



Міністерство освіти та науки України

Національний технічний університет України «Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №6

з дисципліни «Методи оптимізації та планування»

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

Виконав:
студент 2-го курсу ФІОТ
групи ІО-92
Костюк А.В.
Перевірив:
асистент
Регіда П. Г.

Варіант:

Варіант №212

212	-40	20	5	40	-40	-20	$5,4+2,4*x_1+7,3*x_2+9,6*x_3+2,5*x_1*x_1+0,2*x_2*x_2+8,2*x_3*x_3+1,7*x_1*x_2+0,7*x_1*x_3+0,6*x_2*x_3+9,3*x_1*x_2*x_3$
-----	-----	----	---	----	-----	-----	---

Код 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
import timeit

xmin = [-40, 5, -40]
xmax = [20, 40, -20]
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 = [5.4, 2.4, 7.3, 9.6, 2.5, 0.2, 8.2, 1.7, 0.7, 0.6, 9.3]
```

```

    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 = {}; tтабл = {}".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(natural_plan_raw)
y_arr = generate_y(m, natural_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)
print("\nВиконав: студент групи ІО-92 Костюк Антон    Варіант 212")

```

Результат виконання програми:

Перевірка за критерієм Кохрена: m = 3, N = 15
 $G_p = 0.1914893617021277$; $G_t = 0.3346$; $f_1 = 2$; $f_2 = 15$; $q = 0.05$
 $G_p < G_t \Rightarrow$ дисперсії рівномірні \Rightarrow все правильно

Матриця планування для натуралізованих факторів:

x1	x2	x3	x12	x13	x23	x123	x1^2	x2^2	x3^2	y1	y2	y3
-40	+5	-40	-200	+1600	-200	+8000	+1600	+25	+1600	+2	+3	+3
-40	+5	-20	-200	+800	-100	+4000	+1600	+25	+400	+3	-2	-4
-40	+40	-40	-1600	+1600	-1600	+64000	+1600	+1600	+1600	+4	+4	-5
-40	+40	-20	-1600	+800	-800	+32000	+1600	+1600	+400	-4	-1	+1
+20	+5	-40	+100	-800	-200	-4000	+400	+25	+1600	-2	-3	-3
+20	+5	-20	+100	-400	-100	-2000	+400	+25	+400	+1	+3	+2
+20	+40	-40	+800	-800	-1600	-32000	+400	+1600	+1600	+1	+2	+1
+20	+40	-20	+800	-400	-800	-16000	+400	+1600	+400	-1	+4	-5
-61.9	+22.5	-30.0	-1392.75	+1857.0	-675.0	+41782.5	+3831.61	+506.25	+900.0	+0	-3	+4
+41.9	+22.5	-30.0	+942.75	-1257.0	-675.0	-28282.5	+1755.61	+506.25	+900.0	+5	+0	+5
-10.0	-7.775	-30.0	+77.75	+300.0	+233.25	-2332.5	+100.0	+60.451	+900.0	+1	-5	-3
-10.0	+52.775	-30.0	-527.75	+300.0	-1583.25	+15832.5	+100.0	+2785.201	+900.0	+2	-3	+5
-10.0	+22.5	-47.3	-225.0	+473.0	-1064.25	+10642.5	+100.0	+506.25	+2237.29	-4	-2	-1
-10.0	+22.5	-12.7	-225.0	+127.0	-285.75	+2857.5	+100.0	+506.25	+161.29	+3	+5	-4
-10.0	+22.5	-30.0	-225.0	+300.0	-675.0	+6750.0	+100.0	+506.25	+900.0	+3	+5	+5

Рівняння регресії: $y = -8.20 + 0.21x_1 + 0.10x_2 - 0.79x_3 - 0.00x_{12} + 0.01x_{13} - 0.01x_{23} - 0.00x_{123} - 0.00x_1^2 - 0.01x_2^2 - 0.02x_3^2$

Перевірка за критерієм Стюдента: $m = 3$, $N = 15$

Оцінки коефіцієнтів β s: -8.196, 0.209, 0.1, -0.795, -0.005, 0.008, -0.006, -0.0, -0.001, -0.005, -0.016

Коефіцієнти ts: 21.96, 0.56, 0.27, 2.13, 0.01, 0.02, 0.02, 0.00, 0.00, 0.01, 0.04

$f_3 = 30$; $q = 0.05$; $t_{табл} = 2.0423$

β_0 важливий; β_1 неважливий; β_2 неважливий; β_3 важливий; β_{12} неважливий; β_{13} неважливий; β_{23} неважливий; β_{123} неважливий; β_{11} неважливий; β_{22} неважливий; β_{33} неважливий

Рівняння регресії: $y = -8.20 - 0.79x_3$

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

Теоретичні значення Y для різних комбінацій факторів:

$x_1 = 5$	$x_2 = -40$	$x_3 = -200$: $y = 0$
$x_1 = 5$	$x_2 = -20$	$x_3 = -200$: $y = 0$
$x_1 = 40$	$x_2 = -40$	$x_3 = -1600$: $y = 0$
$x_1 = 40$	$x_2 = -20$	$x_3 = -1600$: $y = 0$
$x_1 = 5$	$x_2 = -40$	$x_3 = 100$: $y = 0$
$x_1 = 5$	$x_2 = -20$	$x_3 = 100$: $y = 0$
$x_1 = 40$	$x_2 = -40$	$x_3 = 800$: $y = 0$
$x_1 = 40$	$x_2 = -20$	$x_3 = 800$: $y = 0$
$x_1 = 22.5$	$x_2 = -30.0$	$x_3 = -1392.75$: $y = 0$
$x_1 = 22.5$	$x_2 = -30.0$	$x_3 = 942.75$: $y = 0$
$x_1 = -7.775$	$x_2 = -30.0$	$x_3 = 77.75$: $y = 0$
$x_1 = 52.775$	$x_2 = -30.0$	$x_3 = -527.75$: $y = 0$
$x_1 = 22.5$	$x_2 = -47.3$	$x_3 = -225.0$: $y = 0$
$x_1 = 22.5$	$x_2 = -12.7$	$x_3 = -225.0$: $y = 0$
$x_1 = 22.5$	$x_2 = -30.0$	$x_3 = -225.0$: $y = 0$

$F_p = 2.528641571194763$, $F_t = 2.063$

$F_p > F_t \Rightarrow$ модель неадекватна

Виконав: студент групи ІО-92 Костюк Антон Варіант 212

Висновки:

Під час виконання лабораторної роботи було змодельовано трьохфакторний експеримент при використанні лінійного рівняння регресії, рівняння регресії з ефектом взаємодії та рівняння регресії з квадратичними членами, складено матрицю планування експерименту, було визначено коефіцієнти рівнянь регресії (натуралізовані та нормовані), для форми з квадратичними членами - натуралізовані, виконано перевірку правильності розрахунку коефіцієнтів рівнянь регресії. Також було проведено 3 статистичні перевірки(використання критеріїв Кохрена, Стюдента та Фішера) для кожної форми рівняння регресії. При виявленні неадекватності лінійного рівняння регресії оригіналу було застосовано ефект взаємодії факторів, при неадекватності і такого рівняння регресії було застосовано рівняння регресії з квадратичними членами.