Rainfall-Aware Hybrid Rule-Based Framework for Detecting Sewer Blockages

Technical Report – MS984

Group 3 – Case Study (Jacobs)

Ebrima Khan, Muhammad Qureshi, Daayum Mohsin, Shivangi Sinha, Jahnavi

Contents

1. Executive Summary

Urban sewer networks are complex systems designed to transport wastewater and stormwater safely to treatment facilities. When blockages occur within these systems, water levels may rise abnormally, potentially leading to overflows, flooding, environmental contamination, and costly emergency maintenance. Early detection of such blockages is therefore critical for operational resilience and environmental protection.

The client requested the development of an analytical process capable of automatically raising alerts for potential sewer blockages at specific CSO (Combined Sewer Overflow) locations or across regions. The solution needed to be robust, interpretable, and suitable for operational deployment using historical sensor and rainfall data.

This report presents a rainfall-aware hybrid rule-based detection framework that integrates statistical time-series analysis with engineering-based decision logic. The approach is designed to distinguish genuine blockage behavior from rainfall-driven fluctuations, ensuring reliable alert generation.

2. Problem Definition

The fundamental challenge in blockage detection lies in differentiating between three types of behavior:

1. Normal diurnal sewer level fluctuations
2. Temporary rises caused by rainfall
3. Sustained abnormal increases caused by blockages

Sewer levels naturally vary throughout the day due to usage patterns. Rainfall further complicates interpretation, as heavy rain can cause rapid level increases that resemble blockage conditions. Therefore, a naïve threshold-based approach would generate excessive false alarms.

The desired system must:

- Operate independently for each CSO location
- Adapt to site-specific behaviour
- Filter rainfall effects
- Detect persistent abnormal conditions
- Generate structured alerts with duration and timestamps

The objective is not merely anomaly detection, but operationally meaningful alert generation.

3. Data Description & Analysis Tool

We used python as a tool for computation and analysis along with:

- pandas for data cleaning, merging, rolling windows
- Numpy numerical logic, thresholds
- Matplotlib for time-series visualisations

Three primary datasets were used in the analysis.

3.1 CSO Level Data (2017–2020)

This dataset contains time-series measurements of water levels recorded at multiple CSO sites. Measurements were taken at high frequency and span four years. These values represent the physical water height inside sewer chambers and form the core signal for blockage detection.

Because each site exhibits different depth ranges and operational characteristics, site-specific modelling is necessary.

3.2 Rainfall Data

Hourly rainfall data was used to classify weather conditions. Rainfall is a major confounding factor because storm events can cause rapid but temporary level increases. Without rainfall filtering, the algorithm would misclassify rainfall events as blockages.

Rainfall values were resampled and aligned with sewer timestamps to enable joint analysis.

3.3 Metadata

The metadata file provides location identifiers and asset information. This allows alerts to be associated with specific CSO assets and enables potential regional aggregation.

4. Methodology

The proposed solution follows a structured, multi-phase analytical pipeline.

Phase 1: Data Preparation

All CSO level datasets (2017–2020) were merged into a unified dataset and resampled to consistent 15-minute intervals. This ensured temporal alignment across sites.

Rainfall data was also resampled to 15-minute resolution and rolling rainfall accumulations (1h, 3h, 6h) were computed. The datasets were then merged on timestamp, allowing each sewer measurement to be associated with recent rainfall history.

Data quality checks were performed to identify missing values and irregular intervals.

Phase 2: Feature Engineering

To detect abnormal behavior, dynamic statistical baselines were required.

For each CSO independently, a rolling 24-hour 99th percentile baseline was calculated. This means that at every timestamp, the system computes the 99th percentile of water levels observed over the previous 24 hours.

$$Baseline(t) = Rolling\ 99th\ percentile\ of\ level\ over\ previous\ 24h$$

$$Rain_{6h} < Threshold$$

This approach offers several advantages:

- It adapts to site-specific behaviour.
- It accounts for natural daily cycles.
- It adjusts gradually to seasonal shifts.
- It avoids fixed thresholds that may not generalise.

The rolling percentile acts as a dynamic upper bound of “normal behaviour.”

Phase 3: Hybrid Rule-Based Detection

The detection framework combines statistical anomaly detection with engineering rules.

Three conditions must be satisfied:

1. The observed level exceeds the rolling 99th percentile baseline.
2. The weather is classified as dry (low rainfall accumulation in the past six hours).
3. The abnormal condition persists for at least four consecutive hours.

$$Abnormal(t) = \begin{cases} 1 & \text{if } level(t) > Baseline(t) \\ 0 & \text{otherwise} \end{cases}$$

$$Dry(t) = \text{Rain accumulation over last 6h} < threshold$$

Then:

$$AbnormalDry(t) = Abnormal(t) \wedge Dry(t)$$

Persistence is critical. Short spikes may occur naturally or due to sensor noise. Sustained elevation is more indicative of obstruction or reduced discharge capacity.

Only when all conditions are satisfied does the system generate a blockage alert.

This hybrid structure balances sensitivity and robustness.

5. Alert Generation Process

When a persistent abnormal dry-weather condition is detected, the system groups consecutive timestamps into a single event. For each event, the following information is recorded:

- CSO identifier
- Start time
- End time
- Duration (hours)

These structured event records represent actionable alerts.

In operational deployment, this alert output could be:

- Displayed on dashboards
- Integrated into asset management systems
- Used to trigger maintenance inspections

The system operates independently per site, enabling both site-level and network-wide monitoring.

6. Results and Validation

Visual validation plots demonstrate that the system correctly identifies sustained abnormal behavior. During normal operation, water levels fluctuate below the dynamic baseline. During detected events, levels rise above the baseline and remain elevated for extended periods.

Rainfall-driven increases were successfully excluded when rainfall thresholds were exceeded.

The rolling percentile baseline proved effective in adapting to local site behaviour, while persistence filtering reduced false positives.

7. Strengths

The framework offers several strengths:

- High interpretability
- Site-adaptive thresholds
- Rainfall-aware filtering
- No requirement for labelled blockage data
- Operational simplicity
- Scalability across regions

Unlike machine learning black-box models, this approach provides clear reasoning behind each alert.

8. Limitations and Future Improvements

The framework relies on parameter choices such as persistence duration and rainfall thresholds. These may require calibration based on operational feedback.

Additionally, the system detects ongoing blockages but does not predict future failures. Future work could incorporate predictive modelling techniques or integrate additional sensor types such as flow rate.

Despite these limitations, the approach provides a robust and deployable foundation.

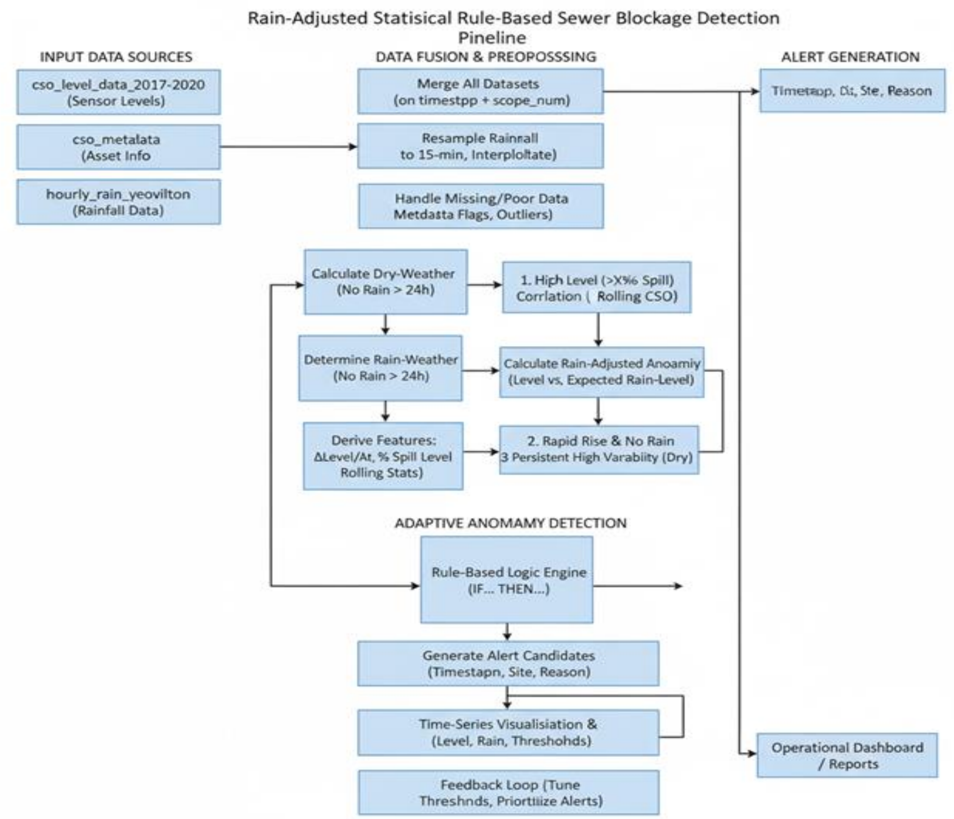
9. Conclusion

This project developed a rainfall-aware hybrid rule-based detection framework capable of identifying potential sewer blockages at specific locations.

By combining rolling statistical baselines, rainfall filtering, and persistence logic, the system generates structured and interpretable alerts suitable for operational deployment.

The framework satisfies the client’s requirement for an algorithm capable of raising alerts across locations while remaining transparent and scalable

Appendix A – Framework Process Flow

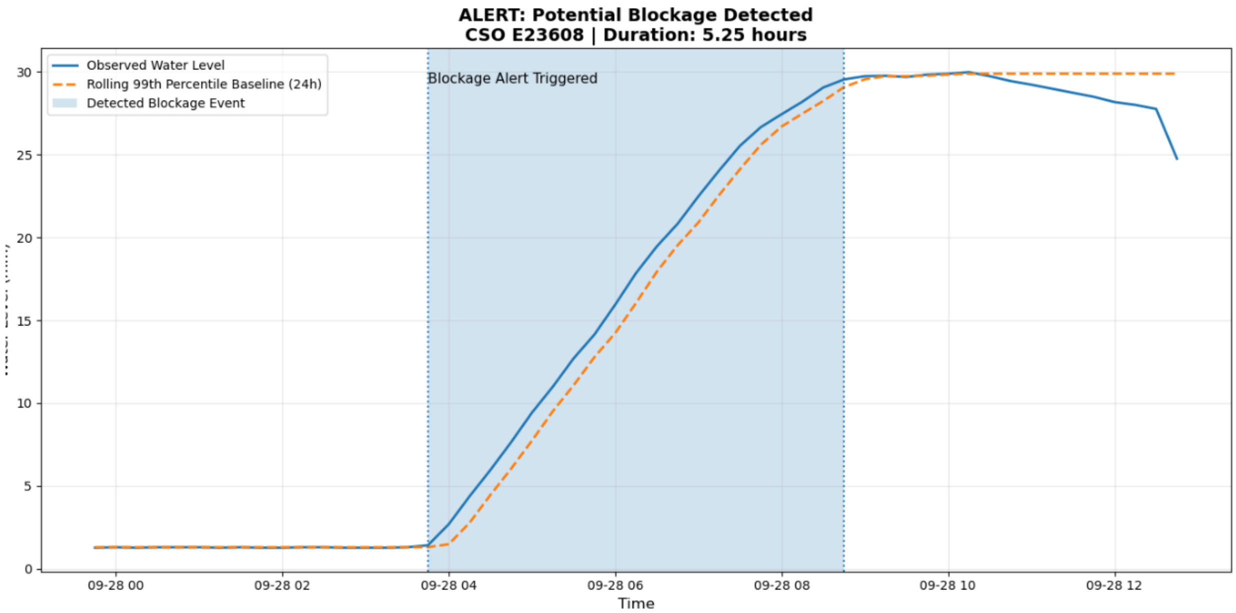


This diagram shows the process flow of the entire hybrid rule-base detection framework which includes:(Input data, Preprocessing, Feature derivation, Rule-based logic, Alerts & outputs), Rainfall explicitly integrated, Rule-based logic engine, Visualization and feedback loop,

Appendix B – Algorithm Pseudocode

```
FOR each CSO:

    Compute rolling 24h 99th percentile baseline

    FOR each timestamp t:
        IF level(t) > baseline(t) AND rainfall_6h(t) < threshold:
            mark as abnormal_dry

    Group consecutive abnormal_dry timestamps

    IF duration >= 4 hours:
        Generate blockage alert
```

Appendix C – Additional Validation Plots



This plot sure why and how an alert occurred. The blue shows the actual measured sewer water level over time. The orange line shows the actual measured sewer water level over time and lastly, the blue shaded area shows time window where the algorithm detected sustained abnormal dry-weather conditions

Appendix D – Alert Generation per CSO

	scope_num	event_group	start_time	end_time	duration_points	duration_hours
0	E23608	289	2019-09-28 03:45:00	2019-09-28 08:45:00	21	5.25
1	E26824	934	2020-02-17 05:00:00	2020-02-17 08:45:00	16	4.00

An alert is generated when sewer level exceeds the rolling 99th percentile baseline during dry weather for a minimum persistence of four hours. Each alert is recorded as a structured event containing location, start time, end time, and duration. Figure X demonstrates one such alert event**.**

Appendix D – Sensitivity Analysis (4h vs 3h vs 6h persistence)

	Persistence (Hours)	Min Intervals	Number of Alerts
0	3	12	10
1	4	16	2
2	6	24	0

A sensitivity analysis was conducted by varying the persistence threshold from 3 to 6 hours. As expected, shorter persistence durations resulted in a higher number of detected events, while longer persistence thresholds reduced alert frequency. The 4-hour threshold provided a balanced trade-off between responsiveness and false alarm reduction.

Appendix E – GitHub Link to Code

https://github.com/KhanJallow/sewer_blockage_detection