Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота 3

з дисципліни «Методи оптимізації та планування експерименту»

Виконав:

Студент 2 курсу ФІОТ

групи ІВ-93

Трибушенко Артем

Перевірив:

Регіда П.Г.

Київ 2021

# Мета: Провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

Варіант: 327



**Роздруківка програми:**

import numpy as np

import pandas as pd

x1\_min = 10

x1\_max = 60

x2\_min = -35

x2\_max = 10

x3\_min = -30

x3\_max = 45

x\_avg\_max = np.mean([x1\_max, x2\_max, x3\_max])

x\_avg\_min = np.mean([x1\_min, x2\_min, x3\_min])

y\_max = 200 + x\_avg\_max

y\_min = 200 + x\_avg\_min

factors\_table = [[1, 1, 1, 1],

[-1, -1, +1, +1],

[-1, +1, -1, +1],

[-1, +1, +1, -1]]

n = 4

m = 3

x1 = np.array([x1\_min, x1\_min, x1\_max, x1\_max])

x2 = np.array([x2\_min, x2\_max, x2\_min, x2\_max])

x3 = np.array([x3\_min, x3\_max, x3\_max, x3\_min])

y1 = np.array([np.random.randint(y\_min, y\_max) for i in range(n)])

y2 = np.array([np.random.randint(y\_min, y\_max) for i in range(n)])

y3 = np.array([np.random.randint(y\_min, y\_max) for i in range(n)])

df\_plan = pd.DataFrame({"x1" : x1, "x2" : x2, "x3" : x3, "y1" : y1, "y2" : y2, "y3" : y3})

ys = ["y1", "y2", "y3"]

y1\_mean\_row, y2\_mean\_row, y3\_mean\_row, y4\_mean\_row = np.mean(df\_plan[ys], axis=1)

mean\_ys\_row = [y1\_mean\_row, y2\_mean\_row, y3\_mean\_row, y4\_mean\_row]

print(f"Середнє по у - {mean\_ys\_row}")

xs = ["x1", "x2", "x3"]

x1\_mean\_col, x2\_mean\_col, x3\_mean\_col = np.mean(df\_plan[xs], axis=0)

print(f"Середнє по X = {x1\_mean\_col}\n, {x2\_mean\_col}\n, {x3\_mean\_col}")

mean\_y = np.mean([y1\_mean\_row, y2\_mean\_row, y3\_mean\_row, y4\_mean\_row])

print(f"Середнє по середнім у = {mean\_y}")

# Підрахунок a1, a2, a3

a1 = sum([df\_plan["x1"][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))]) / len(mean\_ys\_row)

a2 = sum([df\_plan["x2"][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))]) / len(mean\_ys\_row)

a3 = sum([df\_plan["x3"][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))]) / len(mean\_ys\_row)

print("a1 = ", a1, "\na2 = ", round(a2, 3), "\na3 = ", a3)

# Підрахунок a11, a22, a33, a12, a13, a23

a11 = sum(np.square(df\_plan["x1"])) / len(mean\_ys\_row)

a22 = sum(np.square(df\_plan["x2"])) / len(mean\_ys\_row)

a33 = sum(np.square(df\_plan["x3"])) / len(mean\_ys\_row)

print("a11 = ", a11, "\na22 = ", round(a22, 3), "\na33 = ", a33)

a12 = sum([df\_plan["x1"][i] \* df\_plan["x2"][i] for i in range(len(df\_plan["x1"]))]) / len(mean\_ys\_row)

a13 = sum([df\_plan["x1"][i] \* df\_plan["x3"][i] for i in range(len(df\_plan["x1"]))]) / len(mean\_ys\_row)

a23 = sum([df\_plan["x2"][i] \* df\_plan["x3"][i] for i in range(len(df\_plan["x1"]))]) / len(mean\_ys\_row)

a21 = a12

a31 = a13

a32 = a23

print("a21 = a12 =", a21, "\na31 = a13 =", a31, "\na32 = a23 =", a32)

# Коефіцієнти регресії

from numpy.linalg import det

b0 = det(np.array([[mean\_y, x1\_mean\_col, x2\_mean\_col, x3\_mean\_col],

[a1, a11, a12, a13],

[a2, a12, a22, a32],

[a3, a13, a23, a33]])) / det(np.array([[1, x1\_mean\_col, x2\_mean\_col, x3\_mean\_col],

[x1\_mean\_col, a11, a12, a13],

[x2\_mean\_col, a12, a22, a32],

[x3\_mean\_col, a13, a23, a33]]))

b1 = det(np.array([[1, mean\_y, x2\_mean\_col, x3\_mean\_col],

[x1\_mean\_col, a1, a12, a13],

[x2\_mean\_col, a2, a22, a32],

[x3\_mean\_col, a3, a23, a33]])) / det(np.array([[1, x1\_mean\_col, x2\_mean\_col, x3\_mean\_col],

[x1\_mean\_col, a11, a12, a13],

[x2\_mean\_col, a12, a22, a32],

[x3\_mean\_col, a13, a23, a33]]))

b2 = det(np.array([[1, x1\_mean\_col, mean\_y, x3\_mean\_col],

[x1\_mean\_col, a11, a1, a13],

[x2\_mean\_col, a12, a2, a32],

[x3\_mean\_col, a13, a3, a33]])) / det(np.array([[1, x1\_mean\_col, x2\_mean\_col, x3\_mean\_col],

[x1\_mean\_col, a11, a12, a13],

[x2\_mean\_col, a12, a22, a32],

[x3\_mean\_col, a13, a23, a33]]))

b3 = det(np.array([[1, x1\_mean\_col, x2\_mean\_col, mean\_y],

[x1\_mean\_col, a11, a12, a1],

[x2\_mean\_col, a12, a22, a2],

[x3\_mean\_col, a13, a23, a3]])) / det(np.array([[1, x1\_mean\_col, x2\_mean\_col, x3\_mean\_col],

[x1\_mean\_col, a11, a12, a13],

[x2\_mean\_col, a12, a22, a32],

[x3\_mean\_col, a13, a23, a33]]))

print("b0 =",b0, "\nb1 =", round(b1, 2), "\nb2 =", b2, "\nb3 =", b3)

# Підрахунок результатів регресії

regression = df\_plan.loc[:, ["x1", "x2", "x3"]] \* [b1, b2, b3]

y\_not\_finished = regression.apply(sum, axis=1)

mean\_y\_finish = (b0 + y\_not\_finished).tolist()

# Порівняння кінечних результатів регресії з даними на початку

mean = [y1\_mean, y2\_mean, y3\_mean, y4\_mean]

for i in range(len(mean\_y\_finish)):

print(f"given mean of y = {round(mean[i], 5)} <--------> result of regression = {round(mean\_y\_finish[i], 5)}")

# ↑ Результати сходяться. Чудово! ↑

normalized\_data = pd.DataFrame(

{"x0\_norm" : factors\_table[0], "x1\_norm" : factors\_table[1],

"x2\_norm" : factors\_table[2], "x3\_norm" : factors\_table[3],

"y1" : y1, "y2" : y2, "y3" : y3})

print(normalized\_data)

# Знайдемо дисперсії по рядках

df\_ys = normalized\_data[ys]

disps = []

for i in range(4):

disps.append((1/ 3) \* (df\_ys.iloc[i, 0] - mean\_ys\_row[i]) \*\* 2 + (df\_ys.iloc[i, 1] - mean\_ys\_row[i]) \*\* 2 +

(df\_ys.iloc[i, 2] - mean\_ys\_row[i]) \*\* 2)

# За таблицею в 4 рядку 2 стовпчику Gt = 0.7679

Gt = 0.7679

Gp = max(disps) / sum(disps)

print(f"Gp = {Gp} < Gт = {Gt} – Дисперсія однорідна.")

Sb = sum(disps) / n

Sbetas = Sb / (n \* m)

SbetasSquare = Sbetas \*\* (1 / 2)

beta0 = (1 / n) \* np.sum([factors\_table[0][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))])

beta1 = (1 / n) \* np.sum([factors\_table[1][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))])

beta2 = (1 / n) \* np.sum([factors\_table[2][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))])

beta3 = (1 / n) \* np.sum([factors\_table[3][i] \* mean\_ys\_row[i] for i in range(len(mean\_ys\_row))])

print(f"beta0 = {beta0};\nbeta1 = {beta1};\nbeta2 = {round(beta2, 3)};\nbeta3 = {round(beta3, 3)}")

t0 = np.abs(beta0) / SbetasSquare

t1 = np.abs(beta1) / SbetasSquare

t2 = np.abs(beta2) / SbetasSquare

t3 = np.abs(beta3) / SbetasSquare

t\_arr = [t0, t1, t2, t3]

print(f"t0 = {round(t0, 2)};\nt1 = {t1};\nt2 = {round(t2, 2)};\nt3 = {round(t3, 3)}")

# f3 = f1 \* f2

indexes = []

for i, v in enumerate(t\_arr):

if t\_arr[i] > 2.306:

indexes.append(i)

else:

print(f'Коефіцієнт b{i} = {v} приймаємо не значним')

b\_list = [b0, b1, b2, b3]

print(f'y = b{indexes[0]}')

b\_res = [b\_list[indexes[0]] for \_ in range(4)]

for i in b\_res:

print(f'y = {i}')

#Критерій Фішера--------кількість значимих коефіцієнтів--------

d = 1 # кількість значимих коефіцієнтів

s2\_ad = m \* sum([(mean\_ys\_row[i] - b\_res[i]) \*\* 2 for i in range(4)]) / (n - d)

fp = s2\_ad / Sbetas

print(f'Fp = {fp}')

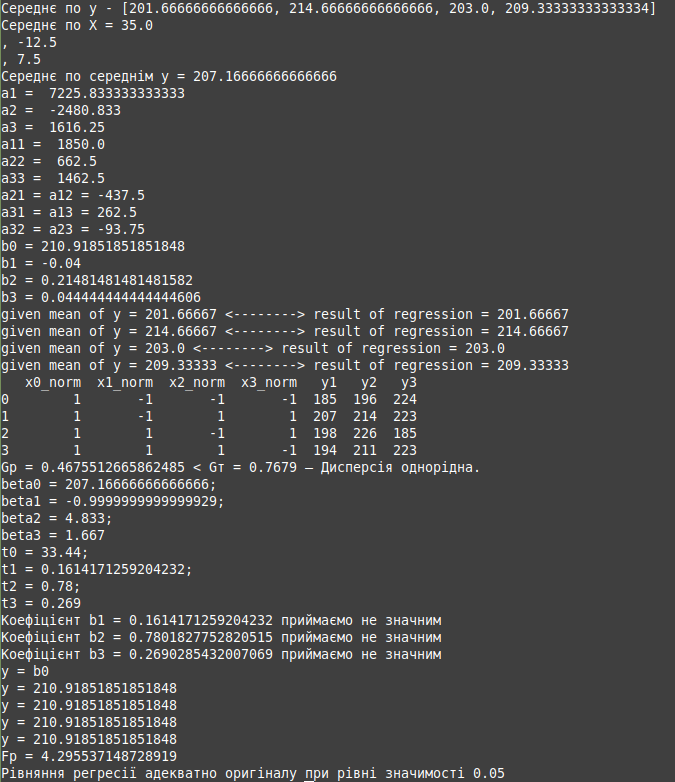
if fp > 4.5:

print('Рівняння регресії неадекватно оригіналу при рівні значимості 0.05')

else:

print('Рівняння регресії адекватно оригіналу при рівні значимості 0.05')

**Результати роботи програми:**



**Контрольні запитання**

1. Що називається дробовим факторним експериментом?

Дробовим факторним експериментом називається експеримент з використанням частини повного факторного експерименту

2. Для чого потрібно розрахункове значення Кохрена?

Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.

3. Для чого перевіряється критерій Стьюдента?

За допомогою критерію Стьюдента перевіряється значущість коефіцієнтів рівняння

4. Чим визначається критерій Фішера і як його застосовувати?

Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваного об”єкта.