Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 2

з дисципліни «Методи наукових досліджень»

на тему «ПРОВЕДЕННЯ ДВОФАКТОРНОГО ЕКСПЕРИМЕТНУ З

ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАВ:

студент 2 курсу

групи ІВ-91

Бойко М. І.

Залікова – 9102

ПЕРЕВІРИВ:

ас. Регіда П. Г.

Київ – 2021

**Лабораторна робота №2**

**Мета**: провести двофакторний експеримент, перевірити однорідність дисперсії за критерієм Романовського, отримати коефіцієнти рівняння регресії, провести натуралізацію рівняння регресії.

**Завдання:**

1. Записати лінійне рівняння регресії.

2. Обрати тип двофакторного експерименту і скласти матрицю

планування для нього з використанням додаткового нульового фактору

(хо=1).

3. Провести експеримент в усіх точках повного факторного простору

(знайти значення функції відгуку y). Значення функції відгуку задати

випадковим чином у відповідності до варіанту у діапазоні 𝑦𝑚𝑖𝑛 ÷ 𝑦𝑚𝑎𝑥

𝑦𝑚𝑎𝑥 = (30 − 𝑁варіанту) ∙ 10,

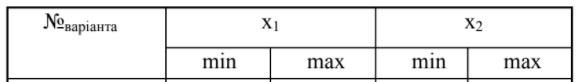
𝑦𝑚𝑖𝑛 = (20 − 𝑁варіанту) ∙ 10.

Варіанти обираються по номеру в списку в журналі викладача

**Група:** ІВ-91

**Номер у списку:** 2

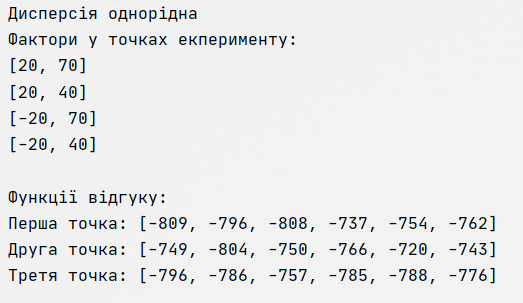
**Варіант – 102**

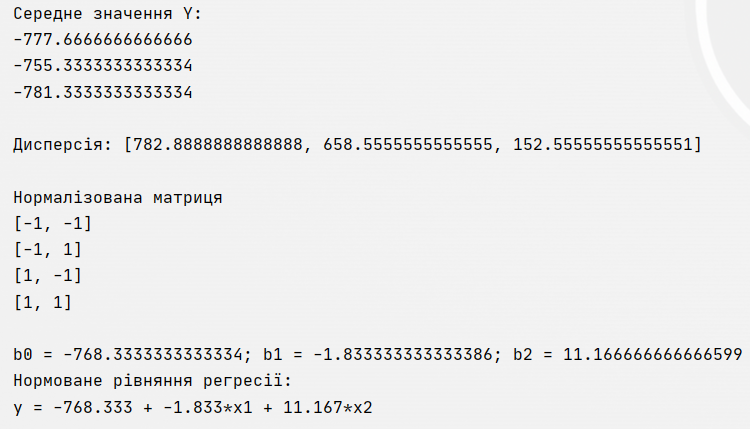
 

**Роздруківка коду програми:**

import numpy as np  
from random import \*  
from math import \*  
  
*# Умова за варіантом:*y\_max = (30 - 102) \* 10  
y\_min = (20 - 102) \* 10  
x1 = [20, 70]  
x2 = [-20, 40]  
m = 6  
  
*# Матриця ПФЕ:*pfeMatrix = [[x1[0], x1[1]],  
 [x1[0], x2[1]],  
 [x2[0], x1[1]],  
 [x2[0], x2[1]]]  
  
*# Критерій Романовського:*romanovskiy\_table = {6: 2.16, 7: 2.3, 8: 2.43,  
 9: 2.5, 10: 2.62, 11: 2.7,  
 12: 2.75, 13: 2.8, 14: 2.85,  
 15: 2.9, 16: 2.94, 17: 2.97,  
 18: 3, 19: 3.05, 20: 3.08}  
  
def findMean(list):  
 mean = 0  
 for i in range(len(list)):  
 mean += list[i]  
 mean /= len(list)  
 return mean  
  
def romanovskiy(disp):  
 sigma = sqrt(2 / m \* (2 \* m - 2) / (m - 4))  
 f\_uv = [(disp[0] / disp[1]),  
 (disp[2] / disp[0]),  
 (disp[2] / disp[1])]  
 theta\_uv = [((m - 2 / m) \* f\_uv[0]),  
 ((m - 2 / m) \* f\_uv[1]),  
 ((m - 2 / m) \* f\_uv[2])]  
 r\_uv = [(abs(theta\_uv[0] - 1) / sigma),  
 (abs(theta\_uv[1] - 1) / sigma),  
 (abs(theta\_uv[2] - 1) / sigma)]  
 for r in r\_uv:  
 if (r < romanovskiy\_table[m]):  
 return True  
 return False  
  
def dispersion(list):  
 mean\_y = findMean(list)  
 disp = 0  
 for i in range(len(list)):  
 disp += (list[i] - mean\_y) \*\* 2  
 disp /= len(list)  
 return disp  
  
def uniformity(m):  
 y = [[randint(0, 100) + y\_min for \_ in range(m)] for i in range(3)]  
  
 disp = [dispersion(y[0]), dispersion(y[1]), dispersion(y[2])]  
  
 if not romanovskiy(disp):  
 if m != 20:  
 m += 1  
 uniformity(m)  
 exit()  
 else:  
 print(  
 **"Дисперсія неодноріда після 20 дослідів**\n**"  
 "Перейти до обчислення коефіцієнтів рівняння регресії неможливо"**)  
 exit()  
  
 print(**"Дисперсія однорідна**\n**"  
 "Фактори у точках екперименту:"**)  
 for x in pfeMatrix:  
 print(x)  
 print(**f"**\n**Функції відгуку:**\n**Перша точка:** {y[0]}\n**Друга точка:** {y[1]}\n**Третя точка:** {y[2]}**"**)  
 print(**f"**\n**Середне значення Y:**\n{findMean(y[0])}\n{findMean(y[1])}\n{findMean(y[2])}**"**)  
 print(**f"**\n**Дисперсія:** {disp}**"**)  
  
 normMatrix = [[-1, -1], [-1, 1], [1, -1], [1, 1]]  
  
 mx1 = (normMatrix[0][0] + normMatrix[1][0] + normMatrix[2][0]) / 3  
 mx2 = (normMatrix[0][1] + normMatrix[1][1] + normMatrix[2][1]) / 3  
 my = (findMean(y[0]) + findMean(y[1]) + findMean(y[2])) / 3  
  
 a1 = (normMatrix[0][0] \*\* 2 + normMatrix[1][0] \*\* 2 + normMatrix[2][0] \*\* 2) / 3  
 a2 = (normMatrix[0][0] \* normMatrix[0][1] + normMatrix[1][0] \* normMatrix[1][1] +  
 normMatrix[2][0] \* normMatrix[2][1]) / 3  
 a3 = (normMatrix[0][1] \*\* 2 + normMatrix[1][1] \*\* 2 + normMatrix[2][1] \*\* 2) / 3  
 a11 = (normMatrix[0][0] \* findMean(y[0]) + normMatrix[1][0] \* findMean(y[1]) + normMatrix[2][  
 0] \* findMean(y[2])) / 3  
 a22 = (normMatrix[0][1] \* findMean(y[0]) + normMatrix[1][1] \* findMean(y[1]) + normMatrix[2][  
 1] \* findMean(y[2])) / 3  
  
 b0\_numerator = np.array([[my, mx1, mx2],  
 [a11, a1, a2],  
 [a22, a2, a3]])  
 b0\_denominator = np.array([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
 b0 = np.linalg.det(b0\_numerator) / np.linalg.det(b0\_denominator)  
 b1\_numerator = np.array([[1, my, mx2],  
 [mx1, a11, a2],  
 [mx2, a22, a3]])  
 b1\_denominator = np.array([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
 b1 = np.linalg.det(b1\_numerator) / np.linalg.det(b1\_denominator)  
 b2\_numerator = np.array([[1, mx1, my],  
 [mx1, a1, a11],  
 [mx2, a2, a22]])  
 b2\_denominator = np.array([[1, mx1, mx2],  
 [mx1, a1, a2],  
 [mx2, a2, a3]])  
 b2 = np.linalg.det(b2\_numerator) / np.linalg.det(b2\_denominator)  
  
 delta\_x1 = abs(x2[0] - x1[0]) / 2  
 delta\_x2 = abs(x2[1] - x1[1]) / 2  
  
 x10 = (x2[0] + x1[0]) / 2  
 x20 = (x2[1] + x1[1]) / 2  
  
 a0 = b0 - b1 \* (x10 / delta\_x1) - b2 \* (x20 / delta\_x2)  
 a1 = b1 / delta\_x1  
 a2 = b2 / delta\_x2  
  
 print(**"**\n**Нормалізована матриця"**)  
 for x in normMatrix:  
 print(x)  
 print(**f"**\n**b0 =** {b0}**; b1 =** {b1}**; b2 =** {b2}**"**)  
 print(**f"Нормоване рівняння регресії:**\n**y =** {round(b0, 3)} **+** {round(b1, 3)}**\*x1 +** {round(b2, 3)}**\*x2**\n**"**)  
  
 print(**"Середні значення нормалізованих Y:"**)  
 for i in range(3):  
 print(**f"**{b0 + b1 \* normMatrix[i][0] + b2 \* normMatrix[i][1]}**"**)  
 print(**"Значення збігаються зі значеннями Yj"**)  
 print(**f"**\n**a0 =** {a0}**; a1 =** {a1}**; a2 =** {a2}**"**)  
 print(**f"Натуралізоване рівняння регресії:**\n**y =** {round(a0, 3)} **+** {round(a1, 3)}**\*x1 +** {round(a2, 3)}**\*x2"**)  
  
 print(**"**\n**Середні значення натуралізованих Y:"**)  
 for i in range(3):  
 print(**f"**{a0 + a1 \* pfeMatrix[i][0] + a2 \* pfeMatrix[i][1]}**"**)  
  
 print(**"Значення збігаються зі значеннями Yj**\n**Коефіцієнти рівняння регресії розраховані вірно"**)  
  
uniformity(m)

**Результат роботи програми:**

****

****

