Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Лабораторна робота №6

з дисципліни «Методи оптимізації та планування експерименту»

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

Виконав:

студент 2-го курсу ФІОТ

групи ІВ-82

Сірокомський Микола

Варiант: 218

Перевірив:  
Регіда П.Г.

Київ – 2020

**Код**

**from** functools **import** partial  
**from** math **import** sqrt  
**from** random **import** randrange  
**from** time **import** process\_time  
  
**from** numpy.linalg **import** solve  
**from** scipy.stats **import** f, t  
  
x1, x2, x3 = [-20, 30], [-35, 15], [-20, 5]  
m, N, l = 2, 15, 1.73 *# кількість повторень кожної комбінації & кількість повторення дослідів*x\_avg = [(max(x1) + max(x2) + max(x3)) / 3, (min(x1) + min(x2) + min(x3)) / 3] *# Xcр(max) & Xср(min)*xo = [(min(x1) + max(x1)) / 2, (min(x2) + max(x2)) / 2, (min(x3) + max(x3)) / 2] *# Xoi*delta\_x = [max(x1) - xo[0], max(x1) - xo[1], max(x1) - xo[2]] *# delta Xi*y\_range = [200 + int(max(x\_avg)), 200 + int(min(x\_avg))] *# Yi(max) & Yi(min)*xn = [[-1, -1, -1, -1, +1, +1, +1, +1, -1.73, 1.73, 0, 0, 0, 0, 0], *# нормовані значення факторів* [-1, -1, +1, +1, -1, -1, +1, +1, 0, 0, -1.73, 1.73, 0, 0, 0],  
 [-1, +1, -1, +1, -1, +1, -1, +1, 0, 0, 0, 0, -1.73, 1.73, 0]]  
  
xx = [[int(x \* y) **for** x, y **in** zip(xn[0], xn[1])], *# нормовані значення факторів для ефекту взаємодії* [int(x \* y) **for** x, y **in** zip(xn[0], xn[2])],  
 [int(x \* y) **for** x, y **in** zip(xn[1], xn[2])]]  
  
xxx = [int(x \* y \* z) **for** x, y, z **in** zip(xn[0], xn[1], xn[2])]  
  
x\_xn = [[round(xn[j][i] \*\* 2, 3) **for** i **in** range(N)] **for** j **in** range(3)] *# нормовані знач. факторів для квад. членів*x = [[min(x1), min(x1), min(x1), min(x1), max(x1), max(x1), max(x1), max(x1), round(-l \* delta\_x[0] + xo[0], 3),  
 round(l \* delta\_x[0] + xo[0], 3), xo[0], xo[0], xo[0], xo[0], xo[0]], *# натуральні значення факторів* [min(x2), min(x2), max(x2), max(x2), min(x2), min(x2), max(x2), max(x2), xo[1], xo[1],  
 round(-l \* delta\_x[1] + xo[1], 3), round(l \* delta\_x[1] + xo[1], 3), xo[1], xo[1], xo[1]],  
 [min(x3), max(x3), min(x3), max(x3), max(x3), min(x3), max(x3), min(x3), xo[2], xo[2], xo[2], xo[2],  
 round(-l \* delta\_x[2] + xo[2], 3), round(l \* delta\_x[2] + xo[2], 3), xo[2]]]  
  
xx2 = [[round(x \* y, 3) **for** x, y **in** zip(x[0], x[1])], *# натуральні значення факторів для ефекту взаємодії* [round(x \* y, 3) **for** x, y **in** zip(x[0], x[2])],  
 [round(x \* y, 3) **for** x, y **in** zip(x[1], x[2])]]  
  
xxx2 = [round(x \* y \* z, 3) **for** x, y, z **in** zip(x[0], x[1], x[2])]  
  
x\_x = [[round(x[j][i] \*\* 2, 3) **for** i **in** range(N)] **for** j **in** range(3)] *# натуральні значення факторів для квадрат. членів*t\_aver = 0  
t\_number= 0  
**for** i **in** range(100):  
 **while True**:  
 start\_time = process\_time()  
 *# формування Y* y = [  
 [round(5.4 + 8.1 \* x[0][j] + 7.5 \* x[1][j] + 9.7 \* x[2][j] + 5.2 \* x[0][j] \* x[0][j] + 0.6 \* x[1][j] \* x[1][j] +  
 4.7 \* x[2][j] \* x[2][j] + 8.1 \* x[0][j] \* x[1][j] + 0.8 \* x[0][j] \* x[2][j] + 7.4 \* x[1][j] \* x[2][j] +  
 0.1 \* x[0][j] \* x[1][j] \* x[2][j] + randrange(0, 10) - 5, 2) **for** i **in** range(m)] **for** j **in** range(N)]  
 arr\_avg = **lambda** arr: round(sum(arr) / len(arr), 4)  
 y\_avg = list(map(arr\_avg, y)) *# середнє значення Y* dispersions = [sum([((y[i][j] - y\_avg[i]) \*\* 2) / m **for** j **in** range(m)]) **for** i **in** range(N)] *# дисперсії по рядках* x\_matrix = x + xx2 + [xxx2] + x\_x *# повна матриця з натуральними значеннями факторів* norm\_matrix = xn + xx + [xxx] + x\_xn *# повна матриця з нормованими значеннями факторів* mx = list(map(arr\_avg, x\_matrix)) *# середні значення х по колонкам* my = sum(y\_avg) / N *# середнє значення Y\_avg  
  
 # ======================================== Форматування таблиці ====================================================* table\_factors\_1 = [**"X1"**, **"X2"**, **"X3"**]  
 table\_factors\_2 = [**"X1X2"**, **"X1X3"**, **"X2X3"**, **"X1X2X3"**, **"X1^2"**, **"X2^2"**, **"X3^2"**]  
 table\_y = [**"Y{}"**.format(i + 1) **for** i **in** range(m)]  
 other = [**"#"**, **"Y"**]  
  
 header\_format\_norm = **"+{0:=^3}"** + **"+{0:=^8}"** \* (len(table\_factors\_1)) + **"+{0:=^8s}"** \* (len(table\_factors\_2))  
 header\_format = **"+{0:=^3}"** + **"+{0:=^8}"** \* (len(table\_factors\_1)) + **"+{0:=^10s}"** \* (  
 len(table\_factors\_2)) + **"+{0:=^10s}"** \* (len(table\_y)) + **"+{0:=^10s}"** row\_format\_norm = **"|{:^3}"** + **"|{:^8}"** \* (len(table\_factors\_1)) + **"|{:^8}"** \* (len(table\_factors\_2))  
 row\_format = **"|{:^3}"** + **"|{:^8}"** \* (len(table\_factors\_1)) + **"|{:^10}"** \* (len(table\_factors\_2)) + **"|{:^10}"** \* (  
 len(table\_y)) + **"|{:^10}"** separator\_format\_norm = **"+{0:-^3s}"** + **"+{0:-^8s}"** \* (len(table\_factors\_1)) + **"+{0:-^8s}"** \* (len(table\_factors\_2))  
 separator\_format = **"+{0:-^3s}"** + **"+{0:-^8s}"** \* (len(table\_factors\_1)) + **"+{0:-^10s}"** \* (  
 len(table\_factors\_2)) + **"+{0:-^10s}"** \* (len(table\_y)) + **"+{0:-^10s}"** my\_sep\_norm = **"|{:^93s}|\n"** my\_sep = **"|{:^140s}|\n" if** m == 2 **else "|{:^151s}|\n"** *# ======================================== Нормальні значення ======================================================* print(header\_format\_norm.format(**"="**) + **"+\n"** + my\_sep\_norm.format(**"Матриця ПФЕ (нормальні значення факторів)"**) +  
 header\_format\_norm.format(**"="**) + **"+\n"** + row\_format\_norm.format(other[0], \*table\_factors\_1, \*table\_factors\_2)  
 + **"|\n"** + header\_format\_norm.format(**"="**) + **"+"**)  
  
 **for** i **in** range(N):  
 print(**"|{:^3}|"**.format(i + 1), end=**""**)  
 **for** j **in** range(3): print(**"{:^+8}|"**.format(xn[j][i]), end=**""**)  
 **for** j **in** range(3): print(**"{:^+8}|"**.format(xx[j][i]), end=**""**)  
 print(**"{:^+8}|"**.format(xxx[i]), end=**""**)  
 **for** j **in** range(3): print(**"{:^+8}|"**.format(x\_xn[j][i]), end=**""**)  
 print()  
  
 print(separator\_format\_norm.format(**"-"**) + **"+\n\n"**)  
  
 *# ======================================== Натуральні значення =====================================================* print(header\_format.format(**"="**) + **"+\n"** + my\_sep.format(**"Матриця ПФЕ (натуральні значення факторів)"**) +  
 header\_format.format(**"="**) + **"+\n"** + row\_format.format(other[0], \*table\_factors\_1, \*table\_factors\_2, \*table\_y,  
 other[1]) + **"|\n"** + header\_format.format(**"="**) + **"+"**)  
  
 **for** i **in** range(N):  
 print(**"|{:^3}|"**.format(i + 1), end=**""**)  
 **for** j **in** range(3): print(**"{:^ 8}|"**.format(x[j][i]), end=**""**)  
 **for** j **in** range(3): print(**"{:^ 10}|"**.format(xx2[j][i]), end=**""**)  
 print(**"{:^ 10}|"**.format(xxx2[i]), end=**""**)  
 **for** j **in** range(3): print(**"{:^ 10}|"**.format(x\_x[j][i]), end=**""**)  
 **for** j **in** range(m): print(**"{:^ 10}|"**.format(y[i][j]), end=**""**)  
 print(**"{:^10.2f}|"**.format(y\_avg[i]))  
  
  
 **def** a(first, second):  
 **return** sum([x\_matrix[first - 1][j] \* x\_matrix[second - 1][j] / N **for** j **in** range(N)])  
  
  
 **def** find\_a(num):  
 **return** sum([y\_avg[j] \* x\_matrix[num - 1][j] / N **for** j **in** range(N)])  
  
  
 **def** check(b\_lst, k):  
 **return** b\_lst[0] + b\_lst[1] \* x\_matrix[0][k] + b\_lst[2] \* x\_matrix[1][k] + b\_lst[3] \* x\_matrix[2][k] + \  
 b\_lst[4] \* x\_matrix[3][k] + b\_lst[5] \* x\_matrix[4][k] + b\_lst[6] \* x\_matrix[5][k] + \  
 b\_lst[7] \* x\_matrix[6][k] + b\_lst[8] \* x\_matrix[7][k] + b\_lst[9] \* x\_matrix[8][k] + \  
 b\_lst[10] \* x\_matrix[9][k]  
  
  
 unknown = [[1, mx[0], mx[1], mx[2], mx[3], mx[4], mx[5], mx[6], mx[7], mx[8], mx[9]],  
 *# ліва частина рівнянь з невідомими для пошуку коефіцієнтів b (приклад в методі)* [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)]]  
 known = [my, find\_a(1), find\_a(2), find\_a(3), find\_a(4), find\_a(5), find\_a(6), find\_a(7), find\_a(8), find\_a(9),  
 find\_a(10)] *# знаходення відомих значень a1, a2, ...* b = solve(unknown, known)  
 print(separator\_format.format(**"-"**) + **f"+\n\n\tОтримане рівняння регресії при m={m}:\n"  
 f"ŷ = {b[0]:.3f} + {b[1]:.3f}\*X1 + {b[2]:.3f}\*X2 + "  
 f"{b[3]:.3f}\*X3 + {b[4]:.3f}\*X1X2 + {b[5]:.3f}\*X1X3 + "  
 f"{b[6]:.3f}\*X2X3 + {b[7]:.3f}\*X1X2X3 + {b[8]:.3f}\*X11^2 + "  
 f"{b[9]:.3f}\*X22^2 + {b[10]:.3f}\*X33^2\n\n\tПеревірка:"**)  
 **for** i **in** range(N): print(**"ŷ{} = {:.3f} ≈ {:.3f}"**.format((i + 1), check(b, i), y\_avg[i]))  
  
  
 *# ======================================== Критерій Кохрена ========================================================* **def** table\_fisher(prob, n, m, d):  
 x\_vec = [i \* 0.001 **for** i **in** range(int(10 / 0.001))]  
 f3 = (m - 1) \* n  
 **for** i **in** x\_vec:  
 **if** abs(f.cdf(i, n - d, f3) - prob) < 0.0001:  
 **return** i  
  
  
 f1, f2 = m - 1, N  
 f3 = f1 \* f2  
 fisher = table\_fisher(0.95, N, m, 1)  
 Gp = max(dispersions) / sum(dispersions)  
 Gt = fisher / (fisher + (m - 1) - 2)  
  
 print(**"\nОднорідність дисперсії (критерій Кохрена): "**)  
 print(**f"Gp = {Gp}\nGt = {Gt}"**)  
 **if** Gp < Gt:  
 print(**"\nДисперсія однорідна (Gp < Gt)"**)  
  
 D\_beta = sum(dispersions) / (N \* N \* m)  
 Sb = sqrt(abs(D\_beta))  
 beta = [sum([(y\_avg[j] \* norm\_matrix[i][j]) / N **for** j **in** range(N)]) **for** i **in** range(len(norm\_matrix))]  
  
 t\_list = [abs(i) / Sb **for** i **in** beta]  
 student = partial(t.ppf, q=1 - 0.025)  
 d, T = 0, student(df=f3)  
 print(**"\nt табличне = "**, T)  
  
 **for** i **in** range(len(t\_list)):  
 **if** t\_list[i] < T:  
 b[i] = 0  
 print(**"\tt{} = {} => коефіцієнт незначимий, його слід виключити з рів-ня регресії"**.format(i, t\_list[i]))  
 **else**:  
 t\_aver += t\_list[i]  
 t\_number+=1  
 print(**"\tt{} = {} => коефіцієнт значимий"**.format(i, t\_list[i]))  
 d += 1  
  
 print(**"\nОтже, кіл-ть значимих коеф. d ="**, d, **"\n\n\tРів-ня регресії з урахуванням критерія Стьюдента:\nŷ = "**,  
 end=**""**)  
 print(**"{:.3f}"**.format(b[0]), end=**""**) **if** b[0] != 0 **else None  
 for** i **in** range(1, 10):  
 print(**" + {:.3f}\*{}"**.format(b[i], (table\_factors\_1 + table\_factors\_2)[i]), end=**""**) **if** b[i] != 0 **else None** print(**"\n\n\tПеревірка при підстановці в спрощене рів-ня регресії:"**)  
 **for** i **in** range(N): print(**"y`{} = {:.3f} ≈ {:.3f}"**.format((i + 1), check(b, i), y\_avg[i]))  
  
 f4 = N - d  
 fisher\_sum = sum([(check(b, i) - y\_avg[i]) \*\* 2 **for** i **in** range(N)])  
 D\_ad = (m / f4) \* fisher\_sum  
  
 fisher = partial(f.ppf, q=1 - 0.05)  
 Fp = D\_ad / sum(dispersions) / N  
 Ft = fisher(dfn=f4, dfd=f3)  
 print(**"\nКритерій Фішера:"**)  
 **if** Fp > Ft:  
 print(**"\tРівняння регресії неадекватне (Ft < Fp)."**)  
 **break  
 else**:  
 print(**"\tРівняння регресії адекватне (Ft > Fp)!"**)  
 print(**"\n\n--- Час виконання програми: %s секунд ---"** % (process\_time() - start\_time))  
 **break  
  
 else**:  
 print(**"Дисперсія неоднорідна (Gp > Gt), збільшуємо m, повторюємо операції"**)  
 m += 1  
print(**"Середнє значеннє значимих коефіціеєнтів після 100 запусків = "** +str(t\_aver/t\_number));