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 №5

«Проведення трьохфакторного експерименту при використанні рівняння регресії з урахуванням квадратичних членів (центральний ортогональний композиційний план)»

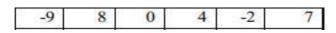
Виконав: студен групи IB-82 Мотора В. С. Залікова книжка No IB-8217 Номер у списку - 015 Перевірив Регіда П.Г.

Київ 2020 р.

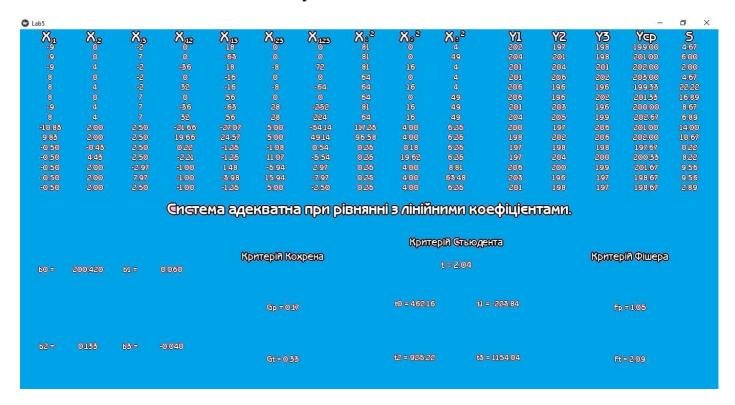
Mema:

Провести трьохфакторний експеримент з урахуванням квадратичних членів ,використовуючи центральний ортогональний композиційний план. Знайти рівняння регресії, яке буде адекватним для опису об'єкту.

Варіант 232



Результат виконання



Лістинг програми:

```
def first(y,m):
    average = []
    dispertion = []
    for i in range(15):
        average.append(sum(y[i]) / m)
        dispertion_current = 0
        for k in range(m):
            dispertion_current += (average[i] - y[i][k]) ** 2
            dispertion.append(dispertion_current / m)
        return average, dispertion
        def kohren(dispertion,f1,f2):
```

```
Gp = max(dispertion) / sum(dispertion)
  Gt = Criteries.get_cohren_value(f2, f1, 0.05)
  return Gp, Gt
def add_y(y,m):
  m += 1
  for i in range(15):
     y[i].append(randint(ymin,ymax))
  return y,m
def start():
  y = [[randint(ymin,ymax) for i in range(m)] for j in range(15)]
  average, dispertion = first(y,m)
  return y, average, dispertion
def get_nat_table(x_k):
  x_n = []
  for i in range(len(x_k)):
     temp = [1,]
     for j in range(3):
       if j == 0:
          if x_k[i][j] == -1:
            temp.append(x_nat['x1min'])
          else:
            temp.append(x_nat['x1max'])
       if j == 1:
          if x_k[i][j] == -1:
            temp.append(x_nat['x2min'])
          else:
            temp.append(x_nat['x2max'])
       if j == 2:
          if x_k[i][j] == -1:
            temp.append(x_nat['x3min'])
          else:
            temp.append(x_nat['x3max'])
     x_n.append(temp)
```

```
x_n[i].append(temp[1] * temp[2])
  x_n[i].append(temp[1] * temp[3])
  x_n[i].append(temp[2] * temp[3])
  x_n[i].append(temp[1] * temp[2] * temp[3])
  x_n[i].append(temp[1]**2)
  x_n[i].append(temp[2]**2)
  x_n[i].append(temp[3]^{**2})
I = 1.215
x01 = (x_nat['x1min'] + x_nat['x1max']) / 2
x02 = (x_nat['x2min'] + x_nat['x2max']) / 2
x03 = (x_nat['x3min'] + x_nat['x3max']) / 2
deltax1 = x_nat['x1max'] - x01
deltax2 = x_nat['x2max'] - x02
deltax3 = x_nat['x3max'] - x03
x_zor = [[1,-1*deltax1+x01, x02, x03],
      [1, I*deltax1+x01, x02, x03],
      [1, x01,-l*deltax2+x02, x03],
      [1, x01, I*deltax2+x02, x03],
      [1, x01, x02,-l*deltax3+x03],
      [1, x01, x02, I*deltax3+x03],
      [1, x01, x02, x03]]
for i in range(7):
   x_zor[i].append(x_zor[i][1] * x_zor[i][2])
   x_zor[i].append(x_zor[i][1] * x_zor[i][3])
  x_zor[i].append(x_zor[i][2] * x_zor[i][3])
  x_zor[i].append(x_zor[i][1] * x_zor[i][2] * x_zor[i][3])
  x_zor[i].append(x_zor[i][1] ** 2)
  x_zor[i].append(x_zor[i][2] ** 2)
  x_zor[i].append(x_zor[i][3] ** 2)
x_n += x_zor
return x_n
```

```
coefs_nat = lstsq(x_n,average,rcond=None)[0]
return coefs_nat

def fisher(m, d, sum_st, y_average, S2b, f1,f2):
    Sad = 0
    for i in range(15):
        Sad += (sum_st[i] - y_average[i]) ** 2
    Sad *= m / (15 - d)
    Fp = Sad / S2b
    f3 = f1 * f2
    f4 = f2 - d
    Ftable = Criteries.get_fisher_value(f3, f4, 0.05)
    return Fp, Ftable
```

Висновок:

Був проведений трьохфакторний експеримент з урахуванням квадратичних членів з використанням центрального ортогонального композиційного плану.

Створена програма працює!