдента:")
    print("Значимі коефіцієнти регресії:", [round(i, 3) for i in sign_coef])
    print("Незначимі коефіцієнти регресії:", [round(i, 3) for i in insign_coef])
    y_st = []
    for i in range(15):
        y_st.append(res[0] + res[1] * x1[i] + res[2] * x2[i] + res[3] * x3[i] + r
es[4] * x1x2[i] + res[5] *
                    x1x3[i] + res[6] * x2x3[i] + res[7] * x1x2x3[i] + res[8] * x1
kv[i] + res[9] *
                    x2kv[i] + res[10] * x3kv[i])
    print("Значення функції відгуку зі значущими коефіцієнтами:\n{}".format(np.ar
ray(list(map(lambda x:round(x, 5), y_st)))))

    print("\nКритерій Фішера")
    sad = m * sum([(y_st[i] - aver_y[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("Математична модель адекватна")
        break
    else:
        print("Математична модель неадекватна")
```

**Результат роботи програми:**

Матриця планування з натуралізованими X:

| X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 | X1^2 | X2^2 | X3^2 |
|---|---|---|---|---|---|---|---|---|---|
| -20 | 30 | 30 | -600 | -600 | 900 | -18000 | 400 | 900 | 900 |
| -20 | 30 | 45 | -600 | -900 | 1350 | -27000 | 400 | 900 | 2025 |
| -20 | 80 | 30 | -1600 | -600 | 2400 | -48000 | 400 | 6400 | 900 |
| -20 | 80 | 45 | -1600 | -900 | 3600 | -72000 | 400 | 6400 | 2025 |
| 30 | 30 | 30 | 900 | 900 | 900 | 27000 | 900 | 900 | 900 |
| 30 | 30 | 45 | 900 | 1350 | 1350 | 40500 | 900 | 900 | 2025 |
| 30 | 80 | 30 | 2400 | 900 | 2400 | 72000 | 900 | 6400 | 900 |
| 30 | 80 | 45 | 2400 | 1350 | 3600 | 108000 | 900 | 6400 | 2025 |
| -38.25 | 55.0 | 37.5 | -2103.75 | -1434.375 | 2062.5 | -78890.625 | 1463.062 | 3025.0 | 1406.25 |
| 48.25 | 55.0 | 37.5 | 2653.75 | 1809.375 | 2062.5 | 99515.625 | 2328.062 | 3025.0 | 1406.25 |
| 5.0 | 11.75 | 37.5 | 58.75 | 187.5 | 440.625 | 2203.125 | 25.0 | 138.062 | 1406.25 |
| 5.0 | 98.25 | 37.5 | 491.25 | 187.5 | 3684.375 | 18421.875 | 25.0 | 9653.062 | 1406.25 |
| 5.0 | 55.0 | 24.525 | 275.0 | 122.625 | 1348.875 | 6744.375 | 25.0 | 3025.0 | 601.476 |
| 5.0 | 55.0 | 50.475 | 275.0 | 252.375 | 2776.125 | 13880.625 | 25.0 | 3025.0 | 2547.726 |
| 5.0 | 55.0 | 37.5 | 275.0 | 187.5 | 2062.5 | 10312.5 | 25.0 | 3025.0 | 1406.25 |

Матриця планування значень Y:

| y1 | y2 | y3 |
|---|---|---|
| -27113.3 | -27118.3 | -27112.3 |
| -32610.3 | -32602.3 | -32603.3 |
| -85074.3 | -85070.3 | -85070.3 |
| -116445.3 | -116440.3 | -116447.3 |
| 71660.7 | 71650.7 | 71659.7 |
| 109058.7 | 109064.7 | 109067.7 |
| 176448.7 | 176443.7 | 176446.7 |
| 259229.7 | 259222.7 | 259226.7 |
| -144709.29375 | -144715.29375 | -144715.29375 |
| 234862.68125 | 234861.68125 | 234863.68125 |
| 19107.89375 | 19110.89375 | 19112.89375 |
| 68046.59375 | 68045.59375 | 68055.59375 |
| 26502.2155625 | 26493.2155625 | 26494.2155625 |
| 62537.5755625 | 62532.5755625 | 62531.5755625 |
| 43021.575 | 43023.575 | 43020.575 |

```
Середні значення Y:
[ -27114.63333  -32605.3      -85071.63333 -116444.3       71657.03333
  109063.7      176446.36667  259226.36667 -144713.29375  234862.68125
   19110.56042   68049.26042   26496.5489    62533.9089    43021.90833]
Дисперсія:
[ 6.88889 12.66667  3.55556  8.66667 20.22222 14.       4.22222  8.22222
  8.       0.66667  4.22222 20.22222 16.22222  6.88889  1.55556]

Рівняння регресії:
y = -43.37491 + 5.73953 * X1 + 5.15378 * X2 + 7.41837 * X3 + 8.09427 * X1X2 +
  + 0.19218 * X1X3 + 3.49351 * X2X3 + 1.90014 * X1X2X3 + 1.09788 * X11^2 + 0.29878 * X22^2 + 8.87548 * X33^2
Експерементальні значення:
[ -27115.24601  -32605.88996  -85072.47201 -116445.08346   71656.64549
  109063.28654  176445.59449  259225.51804 -144711.89191  234862.96891
   19111.05534   68050.48947   26497.41621   62534.74744   43021.88589]

Критерій Кохрена
Gp = 0.14845024469820556
Дисперсія однорідна

Критерій Стьюдента:
Значимі коефіцієнти регресії: [-43.375, 5.74, 5.154, 7.418, 8.094, 0.192, 3.494, 1.9, 1.098, 0.299, 8.875]
Незначимі коефіцієнти регресії: []
Значення функції відгуку зі значущими коефіцієнтами:
[ -27115.24601  -32605.88996  -85072.47201 -116445.08346   71656.64549
  109063.28654  176445.59449  259225.51804 -144711.89191  234862.96891
   19111.05534   68050.48947   26497.41621   62534.74744   43021.88589]

Критерій Фішера
Fp = 0.738127169233098
   19111.05534   68050.48947   26497.41621   62534.74744   43021.88589]

 Критерій Кохрена
 Gp = 0.14845024469820556
 Дисперсія однорідна

 Критерій Стьюдента:
 Значимі коефіцієнти регресії: [-43.375, 5.74, 5.154, 7.418, 8.094, 0.192, 3.494, 1.9, 1.098, 0.299, 8.875]
 Незначимі коефіцієнти регресії: []
 Значення функції відгуку зі значущими коефіцієнтами:
 [ -27115.24601  -32605.88996  -85072.47201 -116445.08346   71656.64549
   109063.28654  176445.59449  259225.51804 -144711.89191  234862.96891
    19111.05534   68050.48947   26497.41621   62534.74744   43021.88589]

 Критерій Фішера
 Fp = 0.738127169233098
 Математична модель адекватна
```