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 №6**

З дисципліни «Методи наукових досліджень»

**Проведення трьохфакторного експерименту при використанні рівняння регресії з квадратичними членами**

ВИКОНАВ:

Студент ІІ курсу ФІОТ

Групи ІВ-91

Липчак Дмитро Олександрович

Номер заліковки: 9118

Номер у списку: 17

ПЕРЕВІРИВ:

ас. Регіда П. Г.

Київ 2021 р.

**Мета роботи:** Провести трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи **рототабельний**

композиційний план.

**Завдання до лабораторної роботи:**

1. Ознайомитися з теоретичними відомостями.

2. Вибрати з таблиці варіантів і записати в протокол інтервали значень х1, х2, х3. Обчислити і записати значення, відповідні кодованим значенням

факторів +1; -1;+ ; - ; 0 для 1, 2, 3.

3. Значення функції відгуку знайти за допомогою підстановки в формулу:

yi = f(х1, х2, х3) + random(10)-5,

де f(х1, х2, х3) вибирається по номеру в списку в журналі викладача.

4. Провести експерименти і аналізуючи значення статистичних перевірок, отримати адекватну модель рівняння регресії. При розрахунках

використовувати натуральні значення факторів.

5. Зробити висновки по виконаній роботі.

*Алгоритм отримання адекватної моделі рівняння регресії*

*1) Вибір рівняння регресії (лінійна форма, рівняння з урахуванням ефекту взаємодії і з урахуванням квадратичних членів);*

*2) Вибір кількості повторів кожної комбінації (m = 2);*

*3) Складення матриці планування експерименту і вибір кількості рівнів (N)*

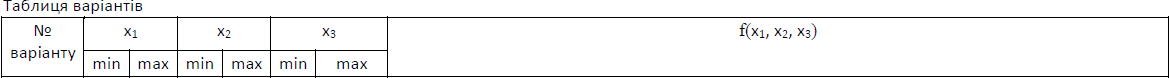
*4) Проведення експериментів;*

*5) Перевірка однорідності дисперсії. Якщо не однорідна – повертаємося на п. 2 і збільшуємо m на 1);*

*6) Розрахунок коефіцієнтів рівняння регресії. При розрахунку використовувати* ***натуральні*** *значення x1, x2 и x3.*

*7) Перевірка нуль-гіпотези. Визначення значимих коефіцієнтів;*

*8) Перевірка адекватності моделі рівняння оригіналу. При неадекватності – повертаємося на п.1, змінивши при цьому рівняння регресії;*

**Виконання роботи**

**Лістинг програми**

from math import fabs  
from random import randrange  
import numpy as np  
from numpy.linalg import solve  
from scipy.stats import f, t  
  
m = 3  
n = 15  
  
x1\_min = 20  
x1\_max = 70  
x2\_min = 25  
x2\_max = 65  
x3\_min = 25  
x3\_max = 35  
  
  
def function(x1, x2, x3):  
 y = 0.6 + 4.0 \* x1 + 2.8 \* x2 + 4.7 \* x3 + 3.1 \* x1 \* x1 + 0.4 \* x2 \* x2 + 5.4 \* x3 \* x3 + 5.7 \* x1 \* x2 + \  
 0.1 \* x1 \* x3 + 8.8 \* x2 \* x3 + 0.1 \* x1 \* x2 \* x3 + randrange(0, 10) - 5  
 return y  
  
  
x01 = (x1\_max + x1\_min) / 2  
x02 = (x2\_max + x2\_min) / 2  
x03 = (x3\_max + x3\_min) / 2  
x1\_delt = x1\_max - x01  
x2\_delt = x2\_max - x02  
x3\_delt = x3\_max - x03  
  
xn = [[-1, -1, -1, +1, +1, +1, -1, +1, +1, +1],  
 [-1, -1, +1, +1, -1, -1, +1, +1, +1, +1],  
 [-1, +1, -1, -1, +1, -1, +1, +1, +1, +1],  
 [-1, +1, +1, -1, -1, +1, -1, +1, +1, +1],  
 [+1, -1, -1, -1, -1, +1, +1, +1, +1, +1],  
 [+1, -1, +1, -1, +1, -1, -1, +1, +1, +1],  
 [+1, +1, -1, +1, -1, -1, -1, +1, +1, +1],  
 [+1, +1, +1, +1, +1, +1, +1, +1, +1, +1],  
 [-1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],  
 [+1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0, 0],  
 [0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],  
 [0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929, 0],  
 [0, 0, -1.73, 0, 0, 0, 0, 0, 0, 2.9929],  
 [0, 0, +1.73, 0, 0, 0, 0, 0, 0, 2.9929],  
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]  
  
x1 = [x1\_min, x1\_min, x1\_min, x1\_min, x1\_max, x1\_max, x1\_max, x1\_max, -1.73 \* x1\_delt + x01, 1.73 \* x1\_delt + x01, x01, x01,  
 x01, x01, x01]  
x2 = [x2\_min, x2\_min, x2\_max, x2\_max, x2\_min, x2\_min, x2\_max, x2\_max, x02, x02, -1.73 \* x2\_delt + x02, 1.73 \* x2\_delt + x02,  
 x02, x02, x02]  
x3 = [x3\_min, x3\_max, x3\_min, x3\_max, x3\_min, x3\_max, x3\_min, x3\_max, x03, x03, x03, x03, -1.73 \* x3\_delt + x03,  
 1.73 \* x3\_delt + x03, x03]  
  
x1x2, x1x3, x2x3, x1x2x3 = [0] \* n, [0] \* n, [0] \* n, [0] \* n  
x1kv, x2kv, x3kv = [0] \* n, [0] \* n, [0] \* n  
  
for i in range(15):  
 x1x2[i] = x1[i] \* x2[i]  
 x1x3[i] = x1[i] \* x3[i]  
 x2x3[i] = x2[i] \* x3[i]  
 x1x2x3[i] = x1[i] \* x2[i] \* x3[i]  
 x1kv[i] = x1[i] \*\* 2  
 x2kv[i] = x2[i] \*\* 2  
 x3kv[i] = x3[i] \*\* 2  
  
list\_for\_a = list(zip(x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3, x1kv, x2kv, x3kv))  
print("")  
  
print("Матриця планування з натуралізованими коефіцієнтами X")  
print(" X1 X2 X3 X1X2 X1X3 X2X3 X1X2X3 X1X1"  
 " X2X2 X3X3")  
for i in range(n):  
 for j in range(len(list\_for\_a[0])):  
 print("{:^12.3f}".format(list\_for\_a[i][j]), end='')  
 print("")  
# вивід матриці планування  
Y = [[function(list\_for\_a[j][0], list\_for\_a[j][1], list\_for\_a[j][2]) for i in range(m)] for j in range(15)]  
print("")  
  
print("Матриця планування Y")  
print(" Y1 Y2 Y3")  
for i in range(n):  
 print(end='')  
 for j in range(len(Y[0])):  
 print("{:^12.3f}".format(Y[i][j]), end='')  
 print("")  
# середні у  
aver\_y = []  
for i in range(len(Y)):  
 aver\_y.append(np.mean(Y[i], axis=0))  
print("")  
  
print("Середні значення відгуку за рядками")  
for i in range(n):  
 print("{:.3f}".format(aver\_y[i]))  
# розрахунок дисперсій  
dispersions = []  
for i in range(len(Y)):  
 a = 0  
 for k in Y[i]:  
 a += (k - np.mean(Y[i], axis=0)) \*\* 2  
 dispersions.append(a / len(Y[i]))  
  
  
def find\_kn(num):  
 a = 0  
 for j in range(n):  
 a += aver\_y[j] \* list\_for\_a[j][num - 1] / n  
 return a  
  
  
def a(first, second):  
 a = 0  
 for j in range(n):  
 a += list\_for\_a[j][first - 1] \* list\_for\_a[j][second - 1] / n  
 return a  
  
  
my = sum(aver\_y) / n  
mx = []  
for i in range(10):  
 number\_lst = []  
 for j in range(n):  
 number\_lst.append(list\_for\_a[j][i])  
 mx.append(sum(number\_lst) / len(number\_lst))  
  
det\_1 = [  
 [1, mx[0], mx[1], mx[2], mx[3], mx[4], mx[5], mx[6], mx[7], mx[8], mx[9]],  
 [mx[0], a(1, 1), a(1, 2), a(1, 3), a(1, 4), a(1, 5), a(1, 6), a(1, 7), a(1, 8), a(1, 9), a(1, 10)],  
 [mx[1], a(2, 1), a(2, 2), a(2, 3), a(2, 4), a(2, 5), a(2, 6), a(2, 7), a(2, 8), a(2, 9), a(2, 10)],  
 [mx[2], a(3, 1), a(3, 2), a(3, 3), a(3, 4), a(3, 5), a(3, 6), a(3, 7), a(3, 8), a(3, 9), a(3, 10)],  
 [mx[3], a(4, 1), a(4, 2), a(4, 3), a(4, 4), a(4, 5), a(4, 6), a(4, 7), a(4, 8), a(4, 9), a(4, 10)],  
 [mx[4], a(5, 1), a(5, 2), a(5, 3), a(5, 4), a(5, 5), a(5, 6), a(5, 7), a(5, 8), a(5, 9), a(5, 10)],  
 [mx[5], a(6, 1), a(6, 2), a(6, 3), a(6, 4), a(6, 5), a(6, 6), a(6, 7), a(6, 8), a(6, 9), a(6, 10)],  
 [mx[6], a(7, 1), a(7, 2), a(7, 3), a(7, 4), a(7, 5), a(7, 6), a(7, 7), a(7, 8), a(7, 9), a(7, 10)],  
 [mx[7], a(8, 1), a(8, 2), a(8, 3), a(8, 4), a(8, 5), a(8, 6), a(8, 7), a(8, 8), a(8, 9), a(8, 10)],  
 [mx[8], a(9, 1), a(9, 2), a(9, 3), a(9, 4), a(9, 5), a(9, 6), a(9, 7), a(9, 8), a(9, 9), a(9, 10)],  
 [mx[9], a(10, 1), a(10, 2), a(10, 3), a(10, 4), a(10, 5), a(10, 6), a(10, 7), a(10, 8), a(10, 9), a(10, 10)]]  
  
det\_2 = [my, find\_kn(1), find\_kn(2), find\_kn(3), find\_kn(4), find\_kn(5), find\_kn(6), find\_kn(7),  
 find\_kn(8), find\_kn(9), find\_kn(10)]  
  
beta = solve(det\_1, det\_2)  
print("")  
  
print("Отримане рівняння регресії")  
print("{:.3f} + {:.3f} \* X1 + {:.3f} \* X2 + {:.3f} \* X3 + {:.3f} \* Х1X2 + {:.3f} \* Х1X3 +\n + {:.3f} \* Х2X3"  
 "+ {:.3f} \* Х1Х2X3 + {:.3f} \* X11^2 + {:.3f} \* X22^2 + {:.3f} \* X33^2 = y"  
 .format(beta[0], beta[1], beta[2], beta[3], beta[4], beta[5], beta[6], beta[7], beta[8], beta[9], beta[10]))  
y\_i = [0] \* n  
print("")  
print("Експериментальні значення")  
for k in range(n):  
 y\_i[k] = beta[0] + beta[1] \* list\_for\_a[k][0] + beta[2] \* list\_for\_a[k][1] + beta[3] \* list\_for\_a[k][2] + \  
 beta[4] \* list\_for\_a[k][3] + beta[5] \* list\_for\_a[k][4] + beta[6] \* list\_for\_a[k][5] + beta[7] \* \  
 list\_for\_a[k][6] + beta[8] \* list\_for\_a[k][7] + beta[9] \* list\_for\_a[k][8] + beta[10] \* list\_for\_a[k][9]  
for i in range(n):  
 print("{:.3f}".format(y\_i[i]))  
print("")  
  
print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка за критерієм Кохрена"))  
Gp = max(dispersions) / sum(dispersions)  
GT = 0.3346  
print("Gp =", Gp)  
if Gp < GT:  
 print("\033[1m\033[30m\033[42mGp={} < GT={} – Дисперсія однорідна\033[0m".format(round(Gp, 3), GT))  
else:  
 print("\033[1m\033[30m\033[41m{}\033[0m".format("Дисперсія неоднорідна"))  
print("")  
print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка значущості коефіцієнтів за критерієм Стьюдента"))  
sb = sum(dispersions) / len(dispersions)  
sbs = (sb / (n \* m)) \*\* 0.5  
  
F3 = (m - 1) \* n  
coef\_1 = []  
coef\_2 = []  
d = 11  
res = [0] \* 11  
for j in range(11):  
 pract\_t = 0  
 for i in range(15):  
 if j == 0:  
 pract\_t += aver\_y[i] / 15  
 else:  
 pract\_t += aver\_y[i] \* xn[i][j - 1]  
 res[j] = beta[j]  
 if fabs(pract\_t / sbs) < t.ppf(q=0.975, df=F3):  
 coef\_2.append(beta[j])  
 res[j] = 0  
 d -= 1  
 else:  
 coef\_1.append(beta[j])  
print("Значущі коефіцієнти регресії", [round(i, 3) for i in coef\_1])  
print("Незначущі коефіцієнти регресії", [round(i, 3) for i in coef\_2])  
st\_y = []  
for i in range(n):  
 st\_y.append(res[0] + res[1] \* x1[i] + res[2] \* x2[i] + res[3] \* x3[i] + res[4] \* x1x2[i] + res[5] \*  
 x1x3[i] + res[6] \* x2x3[i] + res[7] \* x1x2x3[i] + res[8] \* x1kv[i] + res[9] \*  
 x2kv[i] + res[10] \* x3kv[i])  
print("")  
  
print("Значення з отриманими коефіцієнтами")  
for i in range(n):  
 print("{:.3f}".format(st\_y[i]))  
print("")  
  
print("\033[1m\033[30m\033[43m{}\033[0m".format("Перевірка адекватності за критерієм Фішера"))  
Sad = m \* sum([(st\_y[i] - aver\_y[i]) \*\* 2 for i in range(n)]) / (n - d)  
Fp = Sad / sb  
F4 = n - d  
print("Fp =", Fp)  
if Fp < f.ppf(q=0.95, dfn=F4, dfd=F3):  
 print("\033[1m\033[30m\033[42m{}\033[0m".format("При рівні значимості 0.05 рівняння регресії адекватне"))  
else:  
 print("\033[1m\033[30m\033[41m{}\033[0m".format("При рівні значимості 0.05 рівняння регресії неадекватне"))

**Результат роботи програми**

