Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра автоматизованих систем управління



**Звіт**

до лабораторної роботи № 1

з дисципліни

*“Моделювання процесів і смарт-систем”*

Виконав: ст. гр. ОІ-31

**Тесля Микола**

Прийняв: Мельник Р.В.

Львів – 2024

**Варіант 10**

**Тема: Концепція моделі “чорний ящик” та встановлення зажежності між її вхідними і вихідними параметрами.**

*Мета роботи: Зрозуміти концепцію моделі “чорний ящик” та набути навички застосування методу найменших квадратів для визначення залежності між вхідними і вихідними параметрами моделі.*

**Опис побудови прикладу моделі “чорний ящик” (ігрова консоль)**

**Вступ**

Модель "чорний ящик" використовується для аналізу систем, у яких внутрішній механізм роботи залишається невідомим або не розглядається, а взаємодія із зовнішнім середовищем описується через вхідні та вихідні параметри. У цьому випадку ігрова консоль розглядається як "чорний ящик", де користувач вводить певні дані (входи), а консоль генерує відповідні результати (виходи).

**Завдання 1**

**Вибір об'єкта для моделювання**

Об’єктом дослідження обрано ігрову консоль, оскільки вона є добре визначеною системою з чітко заданими входами і виходами, які можуть бути формально описані.

**Вхідні параметри системи**

До входів ігрової консолі відносяться всі фактори, що впливають на її роботу. Основні входи:

* Живлення (електричний кабель, акумулятор).
* Геймпад (контролер) (натискання кнопок, рухи джойстика).
* Інтернет-з'єднання (Wi-Fi або LAN-кабель).
* Ігрові диски, картриджі, цифрові ігри (отримання даних для роботи).
* Підключені аксесуари (VR-шолом, мікрофон, камера, клавіатура).
* Оновлення програмного забезпечення (оновлення ОС, драйверів, ігор).
* Користувацькі налаштування (гучність, яскравість, параметри мережі).
* Хмарні сервіси (онлайн-ігри, збереження, підписки).

**Вихідні параметри системи**

Виходи – це результати роботи консолі, які отримує користувач. Основні виходи:

* Відеосигнал на екран (зображення гри, інтерфейс користувача).
* Звук (музика, звукові ефекти, голосовий чат).
* Сигнали на геймпад (вібрація, світлові індикатори).
* Передача даних в інтернет (онлайн-гра, стрімінг, збереження в хмарі).
* Статус системи (індикатори роботи, повідомлення про помилки).
* Відповідь на натискання кнопок (реакція в грі, перемикання меню).

*Основна модель системи – ігрова консоль*

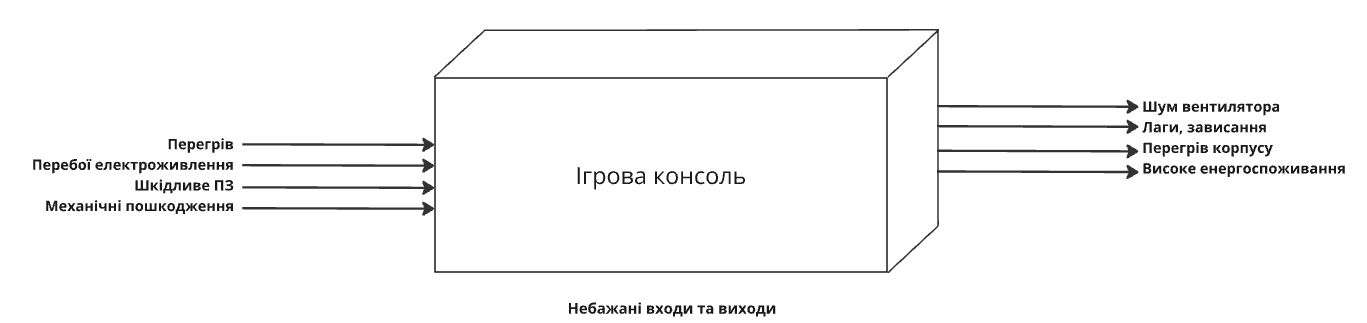
**Небажані входи та виходи**

Окрім основних входів та виходів, існують небажані фактори, які можуть негативно впливати на систему:

Небажані входи:

* Перегрів (через погану вентиляцію або високе навантаження).
* Перебої електроживлення (раптове відключення може призвести до втрати даних).
* Шкідливе ПЗ (віруси при підключенні до інтернету або встановленні неофіційного ПЗ).
* Механічні пошкодження (удари, падіння, потрапляння рідини).

Небажані виходи:

* Шум вентилятора (надмірний рівень шуму).
* Лаги та зависання (через недостатню потужність або погане охолодження).
* Перегрів корпусу (неприємний для користувача, потенційна шкода системі).
* Високе енергоспоживання (особливо в режимі очікування).

**Усунення недоліків системи**

Щоб мінімізувати небажані ефекти, можна застосувати такі рішення:

* Для перегріву → Покращена система охолодження (більше вентиляційних отворів, ефективніші вентилятори).
* Для шуму → Використання тихіших вентиляторів.
* Для перебоїв живлення → Використання джерела безперебійного живлення (ДБЖ).
* Для лагів → Оптимізація програмного забезпечення, використання швидших накопичувачів (SSD).
* Для захисту від вірусів → Контрольоване оновлення ПЗ, обмеження доступу до підозрілих джерел.
* Для економії енергії → Оптимізація режиму очікування.

**Завдання 2**

Для аналізу зв’язку між вхідними та вихідними параметрами моделі використано метод найменших квадратів (МНК).  
Цей метод дозволяє знайти лінійну залежність між вхідними змінними x1, x2 та вихідною змінною y.

Рівняння шуканої залежності має вигляд:

y = a0 ​+ a1​x1 ​+ a2​x2​

Початкові дані

Вхідні дані (після підстановки n=10):

|  |  |  |
| --- | --- | --- |
| x1 | x2 | y |
| 0 | 1.5 | 2.3 |
| 0 | 2.5 | 7 |
| 0 | 3.5 | 1 |
| 1 | 1.5 | 3 |
| 1 | 3.5 | 2 |
| 2 | 1.5 | 8.1 |
| 2 | 2.5 | 5.5 |
| 2 | 2.5 | 7.2 |

Використання методу найменших квадратів

Розрахунок проведено за допомогою розв’язку системи нормальних рівнянь, отриманих на основі методу найменших квадратів.

Отримано рівняння:

y = a0 ​+ a1​x1 ​+ a2​x2​

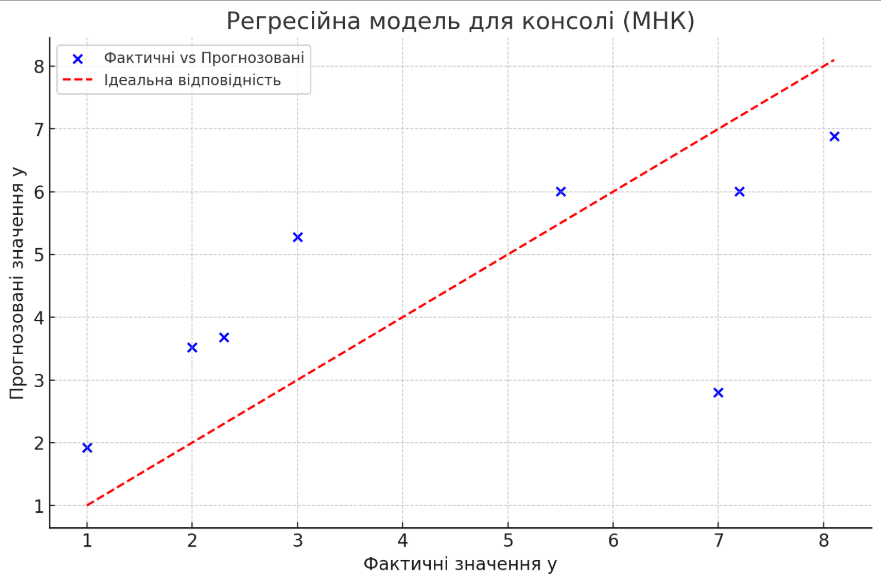
Де знайдені коефіцієнти:

* a0 (вільний член)
* a1 (коефіцієнт при x1)
* a2 (коефіцієнт при x2)

Прогноз вихідного параметра

Для значень x​= 1.5, x2 = 3 обчислено прогнозоване значення y:

y = a0 ​+ a1​(1.5) + a2​(3)

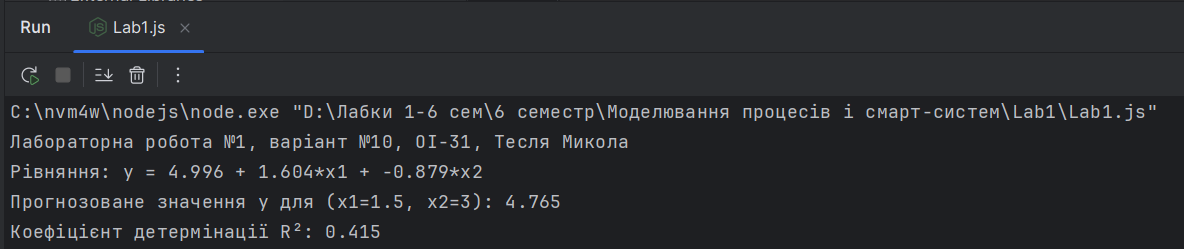


Комп’ютерна програма реалізації методу найменших квадратів

Програма на JavaScript виконує розрахунки методу найменших квадратів, знаходить коефіцієнти рівняння, обчислює прогнозоване значення y та оцінює точність моделі R2.

Код програми:

*console*.log('Лабораторна робота №1, варіант №10, ОІ-31, Тесля Микола');  
  
const *x1* = [0, 0, 0, 1, 1, 2, 2, 2];  
const *x2* = [1.5, 2.5, 3.5, 1.5, 3.5, 1.5, 2.5, 2.5];  
const *y* = [2.3, 7, 1, 3, 2, 8.1, 5.5, 7.2];  
  
function leastSquares(x1, x2, y) {  
 const n = x1.length;  
 let sumX1 = 0, sumX2 = 0, sumY = 0, sumX1X1 = 0, sumX2X2 = 0, sumX1X2 = 0, sumX1Y = 0, sumX2Y = 0;  
  
 for (let i = 0; i < n; i++) {  
 sumX1 += x1[i];  
 sumX2 += x2[i];  
 sumY += y[i];  
 sumX1X1 += x1[i] \* x1[i];  
 sumX2X2 += x2[i] \* x2[i];  
 sumX1X2 += x1[i] \* x2[i];  
 sumX1Y += x1[i] \* y[i];  
 sumX2Y += x2[i] \* y[i];  
 }  
  
 const matrix = [  
 [n, sumX1, sumX2, sumY],  
 [sumX1, sumX1X1, sumX1X2, sumX1Y],  
 [sumX2, sumX1X2, sumX2X2, sumX2Y]  
 ];  
  
 function gaussElimination(mat) {  
 let size = mat.length;  
 for (let i = 0; i < size; i++) {  
 let maxRow = i;  
 for (let k = i + 1; k < size; k++) {  
 if (*Math*.abs(mat[k][i]) > *Math*.abs(mat[maxRow][i])) {  
 maxRow = k;  
 }  
 }  
 [mat[i], mat[maxRow]] = [mat[maxRow], mat[i]];  
  
 for (let k = i + 1; k < size; k++) {  
 let factor = mat[k][i] / mat[i][i];  
 for (let j = i; j <= size; j++) {  
 mat[k][j] -= factor \* mat[i][j];  
 }  
 }  
 }  
  
 let res = new *Array*(size);  
 for (let i = size - 1; i >= 0; i--) {  
 res[i] = mat[i][size] / mat[i][i];  
 for (let k = i - 1; k >= 0; k--) {  
 mat[k][size] -= mat[k][i] \* res[i];  
 }  
 }  
 return res;  
 }  
  
 return gaussElimination(matrix);  
}  
  
const [*a0*, *a1*, *a2*] = leastSquares(*x1*, *x2*, *y*);  
*console*.log(`Рівняння: y = ${*a0*.toFixed(3)} + ${*a1*.toFixed(3)}\*x1 + ${*a2*.toFixed(3)}\*x2`);  
  
const *x1\_test* = 1.5, *x2\_test* = 3;  
const *y\_pred* = *a0* + *a1* \* *x1\_test* + *a2* \* *x2\_test*;  
*console*.log(`Прогнозоване значення y для (x1=1.5, x2=3): ${*y\_pred*.toFixed(3)}`);  
  
function computeR2(x1, x2, y, a0, a1, a2) {  
 let yMean = y.reduce((sum, val) => sum + val, 0) / y.length;  
 let SS\_tot = 0, SS\_res = 0;  
  
 for (let i = 0; i < y.length; i++) {  
 let yPred = a0 + a1 \* x1[i] + a2 \* x2[i];  
 SS\_res += (y[i] - yPred) \*\* 2;  
 SS\_tot += (y[i] - yMean) \*\* 2;  
 }  
 return 1 - (SS\_res / SS\_tot);  
}  
  
const *R2* = computeR2(*x1*, *x2*, *y*, *a0*, *a1*, *a2*);  
*console*.log(`Коефіцієнт детермінації R²: ${*R2*.toFixed(3)}`);

Результат виконання програми

**Висновок**

У ході виконання лабораторної роботи було розглянуто модель "чорного ящика" на прикладі ігрової консолі, визначено її основні вхідні та вихідні параметри, а також виявлено небажані фактори, які можуть негативно впливати на її роботу. Для аналізу залежності між вхідними та вихідними параметрами було застосовано метод найменших квадратів, який дозволив отримати рівняння регресії у вигляді y = a0 ​+ a1​x1​ + a2​x2​​, знайти його коефіцієнти та здійснити прогноз значення вихідного параметра y для заданих вхідних значень. Для оцінки точності отриманої моделі було розраховано коефіцієнт детермінації R2, який підтвердив ефективність апроксимації. Виконані розрахунки та побудований графік підтвердили зв’язок між вхідними та вихідними параметрами системи, а реалізація алгоритму у JavaScript дозволила автоматизувати процес розв’язку системи нормальних рівнянь. Отримані результати демонструють можливість застосування математичних методів моделювання для аналізу складних систем, що дозволяє формалізувати їхню роботу та підвищити ефективність управління параметрами.