Filtering in Preprocessing for Anomaly Detection in Time Series [*†]

Ricardo Fernandes de Araujo[1], Laís Baroni[1], Eduardo Bezerra[1], Gustavo Guedes[1], Esther Pacitti[2], Larissa Pinheiro[3], Diego Salles[1], Fabio Porto[4], Eduardo Ogasawara[1]

[1]Centro Federal de Educação Tecnológica Celso Suckow da Fonseca (CEFET/RJ)

[2]University of Montpellier & INRIA

[3]Universidade Federal do Rio de Janeiro (UFRJ)

[4]Laboratório Nacional de Computação Científica (LNCC)

{ricardo.araujo, lais.baroni, ebezerra, gustavo.guedes}@cefet-rj.br

Esther.Pacitti@lirmm.fr, fporto@lncc.br, eogasawara@ieee.org

Abstract. *Filtering is often adopted as a preprocessing step in time series pipelines, although its impact on anomaly detection remains unclear. We evaluate 14 filters with an ARIMA-based detector on the GECCO Industrial Challenge 2018 and Yahoo! Webscope S5 benchmarks. Filtering improves F1-Score in only 25.36% of the runs; in the remaining 74.64%, it either fails to improve performance or degrades it. The results reveal a trade-off between stability and potential gains. Adaptive filters are more consistent, whereas decomposition-based filters can yield larger gains but also more frequent degradation. Within this ARIMA-based protocol, filtering does not behave uniformly across series and should be treated as a context-dependent engineering choice.*

1. Introduction

Anomaly detection in time series is a key component of many practical applications, such as industrial monitoring, financial systems, energy management, healthcare, and computer networks. In these settings, anomalies are often associated with failures, fraud, or sudden behavioral changes that require attention [Gupta et al., 2014; Ogasawara et al., 2025b; Zamanzadeh Darban et al., 2024]. Improving detection reliability is therefore an important concern in real-world deployments.

Filtering is often adopted as an auxiliary preprocessing step in time series pipelines under the assumption that smoothing fluctuations and reducing noise can make relevant temporal structures more apparent. At first glance, this seems beneficial for downstream modeling and anomaly detection. However, this intuition does not always hold. Once filtering is applied, the original signal is modified, and the change affects more than noise alone. In residual-based ARIMA detection, the model is fitted to the filtered series, so

[*]The authors thank CAPES, FAPERJ, and CNPq for partial support of this research

[†]Generative AI tools were used only for linguistic revision, with a focus on textual clarity. The authors reviewed all suggestions and remain fully responsible for the article's content.

the resulting residuals and, indirectly, the decision threshold are influenced by the transformed signal rather than the original behavior. As a result, filtering may attenuate irrelevant fluctuations, but it may also distort patterns that would otherwise be identified as anomalies. Its effect on detection performance therefore cannot be assumed to be uniformly beneficial [Gea et al., 2021].

Against this background, we investigate the extent to which preprocessing filters affect the performance of an ARIMA-based anomaly detector. Rather than proposing a new detector, we isolate a preprocessing step often treated as harmless through controlled experiments that combine multiple filters with a fixed detector on two complementary benchmarks, so that performance differences can be attributed to preprocessing.

The main contributions of this work are: (i) a systematic evaluation of the impact of multiple filters on ARIMA-based anomaly detection against labeled ground truth across two heterogeneous benchmarks; and (ii) an empirical characterization of the trade-off among reliability, risk, and upside potential across different filters and filter families.

Besides this introduction, Section 2 reviews related work, Section 3 describes the experimental design, Section 4 presents the results, and Section 5 concludes the article.

2. Literature Review

Gupta et al. [2014] and Zamanzadeh Darban et al. [2024] emphasize that anomalous behavior must be interpreted in light of temporal context rather than only isolated values. They also show that the literature spans broad methodological families. This perspective is relevant here because preprocessing can modify the sequential structure of a series and, consequently, the temporal patterns against which deviations are identified.

In forecasting-based approaches, anomaly detection is often carried out through residual analysis, which compares observations with model outputs. ARIMA is widely used in this context [Yaacob et al., 2010; Pena et al., 2013; Zhu and Sastry, 2011], but its behavior is affected by irregular observations. As discussed by Contreras et al. [2003], atypical values can interfere with both the fitting process and the stability of the model. Transformations applied before modeling may therefore alter the conditions under which detection is performed.

Along the filtering axis, Gea et al. [2021] evaluate outlier-treatment methods for ARIMA model performance and warn about the risk of excessive smoothing. More broadly, filtering is frequently adopted as an auxiliary preprocessing step, yet its downstream impact on anomaly detection is rarely assessed under a systematic, comparable, ground-truth-based protocol. Taken together, this suggests that the gap addressed in this article is not a lack of detection methods or filtering techniques, but a lack of systematic experimental evidence on *when* filtering helps, *when* it hurts, and *which filter profiles* fit different temporal regimes within a fixed ARIMA-based protocol.

3. Experimental Design

3.1. Experimental Overview

The study adopts a comparative experimental design. In each run, we apply a filter to a time series, fit an ARIMA model to the filtered series, compute the model's *in-sample* fitted values, and obtain residuals as the difference between the original series and those

fitted values. Residuals are computed against the original series to preserve anomaly signals that filtering might otherwise attenuate. Formally, a run is defined as $e = (s, f)$, where filter f is applied to series s and always compared with the unfiltered *baseline* of the same series. The variation relative to the *baseline* is measured by

$$\Delta F1(e) = F1_{filter}(e) - F1_{baseline}(s), \tag{1}$$

$$\Delta F1_{\%}(e) = 100 \cdot \Delta F1(e)/F1_{baseline}(s), \quad \text{when } F1_{baseline}(s) > 0. \tag{2}$$

The 14 filters evaluated in this study are all available in the TSPredIT [Ogasawara et al., 2025a] framework. The complete set includes average filters such as Moving Average `MA` and Exponential Moving Average `EMA`, adaptive filters such as `Kalman` and `RFilter`, and the robust filter `Winsor`. It also includes smoothing filters such as Hodrick–Prescott `HP`, `LOWESS`, `Smooth`, and `Spline`, as well as decomposition-based filters such as Empirical Mode Decomposition `EMD`, Fast Fourier Transform `FFT`, `REMD`, `Seasonal_Adj`, and `Wavelet`.

3.2. Benchmarks

We use two benchmarks to represent different time series conditions. The GECCO Industrial Challenge 2018 dataset consists of multivariate sensor data from a water-treatment plant [Mao et al., 2017; Inoue et al., 2017]. In our protocol, we use a 24-hour slice, evaluate nine variables individually, and derive ground truth directly from `EVENT`.

The second benchmark is Yahoo! Webscope S5, which contains univariate series distributed across four subsets, with real and synthetic data under different noise levels and regime changes [Ishimtsev et al., 2017]. Four series show consistency problems and are therefore removed, leaving 363 valid series. With 14 filters, plus the *baseline*, this yields 5,445 runs on Yahoo! and 135 on GECCO. The baseline is used only as a reference for comparison and is not included in the aggregate statistics, resulting in 5,082 runs on Yahoo! and 126 on GECCO.

3.3. Framework Integration

To enable the study, we adapt the `arima_filter` detector from the Harbinger framework [Salles et al., 2020] to accept filter objects from *TSPredIT*. All filters are used with their default parameter settings, without per-series or per-domain tuning. The pipeline applies the selected filter to the series, fits an ARIMA model using `auto.arima()` [Hyndman and Athanasopoulos, 2018], and computes residuals by comparing the original series with the corresponding *in-sample* fitted values. Detection is then performed in residual space using the same `arima_filter` criterion in all runs. The decision is based on the absolute residual magnitude (L1 distance), using the default thresholding criterion provided by the Harbinger framework with the fixed sensitivity parameter $k = 3$, as commonly adopted in residual-based detection [Pena et al., 2013; Zhu and Sastry, 2011].

The ARIMA fitting and detection procedure remains the same across all runs, with changes applied only at the preprocessing stage. Differences in performance are therefore associated with the filtering step.

3.4. Metrics and Analysis Criteria

We use Precision, Recall, and F1-Score, with F1 as the main reference. For a set E of runs, we define

$$\text{Improvement}(E) = \frac{1}{|E|} \sum_{e \in E} \mathbf{1}[\Delta F1(e) > 0], \tag{3}$$

$$\text{Degradation}(E) = \frac{1}{|E|} \sum_{e \in E} \mathbf{1}[\Delta F1(e) < 0], \tag{4}$$

$$\text{Potential}(E) = \frac{1}{|E^+|} \sum_{e \in E^+} \Delta F1_{\%}(e), \tag{5}$$

where $E^+ = \{e \in E : \Delta F1(e) > 0\}$.

We report how often improvements and degradations occur, together with their magnitude. When $F1_{baseline} = 0$, percentage variation is not considered to avoid misleading interpretations. In such cases, the run is treated as *no change* in the aggregate analysis.

4. Results

4.1. Global Impact of Filtering

The first experimental question is whether filtering improves anomaly detection in aggregate terms. Under the adopted protocol, the answer is largely negative. Considering GECCO and Yahoo! together, only 25.36% of filter applications improve F1-Score, while 59.16% degrade performance and 15.48% produce no measurable change. Table 1 summarizes this pattern in relative terms and, for the aggregate total, also in absolute counts.

Table 1. Global impact of filtering on F1-Score.

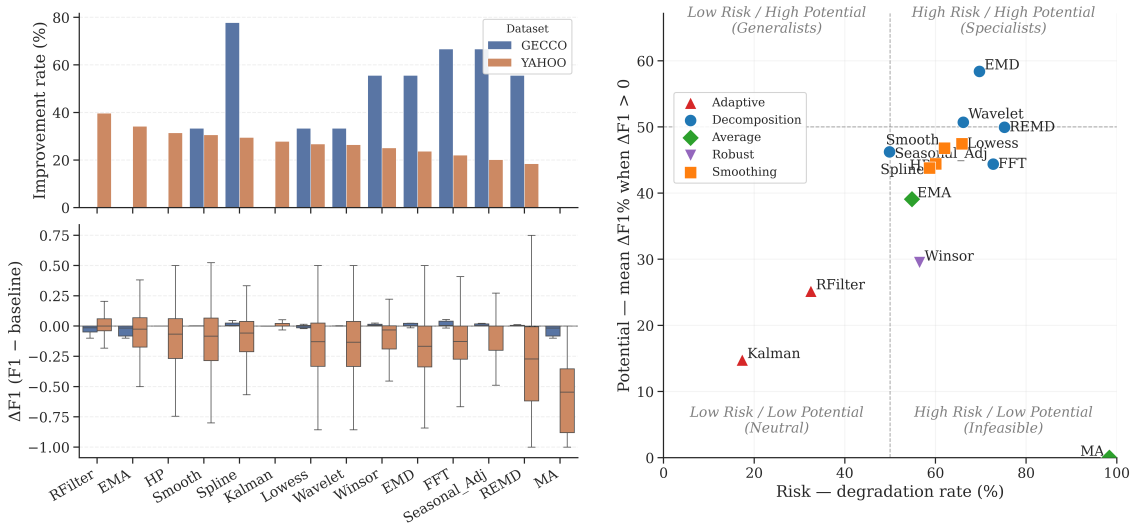
Dataset	Runs	▲	▼	■
GECCO	126	26.98%	26.19%	46.83%
Yahoo!	5,082	25.32%	59.98%	14.70%
Total	5,208	25.36% (1,321)	59.16% (3,081)	15.48% (806)

Overall, the results do not support the idea that filtering before detection consistently improves the signal seen by the model under this protocol. Despite the different profiles of the two benchmarks and the much larger scale of Yahoo!, the proportion of F1-Score improvements is similar in both cases. This suggests that the main practical issue is whether a given filter is more likely to degrade performance than to provide any gain.

4.2. Reliability and Potential

However, an aggregate view alone would hide important differences among filters. To make this explicit, Figure 1(a) shows how often each filter improves F1-Score and the distribution of $\Delta F1$ by filter, highlighting the trade-off between reliability and potential.

Figures 1(a,b) show that adaptive filters, especially `Kalman` and `RFilter`, tend to remain closer to zero variation and are less prone to severe performance degradation. In



(a) Improvement rate and ΔF1 distribution **(b) Risk vs Potential Matrix**

Figure 1. Filter performance profiles. (a) Improvement rate (top, %) and $\Delta F1$ distribution (bottom) by filter and dataset; absent bars indicate 0% improvement. (b) Risk-versus-potential matrix: risk is the per-filter degradation rate (Eq. 5) and potential is the mean $\Delta F1_{\%}$ conditional on improvement (Eq. 6); dashed lines mark the 50% threshold on each axis. Filter categories are distinguished by marker shape.

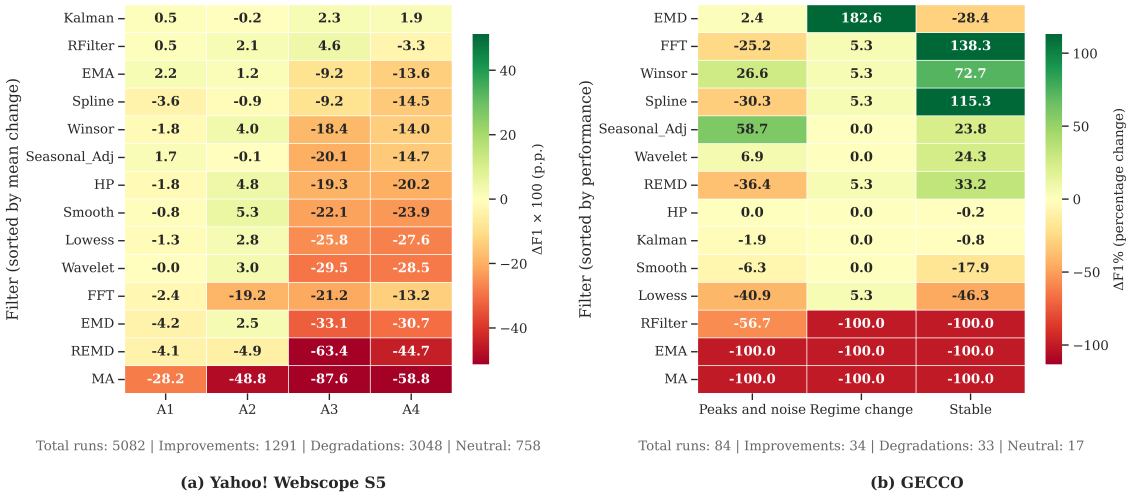
Yahoo!, `Kalman` has a degradation rate of 17.36%, whereas `RFilter` reaches 39.67% improvement and 32.51% degradation. By contrast, decomposition-based filters, such as `EMD`, `Wavelet`, and `REMD`, show more dispersed behavior: large gains appear in some runs, but failures are also more frequent.

This pattern is even clearer in the risk-versus-potential matrix in Figure 1(b), built with 50% thresholds on both axes, where **risk** corresponds to the per-filter degradation rate (Eq. 5) and **potential** to the mean $\Delta F1_{\%}$ conditional on $\Delta F1 > 0$ (Eq. 6). In the evaluated set of filters and benchmarks, no filter lies in the high-potential, low-risk quadrant. Decomposition-based filters appear as *specialists*: they deliver large gains when they match the series profile, but they also carry a high probability of degradation, above 65% in Yahoo! for `EMD`, `Wavelet`, and `REMD`. By contrast, adaptive filters show the most neutral and reliable behavior. In the opposite direction, the simple moving average appears as a prohibitive-risk option in this protocol, with 98.35% degradation in Yahoo!.

4.3. Dependence on Series Regime

The effect of filtering strongly depends on the temporal regime. Figures 2(a) and 2(b) summarize percentage changes in F1-Score by filter across different regimes or subsets.

In Yahoo! Webscope S5, the subsets behave quite differently. A1 (real data with peaks, noise, and varied seasonality) and A2 (synthetic series with low noise and seasonal peaks) show more cases in which filtering helps, with several filters yielding positive results. By contrast, A3 (synthetic series with high noise and seasonal peaks) is dominated by noise, and most filters perform poorly there; adaptive filters are among the few that still behave reasonably well. In A4 (regime-change series), `Kalman` tends to be one of the



(a) Yahoo! Webscope S5 **(b) GECCO**

Figure 2. Mean change in F1-Score by filter across distinct temporal regimes. (a) Yahoo! Webscope S5: mean $\Delta F1 \times 100$ in percentage points (Eq. 2). (b) GECCO: mean $\Delta F1_{\%}$ as percentage change relative to baseline (Eq. 3). Green cells indicate improvement; red cells indicate degradation.

more consistent options. Overall, the effect of filtering depends on the temporal regime, although the pattern is not uniform across cases.

In GECCO, the results are less clear, which is not surprising given the smaller number of series. Under the adopted variable groups, `EMD` appears more often among the cases with the largest gains in regime-change variables. In contrast, `RFilter` performs poorly in both noisy and stable contexts. Additionally, in stable series and in cases where noise is more localized, simple smoothing usually provides limited benefit. In these situations, robust or adaptive filters tend to behave more consistently.

4.4. Discussion

Filtering is treated here as part of the *pipeline* design. Because the ARIMA detector operates on residuals, changes introduced before modeling propagate to the fitted model and to the residuals, thereby affecting the decision rule. In this setting, adaptive filters tend to be more stable, whereas more specialized filters show gains only when the characteristics of the series align with the assumptions of the filter and when some degradation can be tolerated.

5. Conclusion

This article investigates the extent to which preprocessing filters affect the performance of an ARIMA-based anomaly detector. Under the adopted experimental protocol, indiscriminate filtering tends to be harmful: across the two evaluated benchmarks, only 25.36% of applications improve the F1-Score, while 74.64% fail to do so. The results show no universally best filter, but a clear trade-off between reliability and potential. Adaptive filters, such as Kalman and RFilter, tend to be more stable across runs, whereas decomposition-based filters, including EMD and Wavelet, sometimes achieve higher gains but also show more frequent degradation. These conclusions are restricted to the evaluated ARIMA-based setting.

References

Contreras, J., Espínola, R., Nogales, F. J., and Conejo, A. J. (2003). ARIMA models to predict next-day electricity prices. *IEEE Transactions on Power Systems*, 18:1014 – 1020.

Gea, C., Lima, J., Bezerra, E., and Ogasawara, E. (2021). Análise de métodos de tratamento de outliers para predição dos retornos de índices de ações negociados em bolsa. In *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 277–282. SBC.

Gupta, M., Gao, J., Aggarwal, C. C., and Han, J. (2014). Outlier Detection for Temporal Data: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 26:2250 – 2267.

Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Inoue, J., Yamagata, Y., Chen, Y., Poskitt, C. M., and Sun, J. (2017). Anomaly detection for a water treatment system using unsupervised machine learning. In R, G., G, K., V, R., X, W., L, M., S, A., X, N., and G, D., editors, *IEEE International Conference on Data Mining Workshops, ICDMW*, volume 2017-November, pages 1058 – 1065. IEEE Computer Society.

Ishimtsev, V., Bernstein, A., Burnaev, E., and Nazarov, I. (2017). Conformal k-NN Anomaly Detector for Univariate Data Streams. In *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, volume 60 of *Proceedings of Machine Learning Research*, pages 213–227. PMLR.

Mao, Y., Qi, H., Chen, X., and Li, X. (2017). Event Detection with Multivariate Water Parameters in the Water Monitoring Applications. In *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017*, pages 321 – 326.

Ogasawara, E., Alexandrino, F., Gea, C., Santos, D., Salles, R., Birindiba, V., Pacheco, C., Bezerra, E., Pacitti, E., Porto, F., and Carvalho, D. (2025a). tspredit: Time Series Prediction with Integrated Tuning.

Ogasawara, E., Salles, R., Porto, F., and Pacitti, E. (2025b). *Event Detection in Time Series*. Synthesis Lectures on Data Management. Springer Nature Switzerland, Cham, 1 edition.

Pena, E. H. M., De Assis, M. V. O., and Proença, M. L. (2013). Anomaly Detection Using Forecasting Methods ARIMA and HWDS. In *Proceedings - International Conference of the Chilean Computer Science Society, SCCC*, volume 0, pages 63 – 66.

Salles, R., Escobar, L., Baroni, L., Zorrilla, R., Ziviani, A., Kreischer, V., Delicato, F., Pires, P. F., Maia, L., Coutinho, R., Assis, L., and Ogasawara, E. (2020). Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. In *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 73–84. SBC.

Yaacob, A. H., Tan, I. K., Chien, S. F., and Tan, H. K. (2010). ARIMA based network anomaly detection. In *2nd International Conference on Communication Software and Networks, ICCSN 2010*, pages 205 – 209.

Zamanzadeh Darban, Z., Webb, G. I., Pan, S., Aggarwal, C., and Salehi, M. (2024). Deep Learning for Time Series Anomaly Detection: A Survey. *ACM Computing Surveys*, 57:1–42.

Zhu, B. and Sastry, S. (2011). Revisit dynamic ARIMA based anomaly detection. In *Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011*, pages 1263 – 1268.