­­­Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Методи оптимізації та планування

Лабораторна робота №3

**«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»**

Виконав:

студент групи ІВ-81

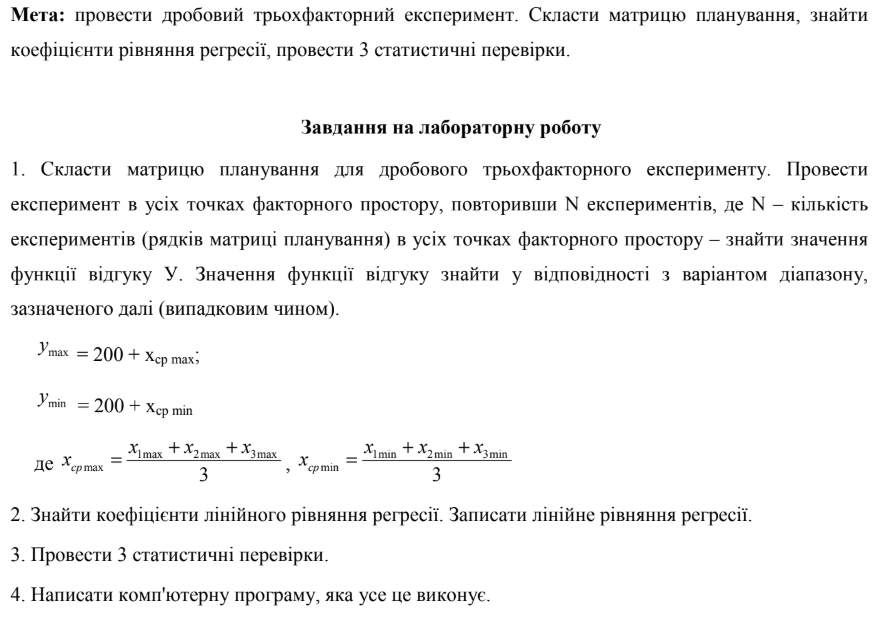
Богданенко В.Ю.

Перевірив:

ас. Регіда П. Г.

Київ

2020 р.



**Варіант завдання:**



**Лістинг програми:**

**import** random  
**import** numpy **as** np  
**from** math **import** sqrt  
**def** lab3():  
 x1min = -20  
 x1max = 30  
 x2min = 30  
 x2max = 80  
 x3min = 30  
 x3max = 45  
 xsmax = (x1max + x2max + x3max) / 3  
 xsmin = (x1min + x2min + x3min) / 3  
 ymax = 200 + xsmax  
 ymin = 200 + xsmin  
 *# print(ymax)  
 # print(ymin)* fisher = [5.3, 4.5, 4.1, 3.8]  
 matrixa = [[x1min, x2min, x3min], [x1min, x2max, x3max], [x1max, x2min, x3max], [x1max, x2max, x3min]]  
 **for** i **in** range(3):  
 **for** j **in** range(len(matrixa)):  
 matrixa[j].append(random.randint(int(ymin), int(ymax)))  
 *# matrixa=[[-25,5,15,15,18,16],[-25,40,25,10,19,13],[75,5,25,11,14,12],[75, 40, 15, 16, 19, 16]]* matrixx1 = [[1, -1, -1, -1], [1, -1, 1, 1], [1, 1, -1, 1], [1, 1, 1, -1]]  
 ynlist = []  
 **for** i **in** range(len(matrixa)):  
 yn = 0  
 **for** j **in** range(3, len(matrixa[0])):  
 yn = yn + matrixa[i][j]  
 ynlist.append(yn / 3)  
 mxlist = []  
 **for** j **in** range(3):  
 mx = 0  
 **for** i **in** range(len(matrixa)):  
 mx = mx + matrixa[i][j]  
 mxlist.append(mx / 4)  
 my = 0  
 **for** i **in** range(len(ynlist)):  
 my = my + ynlist[i]  
 my = my / 4  
 a = []  
 **for** i **in** range(3):  
 an = 0  
 **for** j **in** range(len(ynlist)):  
 an = an + ynlist[j] \* matrixa[j][i]  
 an = an / 4  
 a.append(an)  
 apow2 = []  
 **for** i **in** range(3):  
 apow2n = 0  
 **for** j **in** range(len(matrixa)):  
 apow2n = apow2n + matrixa[j][i] \*\* 2  
 apow2n = apow2n / 4  
 apow2.append(apow2n)  
 a12 = 0  
 a23 = 0  
 a13 = 0  
 **for** i **in** range(len(matrixa)):  
 a12 = a12 + matrixa[i][0] \* matrixa[i][1]  
 a13 = a13 + matrixa[i][0] \* matrixa[i][2]  
 a23 = a23 + matrixa[i][2] \* matrixa[i][1]  
 a12 = a12 / 4  
 a23 = a23 / 4  
 a13 = a13 / 4  
 matrix0 = [[my, mxlist[0], mxlist[1], mxlist[2]], [a[0], apow2[0], a12, a13], [a[1], a12, apow2[1], a23], [a[2], a12, a23, apow2[2]]]  
 matrix1 = [[1, my, mxlist[1], mxlist[2]], [mxlist[0], a[0], a12, a13], [mxlist[1], a[1], apow2[1], a23], [mxlist[2], a[2], a23, apow2[2]]]  
 matrix2 = [[1, mxlist[0], my, mxlist[2]], [mxlist[0], apow2[0], a[0], a13], [mxlist[1], a12, a[1], a23], [mxlist[2], a12, a[2], apow2[2]]]  
 matrix3 = [[1, mxlist[0], mxlist[1], my], [mxlist[0], apow2[0], a12, a[1]], [mxlist[1], a12, apow2[1], a[1]], [mxlist[2], a12, a23, a[2]]]  
 matrix4 = [[1, mxlist[0], mxlist[1], mxlist[2]], [mxlist[0], apow2[0], a12, a13], [mxlist[1], a12, apow2[1], a23], [mxlist[2], a12, a23, apow2[2]]]  
 det0 = np.linalg.det(matrix0)  
 det1 = np.linalg.det(matrix1)  
 det2 = np.linalg.det(matrix2)  
 det3 = np.linalg.det(matrix3)  
 det4 = np.linalg.det(matrix4)  
 b0 = det0 / det4  
 b1 = det1 / det4  
 b2 = det2 / det4  
 b3 = det3 / det4  
 blist = [str(**'%.3f'** % b0), **" + "** + str(**'%.3f'** % b1), **" + "** + str(**'%.3f'** % b2), **" + "** + str(**'%.3f'** % b3)]  
 b1list = [b0, b1, b2, b3]  
 text0 = **"y = "** + str(**'%.3f'** % b0) + **" + "** + str(**'%.3f'** % b1) + **"\*X1 + "** + str(**'%.3f'** % b2) + **"\*X2 + "** + str(**'%.3f'** % b3) + **"\*X3"** print(**' X1 |'**, **'X2 |'**, **'X3 |'**, **' Y1 |'**, **' Y2 |'**, **' Y3'**)  
 print(matrixa[0][0],**'|'**, matrixa[0][1],**'|'**, matrixa[0][2],**'|'**, matrixa[0][3],**'|'**, matrixa[0][4],**'|'**, matrixa[0][5])  
 print(matrixa[1][0],**'|'**, matrixa[1][1],**'|'**, matrixa[1][2],**'|'**, matrixa[1][3],**'|'**, matrixa[1][4],**'|'**, matrixa[1][5])  
 print(matrixa[2][0],**' |'**, matrixa[2][1],**'|'**, matrixa[2][2],**'|'**, matrixa[2][3],**'|'**, matrixa[2][4],**'|'**, matrixa[2][5])  
 print(matrixa[3][0],**' |'**, matrixa[3][1],**'|'**, matrixa[3][2],**'|'**, matrixa[3][3],**'|'**, matrixa[3][4],**'|'**, matrixa[3][5])  
 print(text0)  
 *# КОХРЕНА* S2y1=0  
 S2y2=0  
 S2y3 = 0  
 S2y4 = 0  
 **for** i **in** range(3, len(matrixa[0])):  
 S2y1 = S2y1 + (matrixa[0][i] - ynlist[0]) \*\* 2  
 S2y2 = S2y2 + (matrixa[1][i] - ynlist[1]) \*\* 2  
 S2y3 = S2y3 + (matrixa[2][i] - ynlist[2]) \*\* 2  
 S2y4 = S2y4 + (matrixa[3][i] - ynlist[3]) \*\* 2  
 S2y1 = S2y1 / 3  
 S2y2 = S2y2 / 3  
 S2y3 = S2y3 / 3  
 S2y4 = S2y4 / 3  
 Gp = max(S2y1, S2y2, S2y3, S2y4) / (S2y1 + S2y2 + S2y3 + S2y4)  
 text1 = **""  
 if** Gp < 0.7679:  
 text1 = **"Дисперсія однорідна"  
 else**:  
 text1 = **"Дисперсія неоднорідна"** print(text1)  
 S2b = (S2y4 + S2y3 + S2y2 + S2y1) / 4  
 S2B = S2b / 12  
 SB = sqrt(S2B)  
 Blist = []  
 **for** j **in** range(4):  
 bn = 0  
 **for** i **in** range(len(matrixx1)):  
 bn = bn + ynlist[i] \* matrixx1[i][j]  
 bn = bn / 4  
 Blist.append(bn)  
 tlist = []  
 **for** i **in** range(len(Blist)):  
 tlist.append(abs(Blist[i]) / SB)  
 tlist1 = []  
 d = 0  
 **for** i **in** range(len(tlist)):  
 **if** tlist[i] >= 2.306:  
 tlist1.append(1)  
 d = d + 1  
 **else**:  
 tlist1.append(0)  
 matrixrez = []  
 **for** i **in** range(4):  
 matrixrez.append(matrixx1[i] + matrixa[i][3:6])  
 tlist2 = [**" "**, **"\*X1"**, **"\*X2"**, **"\*X3"**]  
 text2 = **"y = "  
 for** i **in** range(len(tlist2)):  
 **if** tlist1[i] == 1:  
 text2 = text2 + (blist[i]) + (tlist2[i])  
 yslist = []  
 **for** j **in** range(4):  
 ysn = 0  
 **if** tlist1[0] == 1:  
 ysn = b0  
 **for** i **in** range(3):  
 **if** tlist1[i + 1] == 1:  
 ysn = ysn + b1list[i + 1] \* matrixa[j][i]  
 yslist.append(ysn)  
 f4 = 4 - d  
 f3 = 8  
 S2ad = 0  
 **for** i **in** range(4):  
 S2ad = S2ad + (yslist[i] - ynlist[i]) \*\* 2  
 *# print(yslist[i],ynlist[i])* S2ad=S2ad\*3/f4  
 Fp = S2ad / S2B  
 text3 = **""  
  
 if** Fp < fisher[f4 - 1]:  
 text3 = **"Рівняння регресії адикватне оригіналу"  
 else**:  
 text3 = **"Рівняння регресії неадикватне оригіналу"** print(text2)  
 print(text3)  
print(lab3())

**Відповіді на контрольні питання:**

* 1. Що називається дробовим факторним експериментом?

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.

* 1. Для чого потрібно розрахункове значення Кохрена?

Розрахункове значення Кохрена викорстовується для порівння з теоретичним значенням, і якщо воно менше теоретичного, то гіпотеза про однорідність дисперсій приймається.

* 1. Для чого перевіряється критерій Стьюдента?

Якщо теоретичний коефіцієнт bі = 0, це означає , що в апроксимуючому поліномі відповідний доданок ( фактор ) відсутній . Чим менше значення b i , тим менше вплив відповідного фактора . Якщо виконується нерівність ts < tтабл , то приймається нуль — г іпотеза , тобто вважається , що знайдений коефіцієнт β s 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?

Для перевірки адекватності досліджуваного об’єкта 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.

Знайдене шляхом розрахунку Fр порівнюють з табличним значенням Fт , що визначається при рівні значимості q та кількості ступенів свободи f4 = N - d і

f3 = f1 \* f2. Якщо Fр < Fт то отримана математична модель з прийнятим рівнем статистичної значимості q адекватна експериментальним даним .

**Результат виконання роботи:**

