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 №4:

«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ ЕФЕКТУ ВЗАЄМОДІЇ»

Виконав:

студент групи ІВ-82

Сорока Андрій

Залікова книжка № 8222

Перевірив:

Регіда П. Г.

Київ 2020р.

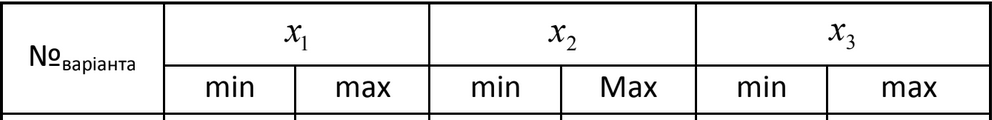
**Лабораторна робота №4**

**Тема:** ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННІ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ ЕФЕКТУ ВЗАЄМОДІЇ.

**Мета:** провести повний трьохфакторний експеремент. Знайти рівняння регресії адекватне об'єкту.

**Виконання:**

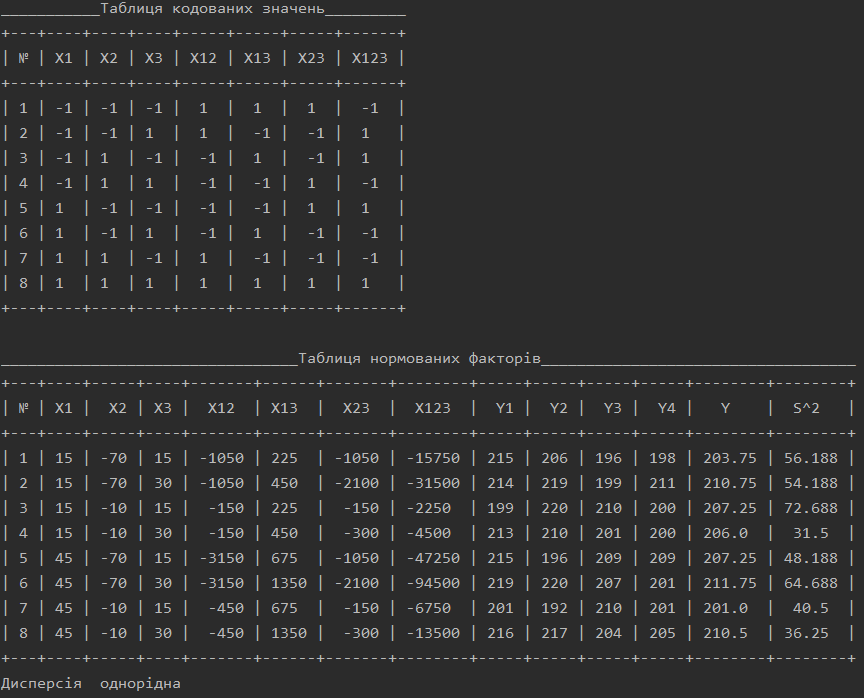
Варіант – 220.

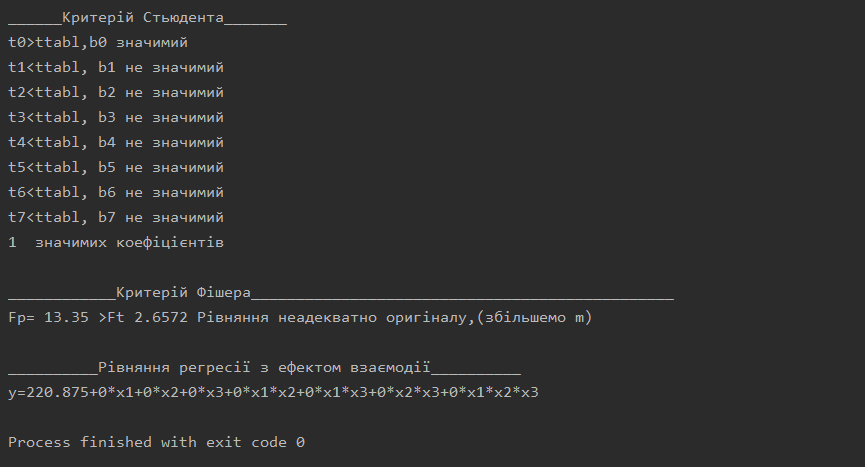


**Код програми**

import random  
from scipy.stats import f, t  
from prettytable import PrettyTable  
from numpy.linalg import solve  
  
x1min = 15  
x1max = 45  
x2min = -70  
x2max = -10  
x3min = 15  
x3max = 30  
  
xAvmax = (x1max + x2max + x3max) / 3  
xAvmin = (x1min + x2min + x3min) / 3  
ymax = int(200 + xAvmax)  
ymin = int(200 + xAvmin)  
  
m = 4  
  
X11 = [-1, -1, -1, -1, 1, 1, 1, 1]  
X22 = [-1, -1, 1, 1, -1, -1, 1, 1]  
X33 = [-1, 1, -1, 1, -1, 1, -1, 1]  
  
  
def sumkf2(x1, x2):  
 xn = []  
 for i in range(len(x1)):  
 xn.append(x1[i] \* x2[i])  
 return xn  
  
  
def sumkf3(x1, x2, x3):  
 xn = []  
 for i in range(len(x1)):  
 xn.append(x1[i] \* x2[i] \* x3[i])  
 return xn  
  
  
def kv(x):  
 xn = []  
 for i in range(len(x)):  
 xn.append(x[i] \* x[i])  
 return xn  
  
  
X12 = sumkf2(X11, X22)  
X13 = sumkf2(X11, X33)  
X23 = sumkf2(X22, X33)  
X123 = sumkf3(X11, X22, X33)  
X8 = kv(X11)  
X9 = kv(X22)  
X10 = kv(X33)  
  
X00 = [1, 1, 1, 1, 1, 1, 1, 1]  
print("\_\_\_\_\_\_\_\_\_\_\_Таблиця кодованих значень\_\_\_\_\_\_\_\_\_")  
table1 = PrettyTable()  
table1.add\_column("№", (1, 2, 3, 4, 5, 6, 7, 8))  
table1.add\_column("X1", X11)  
table1.add\_column("X2", X22)  
table1.add\_column("X3", X33)  
table1.add\_column("X12", X12)  
table1.add\_column("X13", X13)  
table1.add\_column("X23", X23)  
table1.add\_column("X123", X123)  
print(table1)  
for i in range(1, m + 1):  
 globals()['Y%s' % i] = [random.randrange(ymin, ymax, 1) for k in range(8)]  
X1 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max]  
X2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max]  
X3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max]  
X12 = sumkf2(X1, X2)  
X13 = sumkf2(X1, X3)  
X23 = sumkf2(X2, X3)  
X123 = sumkf3(X1, X2, X3)  
X0 = [1] \* 8  
s1, s2, s3, s4, s5, s6, s7, s8 = 0, 0, 0, 0, 0, 0, 0, 0  
for i in range(1, m + 1):  
 s1 += globals()['Y%s' % i][0]  
 s2 += globals()['Y%s' % i][1]  
 s3 += globals()['Y%s' % i][2]  
 s4 += globals()['Y%s' % i][3]  
 s5 += globals()['Y%s' % i][4]  
 s6 += globals()['Y%s' % i][5]  
 s7 += globals()['Y%s' % i][6]  
 s8 += globals()['Y%s' % i][7]  
y1av1 = s1 / m  
y2av2 = s2 / m  
y3av3 = s3 / m  
y4av4 = s4 / m  
y5av5 = s5 / m  
y6av6 = s6 / m  
y7av7 = s7 / m  
y8av8 = s8 / m  
yav = [round(y1av1, 3), round(y2av2, 3), round(y3av3, 3), round(y4av4, 3), round(y5av5, 3), round(y6av6, 3),  
 round(y7av7, 3), round(y8av8, 3)]  
sd1, sd2, sd3, sd4, sd5, sd6, sd7, sd8 = 0, 0, 0, 0, 0, 0, 0, 0  
for i in range(1, m + 1):  
 sd1 += ((globals()['Y%s' % i][0]) - y1av1) \*\* 2  
 sd2 += ((globals()['Y%s' % i][1]) - y2av2) \*\* 2  
 sd3 += ((globals()['Y%s' % i][2]) - y3av3) \*\* 2  
 sd4 += ((globals()['Y%s' % i][3]) - y4av4) \*\* 2  
 sd5 += ((globals()['Y%s' % i][4]) - y5av5) \*\* 2  
 sd6 += ((globals()['Y%s' % i][5]) - y6av6) \*\* 2  
 sd7 += ((globals()['Y%s' % i][6]) - y7av7) \*\* 2  
 sd8 += ((globals()['Y%s' % i][7]) - y8av8) \*\* 2  
  
d1 = sd1 / m  
d2 = sd2 / m  
d3 = sd3 / m  
d4 = sd4 / m  
d5 = sd5 / m  
d6 = sd6 / m  
d7 = sd7 / m  
d8 = sd8 / m  
disper = [round(d1, 3), round(d2, 3), round(d3, 3), round(d4, 3), round(d5, 3), round(d6, 3), round(d7, 3),  
 round(d8, 3)]  
  
  
print("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Таблиця нормованих факторів\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
table2 = PrettyTable()  
table2.add\_column("№", (1, 2, 3, 4, 5, 6, 7, 8))  
table2.add\_column("X1", X1)  
table2.add\_column("X2", X2)  
table2.add\_column("X3", X3)  
table2.add\_column("X12", X12)  
table2.add\_column("X13", X13)  
table2.add\_column("X23", X23)  
table2.add\_column("X123", X123)  
for i in range(1, m + 1):  
 table2.add\_column("Y" + str(i), globals()['Y%s' % i])  
table2.add\_column("Y", yav)  
table2.add\_column("S^2", disper)  
print(table2)  
  
b = [i for i in solve(list(zip(X0, X1, X2, X3, X12, X13, X23, X123)), yav)]  
b0, b1, b2, b3, b4, b5, b6, b7 = round(b[0], 3), round(b[1], 3), round(b[2], 3), round(b[3], 3), round(b[4], 3), round(  
 b[5], 3), round(b[6], 3), round(b[7], 3)  
  
  
dcouple = [d1, d2, d3, d4, d5, d6, d7, d8]  
  
m = 3  
Gp = max(dcouple) / sum(dcouple)  
q = 0.05  
f1 = m - 1  
f2 = N = 8  
fisher = f.isf(\*[q / f2, f1, (f2 - 1) \* f1])  
Gt = round(fisher / (fisher + (f2 - 1)), 4)  
if Gp < Gt:  
 print("Дисперсія однорідна")  
 print("\n\_\_\_\_\_\_Критерій Стьюдента\_\_\_\_\_\_\_")  
 sb = sum(dcouple) / N  
 ssbs = sb / N \* m  
 sbs = ssbs \*\* 0.5  
  
 beta0 = (y1av1 \* 1 + y2av2 \* 1 + y3av3 \* 1 + y4av4 \* 1 + y5av5 \* 1 + y6av6 \* 1 + y7av7 \* 1 + y8av8 \* 1) / 8  
 beta1 = (y1av1 \* (-1) + y2av2 \* (-1) + y3av3 \* (-1) + y4av4 \* (  
 -1) + y5av5 \* 1 + y6av6 \* 1 + y7av7 \* 1 + y8av8 \* 1) / 8  
 beta2 = (y1av1 \* (-1) + y2av2 \* (-1) + y3av3 \* 1 + y4av4 \* 1 + y5av5 \* (-1) + y6av6 \* (  
 -1) + y7av7 \* 1 + y8av8 \* 1) / 8  
 beta3 = (y1av1 \* (-1) + y2av2 \* 1 + y3av3 \* (-1) + y4av4 \* 1 + y5av5 \* (-1) + y6av6 \* 1 + y7av7 \* (  
 -1) + y8av8 \* 1) / 8  
 beta4 = (y1av1 \* 1 + y2av2 \* 1 + y3av3 \* (-1) + y4av4 \* (-1) + y5av5 \* (-1) + y6av6 \* (  
 -1) + y7av7 \* 1 + y8av8 \* 1) / 8  
 beta5 = (y1av1 \* 1 + y2av2 \* (-1) + y3av3 \* 1 + y4av4 \* (-1) + y5av5 \* (-1) + y6av6 \* 1 + y7av7 \* (  
 -1) + y8av8 \* 1) / 8  
 beta6 = (y1av1 \* 1 + y2av2 \* (-1) + y3av3 \* (-1) + y4av4 \* 1 + y5av5 \* 1 + y6av6 \* (-1) + y7av7 \* (  
 -1) + y8av8 \* 1) / 8  
 beta7 = (y1av1 \* (-1) + y2av2 \* 1 + y3av3 \* 1 + y4av4 \* (-1) + y5av5 \* 1 + y6av6 \* (-1) + y7av7 \* (  
 -1) + y8av8 \* 1) / 8  
  
 t0 = abs(beta0) / sbs  
 t1 = abs(beta1) / sbs  
 t2 = abs(beta2) / sbs  
 t3 = abs(beta3) / sbs  
 t4 = abs(beta4) / sbs  
 t5 = abs(beta5) / sbs  
 t6 = abs(beta6) / sbs  
 t7 = abs(beta7) / sbs  
  
 f3 = f1 \* f2  
 ttabl = round(abs(t.ppf(q / 2, f3)), 4)  
  
 d = 8  
 if t0 < ttabl:  
 print("t0<ttabl, b0 не значимий")  
 b0 = 0  
 d = d - 1  
 else:  
 print("t0>ttabl,b0 значимий")  
 if t1 < ttabl:  
 print("t1<ttabl, b1 не значимий")  
 b1 = 0  
 d = d - 1  
 if t2 < ttabl:  
 print("t2<ttabl, b2 не значимий")  
 b2 = 0  
 d = d - 1  
 if t3 < ttabl:  
 print("t3<ttabl, b3 не значимий")  
 b3 = 0  
 d = d - 1  
 if t4 < ttabl:  
 print("t4<ttabl, b4 не значимий")  
 b4 = 0  
 d = d - 1  
 if t5 < ttabl:  
 print("t5<ttabl, b5 не значимий")  
 b5 = 0  
 d = d - 1  
 if t6 < ttabl:  
 print("t6<ttabl, b6 не значимий")  
 b6 = 0  
 d = d - 1  
 if t7 < ttabl:  
 print("t7<ttabl, b7 не значимий")  
 b7 = 0  
 d = d - 1  
  
 print(d, " значимих коефіцієнтів")  
  
 yy1 = b0 + b1 \* x1min + b2 \* x2min + b3 \* x3min + b4 \* x1min \* x2min + b5 \* x1min \* x3min + b6 \* x2min \* x3min + b7 \* x1min \* x2min \* x3min  
 yy2 = b0 + b1 \* x1min + b2 \* x2min + b3 \* x3max + b4 \* x1min \* x2min + b5 \* x1min \* x3max + b6 \* x2min \* x3max + b7 \* x1min \* x2min \* x3max  
 yy3 = b0 + b1 \* x1min + b2 \* x2max + b3 \* x3min + b4 \* x1min \* x2max + b5 \* x1min \* x3min + b6 \* x2max \* x3min + b7 \* x1min \* x2max \* x3min  
 yy4 = b0 + b1 \* x1min + b2 \* x2max + b3 \* x3max + b4 \* x1min \* x2max + b5 \* x1min \* x3max + b6 \* x2max \* x3max + b7 \* x1min \* x2max \* x3max  
 yy5 = b0 + b1 \* x1max + b2 \* x2min + b3 \* x3min + b4 \* x1max \* x2min + b5 \* x1max \* x3min + b6 \* x2min \* x3min + b7 \* x1max \* x2min \* x3min  
 yy6 = b0 + b1 \* x1max + b2 \* x2min + b3 \* x3max + b4 \* x1max \* x2min + b5 \* x1max \* x3max + b6 \* x2min \* x3max + b7 \* x1max \* x2min \* x3max  
 yy7 = b0 + b1 \* x1max + b2 \* x2max + b3 \* x3min + b4 \* x1max \* x2max + b5 \* x1max \* x3min + b6 \* x2max \* x3min + b7 \* x1max \* x2min \* x3max  
 yy8 = b0 + b1 \* x1max + b2 \* x2max + b3 \* x3max + b4 \* x1max \* x2max + b5 \* x1max \* x3max + b6 \* x2max \* x3max + b7 \* x1max \* x2max \* x3max  
 print("\n\_\_\_\_\_\_\_\_\_\_\_\_Критерій Фішера\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")  
 f4 = N - d  
 sad = ((yy1 - y1av1) \*\* 2 + (yy2 - y2av2) \*\* 2 + (yy3 - y3av3) \*\* 2 + (yy4 - y4av4) \*\* 2 + (yy5 - y5av5) \*\* 2 + (  
 yy6 - y6av6) \*\* 2 + (yy7 - y7av7) \*\* 2 + (yy8 - y8av8) \*\* 2) \* (m / (N - d))  
 Fp = sad / sb  
  
  
 Ft = round(abs(f.isf(q, f4, f3)), 4)  
  
 cont = 0  
 if Fp > Ft:  
 print("Fp=", round(Fp, 2), ">Ft", Ft, "Рівняння неадекватно оригіналу,(збільшемо m)")  
 m+=1  
 cont = 1  
 else:  
 print("Fp=", round(Fp, 2), "<Ft", Ft, "Рівняння адекватно оригіналу")  
  
else:  
 print("Дисперсія неоднорідна(збільшемо кількість дослідів)")  
 m += 1  
print("\n\_\_\_\_\_\_\_\_\_\_Рівняння регресії з ефектом взаємодії\_\_\_\_\_\_\_\_\_\_")  
print("y=" + str(b0) + "+" + str(b1) + "\*x1+" + str(b2) + "\*x2+" + str(b3) + "\*x3+" + str(b4) + "\*x1\*x2+" + str(  
 b5) + "\*x1\*x3+" + str(b6) + "\*x2\*x3+" + str(b7) + "\*x1\*x2\*x3")

**Результат виконання роботи програми:**





**Висновок**

Отже, у ході виконання лабораторної роботи № 4 ми провели повний трьохфакторний експеримент при використанні рівняння з ефектом взаємодії. Склали матрицю планування, знайшли коефіцієнти рівняння регресії, провели 3 статистичні перевірки. Була написана тестова програма, результати наведені вище.