

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

Лабораторна робота №6
з дисципліни «Методи оптимізації та планування
експерименту»
на тему: «Проведення трьохфакторного експерименту при
використанні рівняння регресії з квадратичними членами»

Виконав:
студент групи ІО-92
Уткін Владислав
Залікова книжка № ІО-9227
Номер у списку групи - 19

Перевірів:
ас. Регіда П.Г.

Київ - 2021

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

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

Завдання:

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; +1; -1; 0 для $\bar{x}_1, \bar{x}_2, \bar{x}_3$.
3. Значення функції відгуку знайти за допомогою підстановки в формулу:

$$y_i = f(x_1, x_2, x_3) + \text{random}(10) - 5,$$

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.

Порядок виконання роботи:

- 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;

Варіант:

219	15	45	-70	-10	15	30	$9,1+3,9*x_1+5,3*x_2+4,6*x_3+4,8*x_1*x_1+0,7*x_2*x_2+3,6*x_3*x_3+7,0*x_1*x_2+1,0*x_1*x_3+5,7*x_2*x_3+2,5*x_1*x_2*x_3$
-----	----	----	-----	-----	----	----	---

Роздруківка тексту програми:

```
import math
import random
from _decimal import Decimal
from scipy.stats import f, t
import numpy
from itertools import compress
from functools import reduce

xmin = [15, -70, 15]
xmax = [45, -10, 30]
norm_plan_raw = [[-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],
                  [+1.73, 0, 0],
                  [0, -1.73, 0],
                  [0, +1.73, 0],
                  [0, 0, -1.73],
                  [0, 0, +1.73]]

x0 = [(xmax[_] + xmin[_])/2 for _ in range(3)]
dx = [xmax[_] - x0[_] for _ in range(3)]

natur_plan_raw = [[xmin[0],      xmin[1],      xmin[2]],
                  [xmin[0],      xmin[1],      xmax[2]],
                  [xmin[0],      xmax[1],      xmin[2]],
                  [xmin[0],      xmax[1],      xmax[2]],
                  [xmax[0],      xmin[1],      xmin[2]],
                  [xmax[0],      xmin[1],      xmax[2]],
                  [xmax[0],      xmax[1],      xmin[2]],
                  [xmax[0],      xmax[1],      xmax[2]],
                  [-1.73*dx[0]+x0[0], x0[1],      x0[2]],
                  [1.73*dx[0]+x0[0], x0[1],      x0[2]],
                  [x0[0],      -1.73*dx[1]+x0[1], x0[2]],
                  [x0[0],      1.73*dx[1]+x0[1], x0[2]],
                  [x0[0],      x0[1],      -1.73*dx[2]+x0[2]],
                  [x0[0],      x0[1],      1.73*dx[2]+x0[2]],
                  [x0[0],      x0[1],      x0[2]]]

def equation_of_regression(x1, x2, x3, cef, importance=[] * 11):
    factors_array = [1, x1, x2, x3, x1 * x2, x1 * x3, x2 * x3, x1 * x2 * x3, x1 ** 2, x2 ** 2, x3 ** 2]
    return sum([el[0] * el[1] for el in compress(zip(cef, factors_array), importance)])

def func(x1, x2, x3):
    coeffs = [9.1, 3.9, 5.3, 4.6, 4.8, 0.7, 3.6, 7.0, 1.0, 5.7, 2.5]
    return equation_of_regression(x1, x2, x3, coeffs)

def generate_factors_table(raw_array):
    raw_list = [row + [row[0] * row[1], row[0] * row[2], row[1] * row[2], row[0] * row[1] * row[2]] + list(
        map(lambda x: x ** 2, row)) for row in raw_array]
    return list(map(lambda row: list(map(lambda el: round(el, 3), row)), raw_list))

def generate_y(m, factors_table):
```

```

return [[round(func(row[0], row[1], row[2]) + random.randint(-5, 5), 3) for _ in range(m)] for row in factors_table]

def print_matrix(m, N, factors, y_vals, additional_text=""):
    labels_table = list(map(lambda x: x.ljust(10),
        ["x1", "x2", "x3", "x12", "x13", "x23", "x123", "x1^2", "x2^2", "x3^2"] + [
            "y{}".format(i + 1) for i in range(m)]))
    rows_table = [list(factors[i]) + list(y_vals[i]) for i in range(N)]
    print("\nМатриця планування" + additional_text)
    print(" ".join(labels_table))
    print("\n".join([" ".join(map(lambda j: "{:<+10}".format(j), rows_table[i])) for i in range(len(rows_table))]))
    print("\t")

def print_equation(coeffs, importance=[True] * 11):
    x_i_names = list(compress(["", "x1", "x2", "x3", "x12", "x13", "x23", "x123", "x1^2", "x2^2", "x3^2"], importance))
    coefficients_to_print = list(compress(coeffs, importance))
    equation = " ".join(
        ["".join(i for i in zip(list(map(lambda x: "{:+.2f}".format(x), coefficients_to_print)), x_i_names))]
    )
    print("Рівняння регресії: y = " + equation)

def set_factors_table(factors_table):
    def x_i(i):
        with_null_factor = list(map(lambda x: [1] + x, generate_factors_table(factors_table)))
        res = [row[i] for row in with_null_factor]
        return numpy.array(res)

    return x_i

def m_ij(*arrays):
    return numpy.average(reduce(lambda accum, el: accum * el, list(map(lambda el: numpy.array(el), arrays))))

def find_coefficients(factors, y_vals):
    x_i = set_factors_table(factors)
    coeffs = [[m_ij(x_i(column), x_i(row)) for column in range(11)] for row in range(11)]
    y_numpy = list(map(lambda row: numpy.average(row), y_vals))
    free_values = [m_ij(y_numpy, x_i(i)) for i in range(11)]
    beta_coefficients = numpy.linalg.solve(coeffs, free_values)
    return list(beta_coefficients)

def cochrans_criteria(m, N, y_table):
    def get_cochran_value(f1, f2, q):
        partResult1 = q / f2
        params = [partResult1, f1, (f2 - 1) * f1]
        fisher = f.isf(*params)
        result = fisher / (fisher + (f2 - 1))
        return Decimal(result).quantize(Decimal('.0001')).__float__()

    print("Перевірка за критерієм Кохрена: m = {}, N = {}".format(m, N))
    y_variations = [numpy.var(i) for i in y_table]
    max_y_variation = max(y_variations)
    gp = max_y_variation / sum(y_variations)
    f1 = m - 1
    f2 = N
    p = 0.95
    q = 1 - p
    gt = get_cochran_value(f1, f2, q)
    print("Gp = {}; Gt = {}; f1 = {}; f2 = {}; q = {:.2f}".format(gp, gt, f1, f2, q))
    if gp < gt:
        print("Gp < Gt => дисперсії рівномірні => все правильно")
        return True
    else:
        print("Gp > Gt => дисперсії нерівномірні => змінюємо значення m")

```

```
return False
```

```
def student_criteria(m, N, y_table, beta_coefficients):
```

```
    def get_student_value(f3, q):
```

```
        return Decimal(abs(t.ppf(q / 2, f3))).quantize(Decimal('.0001')).__float__()
```

```
    print("\nПеревірка за критерієм Стьюдента: m = {}, N = {}".format(m, N))
```

```
    average_variation = numpy.average(list(map(numpy.var, y_table)))
```

```
    variation_beta_s = average_variation / N / m
```

```
    standard_deviation_beta_s = math.sqrt(variation_beta_s)
```

```
    t_i = [abs(beta_coefficients[i]) / standard_deviation_beta_s for i in range(len(beta_coefficients))]
```

```
    f3 = (m - 1) * N
```

```
    q = 0.05
```

```
    t_our = get_student_value(f3, q)
```

```
    importance = [True if el > t_our else False for el in list(t_i)]
```

```
    # print result data
```

```
    print("Оцінки коефіцієнтів  $\beta$ s: " + ", ".join(list(map(lambda x: str(round(float(x), 3)), beta_coefficients))))
```

```
    print("Коефіцієнти ts: " + ", ".join(list(map(lambda i: "{:.2f}".format(i), t_i))))
```

```
    print("f3 = {}; q = {}; табл = {}".format(f3, q, t_our))
```

```
    beta_i = [" $\beta$ 0", " $\beta$ 1", " $\beta$ 2", " $\beta$ 3", " $\beta$ 12", " $\beta$ 13", " $\beta$ 23", " $\beta$ 123", " $\beta$ 11", " $\beta$ 22", " $\beta$ 33"]
```

```
    importance_to_print = ["важливий" if i else "неважливий" for i in importance]
```

```
    to_print = map(lambda x: x[0] + " " + x[1], zip(beta_i, importance_to_print))
```

```
    print(*to_print, sep="; ")
```

```
    print_equation(beta_coefficients, importance)
```

```
    return importance
```

```
def fisher_criteria(m, N, d, x_table, y_table, b_coefficients, importance):
```

```
    def get_fisher_value(f3, f4, q):
```

```
        return Decimal(abs(f.isf(q, f4, f3))).quantize(Decimal('.0001')).__float__()
```

```
    f3 = (m - 1) * N
```

```
    f4 = N - d
```

```
    q = 0.05
```

```
    theoretical_y = numpy.array([equation_of_regression(row[0], row[1], row[2], b_coefficients) for row in x_table])
```

```
    average_y = numpy.array(list(map(lambda el: numpy.average(el), y_table)))
```

```
    s_ad = m / (N - d) * sum((theoretical_y - average_y) ** 2)
```

```
    y_variations = numpy.array(list(map(numpy.var, y_table)))
```

```
    s_v = numpy.average(y_variations)
```

```
    f_p = float(s_ad / s_v)
```

```
    f_t = get_fisher_value(f3, f4, q)
```

```
    theoretical_values_to_print = list(
```

```
        zip(map(lambda x: "x1 = {0[1]:<10} x2 = {0[2]:<10} x3 = {0[3]:<10}".format(x), x_table), theoretical_y))
```

```
    print("\nПеревірка за критерієм Фішера: m = {}, N = {} для таблиці y_table".format(m, N))
```

```
    print("Теоретичні значення Y для різних комбінацій факторів:")
```

```
    print("\n".join(["{arr[0]}: y = {arr[1]}".format(arr=el) for el in theoretical_values_to_print]))
```

```
    print("Fp = {}, Ft = {}".format(f_p, f_t))
```

```
    print("Fp < Ft => модель адекватна" if f_p < f_t else "Fp > Ft => модель неадекватна")
```

```
    return True if f_p < f_t else False
```

```
m = 3
```

```
N = 15
```

```
natural_plan = generate_factors_table(natur_plan_raw)
```

```
y_arr = generate_y(m, natur_plan_raw)
```

```
while not cochrans_criteria(m, N, y_arr):
```

```
    m += 1
```

```
    y_arr = generate_y(m, natural_plan)
```

```
print_matrix(m, N, natural_plan, y_arr, " для натуралізованих факторів:")
```

```
coefficients = find_coefficients(natural_plan, y_arr)
```

```
print_equation(coefficients)
```

```
importance = student_criteria(m, N, y_arr, coefficients)
```

```
d = len(list(filter(None, importance)))
```

```
fisher_criteria(m, N, d, natural_plan, y_arr, coefficients, importance)
```

Роздруківка результатів виконання програми:

```
C:\Anaconda3\envs\labs\python.exe C:/Users/Влад/PycharmProjects/labs/Lab6.py
Перевірка за критерієм Кохрена: m = 3, N = 15
Gr = 0.1965442764578834; Gt = 0.3346; f1 = 2; f2 = 15; q = 0.05
Gr < Gt => дисперсії рівномірні => все правильно

Матриця планування для натуралізованих факторів:
x1      x2      x3      x12      x13      x23      x123      x1^2      x2^2      x3^2      y1      y2      y3
+15      -70      +15      -1050     +225     -1050     -15750     +225     +4900     +225     -5      -1      +4
+15      -70      +30      -1050     +450     -2100     -31500     +225     +4900     +900     -4      -3      -2
+15      -10      +15      -150      +225     -150     -2250     +225     +100      +225     -5      +4      +5
+15      -10      +30      -150      +450     -300     -4500     +225     +100      +900     +0      +4      +2
+45      -70      +15      -3150     +675     -1050     -47250     +2025     +4900     +225     -1      -5      +3
+45      -70      +30      -3150     +1350     -2100     -94500     +2025     +4900     +900     -3      -5      -2
+45      -10      +15      -450      +675     -150     -6750     +2025     +100      +225     +5      +2      +1
+45      -10      +30      -450      +1350     -300     -13500     +2025     +100      +900     -3      +3      +0
+4.05     -40.0      +22.5     -162.0     +91.125   -900.0     -3645.0     +16.403   +1600.0     +506.25   +0      +5      -5
+55.95     -40.0      +22.5     -2238.0     +1258.875 -900.0     -50355.0     +3130.403 +1600.0     +506.25   -2      -4      +2
+30.0      -91.9      +22.5     -2757.0     +675.0     -2067.75   -62032.5     +900.0     +8445.61   +506.25   -1      -1      -2
+30.0      +11.9      +22.5     +357.0      +675.0     +267.75    +8032.5     +900.0     +141.61    +506.25   +5      +4      +1
+30.0      -40.0      +9.525     -1200.0     +285.75    -381.0     -11430.0     +900.0     +1600.0     +90.726   -4      +2      +3
+30.0      -40.0      +35.475    -1200.0     +1064.25   -1419.0     -42570.0     +900.0     +1600.0     +1258.476 -2      +5      +1
+30.0      -40.0      +22.5     -1200.0     +675.0     -900.0     -27000.0     +900.0     +1600.0     +506.25   -3      -3      -5

Рівняння регресії: y = +11.31 -0.06x1 +0.06x2 -0.83x3 +0.00x12 -0.01x13 +0.01x23 -0.00x123 +0.00x1^2 +0.00x2^2 +0.02x3^2

Перевірка за критерієм Стюдента: m = 3, N = 15
Оцінки коефіцієнтів ps: 11.314, -0.057, 0.063, -0.831, 0.003, -0.009, 0.005, -0.0, 0.004, 0.002, 0.024
Коефіцієнти ts: 28.98, 0.15, 0.16, 2.13, 0.01, 0.02, 0.01, 0.00, 0.01, 0.00, 0.06
f3 = 30; q = 0.05; tтабл = 2.0423
p0 важливий; p1 неважливий; p2 неважливий; p3 важливий; p12 неважливий; p13 неважливий; p23 неважливий; p123 неважливий; p11 неважливий; p22 неважливий; p33 неважливий
Рівняння регресії: y = +11.31 -0.83x3
```

Перевірка за критерієм Фішера: m = 3, N = 15 для таблиці y_table

Теоретичні значення Y для різних комбінацій факторів:

x1 = -70	x2 = 15	x3 = -1050	: y = 0
x1 = -70	x2 = 30	x3 = -1050	: y = 0
x1 = -10	x2 = 15	x3 = -150	: y = 0
x1 = -10	x2 = 30	x3 = -150	: y = 0
x1 = -70	x2 = 15	x3 = -3150	: y = 0
x1 = -70	x2 = 30	x3 = -3150	: y = 0
x1 = -10	x2 = 15	x3 = -450	: y = 0
x1 = -10	x2 = 30	x3 = -450	: y = 0
x1 = -40.0	x2 = 22.5	x3 = -162.0	: y = 0
x1 = -40.0	x2 = 22.5	x3 = -2238.0	: y = 0
x1 = -91.9	x2 = 22.5	x3 = -2757.0	: y = 0
x1 = 11.9	x2 = 22.5	x3 = 357.0	: y = 0
x1 = -40.0	x2 = 9.525	x3 = -1200.0	: y = 0
x1 = -40.0	x2 = 35.475	x3 = -1200.0	: y = 0
x1 = -40.0	x2 = 22.5	x3 = -1200.0	: y = 0

Fp = 2.1681342415683673, Ft = 2.063

Fp > Ft => модель неадекватна

Process finished with exit code 0

|

Висновки:

У ході виконання лабораторної роботи я провів повний трьохфакторний експеримент при використанні рівняння регресії з квадратичними членами. Закріпив отримані знання практичним їх використанням при написанні програми, що реалізує завдання лабораторної роботи. Мета лабораторної роботи досягнута.