ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Автоматизація очищення сирих даних IoT-сенсорів для моніторингу екологічного стану»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Владислав ІВАНЧУК
(підпис)

Виконав: здобувач вищої освіти групи ПДМ-61
Владислав ІВАНЧУК

Керівник: Оксана ЗОЛОТУХІНА
канд. техн. наук, доц.

Рецензент: _____
науковий ступінь, вчене звання *Ім'я, ПРІЗВИЩЕ*

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Іванчуку Владиславу Олександровичу

1. Тема кваліфікаційної роботи: «Автоматизація очищення сирих даних IoT-сенсорів для моніторингу екологічного стану»

керівник кваліфікаційної роботи Оксана ЗОЛОТУХІНА, канд. техн. наук, доц.,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «30» жовтня 2025 р. № 467.

2. Строк подання кваліфікаційної роботи «19» грудня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література, відкриті набори даних екологічного моніторингу, математичні моделі відновлення та фільтрації сигналів, вимоги до точності очищення даних.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз предметної області та специфіки даних IoT-сенсорів.
2. Аналіз методів та побудова математичної моделі очищення даних.
3. Програмна реалізація системи автоматизованого очищення.
4. Експериментальне дослідження та оцінка ефективності.

5. Перелік ілюстративного матеріалу: *презентація*

1. Схема потоків даних у багаторівневій IoT-архітектурі.
2. Характеристики та типові дефекти потоків даних.
3. Порівняльний аналіз методів обробки даних.
4. Узагальнена адитивна математична модель сигналу.
5. Математична модель компонентів спотворень.
6. Блок-схема адаптивного конвеєра очищення.
7. Візуалізація відновлення сигналу в стаціонарних умовах.
8. Візуалізація роботи алгоритму компенсації дрейфу.
9. Порівняльна діаграма ефективності методів.
10. Результат роботи алгоритму очищення на реальних даних
11. Результати апробації алгоритму на реальних даних.

6. Дата видачі завдання «31» жовтня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	31.10 – 03.11.2025	
2	Аналіз архітектури IoT-систем та особливостей даних екологічного моніторингу	04.11 – 06.11.2025	
3	Дослідження та порівняльний аналіз методів попередньої обробки даних	07.11 – 10.11.2025	
4	Розробка математичної моделі та методу очищення даних	11.11 – 24.11.2025	
5	Програмна реалізація модуля очищення даних	25.11 – 06.12.2025	
6	Проведення експериментальних досліджень та оцінка ефективності	07.12 – 13.12.2025	
7	Оформлення роботи: вступ, висновки, реферат	14.12 – 17.12.2025	
8	Розробка демонстраційних матеріалів	17.12 – 19.12.2025	

Здобувач вищої освіти _____ Владислав ІВАНЧУК
(підпис)

Керівник
кваліфікаційної роботи _____ Оксана ЗОЛОТУХІНА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 71 стор., 4 табл., 10 рис., 26 джерел.

Мета роботи – підвищення якості очищення сирих даних IoT-сенсорів для моніторингу екологічного стану за рахунок автоматизації обробки комплексного підходу до відновлення та корекції даних.

Об'єкт дослідження – процес очищення сирих даних IoT-сенсорів для моніторингу екологічного стану.

Предмет дослідження – методи та технології автоматизованого очищення сирих даних IoT-сенсорів для моніторингу екологічного стану.

У роботі проведено аналіз архітектурних рішень побудови систем екологічного моніторингу та досліджено специфіку даних сенсорів у контексті моделі великих даних. Систематизовано типові дефекти вхідних потоків, такі як апаратні шуми, структурні пропуски, аномальні викиди та часовий дрейф сенсорів. Обґрунтовано неефективність класичних методів фільтрації в умовах деградації вимірювального обладнання.

Розроблено адитивну математичну модель вхідного сигналу та метод автоматизованого очищення даних, який реалізується у вигляді послідовного конвеєра. Алгоритм включає етапи лінійної інтерполяції для відновлення цілісності, адаптивного фільтра на основі міжквартильного розмаху для усунення викидів, поліноміального де-трендингу для компенсації дрейфу та медіанної фільтрації для згладжування шуму.

Програмно реалізовано розроблений метод у вигляді модуля на мові Python з використанням бібліотек Pandas та NumPy. Проведено експериментальне дослідження на синтетичних та реальних даних. Результати моделювання підтвердили, що запропонований підхід дозволяє знизити середньоквадратичну похибку відновлення сигналу на 88% порівняно з сирими даними, ефективно компенсуючи систематичні зміщення, з якими не впоралися базові методи.

Практичне значення роботи полягає у створенні універсального інструменту для попередньої обробки екологічних даних, який може бути інтегрований у платформи «Розумного міста» або розгорнутий на периферійних пристроях для підвищення достовірності моніторингу.

КЛЮЧОВІ СЛОВА: IOT, ЕКОЛОГІЧНИЙ МОНІТОРИНГ, ОЧИЩЕННЯ ДАНИХ, ДРЕЙФ СЕНСОРА, АНОМАЛІЇ, PYTHON, ЧАСОВІ РЯДИ, ВЕЛИКІ ДАНІ.

ABSTRACT

Text part of the master's qualification work: 71 pages, 10 pictures, 5 table, 26 sources.

The purpose of the work is to improve the quality of cleaning raw IoT sensor data for environmental monitoring by automating processing based on a comprehensive approach to data recovery and correction.

Object of research – the process of cleaning raw data from IoT sensors for monitoring environmental conditions.

Subject of research – methods and technologies for automated cleaning of raw data from IoT sensors for monitoring environmental conditions

Summary of the work: The thesis analyzes architectural solutions for building environmental monitoring systems and investigates the specifics of sensor data in the context of the 5Vs Big Data model. Typical defects of input streams, such as hardware noise, structural gaps, anomalous outliers, and temporal sensor drift, are systematized. The inefficiency of classical filtering methods under conditions of measuring equipment degradation is substantiated.

An additive mathematical model of the input signal and a method for automated data cleaning, implemented as a sequential pipeline, have been developed. The algorithm includes stages of linear interpolation for restoring integrity, an adaptive filter based on the interquartile range (IQR) for eliminating outliers, polynomial detrending for compensating drift, and median filtering for smoothing noise.

The developed method is implemented as a software module in Python using Pandas and NumPy libraries. Experimental research was conducted on synthetic and real data (Air Quality Data Set). Simulation results confirmed that the proposed approach reduces the root mean square error (RMSE) of signal recovery by 88% compared to raw data, effectively compensating for systematic shifts that basic methods failed to handle.

The practical value lies in creating a universal tool for preprocessing environmental data, which can be integrated into "Smart City" platforms or deployed on peripheral devices (Edge Computing) to increase monitoring reliability.

KEYWORDS: IOT, ENVIRONMENTAL MONITORING, DATA CLEANING, SENSOR DRIFT, ANOMALIES, PYTHON, TIME SERIES, BIG DATA.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АЦП - аналого-цифровий перетворювач

ГДК - гранично допустима концентрація

IoT - Internet of Things (інтернет речей)

IQR - Interquartile Range (міжквартильний розмах)

LoRaWAN - Long Range Wide Area Network (глобальна мережа великого радіусу дії)

LSTM - Long Short-Term Memory (довга короткострокова пам'ять) MOX - Metal Oxide Sensor (метал-оксидний сенсор)

MVP - Minimum Viable Product (мінімально життєздатний продукт)

NB-IoT - Narrowband Internet of Things (вузькосмуговий інтернет речей)

PM - Particulate Matter (дрібнодисперсні тверді частки)

RMSE - Root Mean Square Error (середньоквадратична похибка)

SMA - Simple Moving Average (просте ковзне середнє)

VOC - Volatile Organic Compounds (леткі органічні сполуки)

ВСТУП

В умовах стрімкої урбанізації та зростання навантаження на екосистему питання моніторингу навколишнього середовища набуває критичного значення для сталого розвитку суспільства. Традиційні методи лабораторного контролю не здатні забезпечити необхідну оперативність та просторове покриття, тому відповіддю на цей виклик стало розгортання мереж Інтернету речей (IoT), що дозволяють отримувати дані з високою роздільною здатністю [1].

Однак масове використання бюджетних IoT-сенсорів супроводжується проблемою низької якості вхідних даних. Специфіка роботи таких пристроїв призводить до апаратних збоїв, аномальних викидів [2] та систематичного часового дрейфу показників через деградацію елементів [3]. Використання таких «сирих» даних без обробки компрометує надійність усієї системи моніторингу.

Актуальність теми зумовлена тим, що існуючі прості методи фільтрації є неефективними в умовах комбінованих дефектів. Сучасні дослідження вказують на необхідність переходу до автоматизованих конвеєрів, здатних виконувати калібрування в реальному часі [4]. Тому розробка методу, що комплексно вирішує задачі відновлення цілісності та корекції трендів, є важливим завданням.

Мета роботи: підвищення якості очищення сирих даних IoT-сенсорів для моніторингу екологічного стану за рахунок автоматизації обробки на основі комплексного підходу до відновлення та корекції даних.

Для досягнення поставленої мети передбачається:

1. Провести аналіз архітектури IoT-систем та систематизувати типові дефекти вхідних потоків даних.
2. Здійснити порівняльний аналіз існуючих методів обробки часових рядів та обґрунтувати вибір гібридного підходу.
3. Розробити математичну модель сигналу та алгоритм автоматизованого конвеєра очищення, що включає етап де-трендингу.
4. Виконати програмну реалізацію методу та експериментально підтвердити його ефективність.

Об’єкт дослідження: процес очищення сирих даних IoT-сенсорів для моніторингу екологічного стану.

Предмет дослідження: методи та технології автоматизованого очищення сирих даних IoT-сенсорів.

Методи дослідження охоплюють методи системного аналізу для дослідження архітектури IoT; методи математичної статистики та теорії ймовірностей для моделювання шуму та виявлення викидів; методи чисельного аналізу для інтерполяції та апроксимації трендів; методи імітаційного моделювання для перевірки ефективності алгоритму. Програмна реалізація виконана мовою Python.

Наукова новизна одержаних результатів полягає у вдосконаленні методу попередньої обробки потокових екологічних даних шляхом створення автоматизованого конвеєра, який, на відміну від існуючих підходів, поєднує адаптивний поріг виявлення аномалій з поліноміальною корекцією дрейфу сенсорів, що дозволяє підвищити точність відновлення сигналу в умовах деградації обладнання.

Практичне значення роботи відображається у розробці програмного модуля попередньої обробки даних, який може бути інтегрований у платформи «Розумного міста».

Апробація результатів роботи: робота пройшла апробацію на VI Всеукраїнській науково-технічній конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях» [5] та VI Міжнародній науково-технічній конференції «Сучасний стан та перспективи розвитку IoT» [6]. За темою магістерської роботи опубліковано 2 тези доповідей у збірниках матеріалів конференцій.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Особливості побудови сучасних систем екологічного моніторингу

Сучасні системи екологічного моніторингу все частіше переходять від традиційних стаціонарних постів спостереження до гнучких мереж на базі технологій Інтернету речей (IoT). Така трансформація зумовлена необхідністю отримання даних з високою просторовою та часовою роздільною здатністю для контролю критичних показників, таких як концентрація дрібнодисперсних часток (PM2.5, PM10), рівень вуглекислого газу (CO2) та летких органічних сполук (VOC). Згідно з дослідженнями, використання IoT дозволяє суттєво знизити вартість розгортання мережі при збереженні прийнятної точності вимірювань [1].

Архітектура сучасних IoT-систем екологічного моніторингу, як правило, будується за багаторівневим принципом. Для реалізації завдань даної роботи було обрано чотирирівневу модель, яка забезпечує повний цикл роботи з даними: від їх генерації до прийняття управлінських рішень. Загальну структуру системи та взаємодію її компонентів наведено на рисунку 1.1.

Як видно з рисунка 1.1, система складається з наступних рівнів:

1. Рівень збору даних (Perception Layer). Це фізичний рівень, на якому розташовані сенсори та вимірювальні пристрої. В контексті даної роботи сюди входять датчики температури, вологості, газоаналізатори та лазерні лічильники часток. Їхнє завдання — перетворити фізичні параметри навколишнього середовища у цифрові сигнали. Особливістю цього рівня є використання переважно бюджетних сенсорів, які дозволяють розгорнути щільну мережу, але часто генерують потоки зі значним рівнем шуму, що потребує попередньої обробки [2].

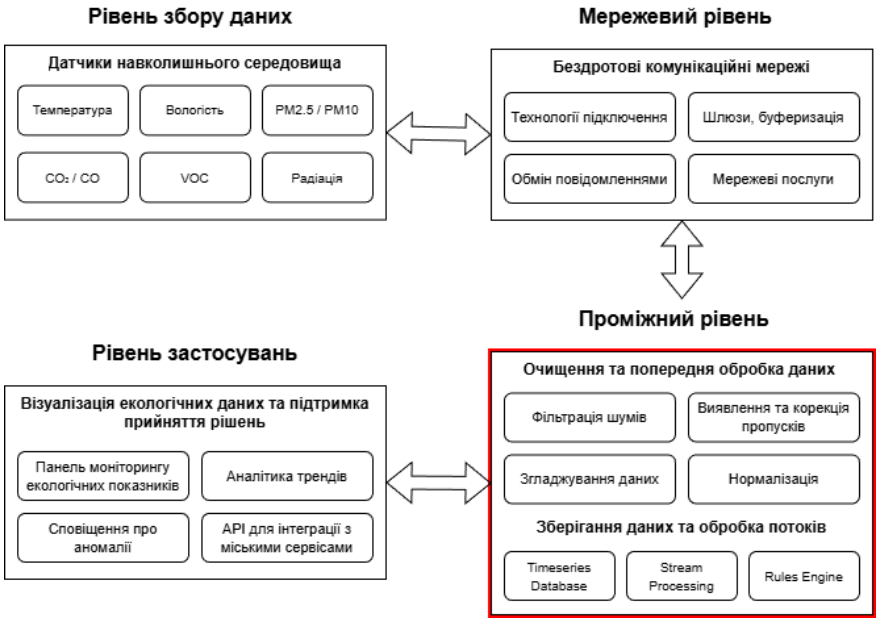


Рис. 1.1 Архітектура системи екологічного моніторингу на базі IoT

2. Мережевий рівень (Network Layer). Відповідає за передачу даних від сенсорів до центрів обробки. Як показано на схемі, зв'язок забезпечується через шлюзи. Вибір технології передачі даних є критичним: для міських умов з щільною забудовою та необхідністю енергоефективності найбільш доцільним є використання технологій LPWAN (Low-Power Wide-Area Network), зокрема протоколу LoRaWAN, який забезпечує стабільну передачу на великі відстані при мінімальному енергоспоживанні [5]. Втрати пакетів на цьому етапі є основною причиною появи пропусків у часових рядах, що відображено у проблематиці роботи.

3. Проміжний рівень (Middleware Layer). Цей рівень є ключовим елементом архітектури, виділеним на рисунку 1.1 як окремий блок обробки. Він виступає сполучною ланкою між «брудними» даними з сенсорів та кінцевими додатками. Функції проміжного рівня включають агрегацію потоків, очищення від шумів та аномалій, а також зберігання даних. Саме тут реалізуються алгоритми фільтрації та корекції дрейфу, необхідність яких обґрунтована у працях з методології оцінювання якості даних [3]. Без ефективного Middleware-шару дані з рівня збору залишаються непридатними для точної аналітики через їх стохастичну природу [4].

4. Рівень застосувань (Application Layer). Верхній рівень схеми, з яким взаємодіють кінцеві користувачі. Він включає дашборди візуалізації та аналітичні модулі. Якість роботи цього рівня (достовірність графіків та прогнозів) напряму залежить від ефективності очищення даних на попередньому етапі.

Важливість такої архітектури полягає в тому, що вона дозволяє ізолювати процеси очищення даних від бізнес-логіки додатків, забезпечуючи гнучкість та масштабованість системи.

1.2 Характеристика та специфіка даних IoT-сенсорів

Дані, що генеруються в системах екологічного моніторингу на базі Інтернету речей, є специфічним класом інформаційних ресурсів, властивості яких кардинально відрізняються від статичних наборів даних у традиційних інформаційних системах. Розуміння цієї специфіки є фундаментальним для побудови архітектури їх збору та попередньої обробки.

Масиви даних, що генеруються IoT-мережами, класифікуються як великі дані (Big Data). У сучасній науковій літературі для їх опису використовується модель «5Vs», основні компоненти якої у контексті екологічного моніторингу наведено на рисунку 1.2.

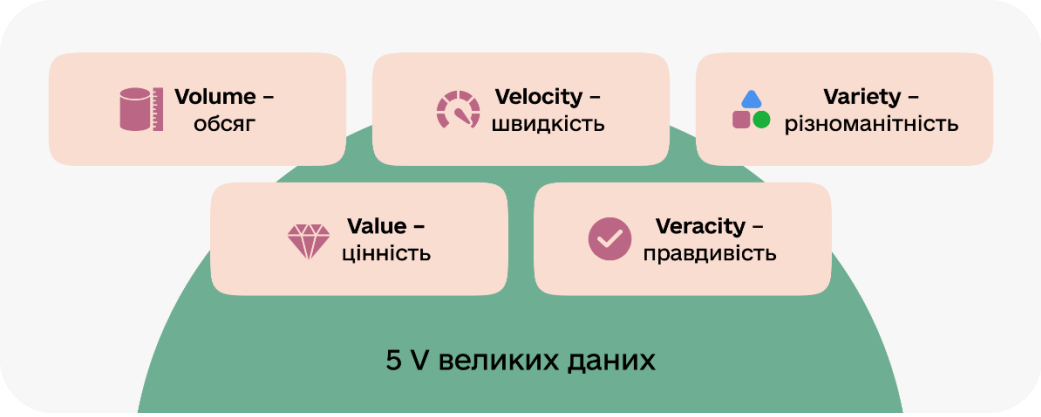


Рис. 1.2 Характеристика даних IoT-сенсорів за моделлю 5Vs

Як видно з рисунка 1.2, специфіка даних визначається п'ятьма ключовими вимірами:

1. Обсяг (Volume): сенсорні мережі генерують величезну кількість вимірювань. Навіть одна станція моніторингу може передавати десятки параметрів щохвилини (температура, вологість, концентрації різних газів), що при масштабуванні на рівень міста формує масивні архіви даних за тривалий період. Це вимагає використання масштабованих сховищ та ефективних алгоритмів компресії.
2. Швидкість (Velocity): дані надходять безперервним потоком у реальному часі. Це вимагає від системи здатності до потокової обробки з мінімальною затримкою, оскільки екологічна ситуація (наприклад, викид шкідливих речовин) може змінюватися миттєво [7]. Традиційні підходи пакетної обробки такі як Batch Processing тут часто є неприйнятними через занадто велику затримку реакції.
3. Різноманітність (Variety): у системах «Розумного міста» дані надходять від різнорідних джерел: стаціонарних постів, мобільних сенсорів на транспорті, персональних гаджетів громадян. Це призводить до значної гетерогенності форматів, одиниць вимірювання та частоти дискретизації, що ускладнює їх інтеграцію [10].
4. Правдивість (Veracity): це найбільш критичний аспект для даної роботи. Ступінь довіри до «сирих» даних IoT є низьким через наявність шумів, аномалій та дрейфу сенсорів. Саме низька достовірність вхідного потоку є головним викликом і обґрунтовує необхідність розробки спеціалізованого алгоритму очищення [9]. Без підвищення показника Veracity подальший аналіз втрачає сенс.
5. Цінність (Value): корисність даних для прийняття рішень. «Сирі» дані мають низьку цінність через зашумленість. Процес очищення та аналітики трансформує їх у цінну інформацію, на основі якої міська влада може приймати рішення про регулювання трафіку або зупинку підприємств.

Окрім моделі 5Vs, для побудови ефективного алгоритму очищення критично важливо враховувати ще три специфічні характеристики екологічних IoT-даних.

По-перше, часова структурованість та автокореляція. Потоки даних IoT являють собою класичні часові ряди, де кожен вимір жорстко прив'язаний до часової мітки. Фундаментальною властивістю таких даних є висока автокореляція, коли поточне значення параметра сильно залежить від попередніх. Крім того, екологічні показники демонструють чітку циклічність, що проявляється у вигляді добових та сезонних коливань температури, інтенсивності трафіку або рівня забруднення. Це створює необхідність у методах обробки, здатних працювати з динамікою методом ковзного вікна, що і було реалізовано у розробленому алгоритмі. Також специфікою є нерівномірність ряду, оскільки інтервали між записами можуть змінюватися через затримки в мережі або перехід сенсорів у режим енергозбереження, що вимагає процедур синхронізації та інтерполяції [8].

По-друге, просторова прив'язка. Вимірювання сенсорів мають сенс лише у контексті конкретної геолокації. Екологічні дані є просторово-часовими, тому значення концентрації забруднювачів у одній точці міста можуть кардинально відрізнятися від значень у сусідньому районі через наявність локальних джерел викидів або особливості забудови та рози вітрів. Хоча в рамках даної роботи розглядається очищення потоку з одного сенсора, розуміння просторової природи даних важливе для інтерпретації аномальних викидів, які можуть бути локальними явищами, а не помилками приладу.

По-третє, контекстуальна залежність. Дані сенсорів не є ізольованими, а їх значення часто корелюють між собою. Наприклад, показники оптичних датчиків пилу PM2.5 суттєво залежать від вологості повітря, оскільки туман може інтерпретуватися приладом як смог, а рівень оксидів азоту NO2 корелює з інтенсивністю трафіку. Цей мультипараметричний контекст створює додаткові виклики для виявлення аномалій, оскільки різкий стрибок одного показника може бути пояснений зміною іншого, а не збоєм обладнання.

Таким чином, екологічні дані, отримані з IoT-сенсорів, формують складний багатовимірний інформаційний простір, у якому поєднуються велика масштабність, висока швидкість надходження, гетерогенність джерел, просторово-часова структуруваність та контекстуальна залежність показників. Сукупність цих властивостей визначає не лише технічні вимоги до побудови архітектури збору та зберігання, але й безпосередньо впливає на методи попередньої обробки, синхронізації та інтерпретації даних. Саме специфіка IoT-потоків, їхня динамічність і комплексність створюють підґрунтя для виникнення широкого спектра типових спотворень, які будуть детально проаналізовані у наступному підрозділі.

1.3 Аналіз типових дефектів та спотворень у потоках екологічних даних

Низький рівень достовірності даних, отриманих від бюджетних IoT-пристроїв, є головною перешкодою для їх безпосереднього використання в системах прийняття рішень. Для розробки ефективного алгоритму очищення необхідно провести детальну декомпозицію вхідного сигналу та проаналізувати природу виникнення спотворень. На основі аналізу досліджень у галузі метрології сенсорних мереж [11, 12], можна виділити чотири класи критичних дефектів, які потребують програмної корекції на рівні Middleware.

Для систематизації конкретних типів спотворень, які знижують показник Veracity, та їх впливу на моніторинг, розроблено таблицю 1.1.

Таблиця 1.1

Основні дефекти якості даних IoT-сенсорів

Тип дефекту	**Природа виникнення**	**Вплив на моніторинг**
Пропуски (Missing Values)	Нестабільність мережі, збої живлення, перезавантаження шлюзу.	Порушення цілісності часових рядів, неможливість аналізу трендів.
Шуми (Sensor Noise)	Електричні наведення, тепловий шум, вібрації, низька якість АЦП.	Зниження точності миттєвих вимірювань, хибні спрацьовування.
Аномальні викиди (Outliers)	Короткочасні збої сенсора, реальні різкі зміни середовища.	Спотворення статистик (середнього, дисперсії), помилкові аларми.
Дрейф (Sensor Drift)	Хімічна деградація сенсора, забруднення, старіння компонентів.	Накопичення систематичної похибки, хибне виявлення трендів.

Розглянемо детальніше фізичну природу цих явищ, оскільки вона визначає вибір математичного апарату для їх усунення.

1. Пропуски даних. Це найбільш поширений структурний дефект. У бездротових сенсорних мережах втрата пакетів є неминучою через інтерференцію радіосигналів в умовах щільної міської забудови. Як зазначають дослідники, відсоток втрат може сягати 5–15% від загального обсягу передачі [13]. З математичної точки зору, наявність пропусків («дірок») у часовому ряді унеможливлює застосування віконних фільтрів та спектрального аналізу, тому першим етапом обробки обов’язково має бути процедура імпутації (відновлення) значень.

2. Вимірювальний шум. Це явище являє собою високочастотні стохастичні флуктуації сигналу навколо істинного значення. У недорогих металооксидних

газових сенсорах типу МОХ шум часто зумовлений температурною нестабільністю нагрівального елемента. Хоча такі коливання зазвичай мають нормальний розподіл із нульовим математичним сподіванням, їх висока амплітуда значно ускладнює фіксацію мікрозмін екологічного стану.

3. Аномальні викиди. Ці артефакти визначаються як різкі та короткочасні відхилення, що не відповідають фізиці вимірюваного процесу. Наприклад, датчик PM2.5 технічно не здатний зафіксувати стрибок концентрації пилу від 10 до 500 мкг/м³ і повернення назад до 10 мкг/м³ за одну секунду. Такі спотворення часто виникають через апаратні помилки, зокрема збій при перетворенні аналогового сигналу в цифровий, або через вплив високої вологості, коли туман помилково сприймається оптичним сенсором як дим. Традиційні методи усереднення лише розмивають такі викиди, тому для їх ефективного усунення необхідні надійні методи, зокрема аналіз міжквартильного розмаху.

4. Дрейф сенсора. Дане явище є найбільш небезпечним типом дефекту для довгострокового моніторингу. Дрейф проявляється як повільне монотонне зміщення нульового рівня або чутливості приладу. Основною причиною є деградація чутливого елемента, що у газових сенсорах виглядає як виснаження хімічного шару або його отруєння домішками, а в оптичних датчиках проявляється через запилення лінз [12]. В результаті сенсор починає показувати зростання забруднення навіть у чистому повітрі. Оскільки дрейф є низькочастотним процесом, звичайні фільтри сприймають його як корисний тренд. Вирішення цієї проблеми вимагає застосування спеціалізованих методів де-трендингу.

Проведений аналіз показав, що дані IoT-сенсорів характеризуються складною структурою спотворень, де випадкові високочастотні помилки у вигляді шуму накладаються на систематичні низькочастотні тренди дрейфу та розриви суцільності через пропуски даних. Існуючі прості методи не здатні комплексно вирішити ці проблеми. Це обґрунтовує необхідність розробки гібридного алгоритму, який поєднував би методи інтерполяції, статистичної детекції аномалій та поліноміальної корекції трендів.

2 АНАЛІЗ МЕТОДІВ ТА ПОБУДОВА МАТЕМАТИЧНОЇ МОДЕЛІ

2.1 Аналіз існуючих методів попередньої обробки сенсорних даних

Проблема низької якості даних у системах Інтернету речей зумовила появу широкого спектру методів їх попередньої обробки. У науковій літературі ці методи класифікують за математичним апаратом та обчислювальною складністю. Для задач екологічного моніторингу, де критичними є вимоги до роботи в реальному часі на пристроях з обмеженими ресурсами, найбільш актуальними є три групи підходів: статистичні фільтри, методи імпутації для відновлення даних та алгоритми на основі машинного навчання.

2.1.1 Статистичні методи згладжування та фільтрації

Ця група методів базується на припущенні, що корисний сигнал є низькочастотним процесом, а шум має високочастотну природу. Найпоширенішим підходом тут є використання віконних фільтрів.

- Просте ковзне середнє. Метод полягає в обчисленні середнього арифметичного значень у вікні фіксованого розміру. Як зазначається в огляді методів обробки IoT-даних, ковзне середнє, або SMA, є найпростішим у реалізації, проте має суттєві недоліки. Головними з них є внесення фазового зсуву в сигнал та чутливість до аномальних викидів, оскільки одне екстремальне значення здатне спотворити результат для всього вікна обробки.
- Медіанна фільтрація. Цей нелінійний метод замінює центральне значення вікна медіаною вибірки. Дослідження показують, що медіанний фільтр є значно ефективнішим за просте усереднення при боротьбі з імпульсними шумами, оскільки він ігнорує екстремальні значення та не розмиває при цьому різкі зміни або фронти корисного сигналу. Однак, як і попередній

метод, він не здатен компенсувати систематичні тренди, такі як дрейф сенсора.

2.1.2. Методи відновлення пропущених даних

Втрата пакетів даних є типовою проблемою бездротових сенсорних мереж. Ігнорування пропусків або заміна їх нулями неминуче призводить до помилок у подальшому аналізі.

- Лінійна інтерполяція. Метод з'єднує дві відомі точки прямою лінією. Для часових рядів екологічних показників, таких як температура або вологість, які мають високу автокореляцію, цей метод забезпечує достатню точність при мінімальних обчислювальних витратах.
- Інтелектуальна імпутація. Існують підходи на основі регресійного аналізу або методу k-найближчих сусідів, проте їхня висока алгоритмічна складність часто робить їх недоцільними для реалізації на рівні проміжного програмного забезпечення.

2.1.3. Методи на основі машинного навчання

Сучасні дослідження, зокрема українських вчених, акцентують увагу на використанні нейронних мереж для виявлення аномалій та очищення даних.

- Глибоке навчання. Моделі на базі рекурентних нейронних мереж RNN та мереж довгої короткострокової пам'яті LSTM здатні виявляти складні нелінійні залежності та прогнозувати значення часового ряду з високою точністю. У роботі [14] показано, що такі моделі ефективно класифікують аномалії у потоках даних.
- Недоліки для IoT. Головним бар'єром для масового впровадження методів машинного навчання на рівні сенсорів є їхня ресурсоємність. Навчання та запуск моделей LSTM вимагає значних обчислювальних потужностей, що суперечить концепції енергоефективних IoT-пристроїв. Крім того, такі

алгоритми потребують великих обсягів розмічених чистих даних для навчання, які часто відсутні на етапі розгортання системи.

Аналіз існуючих підходів свідчить, що жоден із класичних методів у чистому вигляді не здатен комплексно вирішити задачу автоматизованого очищення в умовах наявності комбінованих дефектів, що включають шум, викиди, дрейф та пропуски. Статистичні фільтри не усувають дрейф, а методи машинного навчання є занадто вимогливими до ресурсів. Це обґрунтовує необхідність розробки гібридного методу, який поєднує ефективність статистичних алгоритмів, таких як IQR та медіана, з детермінованими методами корекції трендів.

2.2 Порівняльний аналіз підходів до фільтрації шумів та викидів

На основі проведеного огляду методів попередньої обробки даних можна зробити висновок, що вибір конкретного алгоритму завжди є компромісом між точністю відновлення сигналу, стійкістю до різних типів дефектів та обчислювальною складністю. Для систематизації характеристик розглянутих підходів та виявлення їхніх слабких місць у контексті задач екологічного моніторингу було проведено порівняльний аналіз.

Найбільш поширеними у вбудованих системах залишаються статистичні методи. Їхньою головною перевагою є простота програмної реалізації та висока швидкодія, що дозволяє використовувати їх навіть на мікроконтролерах з обмеженими ресурсами. Проте аналіз показує, що лінійні методи, такі як ковзне середнє, суттєво спотворюють сигнал при наявності аномальних викидів, оскільки одне помилкове значення впливає на результат усього вікна усереднення. Нелінійні підходи, зокрема медіанна фільтрація, ефективно усувають імпульсні шуми, але, як і лінійні фільтри, не здатні розрізняти корисний тренд зміни показника та паразитний дрейф сенсора [4].

Ймовірнісні методи, серед яких стандартом є фільтр Калмана, забезпечують оптимальне згладжування для процесів з нормальним розподілом шуму. Вони добре враховують динаміку системи, проте їх ефективність різко знижується в умовах реальної експлуатації бюджетних IoT-сенсорів. Головною проблемою є те, що фільтр Калмана вимагає точної апріорної інформації про статистичні характеристики шуму, яка часто є невідомою або змінюється в часі. Крім того, цей метод не вирішує проблему відновлення даних при тривалих перервах у зв'язку [2].

Методи машинного навчання та глибокі нейронні мережі демонструють найвищу точність при виявленні складних патернів аномалій та прогнозуванні значень. Вони здатні адаптуватися до нелінійної поведінки сенсорів. Однак, як зазначається у дослідженні [14], головним бар'єром для їх масового впровадження є висока залежність від якості навчальних вибірок та значна ресурсоємність. Розгортання таких моделей на рівні периферійних обчислень часто є економічно недоцільним через підвищене енергоспоживання.

Узагальнені результати порівняльного аналізу переваг та недоліків розглянутих методів наведено в таблиці 2.1.

Окремо варто виділити підхід Data Fusion, або злиття даних, який передбачає використання інформації з декількох різнорідних сенсорів для підвищення достовірності вимірювань. Цей метод дозволяє частково компенсувати похибки окремих пристроїв, проте він є вторинним етапом обробки і неможливий без якісного первинного очищення кожного окремого потоку даних [15].

Проведений аналіз дозволяє стверджувати, що жоден із класичних методів у чистому вигляді не здатен комплексно вирішити задачу автоматизованого очищення екологічних даних. Статистичні фільтри є швидкими, але пропускають дрейф, ймовірнісні методи є складними в налаштуванні, а алгоритми машинного навчання вимагають значних ресурсів. Це обґрунтовує доцільність розробки гібридного методу, який би поєднував алгоритми імпутації для відновлення пропусків, робастну статистику для видалення викидів та

поліноміальну апроксимацію для корекції трендів. Саме такий підхід дозволить нівелювати недоліки окремих методів та забезпечити високу якість даних.

Таблиця 2.1

Порівняльний аналіз методів обробки даних IoT

Категорія методів	**Переваги**	**Недоліки**
Статистичні методи (середнє, медіана, ковзне вікно)	Простота, швидкість	- Спотворюють піки та різкі зміни - Не враховують нелінійні шумові ефекти - Не працюють із дрейфом сенсорів
Фільтр Калмана	Добре згладжує шум	- Не виявляє аномалії - Важко налаштовується для різних сенсорів - Не відновлює великі пропуски
Методи імпутації (інтерполяція, KNN, регресія)	Відновлюють пропуски даних	- Дає неточні результати при довгих провалах - Не відрізняє аномалію від помилки - Залежить від якості сусідніх точок
Алгоритми ML(аномалії, прогнозування)	Виявляють складні закономірності	- Потребують попередньо очищених даних - Висока залежність від навчальних вибірок - Низька ефективність на сирих потоках
Data Fusion (злиття даних сенсорів)	Покращує точність після обробки	- Неможливий без попереднього очищення - Не компенсує шум та дрейф - Потребує уніфікації форматів і калібрування

2.3 Математична модель задачі очищення даних

Розробка алгоритмів попередньої обробки вимагає формалізованого опису процесів, що відбуваються у вимірювальному каналі. На основі аналізу фізичної природи функціонування сенсорів, проведеного у першому розділі, доцільно застосувати принцип адитивної декомпозиції. Він полягає у представленні вхідного потоку даних як суперпозиції істинного значення вимірюваного параметра та низки незалежних компонентів похибок різної природи. Такий підхід дозволяє перейти від роботи з абстрактним «брудним» сигналом до математичного опису окремих типів завад.

У загальному вигляді математична модель виміряного сигналу в дискретні моменти часу *t* описується рівнянням (2.1):

$$Y(t) = S(t) + N(t) + O(t) + \mathrm{D(t)} + E_{miss}(t), \qquad (2.1)$$

де Y(t) – позначає отримане значення із сенсора або сирий сигнал;

S(t) – є істинним значенням вимірюваного екологічного параметра, що підлягає відновленню;

N(t) – описує високочастотну шумову компоненту стохастичної природи;

O(t) – відповідає за компоненту аномальних викидів;

D(t) – представляє низькочастотний тренд дрейфу сенсора;

функція $E_{miss}(t)$ – моделює дискретну втрату даних.

Розглянемо детальний математичний опис та обґрунтування кожної компоненти моделі.

1. Стохастична модель вимірювального шуму. Шумова складова N(t) у вимірювальних системах Інтернету речей формується під впливом великої кількості незалежних випадкових факторів. До них належать теплові процеси в напівпровідникових елементах, електромагнітні наведення в лініях передачі, а також похибки квантування, що виникають у процесі роботи аналого-цифрового

перетворювача (АЦП) - електронного вузла, який трансформує безперервний електричний сигнал сенсора у дискретний цифровий код.

Згідно з центральною граничною теоремою теорії ймовірностей, сукупний вплив значної кількості незалежних випадкових величин, жодна з яких не домінує, формує результуючий розподіл, що асимптотично наближається до нормального (гаусівського). Тому в роботі прийнято обґрунтовану гіпотезу, що N(t) є адитивним білим шумом, щільність ймовірності якого описується формулою нормального розподілу(2.2):

$$N(t) \sim \mathcal{N}(0, \sigma^2), \tag{2.2}$$

де $\mathcal{N}(0, \sigma^2)$ — нормальний розподіл із нульовим математичним сподіванням та дисперсією σ^2;

σ^2 — інтенсивність шуму (варіативність).

Припущення про нормальний розподіл та нульове середнє є критично важливим, оскільки воно дозволяє використовувати методи локального усереднення (такі як ковзне середнє або медіанна фільтрація) як математично оптимальні оцінки для відновлення сигналу на фоні такого типу завад [16].

2. Статистична модель аномальних викидів. Компонента O(t) описує різкі короткочасні відхилення, амплітуда яких виходить за межі фізично можливої динаміки зміни параметра. Для їх ідентифікації використання стандартного відхилення є неефективним через його чутливість до самих викидів. Тому застосовано надійний підхід на основі міжквартильного розмаху. Умова наявності викиду визначається виходом значення за межі адаптивного довірчого інтервалу (2.3):

$$O(t) = \begin{cases} Y(t), & якщо\ Y(t) \notin [Q_1; -k \cdot IQR\ Q_3 + k \cdot IQR] \\ 0, & інакше \end{cases}, \tag{2.3}$$

де Q_1, Q_3 — перший та третій квартилі вибірки;

$IQR = Q_3 - Q_1$ —міжквартильний розмах;

k— коефіцієнт чутливості (типово 1.5–3);

$[Q_1; -k \cdot IQR\ Q_3 + k \cdot IQR]$ —інтервал допустимих значень.

3. Апроксимаційна модель дрейфу. Компонента D(t) являє собою систематичну похибку, що монотонно змінюється через деградацію сенсора. Аналіз показує, що на інтервалах між калібруваннями дрейф має детермінований характер і апроксимується поліномом першого порядку (2.4):

$$\mathrm{D}(t) \approx a \cdot t + b, \tag{2.4}$$

де *a* коефіцієнт а характеризує швидкість дрейфу;

b — початкове зміщення.

Врахування цієї компоненти дозволяє застосувати методи регресійного аналізу для її компенсації.

4. Модель пропусків. Функція $E_{miss}(t)$ формалізує порушення неперервності потоку даних, що виникають внаслідок нестабільності каналу зв'язку або апаратних збоїв. Математично вона визначається як бінарний індикатор, який ідентифікує моменти часу, коли корисний сигнал відсутній. Аналітичний вираз цієї функції має вигляд (2.5):

$$E_{miss}(t) = \begin{cases} 1, & якщо\ Y(t) = \emptyset \\ 0, & інакше \end{cases}, \tag{2.5}$$

де символом $\emptyset$ позначено подію втрати пакету даних або отримання невизначеного значення типу NaN.

Якщо функція набуває значення одиниці, це сигналізує про розрив у часовому ряді. Наявність такої компоненти у моделі є критичною, оскільки вона вказує на необхідність виконання процедури імпутації перед застосуванням будь-яких алгоритмів фільтрації, адже математичні операції над пустими множинами є невизначеними.

Таким чином, розроблена математична модель (2.1) забезпечує вичерпний формалізований опис структури вхідного потоку даних у системах екологічного моніторингу. Представлення виміряного сигналу у вигляді адитивної суміші корисного компонента та незалежних завад різної природи (стохастичного шуму, імпульсних викидів, детермінованого дрейфу та структурних пропусків) створює необхідну теоретичну базу для синтезу ефективного методу очищення. Такий підхід дозволяє перейти від емпіричного підбору фільтрів до побудови обґрунтованого алгоритмічного конвеєра, кожен етап якого буде спрямований на мінімізацію конкретної, математично визначеної складової похибки.

2.4 Метод автоматизованого очищення даних у системах моніторингу

На основі побудованої в попередньому підрозділі математичної моделі розроблено метод попередньої обробки даних, який реалізується у вигляді послідовного автоматизованого конвеєра. На відміну від існуючих підходів, які розглядають задачі фільтрації ізольовано, запропонований метод базується на принципі ієрархічного усунення дефектів. Обробка відбувається від відновлення структурної цілісності ряду до корекції грубих помилок і, нарешті, до тонкого згладжування шуму.

Алгоритмічна реалізація методу складається з чотирьох взаємопов’язаних етапів, порядок виконання яких є суворо детермінованим. Загальну схему алгоритму наведено на рисунку 2.1.

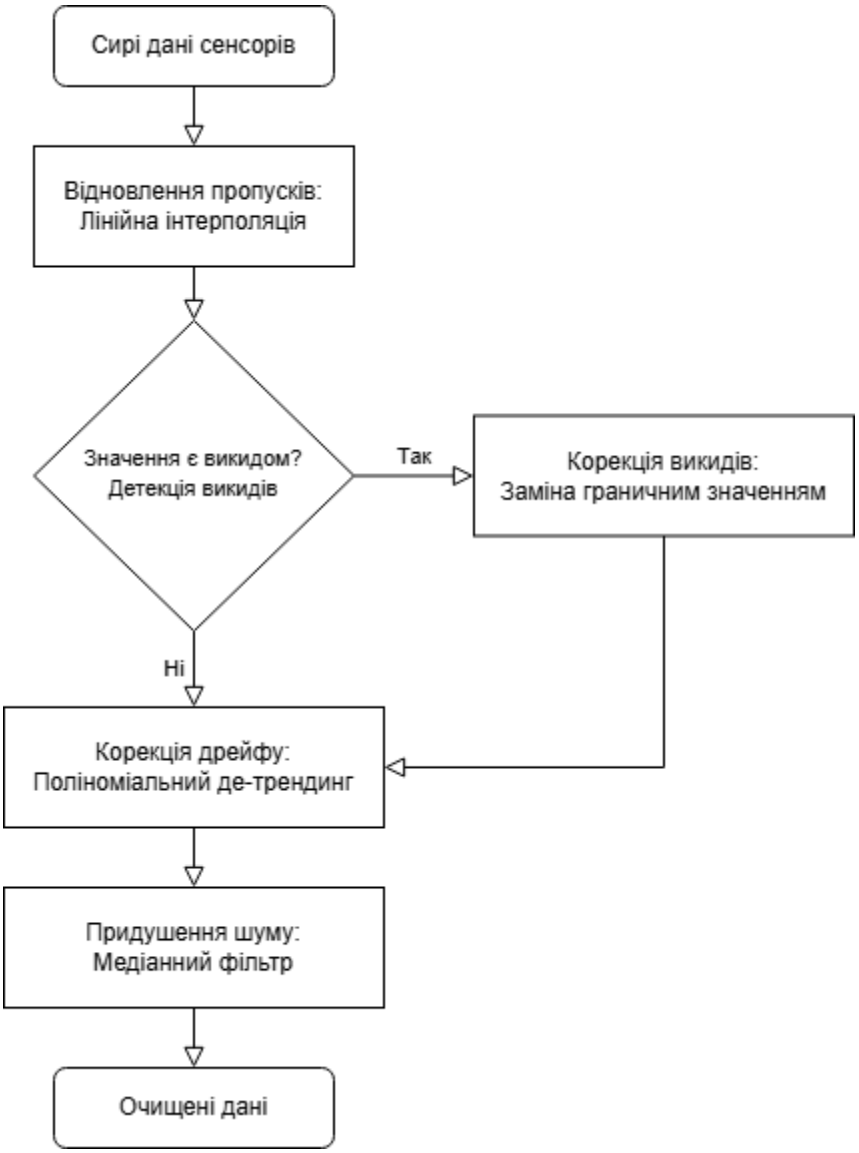


Рис. 2.1 Блок-схема алгоритму автоматизованого очищення даних

Розглянемо детальний опис та математичне обґрунтування кожного етапу.

Етап 1. Імпутація даних. Першочерговим завданням є відновлення неперервності часового ряду. Вхідний потік даних перевіряється на наявність пропусків, що відповідають одиничному значенню функції втрат у математичній моделі. Виконання цього кроку на самому початку є критично важливим, оскільки наявність розривів унеможливлює коректну роботу віконних функцій та статистичних оцінок на наступних етапах.

Для заповнення коротких проміжків, які складають до 5% від довжини вікна спостереження, використовується метод лінійної інтерполяції. Він полягає

у з’єднанні двох відомих точок, що межують з пропуском, прямою лінією та обчисленні проміжних значень на цій прямій. Такий підхід є обчислювально ефективним та забезпечує достатню точність для інерційних екологічних процесів, що підтверджується дослідженнями в галузі відновлення сенсорних даних [13, 18].

Етап 2. Детекція та корекція викидів. На відновленому сигналі застосовується процедура виявлення аномалій. Важливою особливістю методу є усунення аномальних піків саме до етапу оцінки тренду. Це пояснюється тим, що класичні методи регресійного аналізу, які використовуються для виявлення дрейфу, є вкрай чутливими до викидів. Навіть одне екстремальне значення здатне суттєво викривити лінію тренду, що призведе до помилок на етапі де-трендингу [11]. Для реалізації цього етапу використовується адаптивний фільтр на основі міжквартильного розмаху.

Алгоритм розраховує квартилі розподілу у ковзному вікні та формує динамічний довірчий інтервал згідно з рекомендаціями класичної математичної статистики [17]. Значення, що виходять за його межі, класифікуються як викиди та замінюються медіаною локального вікна, що дозволяє зберегти загальну тенденцію сигналу без внесення спотворень.

Етап 3. Корекція дрейфу (Де-трендинг). Цей етап забезпечує основну наукову новизну методу та його перевагу над простими фільтрами. З очищеного від викидів сигналу виділяється низькочастотний тренд, що відповідає дрейфу сенсора. Для цього використовується метод найменших квадратів, який апроксимує систематичну похибку поліномом першого ступеня. Отримана функція тренду віднімається від основного сигналу. Ця операція, відома як де-трендинг, дозволяє вирівняти часовий ряд відносно горизонтальної осі або реального середнього значення, ефективно компенсуючи накопичену похибку деградації сенсора, описану в роботах з метрології газоаналізаторів [12].

Етап 4. Згладжування залишкового шуму. На фінальному етапі виконується придушення високочастотної стохастичної компоненти. Для цього застосовується медіанний фільтр з малим розміром вікна. Вибір медіани замість

середнього арифметичного обумовлений необхідністю збереження різких фронтів реальних змін екологічних показників. Наприклад, у випадку різкого зростання концентрації чадного газу при пожежі, медіанний фільтр збереже крутизну фронту сигналу, тоді як ковзне середнє розмиє його, вносячи небажану затримку в роботу системи оповіщення [16].

Таким чином, запропонований метод являє собою комплексну алгоритмічну процедуру, яка дозволяє автоматизувати процес попередньої обробки «сирих» даних. Ключовою перевагою розробленого підходу є сувора ієрархічна організація етапів, що забезпечує мінімізацію взаємного негативного впливу різних типів завад, зокрема викривлення лінії тренду аномальними викидами. Така структура конвеєра створює необхідну теоретичну та алгоритмічну базу для програмної реалізації системи очищення, опис якої наведено у наступному розділі.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Обґрунтування вибору засобів розробки

Реалізація методу автоматизованого очищення даних IoT-сенсорів вимагає застосування програмних засобів, здатних ефективно працювати з великими масивами числової інформації, забезпечувати високу швидкість виконання математичних операцій та надавати гнучкі інструменти для роботи з часовими рядами. Враховуючи, що задача лежить на перетині інженерії програмного забезпечення, математичного моделювання та науки про дані, вибір технологічного стеку базувався на критеріях продуктивності, швидкості розробки та наявності спеціалізованих бібліотек.

3.1.1 Вибір мови програмування

Як базову мову програмування для реалізації системи було обрано **Python**. Цей вибір зумовлений низкою факторів, що роблять Python домінуючим інструментом у сфері аналізу даних та розробки інтернету речей. Для обґрунтування цього рішення було проведено порівняльний аналіз ключових характеристик мов програмування, результати якого наведено в таблиці 3.1.

1. Баланс між швидкістю розробки та продуктивністю. Традиційно для систем реального часу та вбудованих пристроїв використовують мови низького рівня, такі як C або C++. Вони забезпечують максимальну швидкість виконання коду та прямий доступ до пам'яті. Однак розробка складних алгоритмів обробки даних, зокрема статистичних фільтрів та регресійного аналізу, на C++ вимагає значних витрат часу на написання низькорівневого коду та управління пам'яттю [19]. Python, будучи інтерпретованою мовою високого рівня, дозволяє зосередитися на логіці алгоритму, а не на деталях його реалізації.

Таблиця 3.1

Порівняльний аналіз мов програмування для реалізації системи очищення даних

Критерій порівняння	**Python (з NumPy/Pandas)**	**C / C++**	**Обґрунтування вибору**
Швидкість розробки (Time-to-Market)	Висока	Низька	Python дозволяє реалізувати складні алгоритми в декілька рядків коду завдяки високорівневим абстракціям.
Швидкість виконання обчислень	Середня (Висока при векторизації)	Дуже висока	Використання C-оптимізованих бібліотек (NumPy) нівелює відставання Python у швидкості для матричних операцій.
Управління пам'яттю	Автоматичне (Garbage Collection)	Ручне	Автоматичне управління знижує ризик помилок (витоків пам'яті) на етапі прототипування.
Наявність наукових бібліотек	Дуже велика (SciPy, Pandas, Sklearn)	Обмежена, часто потребує написання з нуля	Наявність готових перевірених реалізацій методів (інтерполяція, регресія) прискорює розробку.
Інтеграція з IoT-платформами	Вбудована підтримка	Потребує додаткових бібліотек	Python є стандартом для написання скриптів обробки на шлюзах та у хмарі.

Хоча сам по собі Python є повільнішим за C++, використання спеціалізованих бібліотек, таких як NumPy та Pandas, написаних на C та Fortran, дозволяє досягти продуктивності на рівні компільованих мов [20]. Це досягається завдяки механізму векторизації обчислень, що є критичним для обробки потоків даних сенсорів.

2. Екосистема та підтримка наукових обчислень. Python має найрозвиненішу у світі екосистему пакетів для наукових досліджень. Наявність перевірених та оптимізованих реалізацій математичних методів для інтерполяції, статистики та апроксимації дозволяє уникнути помилок, можливих при написанні цих алгоритмів з нуля [21].

3. Інтеграційні можливості. У контексті архітектури IoT та рівня проміжного програмного забезпечення модуль очищення часто повинен взаємодіяти з веб-сервісами, брокерами повідомлень протоколу MQTT та базами даних. Python має вбудовану підтримку всіх сучасних протоколів передачі даних, що спрощує інтеграцію розробленого модуля в існуючі хмарні платформи типу AWS IoT або Google Cloud IoT, а також локальні сервери.

3.1.2 Бібліотеки для обробки та аналізу даних

Для реалізації кожного етапу розробленого конвеєра, що включає імпутацію, детектування викидів, де-трендинг та згладжування, було використано спеціалізовані бібліотеки, які складають ядро наукового стеку Python.

NumPy це фундаментальна бібліотека, яка забезпечує підтримку багатовимірних масивів та матриць.

- Обґрунтування використання. Стандартні списки Python є повільними та споживають багато пам'яті через накладні витрати на зберігання типу даних для кожного елемента. Масиви numpy.ndarray зберігають дані одного типу в неперервній області пам'яті, що дозволяє процесору використовувати векторні інструкції SIMD.

- Роль у роботі. У розробленому алгоритмі NumPy використовується для реалізації методу найменших квадратів через функцію np.polyfit() на етапі корекції дрейфу. Ця функція виконує поліноміальну регресію значно швидше, ніж аналогічні алгоритми ітераційного типу [20]. Також модуль numpy.random застосовано для генерації синтетичних даних, що імітують шум та викиди під час експериментального дослідження. Крім того, бібліотека дозволяє виконувати поелементні арифметичні операції над масивами, наприклад віднімання тренду від сигналу, без використання повільних циклів.

Бібліотека Pandas є високорівневою надбудовою над NumPy, розробленою спеціально для роботи з табличними даними та часовими рядами.

- Обґрунтування використання. Робота з даними IoT-сенсорів передбачає маніпулювання часовими мітками, індексацію за часом та обробку пропущених значень. Pandas надає об'єкти DataFrame та Series, які ідеально підходять для цих задач.
- Роль у роботі. Бібліотека забезпечує автоматичне вирівнювання даних за часовою віссю, що важливо при аналізі нерівномірних рядів. Метод .interpolate() став основою першого етапу алгоритму, дозволяючи відновити цілісність ряду однією командою [21]. Метод .rolling() дозволив реалізувати адаптивний IQR та медіанну фільтрацію. Цей метод автоматично створює вікна заданого розміру та застосовує до них агрегуючі функції типу медіани або стандартного відхилення, оптимізуючи цей процес на рівні C-коду. Без Pandas реалізація ковзного вікна вимагала б написання складного та потенційно повільного коду.

Scikit-learn це бібліотека для машинного навчання, яка містить ефективні реалізації алгоритмів класифікації, регресії та кластеризації, а також інструменти для оцінки якості моделей.

- Обґрунтування використання. Хоча основний алгоритм очищення є детермінованим, для наукового обґрунтування його ефективності необхідно використовувати об'єктивні метрики якості.

- Роль у роботі. Бібліотека використана на етапі валідації результатів. Модуль sklearn.metrics надає функцію mean_squared_error, яка була використана для розрахунку середньоквадратичної похибки RMSE між еталонним та очищеним сигналами. Це дозволило отримати кількісну оцінку покращення якості даних та порівняти розроблений метод з аналогами.

Бібліотека Matplotlib це основний інструмент для візуалізації даних у Python.

- Обґрунтування використання. У науковій роботі візуальне представлення результатів є не менш важливим за чисельні показники. Matplotlib дозволяє створювати графіки публікаційної якості з повним контролем над кожним елементом, включаючи осі, легенди, кольори та шрифти.
- Роль у роботі. Бібліотека використана для побудови порівняльних графіків сирого та очищеного сигналів, візуалізації виявленого тренду дрейфу та побудови гістограм ефективності методів. Можливість накладання декількох шарів на один графік дозволила наочно продемонструвати роботу кожного етапу конвеєра [22].

3.1.3 Середовище розробки та виконання

Для програмної реалізації, відлагодження алгоритмів та проведення експериментальних досліджень було обрано хмарне інтерактивне середовище Google Colaboratory. Цей вибір зумовлений необхідністю швидкого прототипування та забезпечення відтворюваності отриманих наукових результатів.

Хмарна архітектура та доступність ресурсів. Google Colaboratory є безкоштовним хмарним сервісом, що надає готове до використання середовище Jupyter Notebook. Ключовою перевагою платформи є відсутність необхідності локального налаштування робочого оточення. Усі необхідні для роботи бібліотеки наукового стеку Python, такі як Pandas, NumPy, Scikit-learn та

Matplotlib, є попередньо встановленими та оптимізованими для роботи на інфраструктурі Google. Це дозволило зосередитися безпосередньо на реалізації алгоритму очищення, не витрачаючи час на адміністрування пакетів та вирішення конфліктів залежностей. Крім того, платформа надає доступ до потужних обчислювальних ресурсів, що дозволяє швидко обробляти великі масиви даних під час моделювання експериментальних сценаріїв.

Інтеграція та збереження даних. Важливим аспектом вибору середовища стала глибока інтеграція з хмарним сховищем Google Drive. У рамках роботи це забезпечило надійне збереження вхідних наборів даних, скриптів та результатів моделювання в єдиному захищеному просторі. Доступ до файлової системи здійснюється через монтування віртуального диску, що дозволяє працювати з файлами так само прозоро, як і в локальній операційній системі. Такий підхід гарантує збереження результатів дослідження незалежно від локального обладнання.

Інтерактивність та візуалізація. Формат записника дозволяє поєднувати в одному документі блоки виконуваного коду, результати обчислень, графічні візуалізації та форматований текстовий опис. Для задачі дослідження методів фільтрації це є критично важливим, оскільки дозволяє спостерігати проміжні результати на кожному етапі конвеєра. Зміна параметрів алгоритму, наприклад розміру вікна згладжування або коефіцієнта чутливості IQR, дозволяє миттєво перерахувати результати та оновити графіки, що значно прискорює процес підбору оптимальних налаштувань.

Таким чином, використання Google Colaboratory забезпечило ефективний, гнучкий та надійний інструментарій для розробки та експериментальної перевірки запропонованого методу очищення даних.

3.1.4 Архітектурна сумісність

Обраний технологічний стек на базі Python, Pandas та NumPy є загальновизнаним галузевим стандартом для обробки даних, що забезпечує високу архітектурну сумісність та переносимість розробленого рішення. Програмний код, створений з використанням цих бібліотек, може бути легко інтегрований у промислові системи без суттєвих модифікацій. Наприклад, перенесення алгоритму із середовища прототипування у виробниче середовище часто реалізується шляхом розгортання мікросервісів у контейнерах Docker.

Крім того, кросплатформеність Python та наявність оптимізованих версій бібліотек для архітектури ARM відкриває можливості для розгортання розробленого модуля безпосередньо на апаратних шлюзах інтернету речей, таких як одноплатні комп'ютери Raspberry Pi. Це дозволяє реалізувати концепцію граничних обчислень, коли первинна обробка та очищення даних відбуваються безпосередньо на межі мережі, зменшуючи навантаження на канали зв'язку та хмарні сервери [19].

Аналіз вимог до системи та функціональних можливостей сучасних інструментальних засобів підтвердив, що екосистема Python є оптимальним вибором для вирішення поставлених завдань. Застосування бібліотек NumPy та Pandas дозволяє ефективно поєднати простоту високорівневого програмування з продуктивністю низькорівневих обчислень, а використання хмарного середовища Google Colaboratory забезпечує гнучкість та ефективність проведення експериментальних досліджень.

3.2 Архітектура програмного модуля

Програмна реалізація системи автоматизованого очищення даних побудована за модульним принципом, що забезпечує гнучкість налаштування параметрів експерименту та можливість незалежного тестування окремих етапів обробки. Архітектура розробленого рішення складається з трьох основних

функціональних блоків: модуля генерації синтетичних даних, модуля попередньої обробки у вигляді конвеєра очищення та модуля оцінки ефективності й візуалізації. Взаємодію цих компонентів та потоки даних у системі відображено на діаграмі компонентів, яку наведено на рисунку 3.1.

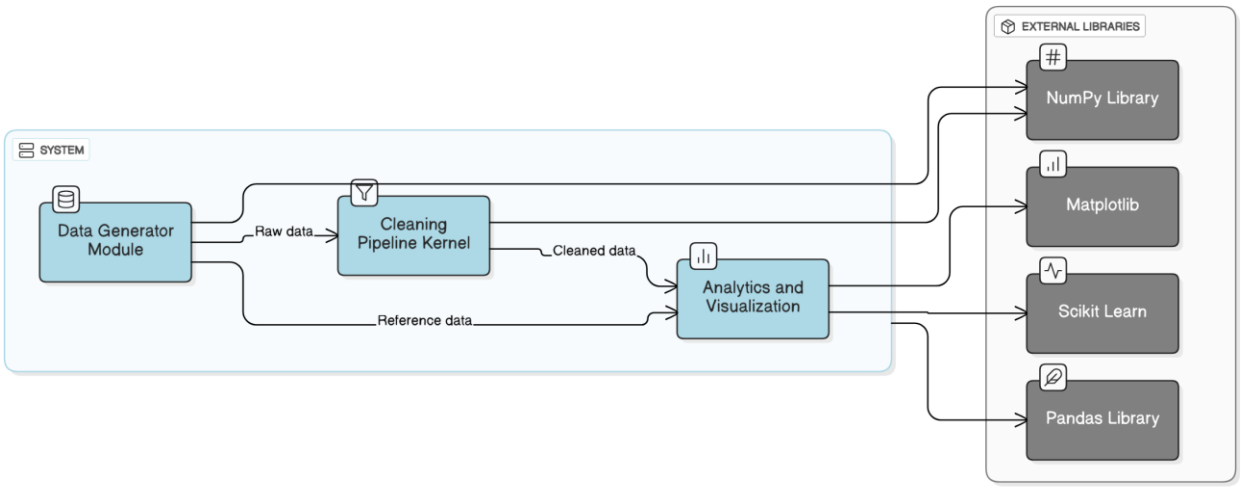


Рис. 3.1 Діаграма компонентів програмного модуля

Першим ключовим елементом архітектури є блок імітаційного моделювання, реалізований у вигляді функції генерації даних. Його головне призначення полягає у створенні контрольованого експериментального середовища, де відомий істинний сигнал, який у задачах машинного навчання часто називають Ground Truth. Наявність еталонного сигналу дозволяє об’єктивно оцінювати якість очищення, що неможливо зробити на реальних даних через відсутність достовірної інформації про істинні значення вимірюваного параметра. Цей модуль програмно формує ідеальний сигнал як комбінацію гармонічних функцій, після чого накладає на нього модельовані спотворення. До таких спотворень належать адитивний білий гаусівський шум із заданою дисперсією, випадкові аномальні викиди, що реалізуються шляхом додавання значних зміщень у випадкових точках, та лінійний дрейф, який моделює деградацію сенсора у часі. Також на цьому етапі відбувається імітація втрати пакетів даних шляхом випадкового видалення частини значень та

присвоєння їм невизначеного статусу NaN, що дозволяє перевірити стійкість системи до порушення цілісності часового ряду.

Центральним ядром системи є блок попередньої обробки, реалізований функцією конвеєра очищення під назвою cleaning_pipeline. Цей компонент приймає на вхід забруднений набір даних у форматі Pandas DataFrame і послідовно пропускає його через каскад фільтрів. Архітектурно цей блок побудований як конвеєр обробки даних, де вихідні дані одного етапу автоматично стають вхідними для наступного. Спочатку виконується модуль інтерполяції, який відновлює неперервність потоку шляхом заповнення пропусків розрахованими значеннями. Далі дані надходять до модуля детекції аномалій, що використовує алгоритм ковзного вікна для розрахунку адаптивних порогів на основі міжквартильного розмаху IQR. Очищений від викидів сигнал передається до модуля де-трендингу, який за допомогою поліноміальної регресії виділяє та компенсує низькочастотну складову дрейфу, повертаючи сигнал до коректної базової лінії. На фінальному етапі працює модуль згладжування на базі медіанного фільтра, який пригнічує залишкові високочастотні шуми. Результатом роботи блоку є очищений часовий ряд та параметри виявленого тренду, які можуть бути використані для діагностики стану сенсора.

Третім компонентом є модуль аналізу та візуалізації, який відповідає за розрахунок метрик якості та графічне представлення результатів. Він виконує порівняння вихідного сигналу конвеєра з еталонним сигналом, згенерованим на першому етапі, та обчислює середньоквадратичну похибку RMSE. Використання бібліотеки Matplotlib дозволяє візуалізувати накладені графіки сирого, еталонного та очищеного сигналів, що дає можливість якісно оцінити роботу алгоритму, зокрема коректність виявлення дрейфу та повноту усунення аномалій. Така архітектура дозволяє проводити серії експериментів із різними параметрами шумів та налаштуваннями фільтрів, забезпечуючи високу точність та відтворюваність досліджень.

3.3 Реалізація методу очищення

Програмна реалізація розробленого методу базується на використанні об'єктно-орієнтованих структур даних бібліотеки Pandas. Алгоритм оформлено у вигляді функції-конвеєра, яка послідовно трансформує вхідний набір даних.

Для деталізації програмної логіки та внутрішньої структури розробленого рішення було спроєктовано діаграму класів, наведену на рисунку 3.2.

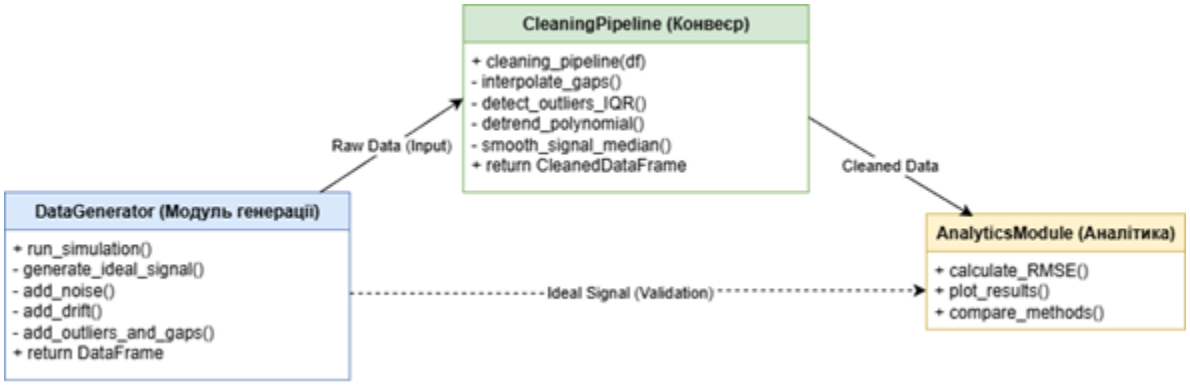


Рис. 3.2 Діаграма класів програмного модуля очищення даних

Як видно з діаграми, архітектура рішення базується на трьох основних класах. Основний клас CleaningPipeline інкапсулює методи обробки: interpolate_gaps() відповідає за відновлення цілісності ряду, detect_outliers_IQR() реалізує логіку виявлення аномалій, а detrend_polynomial() виконує компенсацію дрейфу. Допоміжний клас DataGenerator забезпечує створення тестового середовища, а AnalyticsModule відповідає за верифікацію результатів.

Нижче наведено опис програмної логіки для кожного з чотирьох етапів обробки.

На першому кроці виконується відновлення втрачених значень. У вхідному масиві пропуски позначено стандартним значенням NaN, що розшифровується як "не число". Для їх заповнення обрано метод лінійної інтерполяції, оскільки екологічні процеси, такі як зміна температури або концентрації газів, мають інерційну природу і не змінюються миттєво.

Програмно це реалізовано наступним викликом методу бібліотеки Pandas:

```
# Відновлення пропусків методом лінійної інтерполяції
df['step1'] = df['raw'].interpolate(method='linear')
# Обробка крайових значень
df['step1'] = df['step1'].bfill().ffill()
```

Логіка роботи полягає у тому, що функція interpolate знаходить розрив між двома відомими точками та обчислює проміжні значення, які лежать на прямій лінії, що їх з'єднує. Додаткові методи bfill та ffill застосовуються для заповнення пропусків, що можуть знаходитися на самому початку або в кінці часового ряду, де інтерполяція неможлива через відсутність сусідніх точок з одного боку.

Для виявлення викидів використано підхід на основі ковзного вікна. На відміну від статичних методів, які обчислюють поріг для всього файлу, тут статистичні параметри розраховуються для локальної ділянки даних.

Фрагмент коду, що реалізує розрахунок адаптивних меж:

```
# Розрахунок локальної медіани та відхилення у ковзному вікні
rolling_center = df['step1'].rolling(window=50, center=True).median()
rolling_std = df['step1'].rolling(window=50, center=True).std()
# Формування динамічних меж викидів
lower_bound = rolling_center - 3 * rolling_std
upper_bound = rolling_center + 3 * rolling_std
```

Ключовим елементом тут є метод rolling, який створює віртуальне вікно розміром 50 відліків. У цьому вікні обчислюється медіана як центр розподілу та стандартне відхилення як міра варіабельності. Використання медіани замість середнього арифметичного робить сам фільтр стійким до впливу аномалій. Далі формується логічна маска, яка маркує як викиди всі точки, що виходять за межі розрахованого динамічного коридору.

Етап усунення систематичної похибки сенсора реалізовано за допомогою поліноміальної апроксимації. Завдання полягає у знаходженні параметрів прямої лінії, яка найкраще описує загальний нахил графіку, ігноруючи локальні коливання.

Реалізація за допомогою бібліотеки NumPy виглядає так:

```
# Апроксимація тренду поліномом 1-го ступеня (пряма лінія)
slope, intercept = np.polyfit(time_axis, df['step2'], 1)
# Розрахунок лінії тренду та її віднімання
trend_line = slope * time_axis + intercept
df['step3'] = df['step2'] - trend_line
```

Функція polyfit реалізує метод найменших квадратів. Вона приймає масив часових міток та масив значень сигналу, повертаючи коефіцієнти нахилу та зміщення. Параметр 1 вказує на те, що шукається поліном першого ступеня. Після отримання коефіцієнтів розраховується вектор значень тренду, який віднімається від основного сигналу. Ця операція повертає графік у горизонтальне положення, компенсуючи накопичену похибку сенсора.

На останньому етапі виконується придушення залишкового високочастотного шуму. Для цього застосовано медіанний фільтр з невеликим розміром вікна.

```
# Фінальне згладжування медіанним фільтром
df['final'] = df['step3'].rolling(window=7, center=True).median()
```

Вибір розміру вікна у 7 відліків є емпіричним компромісом між якістю придушення шуму та збереженням корисної деталізації сигналу. Метод median у ковзному вікні ефективно відсіює випадкові флуктуації, не розмиваючи при цьому різкі фронти реальних змін екологічного показника, що є перевагою над простим усередненням.

Наведена програмна реалізація демонструє практичне втілення математичної моделі, описаної у другому розділі. Використання векторизованих операцій бібліотек Pandas та NumPy забезпечує високу ефективність обчислень, дозволяючи обробляти великі масиви даних без використання повільних циклічних конструкцій.

3.4 Візуалізація результатів роботи

Важливим компонентом розробленого програмного комплексу є модуль візуалізації. У задачах обробки часових рядів чисельні метрики не завжди дають повну картину ефективності алгоритму. Графічне представлення дозволяє досліднику візуально оцінити характер усунених артефактів, перевірити коректність виявленого тренду та переконатися, що алгоритм не спотворив корисну складову сигналу.

Для реалізації цього функціоналу використано бібліотеку Matplotlib. Побудова графіків здійснюється за принципом нашарування, що дозволяє сумістити на одній координатній площині вхідні, проміжні та вихідні дані для їх зручного порівняння.

Процес візуалізації розділено на декілька етапів. Спочатку ініціалізується графічне полотно із заданими пропорціями, що забезпечує читабельність часового ряду навіть при великій кількості точок.

```
plt.figure(figsize=(14, 7))
```

Першим шаром відмальовуються вхідні «сирі» дані. Для них обрано світло-сірий колір та найнижчий пріоритет відображення (zorder=1). Це створює візуальний контекст рівня зашумленості, але не відволікає увагу від основного результату.

```
plt.plot(data_cleaned['time'], data_cleaned['raw'],
         color='lightgray', label='Сирі дані', zorder=1)
```

Наступним кроком відображається виявлений тренд дрейфу. Використання пунктирної лінії помаранчевого кольору дозволяє чітко виділити систематичну складову похибки, яку розрахував алгоритм поліноміальної регресії.

```
plt.plot(data_cleaned['time'], detected_trend,
         color='orange', linestyle='--', linewidth=2,
         label='Виявлений дрейф сенсора', zorder=4)
```

Основним елементом візуалізації є результат роботи конвеєра — очищений сигнал. Для нього встановлено насичений синій колір та збільшену товщину лінії, що акцентує увагу на фінальному результаті обробки після видалення шумів та дрейфу.

```
plt.plot(data_cleaned['time'], data_cleaned['step4_cleaned'],
         color='blue', linewidth=2.5,
         label='Результат', zorder=3)
```

Для верифікації точності роботи методу на графік накладається еталонний сигнал, згенерований на етапі моделювання. Він відображається зеленим пунктиром із частковою прозорістю. Близькість синьої (відновленої) та зеленої (істинної) ліній слугує візуальним критерієм якості.

```
plt.plot(data_cleaned['time'], data_cleaned['ideal'],
         color='green', linestyle='--', alpha=0.8,
         label='Істинний сигнал', zorder=2)
```

Завершує побудову додавання заголовка, який містить розрахований відсоток покращення якості, легенди для ідентифікації ліній та відображення готового графіка.

```
plt.title(f'Результат очищення (RMSE знижено на {improvement:.1f}%)')
plt.legend()
plt.show()
```

Розроблений скрипт формує багатошарове зображення, яке несе значне інформаційне навантаження для аналізу роботи методу:

1. Фоновий шар. Сирі дані на задньому плані дозволяють оцінити амплітуду шуму та наявність аномальних викидів, з якими довелося працювати алгоритму.
2. Діагностичний шар. Лінія тренду демонструє роботу модуля де-трендингу. Це критично важливо для підтвердження того, що система правильно ідентифікувала систематичне зміщення сенсора, а не сплутала його з реальним зростанням показника.

3. Основний шар. Результуюча крива демонструє ефективність згладжування та повернення сигналу до горизонтального положення.
4. Валідаційний шар. Накладання істинного сигналу дозволяє миттєво оцінити похибку відновлення без необхідності аналізу числових таблиць.

Окрім графічної інформації, програмний модуль виводить у консоль кількісні показники ефективності, розраховані за допомогою бібліотеки scikit-learn.

Така комбінація візуального та чисельного представлення результатів дозволяє комплексно оцінити роботу системи: графік демонструє характер виправлень - якісну оцінку, а метрика середньоквадратичної похибки надає об’єктивне числове підтвердження точності - кількісну оцінку.

Реалізований модуль візуалізації забезпечує наочне представлення процесів, що відбуваються всередині алгоритму. Використання багатошарової графіки дозволяє детально проаналізувати коректність роботи кожного етапу очищення, зокрема виділення тренду та усунення викидів, що є необхідною умовою для валідації результатів дослідження.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ

4.1 Методика проведення експерименту та характеристика вхідних даних

Експериментальне дослідження ефективності розробленого методу автоматизованого очищення даних проводилося у два етапи. Така стратегія зумовлена необхідністю всебічної верифікації алгоритму як на рівні теоретичної моделі з суворо контрольованими параметрами завад, так і на реальних фізичних сигналах, отриманих із сенсорних мереж. Загальна структура експерименту включає етап математичного моделювання на синтетичних даних та етап натурної апробації на реальних даних. Перший етап необхідний для точного розрахунку метрик похибки, оскільки наявність ідеального еталонного сигналу дозволяє математично точно оцінити якість відновлення. Другий етап дозволяє перевірити адаптивність методу до реальної фізики сенсорів, нелінійності вимірювань та стохастичних факторів навколишнього середовища, які важко відтворити у симуляції.

4.1.1 Моделювання синтетичних даних для верифікації алгоритму

Перша фаза експерименту полягала у створенні програмного генератора сигналів, який імітує поведінку типового сенсора інтернету речей. Для цього було використано адитивну математичну модель, обґрунтовану у другому розділі роботи. Генерація даних відбувалася шляхом накладання різнорідних спотворень на відомий корисний сигнал, що дозволило отримати точний еталон для подальшого порівняння.

У якості базового або істинного сигналу було обрано комбінацію гармонічних функцій, що імітують добові та сезонні цикли зміни екологічних

параметрів, наприклад концентрації газів або температури. Математично такий сигнал описується як сума синусоїд з різними частотами та амплітудами, що дозволяє відтворити як повільні добові тренди, так і швидші локальні зміни. Такий підхід забезпечує наближеність тестових даних до реальних процесів, що відбуваються в атмосфері.

Для перевірки стійкості алгоритму на сформований ідеальний сигнал накладалися три типи завад, характерні для бюджетних сенсорів. Вимірювальний шум генерувався як адитивний білий гаусівський шум із нульовим математичним сподіванням та середньоквадратичним відхиленням 1.5. Це дозволило змоделювати теплові шуми електроніки та випадкові флуктуації, які неминуче присутні у вимірювальному каналі. Аномальні викиди додавалися у випадкові моменти часу до сигналу у вигляді імпульсних завад великої амплітуди, що становила плюс або мінус 15 одиниць. Імовірність появи такого викиду становила 2 відсотки, що імітує рідкісні апаратні збої або короткочасні локальні збурення, наприклад вплив прямого сонячного світла на оптичний сенсор.

Окрему увагу було приділено моделюванню дрейфу сенсора, який є однією з головних проблем довгострокового моніторингу. Для імітації деградації чутливого елемента було введено лінійний тренд дрейфу, параметри якого обиралися таким чином, щоб за час спостереження зміщення нуля сенсора становило значну величину, співмірну з амплітудою корисного сигналу. Крім того, для імітації нестабільності каналу зв'язку з масиву даних випадковим чином видалялося 5 відсотків вимірювань, що створювало розриви у часовому ряді та вимагало перевірки роботи модуля імпутації. Такий комплексний підхід дозволив отримати набір даних, який містить усі типові проблеми моніторингу, але при цьому зберігає інформацію про істинний стан системи.

4.1.2 Характеристика набору реальних даних

Для другого етапу апробації методу в умовах реальної експлуатації було використано відкритий набір даних Air Quality Data Set, опублікований у репозиторії машинного навчання Каліфорнійського університету в Ірвайні [24]. Вибір саме цього набору обумовлений його високою якістю документування та наявністю верифікованих еталонних значень, що є рідкістю для відкритих даних у сфері інтернету речей.

У обраному для дослідження наборі даних представлено результати вимірювань, отримані за допомогою стаціонарного мультисенсорного пристрою газового аналізу. Згідно з описом датасету, обладнання було розгорнуто в польових умовах на рівні дороги у районі італійського міста з високою інтенсивністю автомобільного трафіку. Така локація характеризується значним забрудненням повітря, що забезпечує присутність у даних високої динаміки змін концентрації забруднюючих речовин. Часовий ряд датасету охоплює проміжок тривалістю один рік. Це дозволяє врахувати у дослідженні вплив різноманітних сезонних факторів, таких як температура та відносна вологість повітря, які суттєво впливають на точність показань хімічних сенсорів. Часова роздільна здатність записів становить одну годину, що є стандартом для екологічного моніторингу міського середовища.

Унікальною особливістю даного набору є наявність паралельних вимірювань з двох типів приладів різного класу точності. Зокрема, датасет містить записи бюджетного метал-оксидного сенсора на базі оксиду олова, номінально налаштованого на детектування чадного газу. Принцип дії таких сенсорів базується на зміні електричного опору нагрітого чутливого шару при адсорбції молекул газу на його поверхні. Сенсори цього типу є стандартом для масових рішень інтернету речей через низьку вартість та компактність, але вони мають суттєві метрологічні недоліки. До таких недоліків належать високий рівень власного шуму, нелінійність передавальної характеристики та сильна перехресна чутливість до температури та вологості. Найбільш критичною проблемою є їх схильність до часового дрейфу, викликаного незворотним

хімічним отруєнням активного шару та старінням нагрівального елемента [25]. Саме ці характеристики роблять їх ідеальним об'єктом для тестування розробленого алгоритму.

Окрім даних бюджетного сенсора, датасет містить результати контрольних вимірювань, отриманих на тій самій локації за допомогою професійного сертифікованого газоаналізатора. Його показники у даному дослідженні приймаються за істину або еталон. Наявність такого еталонного сигналу дозволяє не просто візуально оцінити гладкість кривої після очищення, а й об’єктивно перевірити, чи не спотворив алгоритм корисну складову сигналу при видаленні шумів, що є ключовим критерієм коректності роботи методу.

З оригінального багатовимірного набору даних для експерименту було виділено дві цільові часові серії, що відповідають концентрації чадного газу. Перша серія представляє середньогодинний показник опору метал-оксидного сенсора. Це і є сирі вхідні дані для розробленого алгоритму, які містять усі досліджувані типи дефектів, включаючи шуми та дрейф. Друга серія містить істинну концентрацію чадного газу в міліграмах на метр кубічний, виміряну еталонним аналізатором, і використовується виключно для валідації результатів.

В оригінальному наборі даних пропущені значення або помилки приладу маркуються спеціальним числовим кодом мінус двісті. Перед подачею на вхід розробленого конвеєра було проведено процедуру попередньої обробки або парсингу, під час якої всі значення з кодом помилки були замінені на стандартний ідентифікатор відсутності даних NaN. Ця процедура необхідна для коректної активації модуля імпутації, описаного в третьому розділі, який розпізнає та відновлює втрачені фрагменти даних.

4.1.3 Критерії оцінки ефективності

Для кількісної оцінки якості роботи розробленого методу та його порівняння з існуючими аналогами, такими як ковзне середнє та медіанна фільтрація, було обрано метрику середньоквадратичної похибки RMSE. Ця

метрика розраховується як корінь квадратний із середнього арифметичного квадратів різниць між значеннями еталонного сигналу та відновленого сигналу після очищення.

Вибір саме цієї метрики для задач екологічного моніторингу обґрунтований двома ключовими факторами. По-перше, середньоквадратична похибка вимірюється в тих самих фізичних одиницях, що і вихідні дані, що значно полегшує інтерпретацію результатів. Наприклад, похибка в 1.5 міліграма на метр кубічний є зрозумілою для фахівців предметної області. По-друге, завдяки математичній операції піднесенню різниці до квадрату ця метрика накладає більший штраф за значні відхилення. Оскільки однією з головних цілей роботи є усунення аномальних пікових викидів, RMSE є найкращим індикатором успішності досягнення цієї мети [26]. Зниження показника похибки свідчитиме про те, що алгоритм не лише згладив дрібний фоновий шум, а й ефективно усунув грубі помилки вимірювання та компенсував систематичний дрейф сенсора.

Отже, використання двоетапної методики дозволяє верифікувати роботу алгоритму як на еталонних сигналах із відомими параметрами, так і в умовах реальної експлуатації.

4.2 Дослідження ефективності методу на модельованих даних

Перший етап верифікації розробленого програмного забезпечення полягав у тестуванні алгоритму на синтетичних наборах даних. Використання математичного моделювання дозволило чітко розділити вплив різних факторів спотворення на якість відновлення сигналу та отримати точні кількісні оцінки похибки, що неможливо зробити на реальних даних через відсутність еталонних значень. Дослідження проводилося у двох сценаріях, які імітують різні умови експлуатації сенсорного обладнання.

У першій серії експериментів моделювалася робота системи в умовно ідеальному або стаціонарному режимі. Такий сценарій відповідає ситуації, коли

сенсор є новим, відкаліброваним та не має ознак деградації чутливого елемента. Вхідний сигнал містив лише корисну гармонічну складову, на яку накладався високочастотний вимірювальний шум та випадкові аномальні викиди. Компонента дрейфу на цьому етапі була відключена.

Для візуальної оцінки якості фільтрації було побудовано графік часових рядів, на якому суміщено вхідний зашумлений сигнал та результат роботи запропонованого конвеєра (рисунок 4.1).

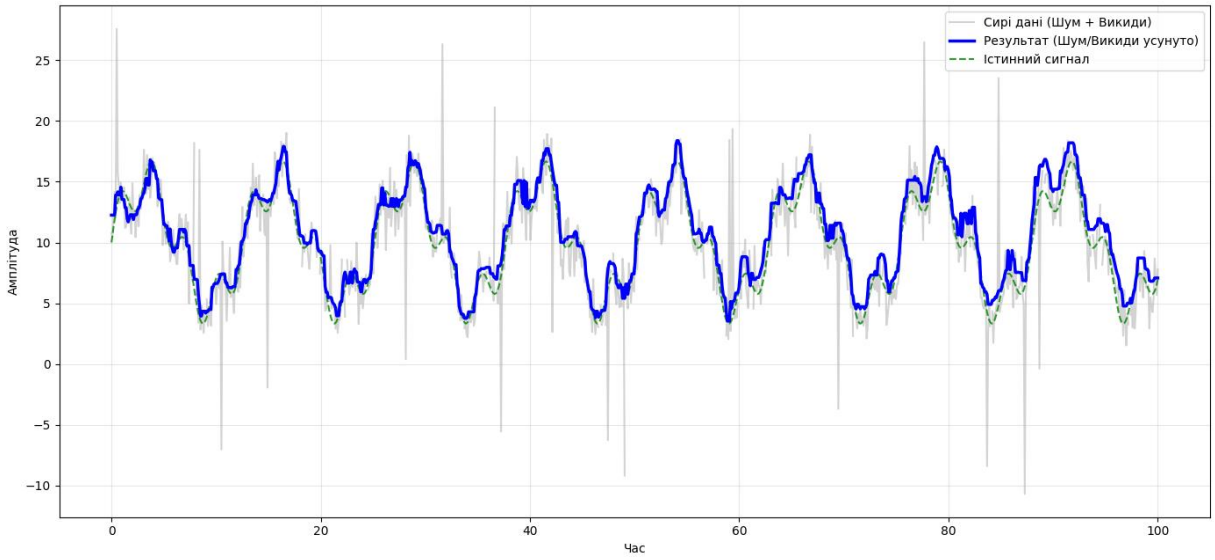


Рис. 4.1 Візуалізація відновлення сигналу в стаціонарних умовах

Як видно з рисунка 4.1, розроблений алгоритм забезпечує ефективне згладжування високочастотних флуктуацій, повертаючи форму сигналу до вигляду, близького до еталонної синусоїди. Важливо відзначити, що завдяки використанню адаптивного фільтра на основі міжквартильного розмаху (IQR) вдалося усунути різкі аномальні викиди без внесення значних фазових спотворень, які характерні для простих лінійних методів.

Для об'єктивної кількісної оцінки та порівняння з аналогами було розраховано значення середньоквадратичної похибки для кожного методу. Результати порівняльного аналізу наведено на діаграмі рисунка 4.2.

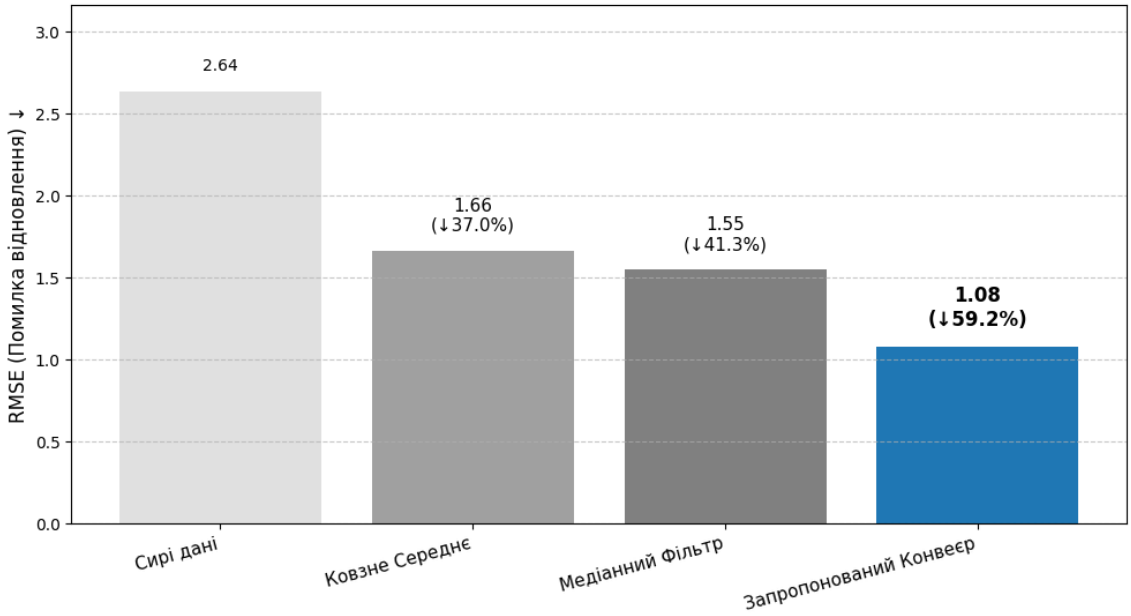


Рис. 4.2 Порівняльна ефективність методів очищення в стаціонарних умовах

Аналіз діаграми показує, що початкова похибка відновлення для сирих даних становила 2.64. Метод простого ковзного середнього показав зниження похибки до рівня 1.66, що становить покращення на 37 відсотків. Це відносно невисокий результат, що пояснюється чутливістю середнього арифметичного до аномальних викидів, які зміщують результат фільтрації. Медіанна фільтрація продемонструвала кращу стійкість до імпульсних завад, знизивши похибку до рівня 1.55. Проте найкращий результат показав запропонований у роботі комплексний конвеєр. Завдяки попередньому етапу детекції аномалій та відновленню пропусків вдалося досягти значення похибки 1.08, що означає зниження рівня помилок на 59.2 відсотка порівняно з вхідним сигналом.

Друга серія експериментів була спрямована на дослідження стійкості алгоритмів до дрейфу нульового рівня, що є типовою проблемою для довгострокового екологічного моніторингу. У вхідний сигнал було додано лінійний тренд, який імітував поступову зміну показників сенсора через старіння компонентів.

Динаміку роботи алгоритму та процес компенсації тренду візуалізовано на рисунку 4.3.

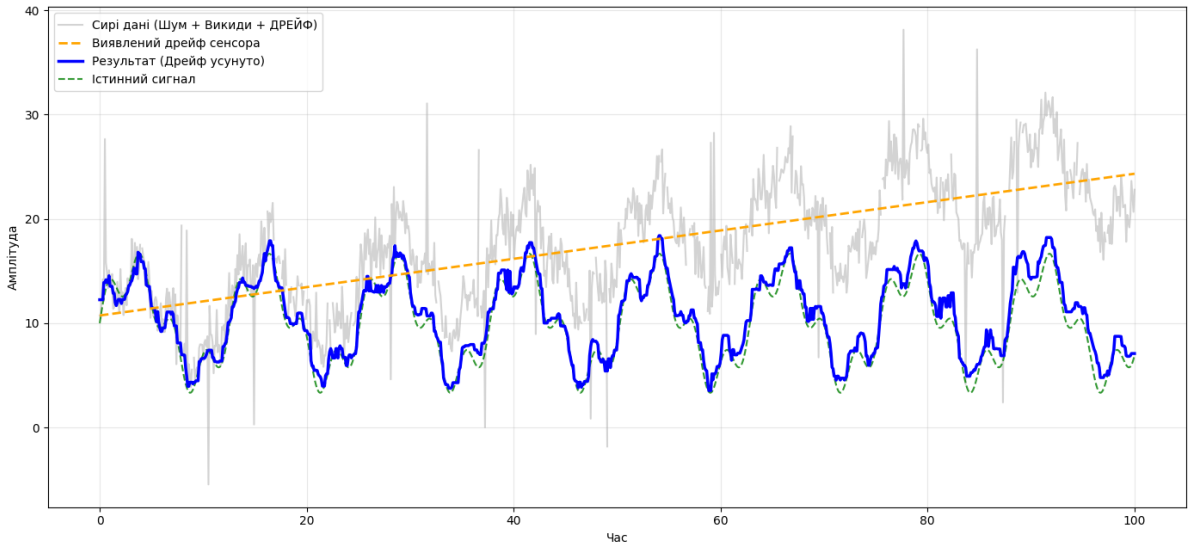


Рис. 4.3 Візуалізація роботи алгоритму компенсації дрейфу на модельованих даних

Графічна візуалізація процесів обробки, представлена на рисунку 4.3, наочно демонструє механізм роботи запропонованого алгоритму в умовах нестаціонарності. На відміну від стандартних підходів, які працюють лише з локальним вікном даних, розроблений конвеєр аналізує глобальну структуру часового ряду. Це дозволило системі автоматично детектувати наявність низькочастотної складової похибки, яку на графіку відображено помаранчевою пунктирною лінією. Застосування процедури поліноміального де-трендингу дозволило математично описати закон зміни дрейфу та відняти цю паразитну складову від вхідного сигналу.

Як наслідок такої операції, результуюча крива, позначена синім кольором, повернулася до горизонтальної базової лінії, що відповідає параметрам істинного сигналу. Важливо відзначити, що при компенсації значного амплітудного зміщення вхідних даних алгоритм зберіг корисну високочастотну динаміку процесу, не спотворивши форму гармонічних коливань. Це свідчить про коректність розділення спектральних компонентів корисного сигналу та тренду дрейфу.

Однак для об'єктивного підтвердження ефективності методу недостатньо лише візуальної оцінки. Необхідно зіставити отримані результати з показниками

роботи альтернативних методів у чисельному вимірі, щоб оцінити масштаб виграшу в точності. Для цього було проведено розрахунок середньоквадратичної похибки відновлення для кожного з досліджуваних алгоритмів. Результати цього порівняльного аналізу систематизовано та представлено у вигляді гістограми на рисунку 4.4.

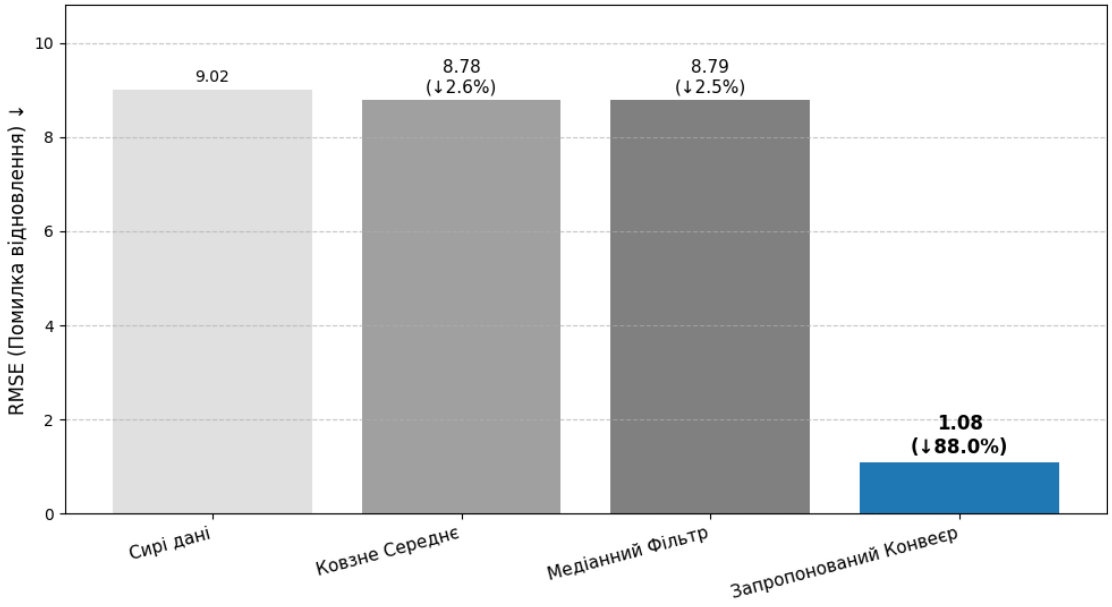


Рис. 4.4 Порівняльна ефективність методів очищення в умовах деградації сенсора

Аналіз діаграми на рисунку 4.4 виявив критичну вразливість класичних методів фільтрації в умовах нестаціонарності. Через наявність дрейфу початкова похибка сирих даних зросла до значення 9.02. Алгоритми ковзного середнього та медіанної фільтрації продемонстрували свою неспроможність розрізнити паразитний тренд дрейфу та корисну низькочастотну складову сигналу. В результаті обидва методи просто відстежували зміщений сигнал, що призвело до накопичення значної систематичної похибки. Значення метрики RMSE для цих методів залишилися на критично високому рівні 8.78 та 8.79 відповідно.

Натомість запропонований конвеєр обробки завдяки блоку де-трендингу дозволив утримувати похибку відновлення на рівні 1.08, що ідентично показнику для ідеальних умов першого сценарію. Таким чином, метод забезпечив зниження похибки на 88 відсотків порівняно з сирим сигналом. Такий значний розрив у

ефективності математично доводить необхідність використання процедур корекції тренду в системах автоматизованого моніторингу.

4.3 Апробація методу на реальних даних екологічного моніторингу

Завершальним етапом експериментального дослідження стала перевірка працездатності розробленого програмного модуля на реальних даних, отриманих з відкритих репозиторіїв. На відміну від математичного моделювання, де параметри завад є заздалегідь відомими та контрольованими, робота з реальними сигналами дозволяє оцінити адаптивність алгоритму до непередбачуваних змін умов вимірювання та складних суперпозицій шумів різної фізичної природи.

Для апробації було обрано фрагмент часового ряду показників метал-оксидного сенсора з набору даних Air Quality Data Set, що відповідає періоду інтенсивних спостережень [24]. Цей тип сенсорів характеризується високою чутливістю до змін навколишнього середовища, що неминуче призводить до появи значної шумової компоненти та нестабільності нульового рівня. Результати обробки цього фрагмента розробленим конвеєром візуалізовано на рисунку 4.5.

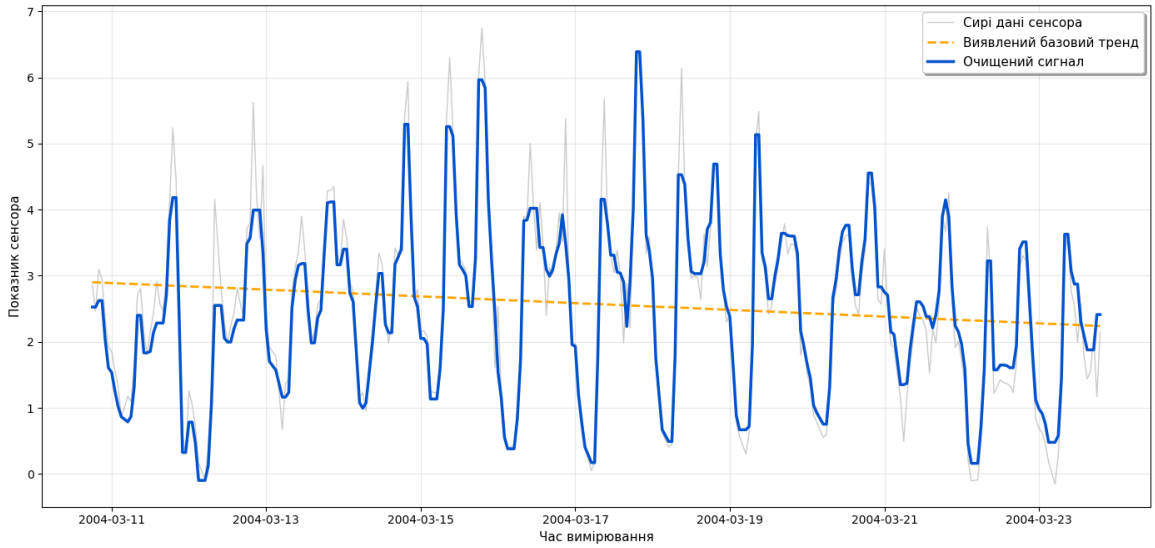


Рис. 4.4 Результат роботи алгоритму очищення на реальних даних

Візуальний аналіз вхідного потоку даних, позначеного на графіку світло-сірим кольором, підтверджує теоретичні припущення щодо низької якості первинних вимірювань бюджетних сенсорів. Сигнал характеризується високою волатильністю та наявністю значної високочастотної складової шуму, яка ускладнює ідентифікацію локальних екстремумів. Крім того, на обраному інтервалі спостерігається чітко виражена тенденція до зниження базового рівня показників, що свідчить про наявність дрейфу сенсора або зміну фонових екологічних умов.

Застосування розробленого методу дозволило автоматично виділити цей низькочастотний тренд, який на графіку позначено помаранчевою пунктирною лінією. Алгоритм поліноміальної апроксимації коректно ідентифікував лінійну складову зміни базової лінії, не реагуючи при цьому на добові коливання концентрації забруднюючих речовин. Це свідчить про правильний вибір параметрів моделі дрейфу та стійкість методу регресії до локальних варіацій сигналу.

Результуюча крива, відображена синім кольором, демонструє сигнал після проходження повного циклу очищення, включаючи етапи імпутації, видалення викидів та медіанного згладжування. Порівняння сирого та очищеного сигналів дозволяє зробити висновок про ефективність придушення високочастотного шуму. Очищений графік є гладким та позбавленим хаотичних осциляцій, що значно підвищує його інформативність для подальшого аналізу. При цьому важливо відзначити, що алгоритм зберіг форму та амплітуду основних піків, які відповідають реальним добовим циклам зміни якості повітря. Це підтверджує, що застосування медіанної фільтрації та адаптивного порогового обмеження не призводить до втрати корисної інформації про динаміку екологічного стану.

Таким чином, апробація на натурних даних підтвердила спроможність розробленого методу працювати в умовах реальних невизначеностей. Система успішно впоралася з задачами відновлення цілісності ряду, компенсації дрейфу

та фільтрації шумів без необхідності ручного налаштування параметрів під конкретний зразок даних.

4.4 Узагальнена оцінка ефективності та порівняльний аналіз результатів

Для формування цілісного уявлення про ефективність розробленого методу необхідно провести узагальнення результатів, отриманих на етапах математичного моделювання та натурної апробації. Ключовим критерієм оцінки є здатність алгоритмів адаптуватися до зміни умов експлуатації, зокрема до появи систематичних похибок типу дрейфу. Для зручності порівняння кількісні показники точності відновлення сигналу для всіх розглянутих сценаріїв зведено в таблицю 4.1.

Таблиця 4.1

Зведені показники ефективності методів очищення даних (RMSE)

Метод обробки	**RMSE (Стаціонарний режим)**	**RMSE (Режим дрейфу)**	**Зниження похибки (з дрейфом)**
Сирий вхідний сигнал	2.64	9.02	—
Просте ковзне середнє	1.66	8.78	2.6%
Медіанна фільтрація	1.55	8.79	2.5%
Запропонований конвеєр	**1.08**	**1.08**	**88.0%**

Детальний аналіз наведених даних дозволяє зробити низку важливих висновків щодо меж застосування досліджуваних алгоритмів.

У стаціонарних умовах, коли відсутній дрейф сенсора, усі методи демонструють позитивну динаміку покращення якості даних. Проте навіть у цьому спрощеному сценарії запропонований конвеєр забезпечує найменшу похибку відновлення на рівні 1.08.

Це пояснюється використанням адаптивного порогового фільтра, який усуває аномальні викиди ефективніше, ніж лінійне згладжування, що дозволяє досягти покращення якості на 59% відносно сирих даних.

Однак найбільш показовими є результати роботи в умовах деградації сенсора. Як свідчать дані таблиці, поява лінійного тренду призводить до катастрофічного зростання похибки для класичних методів. Ковзне середнє та медіанний фільтр, не маючи механізмів оцінки глобальних трендів, фактично транслюють похибку дрейфу на вихід системи. Їхня ефективність у цьому режимі становить менше трьох відсотків, що робить їх використання недоцільним для довготривалого моніторингу без частих процедур апаратного калібрування.

Критично важливою перевагою розробленого методу є стабільність його показників. Значення середньоквадратичної похибки залишається незмінним на рівні 1.08 незалежно від наявності чи відсутності дрейфу. Це свідчить про повну компенсацію систематичної складової похибки блоку де-трендингу. Загальне зниження рівня помилок на 88 відсотків у складному сценарії підтверджує гіпотезу дослідження про необхідність комплексного підходу до очищення.

Результати апробації на реальних даних, наведені у попередньому підрозділі, корелюють із висновками математичного моделювання. Виділений на реальному сигналі тренд за своєю формою відповідає модельованому дрейфу, а візуальна якість очищеного сигналу підтверджує здатність алгоритму працювати в умовах високої невизначеності.

Таким чином, проведене дослідження довело, що інтеграція методів імпутації, робастної статистики та поліноміальної апроксимації в єдиний

автоматизований конвеєр дозволяє створити універсальний інструмент попередньої обробки. Такий інструмент забезпечує високу достовірність даних як для нових, так і для деградованих сенсорів, що дозволяє подовжити термін їх експлуатації та знизити витрати на обслуговування мережі моніторингу.

4.5 Практичні рекомендації щодо впровадження системи

Запропонований алгоритмічний конвеєр може бути інтегрований у різні рівні архітектури інтернету речей залежно від вимог до швидкодії та доступних обчислювальних ресурсів.

Першим сценарієм впровадження є розгортання програмного модуля на рівні хмарного проміжного програмного забезпечення. У цьому варіанті алгоритм функціонує як мікросервіс, який приймає потоки сирих даних від шлюзів через протокол MQTT, виконує їх очищення та зберігає результат у базі даних часових рядів. Такий підхід є оптимальним для централізованих систем розумного міста, де пріоритетом є глибокий аналіз історичних даних та формування довгострокових прогнозів. Реалізація на мові Python дозволяє легко інтегрувати модуль у контейнерні середовища оркестрації типу Kubernetes.

Другим, більш перспективним сценарієм, є реалізація концепції граничних обчислень. Оскільки розроблений алгоритм базується на векторизованих операціях та не потребує ресурсів для навчання нейронних мереж, він може бути розгорнутий безпосередньо на апаратних шлюзах або концентраторах даних, побудованих на базі одноплатних комп'ютерів архітектури ARM, таких як Raspberry Pi. Перенесення обчислень на край мережі дозволяє відфільтрувати шуми та стиснути дані безпосередньо у місці їх виникнення, що суттєво знижує навантаження на канали зв'язку та зменшує обсяг трафіку, що передається у хмару.

Для забезпечення ефективної роботи системи критично важливо правильно обрати параметри алгоритму, де ключовим фактором є частота дискретизації сенсора. Розмір ковзного вікна для детекції аномалій та

згладжування повинен обиратися як компроміс між якістю придушення шуму та збереженням корисної динаміки сигналу. Емпірично встановлено, що оптимальна ширина вікна повинна охоплювати часовий інтервал, достатній для нівелювання короткочасних стохастичних флуктуацій, але при цьому бути значно меншою за характерний час зміни добового тренду екологічного показника.

Для систем із низькою частотою дискретизації (наприклад, погодинні вимірювання, характерні для набору Air Quality Data Set) рекомендовано використовувати вікно шириною від 3 до 5 годин. Зменшення вікна нижче цього діапазону призведе до пропуску високочастотного шуму, а збільшення понад 5 годин може спричинити надмірне згладжування, через що будуть втрачені локальні екстремуми, які несуть інформацію про пікові забруднення [24].

Варто також звернути увагу на обмеження методу. Алгоритм лінійної інтерполяції забезпечує високу точність відновлення лише для короткочасних пропусків даних. У випадку тривалих збоїв зв'язку, коли дані відсутні протягом значного періоду часу, лінійна апроксимація може призвести до втрати інформації про локальні екстремуми. У таких ситуаціях рекомендується використовувати додаткові джерела даних від сусідніх станцій моніторингу для кореляційного відновлення або маркувати такі ділянки як недостовірні.

У четвертому розділі проведено комплексне експериментальне дослідження розробленого методу автоматизованого очищення даних. Для забезпечення об'єктивності оцінювання використано двоетапну методику, що включає верифікацію на синтетичних даних з відомим еталонним сигналом та апробацію на реальному наборі даних екологічного моніторингу Air Quality Data Set.

За результатами проведених експериментів можна зробити наступні висновки:

1. Ефективність у стаціонарних умовах. Результати моделювання підтвердили, що за відсутності дрейфу сенсора запропонований метод забезпечує зниження похибки відновлення (RMSE) на 59% порівняно з

сирим сигналом, перевершуючи показники стандартних фільтрів ковзного середнього та медіани.

2. Стійкість до деградації сенсора. В умовах появи дрейфу нульового рівня ефективність методу зростає, забезпечуючи зниження середньоквадратичної похибки на 88%. Це підтверджує критичну важливість впровадження процедури поліноміального де-трендингу для довгострокового моніторингу.
3. Апробація на реальних даних. Перевірка на натурних даних підтвердила здатність алгоритму коректно ідентифікувати та компенсувати тренди в реальних умовах без втрати корисної інформації про добові цикли забруднення.
4. Практична цінність. На основі отриманих результатів сформовано комплекс практичних рекомендацій щодо налаштування параметрів алгоритму для його впровадження у системи «Розумного міста».

ВИСНОВКИ

У магістерській роботі вирішено актуальне науково-прикладне завдання підвищення якості екологічного моніторингу шляхом створення та програмної реалізації методу автоматизованого очищення даних сенсорів інтернету речей. Основні наукові та практичні результати роботи полягають у наступному:

1. Обґрунтовано вибір чотирирівневої моделі системи моніторингу. Визначено, що попереднє очищення даних доцільно виконувати на проміжному рівні перед їх збереженням та аналізом.
2. Систематизовано типові дефекти даних сенсорів: апаратні шуми, пропуски, викиди та дрейф. Встановлено, що низька достовірність первинних вимірювань є основною проблемою бюджетного обладнання.
3. Розроблено адитивну математичну модель вхідного сигналу, яка враховує корисну складову та чотири компоненти спотворень: шум, викиди, дрейф та втрату даних.
4. Сформовано метод автоматизованого очищення даних у вигляді конвеєра, що включає лінійну інтерполяцію, фільтрацію викидів (IQR), корекцію дрейфу та медіанне згладжування.
5. Реалізовано програмний модуль мовою Python (Pandas, NumPy). Його архітектура забезпечує потокову обробку даних та можливість інтеграції у хмарні платформи або периферійні пристрої.
6. Експериментально підтверджено ефективність методу: в умовах дрейфу сенсора середньоквадратична похибка (RMSE) знизилася на 88% (з 9.02 до 1.08), що перевищує показники базових методів фільтрації.
7. Проведено апробацію на реальному наборі даних Air Quality Data Set. Підтверджено здатність алгоритму компенсувати тренд дрейфу та відновлювати динаміку показників без втрати інформативності.

8. Сформовано практичні рекомендації щодо налаштування параметрів алгоритму (розміру вікна та коефіцієнтів чутливості) залежно від типу сенсорів та умов експлуатації.

Робота пройшла апробацію на наступних конфереціях:

1. Іванчук В.О., Золотухіна О.А. Інтелектуальний аналіз та обробка IoT-даних в системах моніторингу навколишнього середовища. VI Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С.561.

2. Іванчук В.О., Золотухіна О.А. Підвищення якості моніторингу екологічного стану розумних міст шляхом очищення IoT-даних. VI Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT», 15 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С 226.

ПЕРЕЛІК ПОСИЛАНЬ

1. Мартинюк О. С., Федусенко О. В. Проєктування системи екологічного моніторингу з використанням технологій Інтернету речей. Вчені записки ТНУ імені В.І. Вернадського. Серія: Технічні науки. 2020. Т. 31 (70), № 2, ч. 2. С. 134–140.
2. Кучук Г. А., Коваленко А. А. Метод попередньої обробки інформаційних потоків в мережах Інтернету речей. Сучасні інформаційні системи. 2020. Т. 4, № 4. С. 120–125.
3. Говорущенко Т. О. Методологія оцінювання якості даних в інформаційних системах : монографія. Хмельницький : ХНУ, 2017. 226 с.
4. Лисенко О. І., Вальченко О. О. Методи підвищення достовірності даних в сенсорних мережах моніторингу. Вісник НТУУ "КПІ". Серія: Радіотехніка, Радіоапаратобудування. 2019. № 76. С. 67–74.
5. Іванчук В.О., Золотухіна О.А. Інтелектуальний аналіз та обробка IoT-даних в системах моніторингу навколишнього середовища. *VI Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях»*, 24 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025. С.561.
6. Іванчук В.О., Золотухіна О.А. Підвищення якості моніторингу екологічного стану розумних міст шляхом очищення IoT-даних. *VI Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT»,* 15 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025. С.226.
7. Шевченко В. Л. Технології бездротових сенсорних мереж для екологічного моніторингу. Системи управління, навігації та зв’язку. 2021. № 2 (64). С. 128–134.
8. Analysis of Time Series Data Generated From the Internet of Things Using Deep Learning Models / V. Bhatnagar et al. IEEE Access. 2023. Vol. 11. P. 2345–2358.

9. Data quality in Internet of Things: A state-of-the-art survey / A. Karkouch et al. Journal of Network and Computer Applications. 2016. Vol. 73. P. 57–81.
10. Data Quality Management in the Internet of Things: A Review / A. Ebadat et al. Sensors. 2021. Vol. 21, no. 18. P. 6196.
11. Семенов С. Г., Гавриленко С. Ю., Челак В. В. Методи обробки аномальних значень у часових рядах інформаційних систем моніторингу. Системи обробки інформації. 2022. № 1 (168). С. 34–41.
12. Осадчий С. І., Зіньковський Ю. Ф. Дослідження метрологічних характеристик та часового дрейфу напівпровідникових сенсорів газу. Вимірювальна техніка та метрологія. 2021. Вип. 82. С. 45–51.
13. Ткачук А. Г., Климчук О. А. Проблема відновлення пропущених даних у бездротових сенсорних мережах екологічного моніторингу. Вісник Вінницького політехнічного інституту. 2023. № 2. С. 56–62.
14. Казмірчук С. В., Ільїн В. О. Застосування методів глибинного навчання для виявлення аномалій у часових рядах телекомунікаційних систем. Телекомунікаційні та інформаційні технології. 2023. № 1 (78). С. 45–54. URL: https://tit.dut.edu.ua/index.php/telecommunication/article/view/2456.
15. V. J. et al. Survey on IoT Data Preprocessing. Turkish Journal of Computer and Mathematics Education. 2021. Vol. 12, No. 10. P. 2574–2579.
16. S. Liu et al. A Survey on Information Fusion Techniques in Internet of Things: Computational Methods and Applications. Information Fusion. 2020. Vol. 60. P. 125–153.
17. Горбань І. І. Теорія ймовірностей і математична статистика для наукових працівників та інженерів : підручник. Київ : ІПММС НАН України, 2020. 244 с.
18. Idrees A. K. et al. A distributed approach for data restoration in wireless sensor networks based on spatial-temporal correlation. IEEE Transactions on Industrial Informatics. 2021. Vol. 17, no. 3. P. 2039–2049.

19. Rasheed H. et al. A Comparison of Python and C++ for Data Science and Machine Learning Applications. International Journal of Computer Applications. 2021. Vol. 174, no. 12. P. 15–22.
20. Harris C. R. et al. Array programming with NumPy. Nature. 2020. Vol. 585. P. 357–362.
21. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. 3rd ed. Sebastopol : O'Reilly Media, 2022. 550 p.
22. Hunter J. D. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering. 2007. Vol. 9, no. 3. P. 90–95.
23. Kluyver T. et al. Jupyter Notebooks - a publishing format for reproducible computational workflows. Positioning and Power in Academic Publishing: Players, Agents and Agendas. 2016. P. 87–90.
24. De Vito S. et al. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. Sensors and Actuators B: Chemical. 2008. Vol. 129, no. 2. P. 750–757.
25. Spinelle L. et al. Field calibration of a cluster of low-cost available sensors for air quality monitoring. Part A: Ozone and nitrogen dioxide. Sensors and Actuators B: Chemical. 2015. Vol. 215. P. 249–257.
26. Chai T., Draxler R. R. Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature. Geoscientific Model Development. 2014. Vol. 7, no. 3. P. 1247–1250.

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Магістерська робота

«Автоматизація очищення сирих даних IoT-сенсорів для моніторингу екологічного стану»

Виконав: студент групи ПДМ-61 Владислав ІВАНЧУК

Керівник: канд. техн. наук, доцент кафедри ІПЗ Оксана ЗОЛОТУХІНА

Київ - 2025

МЕТА, ОБ’ЄКТА ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: підвищення якості очищення сирих даних IoT-сенсорів для моніторингу екологічного стану за рахунок автоматизації обробки комплексного підходу до відновлення та корекції даних.

Об’єкт дослідження: процес очищення сирих даних IoT-сенсорів для моніторингу екологічного стану.

Предмет дослідження: методи та технології автоматизованого очищення сирих даних IoT-сенсорів для моніторингу екологічного стану.

СХЕМА ПОТОКІВ ДАНИХ У БАГАТОРІВНЕВІЙ IOT -АРХІТЕКТУРІ МОНІТОРИНГУ ЕКОЛОГІЧНОГО СТАНУ

3

ХАРАКТЕРИСТИКИ ТА ТИПОВІ ДЕФЕКТИ ПОТОКІВ IoT-ДАНИХ

Тип дефекту	**Природа виникнення**	**Вплив на моніторинг**
Пропуски (Missing Values)	Нестабільність мережі, збої живлення, перезавантаження шлюзу.	Порушення цілісності часових рядів, неможливість аналізу трендів.
Шуми (Sensor Noise)	Електричні наведення, тепловий шум, вібрації, низька якість АЦП.	Зниження точності миттєвих вимірювань, хибні спрацьовування.
Аномальні викиди (Outliers)	Короткочасні збої сенсора, реальні різкі зміни середовища.	Спотворення статистик (середнього, дисперсії), помилкові аларми.
Дрейф (Sensor Drift)	Хімічна деградація сенсора, забруднення, старіння компонентів.	Накопичення систематичної похибки, хибне виявлення трендів.

4

ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ ОБРОБКИ ДАНИХ

Категорія методів	**Переваги**	**Недоліки**
Статистичні методи (середнє, медіана, ковзне вікно)	Простота, швидкість	- Спотворюють піки та різкі зміни - Не враховують нелінійні шумові ефекти - Не працюють із дрейфом сенсорів
Фільтр Калмана	Добре згладжує шум	- Не виявляє аномалії - Важко налаштовується для різних сенсорів - Не відновлює великі пропуски
Методи імпутації (інтерполяція, KNN, регресія)	Відновлюють пропуски даних	- Дає неточні результати при довгих провалах - Не відрізняє аномалію від помилки - Залежить від якості сусідніх точок
Алгоритми ML(аномалії, прогнозування)	Виявляють складні закономірності	- Потребують попередньо очищених даних - Висока залежність від навчальних вибірок - Низька ефективність на сирих потоках
Data Fusion(злиття даних сенсорів)	Покращує точність після обробки	- Неможливий без попереднього очищення - Не компенсує шум та дрейф - Потребує уніфікації форматів і калібрування

5

МАТЕМАТИЧНА МОДЕЛЬ ОЧИЩЕННЯ IOT-ДАНИХ

$$Y(t) = S(t) + N(t) + O(t) + \mathrm{D(t)} + E_{miss}\,(t),$$

Y(t) – позначає отримане значення із сенсора або сирий сигнал;

S(t) – є істинним значенням вимірюваного екологічного параметра, що підлягає відновленню;

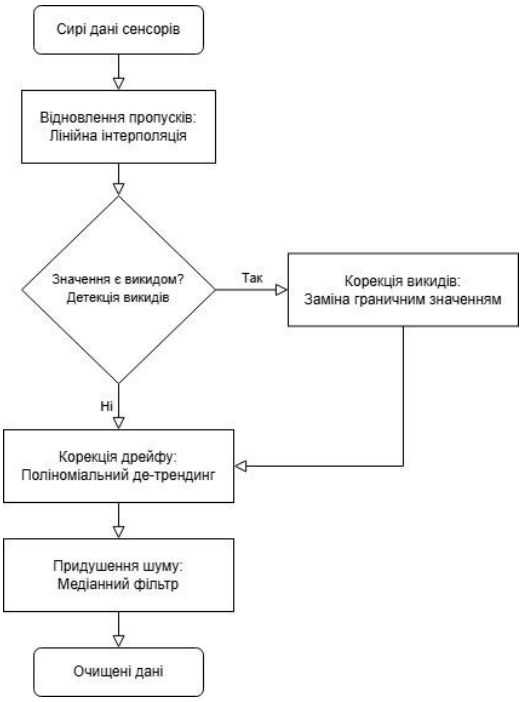
N(t) – описує високочастотну шумову компоненту стохастичної природи;

O(t) – відповідає за компоненту аномальних викидів;

D(t) – представляє низькочастотний тренд дрейфу сенсора;

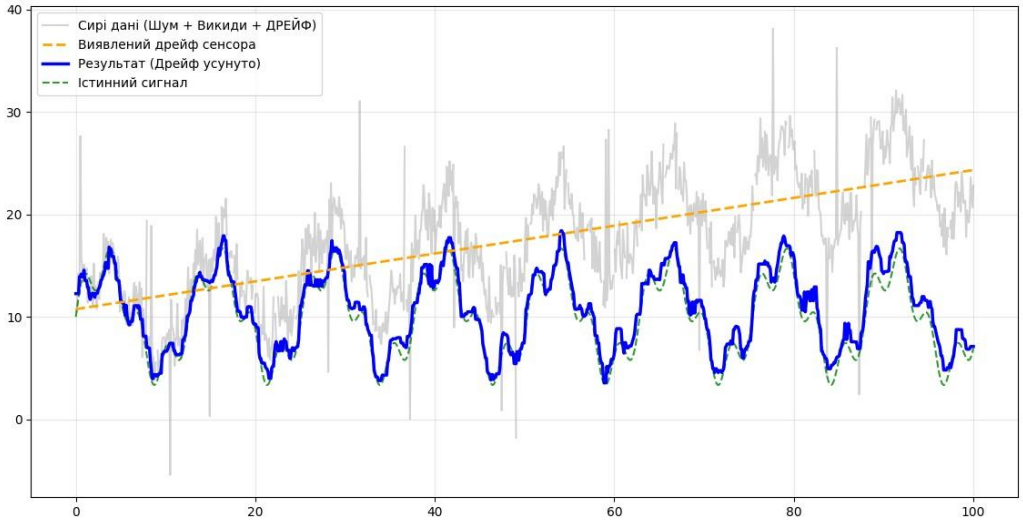
функція $E_{miss}\,(t)$ – моделює дискретну втрату даних.

АДАПТИВНИЙ КОНВЕЄР ОЧИЩЕННЯ IoT-ДАНИХ



7

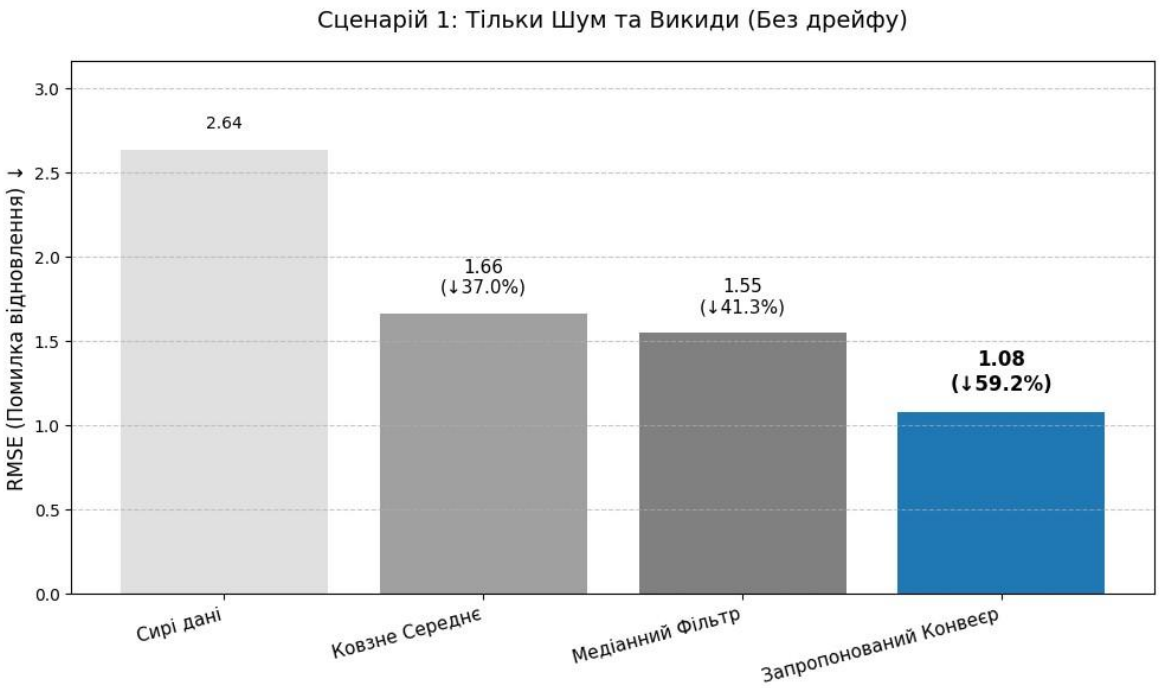
ВІЗУАЛІЗАЦІЯ РОБОТИ АЛГОРИТМУ



Метрика	Значення RMSE
Вхідні (сирі) дані	9.02
Вихідні (очищені) дані	1.08
Покращення якості	88.0%

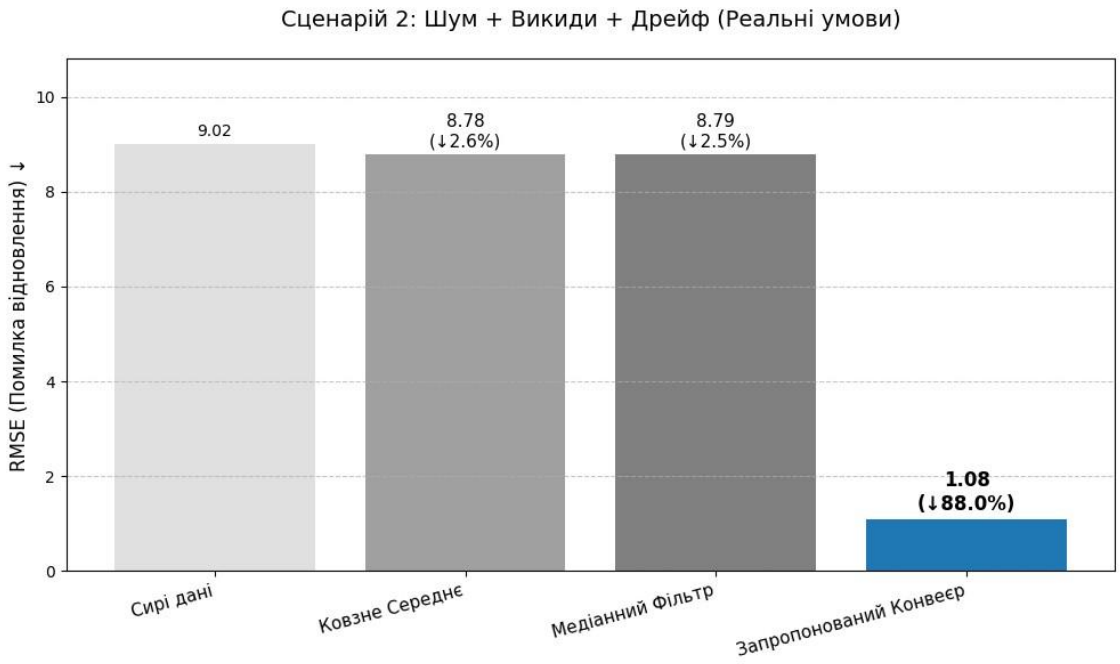
8

ПОРІВНЯЛЬНИЙ АНАЛІЗ ЕФЕКТИВНОСТІ МЕТОДІВ



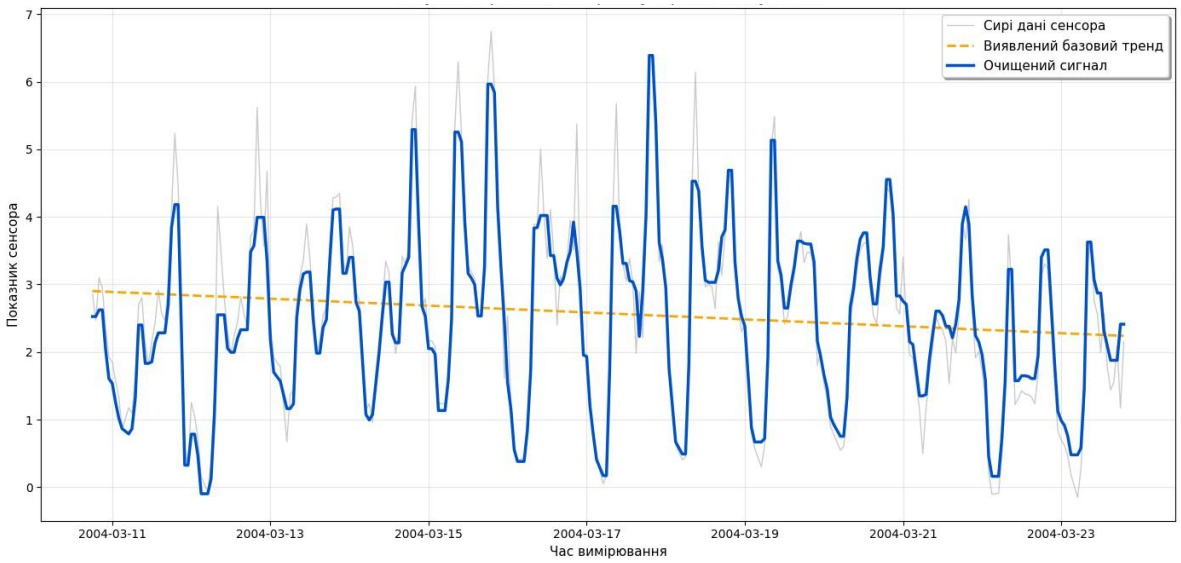
9

ПОРІВНЯЛЬНИЙ АНАЛІЗ ЕФЕКТИВНОСТІ МЕТОДІВ



10

РЕЗУЛЬТАТ РОБОТИ АЛГОРИТМУ ОЧИЩЕННЯ НА РЕАЛЬНИХ ДАНИХ



11

ВИСНОВКИ

1. Обґрунтовано вибір чотирирівневої моделі системи моніторингу. Визначено, що попереднє очищення даних доцільно виконувати на проміжному рівні перед їх збереженням та аналізом.
2. Систематизовано типові дефекти даних сенсорів: апаратні шуми, пропуски, викиди та дрейф. Встановлено, що низька достовірність первинних вимірювань є основною проблемою бюджетного обладнання.
3. Розроблено математичну модель вхідного сигналу, яка враховує корисну складову та чотири компоненти спотворень: шум, викиди, дрейф та втрату даних.
4. Сформовано метод автоматизованого очищення даних у вигляді конвеєра, що включає лінійну інтерполяцію, фільтрацію викидів, корекцію дрейфу та медіанне згладжування.
5. Реалізовано програмний модуль мовою Python. Його архітектура забезпечує потокову обробку даних та можливість інтеграції у хмарні платформи або периферійні пристрої.
6. Експериментально підтверджено ефективність методу: в умовах дрейфу сенсора середньоквадратична похибка знизилася на 88%, що перевищує показники базових методів фільтрації.
7. Проведено апробацію на реальному наборі даних Air Quality Data Set. Підтверджено здатність алгоритму компенсувати тренд дрейфу та відновлювати динаміку показників без втрати інформативності.

12

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Іванчук В.О., Золотухіна О.А. Інтелектуальний аналіз та обробка IoT-даних в системах моніторингу навколишнього середовища. VI Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С.561.
2. Іванчук В.О., Золотухіна О.А. Підвищення якості моніторингу екологічного стану розумних міст шляхом очищення IoT-даних. VI Міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку IoT», 15 квітня 2025 р., Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2025. С 226.

ДОДАТОК Б. ЛІСТИНГ ПРОГРАМНОГО КОДУ

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error


def run_simulation_and_get_metrics(with_drift=False, return_full_data=False):
    np.random.seed(42)
    n_points = 1000
    time = np.linspace(0, 100, n_points)

    ideal = 10 + 5 * np.sin(time * 0.5) + 2 * np.sin(time * 2)
    noise = np.random.normal(0, 1.5, n_points)

    drift = np.linspace(0, 15, n_points) if with_drift else np.zeros(n_points)

    raw = ideal + noise + drift

    df = pd.DataFrame({'time': time, 'ideal': ideal, 'raw': raw})

    n_missing = int(n_points * 0.05)
    df.loc[np.random.choice(df.index, n_missing, replace=False), 'raw'] = np.nan

    n_outliers = 20
    df.loc[np.random.choice(df.index, n_outliers, replace=False), 'raw'] += np.random.choice([-15, 15], n_outliers)

    rmse_raw = np.sqrt(mean_squared_error(df['ideal'], df['raw'].fillna(df['raw'].mean())))

    sma = df['raw'].interpolate().rolling(10).mean().bfill().ffill()
    rmse_sma = np.sqrt(mean_squared_error(df['ideal'], sma))

    median = df['raw'].interpolate().rolling(10).median().bfill().ffill()
    rmse_median = np.sqrt(mean_squared_error(df['ideal'], median))

    step1 = df['raw'].interpolate(method='linear').bfill().ffill()

    roll_med = step1.rolling(50, center=True).median().bfill().ffill()
    roll_std = step1.rolling(50, center=True).std().bfill().ffill()
    mask = (step1 < (roll_med - 3*roll_std)) | (step1 > (roll_med + 3*roll_std))
    step1_no_outliers = step1.copy()
    step1_no_outliers[mask] = np.nan
    step2 = step1_no_outliers.interpolate(method='linear').bfill().ffill()

    x = np.arange(len(step2))
    slope, intercept = np.polyfit(x, step2, 1)
    trend_line = slope * x + intercept
    step3 = step2 - (slope * x)

    final = step3.rolling(7, center=True).median().bfill().ffill()
    rmse_pipeline = np.sqrt(mean_squared_error(df['ideal'], final))

    if return_full_data:
        df['step4_cleaned'] = final
        return df, trend_line, rmse_pipeline, rmse_raw
```

```
    return np.array([rmse_raw, rmse_sma, rmse_median, rmse_pipeline])


def plot_chart(rmse_values, title_text):
    methods = ['Сирі дані', 'Ковзне Середнє', 'Медіанний Фільтр', 'Запропонований Конвеєр']
    colors = ['#E0E0E0', '#A0A0A0', '#808080', '#1f77b4']

    base_rmse = rmse_values[0]
    improvement = (1 - rmse_values / base_rmse) * 100

    plt.figure(figsize=(10, 6))
    bars = plt.bar(methods, rmse_values, color=colors)

    plt.title(title_text, fontsize=14, pad=20)
    plt.ylabel('RMSE (Помилка відновлення) ↓', fontsize=12)

    plt.ylim(0, max(rmse_values) * 1.2)
    plt.grid(axis='y', linestyle='--', alpha=0.7)
    plt.xticks(rotation=15, ha='right', fontsize=11)

    for i, bar in enumerate(bars):
        yval = bar.get_height()

        if i == 0:
            label = f'{yval:.2f}'
            fontsize = 10
            fw = 'normal'
        else:
            label = f'{yval:.2f}\n(↓{improvement[i]:.1f}%)'
            fontsize = 11
            fw = 'normal'
            if i == 3:
                fontsize = 12
                fw = 'bold'

        plt.text(bar.get_x() + bar.get_width()/2, yval + 0.1, label,
                 ha='center', va='bottom', fontsize=fontsize, fontweight=fw)

    plt.tight_layout()
    plt.show()


def plot_scenario1_detailed_process(df, improvement_pct):
    plt.figure(figsize=(14, 7))

    plt.plot(df['time'], df['raw'], color='lightgray', label='Сирі дані (Шум + Викиди)', zorder=1)

    plt.plot(df['time'], df['step4_cleaned'], color='blue', linewidth=2.5, label='Результат (Шум/Викиди усунуто)', zorder=3)

    plt.plot(df['time'], df['ideal'], color='green', linestyle='--', alpha=0.8, label='Істинний сигнал', zorder=2)

    plt.title(f'Деталізація Сценарію 1: Тільки Шум та Викиди (Без дрейфу)\nЗниження помилки RMSE на {improvement_pct:.1f}%', fontsize=14)
    plt.xlabel('Час')
    plt.ylabel('Амплітуда')
    plt.legend()
    plt.grid(True, alpha=0.3)
    plt.tight_layout()
    plt.show()


def plot_detailed_process(df, trend_line, improvement_pct):
    plt.figure(figsize=(14, 7))
```

```
    plt.plot(df['time'], df['raw'], color='lightgray',
label='Сирі дані (Шум + Викиди + ДРЕЙФ)',
zorder=1)


    plt.plot(df['time'], trend_line, color='orange',
linestyle='--', linewidth=2, label=f'Виявлений дрейф
сенсора', zorder=4)


    plt.plot(df['time'], df['step4_cleaned'], color='blue',
linewidth=2.5, label='Результат (Дрейф усунуто)',
zorder=3)


    plt.plot(df['time'], df['ideal'], color='green',
linestyle='--', alpha=0.8, label='Істинний сигнал',
zorder=2)


    plt.title(f'Деталізація Сценарію 2: Повний цикл (3
дрейфом)\nЗниження помилки RMSE на
{improvement_pct:.1f}%', fontsize=14)

    plt.xlabel('Час')

    plt.ylabel('Амплітуда')

    plt.legend()

    plt.grid(True, alpha=0.3)

    plt.tight_layout()

    plt.show()


print("Запуск Сценарію 1 (Метрики)...")

rmse_no_drift =
run_simulation_and_get_metrics(with_drift=False)

plot_chart(rmse_no_drift, 'Сценарій 1: Тільки Шум
та Викиди (Метрики)')


print("Запуск Сценарію 1 (Деталізація)...")

data_s1, _, rmse_res_s1, rmse_raw_val_s1 =
run_simulation_and_get_metrics(with_drift=False,
return_full_data=True)

improv_pct_s1 = (1 - rmse_res_s1 / rmse_raw_val_s1)
* 100

plot_scenario1_detailed_process(data_s1,
improv_pct_s1)


print("\nЗапуск Сценарію 2 (Метрики)...")

rmse_with_drift =
run_simulation_and_get_metrics(with_drift=True)

plot_chart(rmse_with_drift, 'Сценарій 2: Шум +
Викиди + Дрейф (Метрики)')


print("Запуск Сценарію 2 (Деталізація)...")

data_cleaned, detected_trend, rmse_res, rmse_raw_val
= run_simulation_and_get_metrics(with_drift=True,
return_full_data=True)

improv_pct = (1 - rmse_res / rmse_raw_val) * 100

plot_detailed_process(data_cleaned, detected_trend,
improv_pct)
```