Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 3**

з дисципліни «Методи оптимізації та планування експерименту» на тему

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ

ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

ВИКОНАВ:

студент ІІ курсу ФІОТ

групи ІВ-81

Ковальков Дмитро

ПЕРЕВІРИВ:

Регіда П. Г.

Київ – 2020

*Варіант 115*

|  |  |  |
| --- | --- | --- |
|  | *Min* | *Max* |
| *x1* | 10 | 50 |
| *x2* | -20 | 60 |
| *x3* | -20 | 20 |

**Код програми**

import random  
import numpy.linalg as l  
import math  
  
Kohren = (  
 (9985, 9750, 9392, 9057, 8772, 8534, 8332, 8159, 8010, 7880),  
 (9669, 8709, 7977, 7457, 7071, 6771, 6530, 6333, 6167, 6025),  
 (9065, 7679, 6841, 6287, 5892, 5598, 5365, 5175, 5017, 4884),  
 (8412, 6838, 5981, 5440, 5063, 4783, 4564, 4387, 4241, 4118),  
 (7808, 6161, 5321, 4803, 4447, 4184, 3980, 3817, 3682, 3568),  
 (7271, 5612, 4800, 4307, 3974, 3726, 3535, 3384, 3259, 3154),  
 (6798, 5157, 4377, 3910, 3595, 3362, 3185, 3043, 2926, 2829),  
 (6385, 4775, 4027, 3584, 3286, 3067, 2901, 2768, 2659, 2568),  
 (6020, 4450, 3733, 3311, 3029, 2823, 2666, 2541, 2439, 2353))  
  
Student = [12.71, 4.303, 3.182, 2.776, 2.571, 2.447, 2.365, 2.306, 2.262, 2.228, 2.201, 2.179, 2.16, 2.145, 2.131, 2.12] #Для довірчої ймовірності 95%  
  
Fisher = (  
 (164.4, 199.5, 215.7, 224.6, 230.2, 234),  
 (18.5, 19.2, 19.2, 19.3, 19.3, 19.3),  
 (10.1, 9.6, 9.3, 9.1, 9, 8.9),  
 (7.7, 6.9, 6.6, 6.4, 6.3, 6.2),  
 (6.6, 5.8, 5.4, 5.2, 5.1, 5),  
 (6, 5.1, 4.8, 4.5, 4.4, 4.3),  
 (5.5, 4.7, 4.4, 4.1, 4, 3.9),  
 (5.3, 4.5, 4.1, 3.8, 3.7, 3.6),  
 (5.1, 4.3, 3.9, 3.6, 3.5, 3.4),  
 (5, 4.1, 3.7, 3.5, 3.3, 3.2),  
 (4.8, 4, 3.6, 3.4, 3.2, 3.1),  
 (4.8, 3.9, 3.5, 3.3, 3.1, 3))  
  
x1 = [10, 50]  
x2 = [-20, 60]  
x3 = [-20, 20]  
  
m = 3  
N = 4  
q = 0.05  
  
x\_matr = [[min(x1), min(x1), max(x1), max(x1)],  
 [min(x2), max(x2), min(x2), max(x2)],  
 [min(x3), max(x3), max(x3), min(x3)]]  
  
x\_avg\_max = (max(x1) + max(x2) + max(x3)) / 3  
x\_avg\_min = (min(x1) + min(x2) + min(x3)) / 3  
  
y\_max = 200 + x\_avg\_max  
y\_min = 200 + x\_avg\_min  
  
while True:  
 y = [[round(random.uniform(y\_min, y\_max), 4) for i in range(m)] for j in range(4)]  
  
 print("y1, y2, y3, ..., ym:")  
 for i in y: print(i)  
 print("\n")  
 for i in range(len(x\_matr)): print(f"{x\_matr[i]} - x{i+1}")  
  
 avg\_arr = lambda arr: sum(arr) / len(arr)  
  
 y\_avg\_arr = list(map(avg\_arr, y))  
 print(f"\nсереднє y(i): {y\_avg\_arr}")  
  
 mx\_arr = list(map(avg\_arr, x\_matr))  
 print(f"\nmx(i): {mx\_arr}")  
  
 my = avg\_arr(y\_avg\_arr)  
 print(f"my = {my}")  
  
 a = lambda x: (x[0] \* y\_avg\_arr[0] + x[1] \* y\_avg\_arr[1] + x[2] \* y\_avg\_arr[2] + x[3] \* y\_avg\_arr[3]) / 4  
  
 a1 = a(x\_matr[0])  
 a2 = a(x\_matr[1])  
 a3 = a(x\_matr[2])  
  
 a11 = (x\_matr[0][0] \* x\_matr[0][0] + x\_matr[0][1] \* x\_matr[0][1] + x\_matr[0][2] \* x\_matr[0][2] + x\_matr[0][3] \*  
 x\_matr[0][3]) / 4  
 a22 = (x\_matr[1][0] \* x\_matr[1][0] + x\_matr[1][1] \* x\_matr[1][1] + x\_matr[1][2] \* x\_matr[1][2] + x\_matr[1][3] \*  
 x\_matr[1][3]) / 4  
 a33 = (x\_matr[2][0] \* x\_matr[2][0] + x\_matr[2][1] \* x\_matr[2][1] + x\_matr[2][2] \* x\_matr[2][2] + x\_matr[2][3] \*  
 x\_matr[2][3]) / 4  
 a12 = a21 = (x\_matr[0][0] \* x\_matr[1][0] + x\_matr[0][1] \* x\_matr[1][1] + x\_matr[0][2] \* x\_matr[1][2] + x\_matr[0][3] \*  
 x\_matr[1][3]) / 4  
 a13 = a31 = (x\_matr[0][0] \* x\_matr[2][0] + x\_matr[0][1] \* x\_matr[2][1] + x\_matr[0][2] \* x\_matr[2][2] + x\_matr[0][3] \*  
 x\_matr[2][3]) / 4  
 a23 = a32 = (x\_matr[1][0] \* x\_matr[2][0] + x\_matr[1][1] \* x\_matr[2][1] + x\_matr[1][2] \* x\_matr[2][2] + x\_matr[1][3] \*  
 x\_matr[2][3]) / 4  
  
 print(f"""\na1 = {a1}, a2 = {a2}, a3= {a3},  
 a11 = {a11}, a12 = {a12}, a13 = {a13},  
 a21 = {a21}, a22 = {a22}, a23= {a23},  
 a31 = {a31}, a32 = {a32}, a33= {a33}\n""")  
  
 det = lambda sq\_matr: round(l.det(sq\_matr), 4)  
  
 mm = [[1, mx\_arr[0], mx\_arr[1], mx\_arr[2]],  
 [mx\_arr[0], a11, a12, a13],  
 [mx\_arr[1], a12, a22, a32],  
 [mx\_arr[2], a13, a23, a33]]  
  
 m0 = [[my, mx\_arr[0], mx\_arr[1], mx\_arr[2]],  
 [a1, a11, a12, a13],  
 [a2, a12, a22, a32],  
 [a3, a13, a23, a33]]  
  
 m1 = [[1, my, mx\_arr[1], mx\_arr[2]],  
 [mx\_arr[0], a1, a12, a13],  
 [mx\_arr[1], a2, a22, a32],  
 [mx\_arr[2], a3, a23, a33]]  
  
 m2 = [[1, mx\_arr[0], my, mx\_arr[2]],  
 [mx\_arr[0], a11, a1, a13],  
 [mx\_arr[1], a12, a2, a32],  
 [mx\_arr[2], a13, a3, a33]]  
  
 m3 = [[1, mx\_arr[0], mx\_arr[1], my],  
 [mx\_arr[0], a11, a12, a1],  
 [mx\_arr[1], a12, a22, a2],  
 [mx\_arr[2], a13, a23, a3]]  
  
 b = [det(m0) / det(mm), det(m1) / det(mm), det(m2) / det(mm), det(m3) / det(mm)]  
  
 print(f"\nb(i): {b}")  
  
 y1\_avg = b[0] + b[1] \* min(x1) + b[2] \* min(x2) + b[3] \* min(x3)  
 y2\_avg = b[0] + b[1] \* min(x1) + b[2] \* max(x2) + b[3] \* max(x3)  
 y3\_avg = b[0] + b[1] \* max(x1) + b[2] \* min(x2) + b[3] \* max(x3)  
 y4\_avg = b[0] + b[1] \* max(x1) + b[2] \* max(x2) + b[3] \* min(x3)  
  
 D1 = (pow((y[0][0] - y1\_avg), 2) + pow((y[0][1] - y1\_avg), 2) + pow((y[0][2] - y1\_avg), 2)) / 3  
 D2 = (pow((y[1][0] - y2\_avg), 2) + pow((y[1][1] - y2\_avg), 2) + pow((y[1][2] - y2\_avg), 2)) / 3  
 D3 = (pow((y[2][0] - y3\_avg), 2) + pow((y[2][1] - y3\_avg), 2) + pow((y[2][2] - y3\_avg), 2)) / 3  
 D4 = (pow((y[3][0] - y4\_avg), 2) + pow((y[3][1] - y4\_avg), 2) + pow((y[3][2] - y4\_avg), 2)) / 3  
 D = [D1, D2, D3, D4]  
 print(f"\nD(i): {D}")  
 f1 = m - 1  
 f2 = N  
 print(f"f1 = m - 1 = {f1}")  
 print(f"f2 = N = {f2}")  
 Gp = max(D) / sum(D)  
 Gt = Kohren[f2-2][f1-1]\*0.0001  
 print(".................................................................................................................")  
 print("\nОднорідність дисперсії (критерій Кохрена): ")  
 print(f"Gp = {Gp}")  
 print(f"Gt = {Gt}")  
 if Gp < Gt:  
 print("Дисперсія однорідна, ослільки Gp < Gt")  
 break  
 else:  
 print("Дисперсія неоднорідна, оскільки Gp > Gt; збільшуємо m і повторюємо операції")  
 m+=1  
  
  
print(".................................................................................................................")  
print("\n\nНуль-гіпотеза (критерій Стьюдента):")  
Sb = sum(D) / N  
  
Sbs = math.sqrt(Sb / (N \* m))  
print(f"S{{beta}} = {Sbs}")  
  
beta = [(y1\_avg + y2\_avg + y3\_avg + y4\_avg)/4,  
 (-y1\_avg - y2\_avg + y3\_avg + y4\_avg)/4,  
 (-y1\_avg + y2\_avg - y3\_avg + y4\_avg)/4,  
 (-y1\_avg + y2\_avg + y3\_avg - y4\_avg)/4]  
  
print(f"beta(i): {beta}")  
  
t=[]  
for i in beta:  
 t.append(abs(i)/Sbs)  
print(f"t(i): {t}")  
  
f3 = f1\*f2  
print(f"f3 = f1 \* f2 = {f3}")  
t\_kr = Student[f3]  
print(f"t\_kr = {t\_kr}")  
print(f"t(i) < t\_kr: {[i for i in t if i<=t\_kr]}")  
print(f"\nНезначимі коефіцієнти: {[b[i] for i in range(len(t)) if t[i]<=t\_kr]}")  
print(f"Значимі коефіцієнти: {[b[i] for i in range(len(t)) if t[i]>t\_kr]}")  
t\_final = list(filter(lambda x: x<t\_kr, t))  
  
for i in range(len(t)):  
 if t[i] <= t\_kr:  
 b[i] = 0  
  
y\_t1 = b[0] + b[1] \* x\_matr[0][0] + b[2] \* x\_matr[1][0] + b[3]\* x\_matr[2][0]  
y\_t2 = b[0] + b[1] \* x\_matr[0][1] + b[2] \* x\_matr[1][1] + b[3]\* x\_matr[2][1]  
y\_t3 = b[0] + b[1] \* x\_matr[0][2] + b[2] \* x\_matr[1][2] + b[3]\* x\_matr[2][2]  
y\_t4 = b[0] + b[1] \* x\_matr[0][3] + b[2] \* x\_matr[1][3] + b[3]\* x\_matr[2][3]  
y\_t = [y\_t1, y\_t2, y\_t3, y\_t4]  
print(f"Середнє у(i): {y\_t}")  
  
print(".................................................................................................................")  
print("\nАдекватність моделі (критерій Фішера):")  
d = N - len(t\_final)  
f4 = N-d  
print(f"f4 = N - d = {f4}")  
  
fisher\_sum = 0  
for i in range(0, N):  
 fisher\_sum += pow((y\_t[i] - y\_avg\_arr[i]), 2)  
D\_ad = (m/(N-d))\*fisher\_sum  
Fp = D\_ad/Sb  
print(f"Fp = {Fp}")  
Ft = Fisher[f3-1][f4-1]  
print(f"Ft = {Ft}")  
if Ft > Fp:  
 print(f"Ft > Fp\nОтже, дане рівняння регресії адекватно оригіналу при рівні значимості {q}")  
else:  
 print(f"Ft < Fp\nОтже, дане рівняння регресії неадекватно оригіналу при рівні значимості {q}")

**Приклад роботи програми**

y1, y2, y3, ..., ym:

[219.2966, 220.9399, 226.728]

[224.344, 229.7332, 242.4129]

[213.0748, 206.2092, 196.0705]

[238.3164, 216.7277, 198.695]

[10, 10, 50, 50] - x1

[-20, 60, -20, 60] - x2

[-20, 20, 20, -20] - x3

середнє y(i): [222.32150000000001, 232.16336666666666, 205.11816666666667, 217.91303333333335]

mx(i): [30.0, 20.0, 0.0]

my = 219.37901666666667

a1 = 6424.102166666667, a2 = 4613.947666666667, a3= -14.7650000000001,

a11 = 1300.0, a12 = 600.0, a13 = 0.0,

a21 = 600.0, a22 = 2000.0, a23= 0.0,

a31 = 0.0, a32 = 0.0, a33= 400.0

b(i): [228.34455, -0.39317083333320313, 0.14147958333320312, -0.0369125]

D(i): [10.158693606666672, 57.367046215555625, 48.78621328222223, 262.3450638822221]

f1 = m - 1 = 2

f2 = N = 4

..............................................................................................

Однорідність дисперсії (критерій Кохрена):

Gp = 0.6928303243128857

Gt = 0.7679

Дисперсія однорідна, ослільки Gp < Gt

..............................................................................................

Нуль-гіпотеза (критерій Стьюдента):

S{beta} = 2.808680803133188

beta(i): [219.37901666666795, -7.863416666664072, 5.65918333332813, -0.7382499999999936]

t(i): [78.10749317684747, 2.799683274045288, 2.014890167303846, 0.2628458168605163]

f3 = f1 \* f2 = 8

t\_kr = 2.262

t(i) < t\_kr: [2.014890167303846, 0.2628458168605163]

Незначимі коефіцієнти: [0.14147958333320312, -0.0369125]

Значимі коефіцієнти: [228.34455, -0.39317083333320313]

Середнє у(i): [224.41284166666796, 224.41284166666796, 208.68600833333983, 208.68600833333983]

..............................................................................................

Адекватність моделі (критерій Фішера):

f4 = N - d = 2

Fp = 2.5719079531598834

Ft = 4.5

Ft > Fp

Отже, дане рівняння регресії адекватно оригіналу при рівні значимості 0.05

Process finished with exit code 0

**Контрольні запитання**

1. Що називається дробовим факторним експериментом?

Дробовий факторний експеримент – це частина ПФЕ, який мінімізує число дослідів, за рахунок тієї інформації, яка не дуже істотна для побудови лінійної моделі.

1. Для чого потрібно розрахункове значення Кохрена?

Розрахункове значення Кохрена показує, яку частку в загальній сумі дисперсій у рядках має максимальна з них.

1. Для чого перевіряється критерій Стьюдента?

Критерій Стьюдента використовується для перевірки значущості коефіцієнтів.

1. Чим визначається критерій Фішера і як його застосовувати?

Критерій Фішера використовується для перевірки адекватності рівняння регресії.