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

Методи оптимізації та планування експерименту
Лабораторна робота №3
«Проведення трьохфакторного експерименту з використанням лінійного рівняння регресії»

ВИКОНАВ:
студент ІІ курсу ФІОТ
групи ІО-91
Сниченков Д. А.
варіант – 21

ПЕРЕВІРИВ:
Регіда П. Г.

Київ – 2021

Текст програми

```
from random import randint
import numpy as np


def find_a(a, b=None):
    denominator = len(a)
    if b is None:
        nominator = sum(a[i] ** 2 for i in range(len(a)))
    else:
        nominator = sum(a[i] * b[i] for i in range(len(a)))

    return nominator / denominator


def cramer(arr, ins, pos):
    matrix = np.insert(np.delete(arr, pos, 1), pos, ins, 1)
    nominator = np.linalg.det(matrix)
    denominator = np.linalg.det(arr)

    return nominator / denominator


def get_dispersion(y, y_r):
    return [round(sum([(y[i][j] - y_r[i]) ** 2 for j in range(len(y[i]))]) / 3, 3) for i in
range(len(y_r))]


def cochran(dispersion, m):
    g_p = max(dispersion) / sum(dispersion)
    g_t = [.9065, .7679, .6841, .6287, .5892, .5598, .5365, .5175, .5017, .4884]

    if g_p < g_t[m - 2]:
        return [round(g_p, 4), g_t[m - 2]]
    else:
        return


def student(dispersion, m, y_r, x_n):
    table = {
        8: 2.306,
        12: 2.179,
        16: 2.120,
        20: 2.086,
        24: 2.064,
        28: 2.048,
        'inf': 1.960
    }

    x_nt = x_n.T
    n = len(y_r)

    s_b = sum(dispersion) / len(y_r)
    s_beta = (s_b / (m * n)) ** (1 / 2)

    beta = [sum([y_r[j] * x_nt[i][j] for j in range(n)]) / n for i in range(n)]
    t = [abs(beta[i]) / s_beta for i in range(len(beta))]

    f3 = n * (m - 1)

    if f3 > 30:
        t_t = table['inf']
    elif f3 > 0:
        t_t = table[f3]
    else:
        return

    result = []
    for i in t:
        if i < t_t:
            result.append(False)
        else:
            result.append(True)

    return result
```

```
def fisher(y_r, y_st, b_det, dispersion, m):
    table = {
        8: [5.3, 4.5, 4.1, 3.8, 3.7, 3.6],
        12: [4.8, 3.9, 3.5, 3.3, 3.1, 3.0],
        16: [4.5, 3.6, 3.2, 3.0, 2.9, 2.7],
        20: [4.5, 3.5, 3.1, 2.9, 2.7, 2.6],
        24: [4.3, 3.4, 3.0, 2.8, 2.6, 2.5]
    }

    n = len(y_r)
    s_b = sum(dispersion) / n
    d = 0
    for b in b_det:
        if b:
            d += 1

    f4 = n - d
    f3 = n * (m - 1)
    s_ad = (m / f4) * sum([(y_st[i] - y_r[i]) ** 2 for i in range(n)])
    f_ap = s_ad / s_b
    f_t = table[f3][f4 - 1]

    if f_ap < f_t:
        return f"\nРівняння регресії адекватно оригіналу:\nFap < Ft: {round(f_ap, 2)} < {f_t}"
    else:
        return f"\nРівняння регресії неадекватно оригіналу: \nFap > Ft: {round(f_ap, 2)} > {f_t}"


def experiment(m, x1_min, x1_max, x2_min, x2_max, x3_min, x3_max):
    y_min = round((x1_min + x2_min + x3_min) / 3) + 200
    y_max = round((x1_max + x2_max + x3_max) / 3) + 200

    x_norm = np.array([
        [1, -1, -1, -1],
        [1, -1, 1, 1],
        [1, 1, -1, 1],
        [1, 1, 1, -1]
    ])

    x = np.array([
        [x1_min, x2_min, x3_min],
        [x1_min, x2_max, x3_max],
        [x1_max, x2_min, x3_max],
        [x1_max, x2_max, x3_min]
    ])

    y = [[randint(y_min, y_max) for _ in range(m)] for _ in range(len(x))]
    y_r = [round(sum(y[i]) / len(y[i]), 2) for i in range(len(y))]
    n = len(y_r)

    dispersion = get_dispersion(y, y_r)
    cochran_cr = cochran(dispersion, m)

    if cochran_cr is None:
        raise Exception("Need more experiments")
    else:
        pass

    m_x = [sum(i) / len(i) for i in x.T]
    m_y = sum(y_r) / n

    x_t = x.T

    a_1 = find_a(x_t[0], y_r)
    a_2 = find_a(x_t[1], y_r)
    a_3 = find_a(x_t[2], y_r)

    a_11 = find_a(x_t[0])
    a_22 = find_a(x_t[1])
    a_33 = find_a(x_t[2])

    a_12 = a21 = find_a(x_t[0], x_t[1])
    a_13 = a31 = find_a(x_t[0], x_t[2])
    a_23 = a32 = find_a(x_t[1], x_t[2])

    b_delta = np.array([
```

```
        [1, m_x[0], m_x[1], m_x[2]],
        [m_x[0], a_11, a_12, a_13],
        [m_x[1], a21, a_22, a_23],
        [m_x[2], a31, a32, a_33]
    ])

    b_set = np.array([m_y, a_1, a_2, a_3])
    b = [cramer(b_delta, b_set, i) for i in range(n)]

    b_det = student(dispersion, m, y_r, x_norm)
    b_cut = b.copy()

    if b_det is None:
        raise Exception("Need more experiments")
    else:
        for i in range(n):
            if not b_det[i]:
                b_cut[i] = 0

        y_st = [round(b_cut[0] + x[i][0] * b_cut[1] + x[i][1] * b_cut[2] + x[i][2] * b_cut[3], 2) for i in
range(n)]

    fisher_cr = fisher(y_r, y_st, b_det, dispersion, m)

    print(f"\nМатриця планування для m = {m}:")
    for i in range(m):
        print(f"Y{i + 1} - {np.array(y).T[i]}")

    print(f"\nСередні значення функції відгуку за рядками:\nY_R: {y_r}")
    print(f"\nКоефіцієнти рівняння регресії:")
    for i in range(len(b)):
        print(f"b{i} = {round(b[i], 3)}")

    print("\nДисперсії по рядках:\nS²{y} = ", dispersion, sep="")
    print(f"\nЗа критерієм Кохрена дисперсія однорідна:\nGp < Gt - {cochran_cr[0]} < {cochran_cr[1]}")

    print(f"\nЗа критерієм Стьюдента коефіцієнти ", end="")
    for i in range(len(b_det)):
        if not b_det[i]:
            print(f"b{i} ", end="")
    print("приймаємо незначними")

    print(f"\nОтримані функції відгуку зі спрощеними коефіцієнтами:\nY_St - {y_st}")
    print(fisher_cr)


def start_experiment(x1_min, x1_max, x2_min, x2_max, x3_min, x3_max):
    global m
    try:
        experiment(m, x1_min, x1_max, x2_min, x2_max, x3_min, x3_max)
    except:
        print("Збільшуємо кількість експериментів")
        m += 1
        start_experiment(x1_min, x1_max, x2_min, x2_max, x3_min, x3_max)


if __name__ == "__main__":
    x1_min, x1_max = 10, 40
    x2_min, x2_max = -30, 45
    x3_min, x3_max = -30, -10

    m = 3
    start_experiment(x1_min, x1_max, x2_min, x2_max, x3_min, x3_max)
```

Результат виконання програми

```
Матриця планування для m = 3:
Y1 - [221 189 189 206]
Y2 - [215 220 195 210]
Y3 - [205 191 189 194]

Середні значення функції відгуку за рядками:
Y_R: [213.67, 200.0, 191.0, 203.33]

Коефіцієнти рівняння регресії:
b0 = 197.125
b1 = -0.322
b2 = -0.009
b3 = -0.65

Дисперсії по рядках:
S²{y} = [43.556, 200.667, 8.0, 46.222]

За критерієм Кохрена дисперсія однорідна:
Gp < Gt - 0.6724 < 0.7679

За критерієм Стьюдента коефіцієнти b1 b2 приймаємо незначними

Отримані функції відгуку зі спрощеними коефіцієнтами:
Y_St - [216.63, 203.63, 203.63, 216.63]

Рівняння регресії неадекватно оригіналу:
Fap > Ft: 7.2 > 4.5
>
```