

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 №3
З дисципліни «Методи оптимізації та планування»
ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З
ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ

ВИКОНАВ:
Студент II курсу ФІОТ
Групи ІВ-91
Каптур Максим Ігорович
Номер залікової книжки - 9112

ПЕРЕВІРИВ:
Регіда П.Г.

Київ 2021 р.

Мета:

Провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

Варіант завдання:

Варіант	X ₁		X ₂		X ₃	
	min	max	min	max	min	max
112	-40	20	-35	15	20	25

Лістинг програми:

```
from random import *
import numpy as np
from numpy.linalg import solve
from scipy.stats import f, t
from functools import partial

class Laba_3:

    def __init__(self, n, m):
        self.n = n
        self.m = m
        self.X_Min = (-40 - 35 + 20) / 3
        self.X_Max = (20 + 15 + 25) / 3
        self.Y_Max = round(200 + self.X_Max)
        self.Y_Min = round(200 + self.X_Min)
        self.x_Norm = [[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]]
        self.X_range = [(-40, 20), (-35, 15), (20, 25)]
        self.Y = np.zeros(shape=(self.n, self.m))
        self.Y_new = []
        for i in range(self.n):
            for j in range(self.m):
                self.Y[i][j] = randint(self.Y_Min, self.Y_Max)
        self.Y_av = [round(sum(i) / len(i), 2) for i in self.Y]
        self.x_Norm = self.x_Norm[:len(self.Y)]
        self.x = np.ones(shape=(len(self.x_Norm), len(self.x_Norm[0])))
        for i in range(len(self.x_Norm)):
            for j in range(1, len(self.x_Norm[i])):
                if self.x_Norm[i][j] == -1:
                    self.x[i][j] = self.X_range[j - 1][0]
                else:
                    self.x[i][j] = self.X_range[j - 1][1]
        self.f1 = m - 1
        self.f2 = n
        self.f3 = self.f1 * self.f2
        self.q = 0.05
```

```

def regres(self, x, b):
    Y = sum([x[i] * b[i] for i in range(len(x))])
    return Y

def koefs(self):
    mx1 = sum(self.x[:, 1]) / self.n
    mx2 = sum(self.x[:, 2]) / self.n
    mx3 = sum(self.x[:, 3]) / self.n
    my = sum(self.Y_av) / self.n
    a12 = sum([self.x[i][1] * self.x[i][2] for i in range(len(self.x))]) / self.n
    a13 = sum([self.x[i][1] * self.x[i][3] for i in range(len(self.x))]) / self.n
    a23 = sum([self.x[i][2] * self.x[i][3] for i in range(len(self.x))]) / self.n
    a11 = sum([i ** 2 for i in self.x[:, 1]]) / self.n
    a22 = sum([i ** 2 for i in self.x[:, 2]]) / self.n
    a33 = sum([i ** 2 for i in self.x[:, 3]]) / self.n
    a1 = sum([self.Y_av[i] * self.x[i][1] for i in range(len(self.x))]) / self.n
    a2 = sum([self.Y_av[i] * self.x[i][2] for i in range(len(self.x))]) / self.n
    a3 = sum([self.Y_av[i] * self.x[i][3] for i in range(len(self.x))]) / self.n
    X = [[1, mx1, mx2, mx3], [mx1, a11, a12, a13], [mx2, a12, a22, a23], [mx3,
a13, a23, a33]]
    Y = [my, a1, a2, a3]
    B = [round(i, 2) for i in solve(X, Y)]
    print('\nНаше рівняння регресії')
    print(f'Y = {B[0]} + {B[1]}*x1 + {B[2]}*x2 + {B[3]}*x3')

    return B

def dispers(self):
    result = []
    for i in range(self.n):
        s = sum([(self.Y_av[i] - self.Y[i][j]) ** 2 for j in range(self.m)]) /
self.m
        result.append(s)
    return result

def KOHREN(self):
    q1 = self.q / self.f1
    fisher_value = f.ppf(q=1 - q1, dfn=self.f2, dfd=(self.f1 - 1) * self.f2)
    G_cr = fisher_value / (fisher_value + self.f1 - 1)
    s = self.dispers()
    Gp = max(s) / sum(s)
    return Gp, G_cr

def Student(self):
    def bs():
        result = [sum(1 * Y for Y in self.Y_av) / self.n]
        for i in range(3):
            b = sum(j[0] * j[1] for j in zip(self.x[:, i], self.Y_av)) / self.n
            result.append(b)
        return result

    S_kv = self.dispers()
    s_kv_aver = sum(S_kv) / self.n

    s_Bs = (s_kv_aver / self.n / self.m) ** 0.5
    Bs = bs()
    ts = [abs(B) / s_Bs for B in Bs]
    return ts

def FISHER(self, d):
    S_ad = self.m / (self.n - d) * sum([(self.Y_new[i] - self.Y_av[i]) ** 2 for i

```

```

in range(len(self.Y))]
    S_kv = self.dispers()
    S_kv_aver = sum(S_kv) / self.n
    F_p = S_ad / S_kv_aver
    return F_p

def all_together(self):

    Student = partial(t.ppf, q=1 - 0.025)
    t_student = Student(df=self.f3)

    print('\nКритерій Кохрена, перевірка')
    Gp, G_kr = self.KOHNEN()
    print(f'Gp = {Gp}')
    if Gp < G_kr:
        print(f'З ймовірністю {1-self.q} дисперсії однорідні.')
    else:
        print("Либо больше, либо ничего (дослідів)")
        self.m += 1
        FractionalExperiment(self.n, self.m)

    ts = self.Student()
    print('\nПеревірка значущості коефіцієнтів за критерієм Стьюдента')
    print('Критерій Стьюдента:\n', ts)
    result = [t for t in ts if t > t_student]
    B = self.koefs()
    final_k = [B[ts.index(i)] for i in ts if i in result]
    print('Коефіцієнти {} статистично незначущі, тому ми виключаємо їх з
рівняння.'.format(
        [i for i in B if i not in final_k]))

    for j in range(self.n):
        self.Y_new.append(self.regres([self.x[j][ts.index(i)] for i in ts if i in
result], final_k))

    print(f'\nЗначення "Y" з коефіцієнтами {final_k}')
    print(self.Y_new)

    d = len(result)
    f4 = self.n - d
    F_p = self.FISHER(d)

    FISHER = partial(f.ppf, q=1 - 0.05)
    f_t = FISHER(dfn=f4, dfd=self.f3) # табличне знач
    print('\nПеревірка адекватності за критерієм Фішера')
    print('Fp =', F_p)
    print('F_t =', f_t)
    if F_p < f_t:
        print('Математична модель адекватна експериментальним даним')
    else:
        print('Математична модель не адекватна експериментальним даним')

experiment = Laba_3(7, 8)
experiment.all_together()

```

Контрольні запитання:

1. Що називається дробовим факторним експериментом?
Дробовим факторним експериментом називається експеримент з використанням частини повного факторного експерименту.
2. Для чого потрібно розрахункове значення Кохрена?
Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.
3. Для чого перевіряється критерій Стюдента?
За допомогою критерію Стюдента перевіряється значущість коефіцієнтів рівняння регресії.
4. Чим визначається критерій Фішера і як його застосовувати?
Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваному об'єкту.

Результат виконання роботи:

```
C:\Users\Максим\AppData\Local\Programs\Python\Python39\python.exe C:/Users/Максим/PycharmProjects/Kurs_2/Laba_2_MND.py

Критерій Кохрена, перевірка
Gr = 0.23738865745858875
З ймовірністю 0.95 дисперсії однорідні.

Перевірка значущості коефіцієнтів за критерієм Стюдента
Критерій Стюдента:
[144.89505224578826, 144.89505224578826, 2092.8615644286224, 1936.4577758084868]

Наше рівняння регресії
Y = 188.42 + -0.02*x1 + 0.08*x2 + 0.68*x3
Коефіцієнти [-0.02] статистично незначущі, тому ми виключаємо їх з рівняння.

Значення "Y" з коефіцієнтами [188.42, 188.42, 0.08, 0.68]
[387.64, 395.03999999999996, 391.03999999999996, 391.64, 391.03999999999996, 391.64, 387.64]

Перевірка адекватності за критерієм Фішера
Fr = 6036.404615387623
F_t = 2.7939488515842408
Математична модель не адекватна експериментальним даним

Process finished with exit code 0
```

Висновок:

В даній лабораторній роботі я провів дробовий трьохфакторний експеримент з трьома статистичними перевірками і отримав коефіцієнти рівняння регресії.