НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського”

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 4

з дисципліни "Методи оптимізації та планування експерименту" на тему:

ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ ПРИ ВИКОРИСТАННЯМ РІВНЯННЯ РЕГРЕСІЇ З УРАХУВАННЯМ ЕФЕКТУ ВЗАЄМОДІЇ

Виконала Дуплій Наталя Володимирівна

Факультет ІОТ,

Група ІВ-82

Залікова книжка № ІВ-8209

Київ – 2020 р.

**Варіант**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №варіанта | X1 | | X2 | | X3 | |
| min | max | min | max | min | max |
| 208 | -5 | 15 | -35 | 10 | -35 | -10 |

**Короткі теоретичні відомості**

Рівняння регресії ПФЕ з урахуванням ефекту взаємодії для трьох факторів має вигляд:

Це рівняння повинне відображати з певною точністю значення функції відгуку для значень факторів, які знаходяться у певних межах, і зокрема, для точок плану:

**Результати підготовки та виконання роботи**

Напишемо рівняння регресії з ефектом взаємодії:

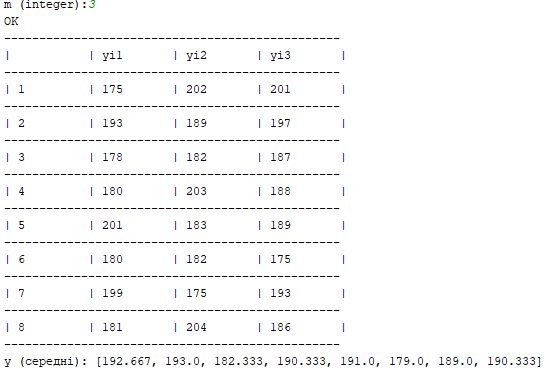
Інтервали значень :

Значення, відповідні кодованим:

Складаємо матрицю планування для повного трьохфакторного експерименту з використанням додаткового нульового чинника() і заповнюємо таблицю кодованими значеннями .

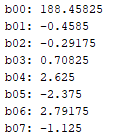
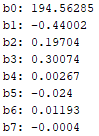
Обираємо m=3

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | +1 | -1 | -1 | -1 | +1 | +1 | +1 | -1 |  |  |  |  |  |
| **2** | +1 | -1 | +1 | +1 | -1 | -1 | +1 | -1 |  |  |  |  |  |
| **3** | +1 | +1 | -1 | +1 | -1 | +1 | -1 | -1 |  |  |  |  |  |
| **4** | +1 | +1 | +1 | -1 | +1 | -1 | -1 | -1 |  |  |  |  |  |
| **5** | +1 | -1 | -1 | +1 | +1 | -1 | -1 | +1 |  |  |  |  |  |
| **6** | +1 | -1 | +1 | -1 | -1 | +1 | -1 | +1 |  |  |  |  |  |
| **7** | +1 | +1 | -1 | -1 | -1 | -1 | +1 | +1 |  |  |  |  |  |
| **8** | +1 | +1 | +1 | +1 | +1 | +1 | +1 | +1 |  |  |  |  |  |

Проводимо експеримент у всіх точках плану

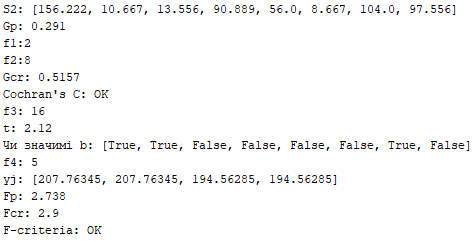
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **1** | +1 | -1 | -1 | -1 | +1 | +1 | +1 | -1 | 175 | 202 | 201 | 192.667 | 156.222 |
| **2** | +1 | -1 | +1 | +1 | -1 | -1 | +1 | -1 | 193 | 189 | 197 | 193.0 | 10.667 |
| **3** | +1 | +1 | -1 | +1 | -1 | +1 | -1 | -1 | 178 | 182 | 187 | 182.333 | 13.556 |
| **4** | +1 | +1 | +1 | -1 | +1 | -1 | -1 | -1 | 180 | 203 | 188 | 190.333 | 90.889 |
| **5** | +1 | -1 | -1 | +1 | +1 | -1 | -1 | +1 | 201 | 183 | 189 | 191.0 | 56.0 |
| **6** | +1 | -1 | +1 | -1 | -1 | +1 | -1 | +1 | 180 | 182 | 175 | 179.0 | 8.667 |
| **7** | +1 | +1 | -1 | -1 | -1 | -1 | +1 | +1 | 199 | 175 | 193 | 189.0 | 104.0 |
| **8** | +1 | +1 | +1 | +1 | +1 | +1 | +1 | +1 | 181 | 204 | 186 | 190.333 | 97.556 |

Середні значення функції відгуку та коефіцієнти рівняння регресії знайдені за допомогою програми.



Рівняння регресії:

Проводимо статистичні перевірки.



Дисперсія однорідна за критерієм Кохрена. Коефіцієнти b0, b1, b6 значимі, b2, b3, b4, b5, b7 незначимі за критерієм Стьюдента. Модель адекватна за критерієм Фішера.

Скореговане рівняння регресії:

**Висновки**

Проведено повний трьохфакторний експеримент. Знайдено рівняння регресії адекватне об'єкту.

**Лістинг програми**

**import** random  
**import** numpy  
**import** copy  
**from** scipy.stats **import** t  
**from** scipy.stats **import** f  
**import** math  
  
**def** cochran(f1, f2, q):  
 fish = f.isf(q/f2, f1, (f2 - 1)\*f1)  
 result = fish/(fish + f2 - 1)  
 **return** result  
  
**def** part2(yi, x, y):  
 **global** b, m  
 k = len(x[0])  
 mx = [[] **for** i **in** range(k)]  
 mx[0].append(k)  
 **for** i **in** range(1, k):  
 suma = sum(x[i])  
 mx[0].append(suma)  
 mx[i].append(suma)  
 **for** j **in** range(1, k):  
 mx[i].append(sum([x[i][l] \* x[j][l] **for** l **in** range(k)]))  
  
 det = numpy.linalg.det(mx)  
 delta = round(det, 3)  
  
 my = [sum(yi)]  
 **for** i **in** range(1, k):  
 my.append(round(sum([yi[j]\*x[i][j] **for** j **in** range(k)]), 5))  
  
 b = [copy.deepcopy(mx) **for** i **in** range(k+1)]  
 **for** i **in** range(k):  
 **for** j **in** range(k):  
 b[i][j][i] = my[j]  
 b[i] = round(numpy.linalg.det(b[i])/delta, 5)  
  
 print(**"b"** + str(i) + **": "** + str(b[i]))  
  
 x0 = [[-1, -1, 1, 1, -1, -1, 1, 1],  
 [-1, 1, -1, 1, -1, 1, -1, 1],  
 [-1, 1, 1, -1, 1, -1, -1, 1]]  
 x0.append([x0[0][i] \* x0[1][i] **for** i **in** range(k)])  
 x0.append([x0[0][i] \* x0[2][i] **for** i **in** range(k)])  
 x0.append([x0[1][i] \* x0[2][i] **for** i **in** range(k)])  
 x0.append([x0[0][i] \* x0[1][i] \* x0[2][i] **for** i **in** range(k)])  
  
 b0 = [round(sum(yi)/k, 5)]  
 print(**"b0"** + str(0) + **": "** + str(b0[0]))  
 **for** i **in** range(1, k):  
 b0.append(round(sum([yi[j] \* x0[i-1][j]/k **for** j **in** range(k)]), 5))  
 print(**"b0"** + str(i) + **": "** + str(b0[i]))  
  
 S2 = []  
 **for** i **in** range(len(y)):  
 S2.append(sum([(y[i][j] - yi[i])\*\*2 **for** j **in** range(len(y[i]))]))  
 S2[i] = round(S2[i]/len(y[i]), 3)  
 print(**"S2: "** + str(S2))  
  
 Gp = round(max(S2)/sum(S2), 3)  
 print(**"Gp: "** + str(Gp))  
  
 f1 = m - 1  
 f2 = k  
  
 print(**"f1:"** + str(f1))  
 print(**"f2:"** + str(f2))  
  
 alpha = 0.05  
  
 Gcr = round(cochran(f1, f2, alpha), 4)  
 print(**"Gcr: "** + str(Gcr))  
 **if** Gp < Gcr:  
 print(**"Cochran's C: OK"**)  
 **else**:  
 print(**"Cochran's C: :("**)  
 m += 1  
 **return** part1(x)  
  
 S2v = sum(S2) / 4  
  
 S2b = round(S2v / (4 \* m), 3)  
 Sb = round(math.sqrt(S2b), 3)  
  
 f3 = f1 \* f2  
 print(**"f3: "** + str(f3))  
 tcr = round(t.ppf(1 - alpha / 2, df=f3), 3)  
 print(**"t: "** + str(tcr))  
 bs = []  
 ts = []  
 d = 0  
 **for** i **in** range(k):  
 bs.append(round(sum([yi[j] \* x[i][j] **for** j **in** range(4)]) / 4, 3))  
 ts.append(round(bs[i] / Sb, 3))  
 **if** ts[i] > tcr:  
 ts[i] = **True** d += 1  
 **else**:  
 ts[i] = **False** print(**"Чи значимі b: "** + str(ts))  
  
 f4 = k - d  
 print(**"f4: "** + str(f4))  
 x = [[-30, -30, 0, 0],  
 [10, 60, 10, 60],  
 [10, 35, 35, 10]]  
 yj = []  
 **for** i **in** range(4):  
 yj.append(0)  
 **for** j **in** range(4):  
 **if** ts[j]:  
 **if** j == 0:  
 yj[i] += b[0]  
 **else**:  
 yj[i] += b[j] \* x[j - 1][i]  
 print(**"yj: "** + str(yj))  
  
 S2ad = round(m \* sum([(yj[i] - yi[i]) \*\* 2 **for** i **in** range(4)]) / f4, 3)  
  
 Fp = round(S2ad / S2v, 3)  
 print(**"Fp: "** + str(Fp))  
 Fcr = round(f.ppf(1 - alpha, f4, f3), 1)  
 print(**"Fcr: "** + str(Fcr))  
 **if** Fp < Fcr:  
 print(**"F-criteria: OK"**)  
 **else**:  
 print(**"F-criteria: :("**)  
  
**def** print\_line(m):  
 print(**"-"** \* 12 \* (m+1))  
  
  
**def** part1(x):  
 print\_line(m)  
 print(**"| "** + **'{:<10}'**.format(**""**), end=**""**)  
 **for** i **in** range(1, m+1):  
 print(**"| "** + **'{:<10}'**.format(**"yi"** + str(i)), end=**""**)  
 print(**"|"**)  
 print\_line(m)  
  
 y = []  
  
 **for** j **in** range(1, k+1):  
 y.append([])  
 print(**"| "** + **'{:<10}'**.format(j), end=**""**)  
 **for** i **in** range(1, m+1):  
 r = round(random.random() \* (max\_num - min\_num) + min\_num)  
 y[j-1].append(r)  
 print(**"| "** + **'{:<10}'**.format(r), end=**""**)  
 print(**"|"**)  
 print\_line(m)  
  
 yi = []  
 **for** i **in** range(k):  
 yi.append(round(1/m \* sum(y[i]), 3))  
  
 print(**"y (середні): "** + str(yi))  
  
 part2(yi, x, y)  
  
max\_num = 205  
min\_num = 175  
x = [[1, 1, 1, 1, 1, 1, 1, 1],  
 [-5, -5, 15, 15, -5, -5, 15, 15],  
 [-35, 10, -35, 10, -35, 10, -35, 10],  
 [-35, -10, -10, -35, -10, -35, -35, -10]]  
k = len(x[0])  
x.append([x[1][i] \* x[2][i] **for** i **in** range(k)])  
x.append([x[1][i] \* x[3][i] **for** i **in** range(k)])  
x.append([x[2][i] \* x[3][i] **for** i **in** range(k)])  
x.append([x[1][i] \* x[2][i] \* x[3][i] **for** i **in** range(k)])  
**while True**:  
 m = input(**"m (integer):"**)  
 **if** m.isnumeric():  
 print(**"OK"**)  
 m = int(m)  
 **break  
 else**:  
 print(**"m must be integer"**)  
  
part1(x)

**Результати виконання**

