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»

на тему «Проведення трьохфакторного експерименту при використанні рівняння регресії з квадратичними членами.»

Виконав:
студент 2 курсу
групи ІВ-91
Охочий Р.О.
Номер у списку групи: 21

ПЕРЕВІРИВ:
ас. Регіда П.Г.

Київ - 2021

Хід роботи

Мета: Провести трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи рототабельний композиційний план.

Завдання:

Завдання до лабораторної роботи:

1. Ознайомитися з теоретичними відомостями.
2. Вибрати з таблиці варіантів і записати в протокол інтервали значень x_1, x_2, x_3. Обчислити і записати значення, відповідні кодованим значенням факторів +1; -1;+ l; - l; 0 для $\bar{x}_1$, $\bar{x}_2$, $\bar{x}_3$.
3. Значення функції відгуку знайти за допомогою підстановки в формулу:

$$y_i = f(x_1, x_2, x_3) + random(10) - 5,$$

де $f(x_1, x_2, x_3)$ 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.

121	10	40	-30	45	-30	-10	5,9+4,0*x1+3,5*x2+8,2*x3+4,3*x1*x1+0,7*x2*x2+9,3*x3*x3+6,1*x1*x2+0,5*x1*x3+8,6*x2*x3+3,1*x1*x2*x3

Код програми

```
from numpy.linalg import solve
from scipy.stats import f, t
import math
import random
import numpy as np



def Lab6(k):
  m = k
  n = 15
  d = 11

  x1min = 10
  x1max = 40
  x2min = -30
  x2max = 45
  x3min = -30
  x3max = -10

  print("x1 min =", x1min, ", x1 max = ", x1max)
  print("x2 min =", x2min, ", x2 max = ", x2max)
  print("x3 min =", x3min, ", x3 max = ", x3max)
  print("-------------------------------------------------------------------------------
")



  x01 = (x1max + x1min) / 2
  x02 = (x2max + x2min) / 2
  x03 = (x3max + x3min) / 2
  deltax1 = x1max - x01
  deltax2 = x2max - x02
  deltax3 = x3max - 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 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max, -1.73 * deltax1 + x01, 1.73
* deltax1 + x01, x01, x01, x01, x01, x01]
  x2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max, x02, x02, -1.73 * deltax2 +
x02, 1.73 * deltax2 + x02, x02, x02, x02]
  x3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max, x03, x03, x03, x03, -1.73 *
deltax3 + x03, 1.73 * deltax3 + 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_A = list(zip(x1, x2, x3, x1x2, x1x3, x2x3, x1x2x3, x1kv, x2kv, x3kv))

  print("Матриця планування з натуралізованими коефіцієнтами X:")
  for i in range(n):
      print(end=' ')
      for j in range(len(List_A[0])):
          print(round(List_A[i][j], 3), end='    ')
      print()
  print("----------------------------------------------------------------------------------
------------")



  def function(X1, X2, X3):
    y = 5.9 + 4 * X1 + 3.5 * X2 + 8.2 * X3 + 4.3 * X1 * X1 + 0.7 * X2 * X2 + 9.3 * X3 * X3 +
6.1 * X1 * X2 + 0.5 * X1 * X3 + 8.6 * X2 * X3 + 3.1 * X1 * X2 * X3 + random.randrange(0, 10)
- 5
    return y
  Y = [[function(List_A[j][0], List_A[j][1], List_A[j][2]) for i in range(m)] for j in
range(15)]


  print("Матриця планування Y:")
  for i in range(n):
      print(end=' ')
      for j in range(len(Y[0])):
          print(round(Y[i][j], 3), end='    ')
```

```
      print()
  print("-----------------------------------------------------------------------------------
------------")



  Y_average = []
  for i in range(len(Y)):
      Y_average.append(np.mean(Y[i], axis=0))
  print("Середні значення відгуку по рядкам:")
  print(Y_average)
  print("-----------------------------------------------------------------------------------
------------")



  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_known(num):
    a = 0
    for j in range(n):
        a += Y_average[j] * List_A[j][num - 1] / n
    return a

  def a(first, second):
    a = 0
    for j in range(n):
        a += List_A[j][first - 1] * List_A[j][second - 1] / n
    return a




  my = sum(Y_average) / n
  mx = []
  for i in range(10):
      number_lst = []
      for j in range(n):
          number_lst.append(List_A[j][i])
      mx.append(sum(number_lst) / len(number_lst))

  det1 = [[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)]]

  det2 = [my, find_known(1), find_known(2), find_known(3), find_known(4), find_known(5),
find_known(6), find_known(7), find_known(8), find_known(9), find_known(10)]
  beta = solve(det1, det2)


  print("Рівняння регресії:")
  print("{:.3f} + {:.3f} * X1 + {:.3f} * X2 + {:.3f} * X3 + {:.3f} * X1X2 + {:.3f} * X1X3 +
{:.3f} * X2X3+ {:.3f} * X1X2X3 + {:.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_A[k][0] + beta[2] * List_A[k][1] + beta[3] *
List_A[k][2] + \
               beta[4] * List_A[k][3] + beta[5] * List_A[k][4] + beta[6] * List_A[k][5] +
beta[7] * \
               List_A[k][6] + beta[8] * List_A[k][7] + beta[9] * List_A[k][8] + beta[10] *
List_A[k][9]
  print(y_i)
  print("-------------------------------------------------------------------------------
------------")


  print("Перевірка однорідності за критерієм Кохрена:")
  Gp = max(dispersions) / sum(dispersions)
  Gt = 0.3346
  print("Gp =", Gp)
  if Gp < Gt:
    print("Дисперсія однорідна")
    print("-------------------------------------------------------------------------------
--------------")
  else:
    print("Дисперсія неоднорідна")
    print("-------------------------------------------------------------------------------
--------------")

  print("Оцінка значимості коефіцієнтів за критерієм Стюдента")
  sb = sum(dispersions) / len(dispersions)
  sbs = (sb / (n * m)) ** 0.5
  F3 = (m - 1) * n
  coefs1 = []
  coefs2 = []

  res = [0] * 11
  for j in range(11):
      t_pract = 0
      for i in range(15):
          if j == 0:
              t_pract += Y_average[i] / 15
          else:
              t_pract += Y_average[i] * xn[i][j - 1]
          res[j] = beta[j]
```

```
      if math.fabs(t_pract / sbs) < t.ppf(q=0.975, df=F3):
          coefs2.append(beta[j])
          res[j] = 0
          d-=1
      else:
          coefs1.append(beta[j])

  print("Значущі коефіцієнти:", [round(i, 3) for i in coefs1])
  print("----------------------------------------------------------------------------------------
------------")
  print("Незначущі коефіцієнти:", [round(i, 3) for i in coefs2])
  print("----------------------------------------------------------------------------------------
------------")

  y_st = []
  for i in range(n):
      y_st.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("Значення з отриманими коефіцієнтами:")
  print(y_st)
  print("----------------------------------------------------------------------------------------
------------")

  print("Перевірка на адекватність за критерієм Фішера")
  Sad = m * sum([(y_st[i] - Y_average[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("Рівняння регресії адекватне при рівні значимості 0.05")
    print("----------------------------------------------------------------------------------------
--------------")
  else:
    print("Рівняння регресії неадекватне при рівні значимості 0.05")
    print("----------------------------------------------------------------------------------------
--------------")

Lab6(3)
```

Результати роботи програми

```
x1 min = 10 , x1 max =  40
x2 min = -30 , x2 max =  45
x3 min = -30 , x3 max =  -10
---------------------------------------------------------------------------
Матриця планування з натуралізованими коефіцієнтами X:
 10   -30   -30   -300   -300   900   9000   100   900   900
 10   -30   -10   -300   -100   300   3000   100   900   100
 10   45   -30   450   -300   -1350   -13500   100   2025   900
 10   45   -10   450   -100   -450   -4500   100   2025   100
 40   -30   -30   -1200   -1200   900   36000   1600   900   900
 40   -30   -10   -1200   -400   300   12000   1600   900   100
 40   45   -30   1800   -1200   -1350   -54000   1600   2025   900
 40   45   -10   1800   -400   -450   -18000   1600   2025   100
 -0.95   7.5   -20.0   -7.125   19.0   -150.0   142.5   0.902   56.25   400.0
 50.95   7.5   -20.0   382.125   -1019.0   -150.0   -7642.5   2595.903   56.25   400.0
 25.0   -57.375   -20.0   -1434.375   -500.0   1147.5   28687.5   625.0   3291.891   400.0
 25.0   72.375   -20.0   1809.375   -500.0   -1447.5   -36187.5   625.0   5238.141   400.0
 25.0   7.5   -37.3   187.5   -932.5   -279.75   -6993.75   625.0   56.25   1391.29
 25.0   7.5   -2.7   187.5   -67.5   -20.25   -506.25   625.0   56.25   7.29
 25.0   7.5   -20.0   187.5   -500.0   -150.0   -3750.0   625.0   56.25   400.0
```

```
Матриця планування Y:
 42781.9   42785.9   42788.9
 11849.9   11851.9   11848.9
 -40691.1   -40692.1   -40686.1
 -12225.1   -12228.1   -12222.1
 127118.9   127118.9   127114.9
 40674.9   40674.9   40677.9
 -151887.1   -151882.1   -151889.1
 -39425.1   -39422.1   -39423.1
 2747.393   2740.393   2745.393
 -8164.582   -8162.582   -8162.582
 98256.973   98251.973   98254.973
 -103573.152   -103571.152   -103575.152
 -7914.813   -7921.813   -7920.813
 2268.157   2273.157   2270.157
 -5603.225   -5603.225   -5610.225
```

```
Середні значення відгуку по рядкам:
[42785.56666666667, 11850.233333333332, -40689.76666666666, -12225.1, 127117.56666666665, 40675.9, -151886.1, -39423.433
33333333, 2744.3932499999996, -8163.2484166666645, 98254.64010416665, -103573.1515625, -7919.146333333333, 2270.49033333
33336, -5605.558333333334]
```

```
Рівняння регресії:
7.021 + 3.825 * X1 + 3.481 * X2 + 8.055 * X3 + 6.101 * X1X2 + 0.492 * X1X3 + 8.599 * X2X3+ 3.100 * X1X2X3 + 4.301 * X11^
2 + 0.700 * X22^2 + 9.292 * X33^2 = Y
```

```
Експериментальні значення:
[42784.24536667381, 11849.930677250253, -40691.10796432385, -12225.42265374707, 127118.0663910939, 40677.41836833767, -1
51885.62027323715, -39421.934962659965, 2746.380153380985, -8165.471974371853, 98254.49865831858, -103573.24677097391, -
7918.087037476722, 2269.1943831546155, -5605.556653085691]
```

```
Перевірка однорідності за критерієм Кохрена:
Gp = 0.13687150837988832
Дисперсія однорідна
```

```
Оцінка значимості коефіцієнтів за критерієм Стюдента
Значущі коефіцієнти: [7.021, 3.825, 3.481, 8.055, 6.101, 0.492, 8.599, 3.1, 4.301, 0.7, 9.292]
```

```
Незначущі коефіцієнти: []
```

```
Значення з отриманими коефіцієнтами:
[42784.24536667381, 11849.930677250253, -40691.10796432385, -12225.42265374707, 127118.0663910939, 40677.41836833767, -1
51885.62027323715, -39421.934962659965, 2746.380153380985, -8165.471974371853, 98254.49865831858, -103573.24677097391, -
7918.087037476722, 2269.1943831546155, -5605.556653085691]
```

```
Перевірка на адекватність за критерієм Фішера
Fp = 2.898026918931638
Рівняння регресії неадекватне при рівні значимості 0.05
```

Висновок: у ході виконання лабораторної роботи проведено трьохфакторний експеримент і отримано адекватну модель – рівняння регресії, використовуючи рототабельний композиційний план. Кінцева мета роботи досягнута.