Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Методи оптимізації та планування експерименту**

Лабораторна робота №3:

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ

ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»

Виконала:

студентка групи ІВ-82

Чуясова Тетяна Андріївна

Номер у списку групи = 26

Перевірив:

Регіда П. Г.

Київ 2020р.

**Лабораторна робота №3**

**Тема:** ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ

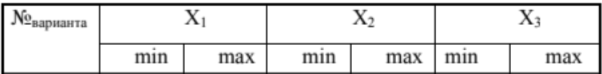
ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ

**Мета:** провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти

коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

**Варіант:**

Варіант =226





**Код програми:**

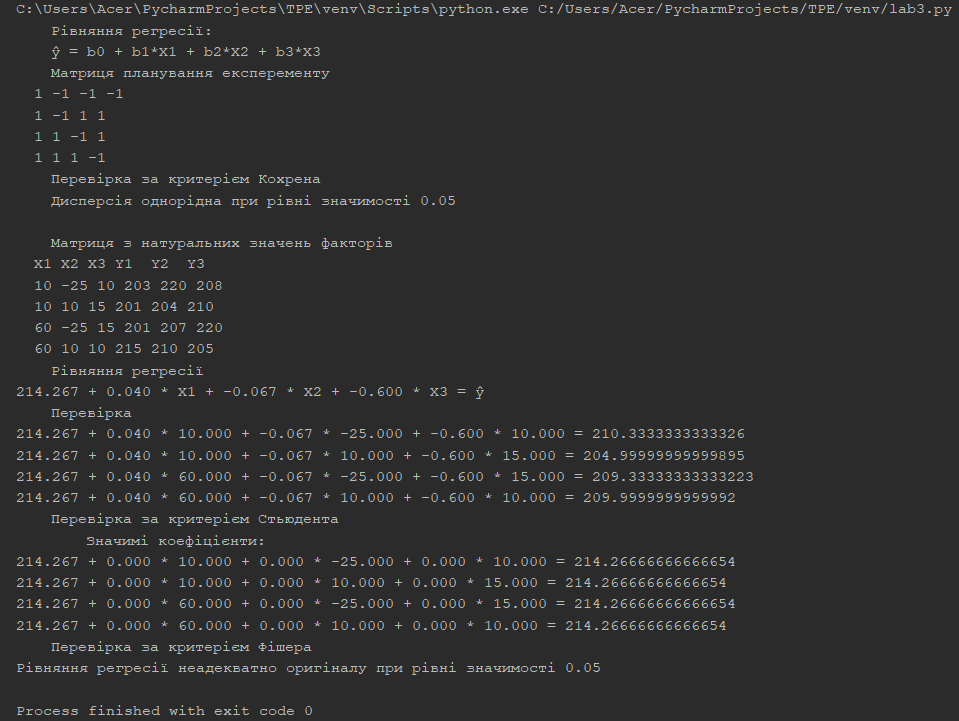
Файл lab3.py

import math  
from func\_tolab3 import genermat, aver\_y, aver\_x, det, Critical,x1\_min,x1\_max,x2\_min,x2\_max,x3\_min,x3\_max,y\_min,y\_max,m,N,p  
  
print("\tРівняння регресії: \n\tŷ = b0 + b1\*X1 + b2\*X2 + b3\*X3")  
print("\tМатриця планування експеременту")  
matrix\_pfe = [[1, -1, -1, -1],  
 [1, -1, 1, 1],  
 [1, 1, -1, 1],  
 [1, 1, 1, -1]]  
  
for i in range(len(matrix\_pfe)):  
 print(" ", end=" ")  
 for j in range(len(matrix\_pfe[i])):  
 print(matrix\_pfe[i][j], end=" ")  
 print(" ")  
  
  
a = True  
while a:  
 y\_matrix = genermat()  
 x\_matrix = [[x1\_min, x2\_min, x3\_min], [x1\_min, x2\_max, x3\_max], [x1\_max, x2\_min, x3\_max], [x1\_max, x2\_max, x3\_min]]  
 matrix = []  
 average\_y = aver\_y(y\_matrix)  
 average\_x = aver\_x(x\_matrix)  
 a1, a2, a3, a11, a22, a33, a12, a13, a23 = 0, 0, 0, 0, 0, 0, 0, 0, 0  
 for i in range(4):  
 a1 += x\_matrix[i][0] \* average\_y[i] / 4  
 a2 += x\_matrix[i][1] \* average\_y[i] / 4  
 a3 += x\_matrix[i][2] \* average\_y[i] / 4  
 a11 += x\_matrix[i][0] \*\* 2 / 4  
 a22 += x\_matrix[i][1] \*\* 2 / 4  
 a33 += x\_matrix[i][2] \*\* 2 / 4  
 a12 += x\_matrix[i][0] \* x\_matrix[i][1] / 4  
 a13 += x\_matrix[i][0] \* x\_matrix[i][2] / 4  
 a23 += x\_matrix[i][1] \* x\_matrix[i][2] / 4  
  
 a21 = a12  
 a31 = a13  
 a32 = a23  
 my = sum(average\_y) / len(average\_y)  
 b0\_numerator = [[my, average\_x[0], average\_x[1], average\_x[2]], [a1, a11, a12, a13], [a2, a21, a22, a23],  
 [a3, a31, a32, a33]]  
 b1\_numerator = [[1, my, average\_x[1], average\_x[2]], [average\_x[0], a1, a12, a13], [average\_x[1], a2, a22, a23],  
 [average\_x[2], a3, a32, a33]]  
 b2\_numerator = [[1, average\_x[0], my, average\_x[2]], [average\_x[0], a11, a1, a13], [average\_x[1], a21, a2, a23],  
 [average\_x[2], a31, a3, a33]]  
 b3\_numerator = [[1, average\_x[0], average\_x[1], my], [average\_x[0], a11, a12, a1], [average\_x[1], a21, a22, a2],  
 [average\_x[2], a31, a32, a3]]  
 b\_denominator = [[1, average\_x[0], average\_x[1], average\_x[2]], [average\_x[0], a11, a12, a13],  
 [average\_x[1], a21, a22, a23], [average\_x[2], a31, a32, a33]]  
 b0 = det(b0\_numerator) / det(b\_denominator)  
 b1 = det(b1\_numerator) / det(b\_denominator)  
 b2 = det(b2\_numerator) / det(b\_denominator)  
 b3 = det(b3\_numerator) / det(b\_denominator)  
 f1 = m - 1  
 f2 = N  
 q = 1 - p  
 dispersion\_y = [0, 0, 0, 0]  
 for i in range(m):  
 dispersion\_y[0] += ((y\_matrix[0][i] - average\_y[0]) \*\* 2) / 3  
 dispersion\_y[1] += ((y\_matrix[1][i] - average\_y[1]) \*\* 2) / 3  
 dispersion\_y[2] += ((y\_matrix[2][i] - average\_y[2]) \*\* 2) / 3  
 dispersion\_y[3] += ((y\_matrix[3][i] - average\_y[3]) \*\* 2) / 3  
  
 Gp = max(dispersion\_y) / sum(dispersion\_y)  
 print("\tПеревірка за критерієм Кохрена")  
 Gt = Critical.get\_cohren\_value(f2, f1, q)  
 if Gt > Gp:  
 print("\tДисперсія однорідна при рівні значимості {:.2f}\n\t".format(q))  
 a = False  
 else:  
 print("\tДисперсія не однорідна при рівні значимості {:.2f}!".format(q))  
 m += 1  
  
for i in range(4):  
 matrix.append(x\_matrix[i] + y\_matrix[i])  
print("\tМатриця з натуральних значень факторів")  
print(" X1 X2 X3 Y1 Y2 Y3 ")  
for i in range(len(matrix)):  
 print(" ", end=" ")  
 for j in range(len(matrix[i])):  
 print(matrix[i][j], end=" ")  
 print(" ")  
  
print("\tРівняння регресії")  
print("{:.3f} + {:.3f} \* X1 + {:.3f} \* X2 + {:.3f} \* X3 = ŷ".format(b0, b1, b2, b3))  
print("\tПеревірка")  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = ".format(b0, b1, x1\_min, b2, x2\_min, b3, x3\_min)  
 + str(b0 + b1 \* x1\_min + b2 \* x2\_min + b3 \* x3\_min))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = ".format(b0, b1, x1\_min, b2, x2\_max, b3, x3\_max)  
 + str(b0 + b1 \* x1\_min + b2 \* x2\_max + b3 \* x3\_max))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = ".format(b0, b1, x1\_max, b2, x2\_min, b3, x3\_max)  
 + str(b0 + b1 \* x1\_max + b2 \* x2\_min + b3 \* x3\_max))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = ".format(b0, b1, x1\_max, b2, x2\_max, b3, x3\_min)  
 + str(b0 + b1 \* x1\_max + b2 \* x2\_max + b3 \* x3\_min))  
  
print("\tПеревірка за критерієм Стьюдента")  
f3 = f1 \* f2  
S\_2b = sum(dispersion\_y) / (N \* N \* m)  
S\_b = math.sqrt(S\_2b)  
beta\_0 = (average\_y[0] + average\_y[1] + average\_y[2] + average\_y[3]) / N  
beta\_1 = (-average\_y[0] - average\_y[1] + average\_y[2] + average\_y[3]) / N  
beta\_2 = (-average\_y[0] + average\_y[1] - average\_y[2] + average\_y[3]) / N  
beta\_3 = (-average\_y[0] + average\_y[1] + average\_y[2] - average\_y[3]) / N  
t\_0 = math.fabs(beta\_0) / S\_b  
t\_1 = math.fabs(beta\_1) / S\_b  
t\_2 = math.fabs(beta\_2) / S\_b  
t\_3 = math.fabs(beta\_3) / S\_b  
Tt = Critical.get\_student\_value(f1 \* f2, q)  
t\_lst = [t\_0, t\_1, t\_2, t\_3]  
b\_lst = [b0, b1, b2, b3]  
for i in range(4):  
 if t\_lst[i] > Tt:  
 continue  
 else:  
 t\_lst[i] = 0  
for j in range(4):  
 if t\_lst[j] != 0:  
 continue  
 else:  
 b\_lst[j] = 0  
print("\t\tЗначимі коефіцієнти: ")  
yj1 = b\_lst[0] + b\_lst[1] \* x1\_min + b\_lst[2] \* x2\_min + b\_lst[3] \* x3\_min  
yj2 = b\_lst[0] + b\_lst[1] \* x1\_min + b\_lst[2] \* x2\_max + b\_lst[3] \* x3\_max  
yj3 = b\_lst[0] + b\_lst[1] \* x1\_max + b\_lst[2] \* x2\_min + b\_lst[3] \* x3\_max  
yj4 = b\_lst[0] + b\_lst[1] \* x1\_max + b\_lst[2] \* x2\_max + b\_lst[3] \* x3\_min  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = "  
 "".format(b\_lst[0], b\_lst[1], x1\_min, b\_lst[2], x2\_min, b\_lst[3],  
 x3\_min) + str(yj1))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = "  
 "".format(b\_lst[0], b\_lst[1], x1\_min, b\_lst[2], x2\_max, b\_lst[3],  
 x3\_max) + str(yj2))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = "  
 "".format(b\_lst[0], b\_lst[1], x1\_max, b\_lst[2], x2\_min, b\_lst[3],  
 x3\_max) + str(yj3))  
print("{:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} + {:.3f} \* {:.3f} = "  
 "".format(b\_lst[0], b\_lst[1], x1\_max, b\_lst[2], x2\_max, b\_lst[3],  
 x3\_min) + str(yj4))  
print("\tПеревірка за критерієм Фішера")  
for i in range(3):  
 if b\_lst[i] == 0:  
 del b\_lst[i]  
  
d = len(b\_lst)  
f4 = N - d  
S\_2ad = m \* ((yj1 - average\_y[0]) \*\* 2 + (yj2 - average\_y[1]) \*\* 2 + (yj3 - average\_y[2]) \*\* 2 + (  
 yj4 - average\_y[3]) \*\* 2) / f4  
Fp = S\_2ad / S\_2b  
Ft = Critical.get\_fisher\_value(f1 \* f2, f4, q)  
if Fp > Ft:  
 print("Рівняння регресії неадекватно оригіналу при рівні значимості {:.2f}".format(q))  
else:  
 print("Рівняння регресії адекватно оригіналу при рівні значимості {:.2f}".format(q))

Файл func\_tolab3.py

x1\_min = 10  
x1\_max = 60  
x2\_min = -25  
x2\_max = 10  
x3\_min = 10  
x3\_max = 15  
y\_min = 200 + int((x1\_min + x2\_min + x3\_min) / 3)  
y\_max = 200 + int((x1\_max + x2\_max + x3\_max) / 3)  
m = 3  
N = 4  
p = 0.95  
  
def genermat():  
 from random import randrange  
 matrix\_with\_y = [[randrange(y\_min, y\_max) for y in range(3)] for x in range(4)]  
 return matrix\_with\_y  
  
  
def aver\_y(lst):  
 average = []  
 for k in range(len(lst)):  
 average.append(sum(lst[k]) / len(lst[k]))  
 return average  
  
  
def aver\_x(lst):  
 average = [0, 0, 0]  
 for k in range(4):  
 average[0] += lst[k][0] / 4  
 average[1] += lst[k][1] / 4  
 average[2] += lst[k][2] / 4  
 return average  
  
  
def det(a):  
 from numpy.linalg import det  
 return det(a)  
  
  
class Critical:  
 @staticmethod  
 def get\_cohren\_value(size\_of\_selections, qty\_of\_selections, significance):  
 from \_pydecimal import Decimal  
 from scipy.stats import f  
 size\_of\_selections += 1  
 partResult1 = significance / (size\_of\_selections - 1)  
 params = [partResult1, qty\_of\_selections, (size\_of\_selections - 1 - 1) \* qty\_of\_selections]  
 fisher = f.isf(\*params)  
 result = fisher / (fisher + (size\_of\_selections - 1 - 1))  
 return Decimal(result).quantize(Decimal('.0001')).\_\_float\_\_()  
  
 @staticmethod  
 def get\_student\_value(f3, significance):  
 from \_pydecimal import Decimal  
 from scipy.stats import t  
 return Decimal(abs(t.ppf(significance / 2, f3))).quantize(Decimal('.0001')).\_\_float\_\_()  
  
 @staticmethod  
 def get\_fisher\_value(f3, f4, significance):  
 from \_pydecimal import Decimal  
 from scipy.stats import f  
 return Decimal(abs(f.isf(significance, f4, f3))).quantize(Decimal('.0001')).\_\_float\_\_()

**Результат виконання роботи програми:**



**Висновок:**

У ході даної лабораторної роботи був проведений дробовий трьохфакторний експеримент, складена матриця планування, знайдені

коефіцієнти рівняння регресії, проведені перевірки Кохрена, Стьюдента та Фішера. Розроблена текстова програма, результати її роботи наведені вище.

**Контрольні запитання**

1. *Що називається дробовим факторним експериментом?*

У деяких випадках немає необхідності проводити повний факторний експеримент (ПФЕ). Якщо буде використовуватися лінійна регресія, то можливо зменшити кількість рядків матриці ПФЕ до кількості коефіцієнтів регресійної моделі. Кількість дослідів слід скоротити, використовуючи для планування так звані регулярні дробові репліки від повного факторного експерименту, що містять відповідну кількість дослідів і зберігають основні властивості матриці планування – це означає дробовий факторний експеримент (ДФЕ).

1. *Для чого потрібно розрахункове значення Кохрена?*

Для перевірки дисперсії на однорідність.

1. *Для чого перевіряється критерій Стьюдента?*

Для перевірки значущості коефіцієнтів регресії. Тобто, Якщо виконується нерівність *ts*< *tтабл*, то приймається нуль-гіпотеза, тобто вважається, що знайдений коефіцієнт *βs* є статистично незначущим і його слід виключити з рівняння регресії. Якщо *ts* > *tтабл* то гіпотеза не підтверджується, тобто *βs* – значимий коефіцієнт і він залишається в рівнянні регресії.

1. *Чим визначається критерій Фішера і як його застосовувати?*

Отримане рівняння регресії необхідно перевірити на адекватність досліджуваному об'єкту. Для цієї мети необхідно оцінити, наскільки відрізняються середні значення у вихідної величини, отриманої в точках факторного простору, і значення у, отриманого з рівняння регресії в тих самих точках факторного простору. Для цього використовують дисперсію адекватності. Адекватність моделі перевіряють за F-критерієм Фішера, який дорівнює відношенню дисперсії адекватності до дисперсії відтворюваності.