Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

**ЛАБОРАТОРНА РОБОТА № 3**

з дисципліни «**Методи оптимізації та планування експерименту**»

# на тему : Проведення трьохфакторного експерименту з використанням лінійного рівняння регресії.

Виконав:

студент ІІ курсу ФІОТ

групи ІВ-82

Онищук В. В.

Перевірив:

Регіда П. Г.

Київ - 2020

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

**Варіант:**



**Лістинг програми:**

**import** datetime  
**from** random **import** \*  
**import** numpy **as** np  
**from** numpy.linalg **import** solve  
**from** scipy.stats **import** f, t  
**from** functools **import** partial  
**from** datetime **import** datetime  
  
start\_time = datetime.now()  
  
  
**class** FractionalExperiment:  
 *#Проведення дробового трьохфакторного експерименту* **def** \_\_init\_\_(self, n, m):  
 start\_time = datetime.now()  
 self.n = n  
 self.m = m  
 self.x\_av\_min = (-10 + -20 + -20) / 3  
 self.x\_av\_max = (50 + 60 + 5) / 3  
 self.y\_max = round(200 + self.x\_av\_max)  
 self.y\_min = round(200 + self.x\_av\_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 = [(-10, 50), (-20, 60), (-20,5)]  
 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** regression(self, x, b):  
 *#Підстановка коефіцієнтів у рівняння регресії* y = sum([x[i] \* b[i] **for** i **in** range(len(x))])  
 **return** y  
  
 **def** count\_koefs(self):  
 *#Розрахунок коефіцієнтів рівняння регресії* 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(**'\nРівняння регресії'**)  
 print(**f'y = {B[0]} + {B[1]}\*x1 + {B[2]}\*x2 + {B[3]}\*x3'**)  
  
 **return** B  
  
 **def** dispersion(self):  
 *#Розрахунок дисперсії* res = []  
 **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  
 res.append(s)  
 **return** res  
  
 **def** kohren(self):  
 *#Перевірка однорідності дисперсій за критерієм Кохрена* 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.dispersion()  
 Gp = max(s) / sum(s)  
 **return** Gp, G\_cr  
  
 **def** student(self):  
 *#Перевірка знащущості коефіцієнтів за критерієм Стьюдента* **def** bs():  
 res = [sum(1 \* y **for** y **in** self.y\_av) / self.n]  
 **for** i **in** range(3): *# 4 - к-сть факторів* b = sum(j[0] \* j[1] **for** j **in** zip(self.x[:, i], self.y\_av)) / self.n  
 res.append(b)  
 **return** res  
  
 S\_kv = self.dispersion()  
 s\_kv\_aver = sum(S\_kv) / self.n  
  
 *# статиcтична оцінка дисперсії* 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):  
 *#Перевірка адекватності за критерієм Фішера* 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.dispersion()  
 S\_kv\_aver = sum(S\_kv) / self.n  
 F\_p = S\_ad / S\_kv\_aver  
 **return** F\_p  
  
 **def** check(self):  
 *#Проведення статистичних перевірок* student = partial(t.ppf, q=1 - 0.025)  
 t\_student = student(df=self.f3)  
  
 print(**'\nПеревірка за критерієм Кохрена'**)  
 Gp, G\_kr = self.kohren()  
 print(**f'Gp = {Gp}'**)  
 **if** Gp < G\_kr:  
 print(**f'З ймовірністю {1-self.q} дисперсії однорідні.'**)  
 **else**:  
 print(**"Необхідно збільшити кількість дослідів"**)  
 self.m += 1  
 FractionalExperiment(self.n, self.m)  
  
 ts = self.student()  
 print(**'\nПеревірка значущості коефіцієнтів за критерієм Стьюдента'**)  
 print(**'Критерій Стьюдента:\n'**, ts)  
 res = [t **for** t **in** ts **if** t > t\_student]  
 B = self.count\_koefs()  
 final\_k = [B[ts.index(i)] **for** i **in** ts **if** i **in** res]  
 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.regression([self.x[j][ts.index(i)] **for** i **in** ts **if** i **in** res], final\_k))  
  
 print(**f'\nЗначення "y" з коефіцієнтами {final\_k}'**)  
 print(self.y\_new)  
  
 d = len(res)  
 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 = FractionalExperiment(7, 8)  
experiment.check()  
print(**"Час виконання обрахунків: %s секунд"** % (datetime.now() - start\_time))

**Результат програми:**

E:\Vlad\МОПЕ\venv\Scripts\python.exe E:/Vlad/МОПЕ/lab3/lab3.py

Перевірка за критерієм Кохрена

Gp = 0.18795671427573488

З ймовірністю 0.95 дисперсії однорідні.

Перевірка значущості коефіцієнтів за критерієм Стьюдента

Критерій Стьюдента:

[96.55261575368972, 96.55261575368972, 1527.7372283309192, 1365.122847817138]

Рівняння регресії

y = 211.15 + 0.03\*x1 + -0.01\*x2 + 0.08\*x3

Коефіцієнти [0.03] статистично незначущі, тому ми виключаємо їх з рівняння.

Значення "y" з коефіцієнтами [211.15, 211.15, -0.01, 0.08]

[420.9, 422.09999999999997, 422.9, 420.09999999999997, 422.9, 420.09999999999997, 420.9]

Перевірка адекватності за критерієм Фішера

Fp = 3109.573963195291

F\_t = 2.7939488515842408

Математична модель не адекватна експериментальним даним

Час виконання обрахунків: 0:00:00.062516 секунд

Process finished with exit code 0

**Відповіді на контрольні запитання:**

1. Що називається дробовим факторним експериментом?

У деяких випадках немає необхідності проводити повний факторний експеримент (ПФЕ). 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 (ДФЕ).

2. Для чого потрібно розрахункове значення Кохрена?

Для перевірки однорідності дисперсії. Якщо виконується умова Gp <Gт, то з обраним рівнем статистичної значимості q (з ймовірністю p = 1 - q) усі дисперсії у рядках визнаються однорідними. Якщо Gp> Gт, то гіпотезу відкидають, тобто m недостатньо (m - кількість дослідів). Тоді необхідно збільшити кількість дослідів: m = m +1.

1. Для чого перевіряється критерій Стьюдента?

Для перевірки значущості коефіцієнтів. Якщо виконується нерівність ts< tтабл, то приймається нуль-гіпотеза, тобто вважається, що знайдений коефіцієнт βs 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?

Адекватність моделі перевіряють за F-критерієм Фішера, який дорівнює відношенню дисперсії адекватності до дисперсії відтворюваност.Використовують для преревірки адекватності математичної моделі експериментальним даним. Якщо Fp< Fт то отримана математична модель з прийнятим рівнем статистичної значимості q адекватна експериментальним даним.