DO WE REALLY NEED COMPLEX MODELS? REVISITING PCA FOR TIME-SERIES ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series anomaly detection increasingly relies on complex deep or foundation-model architectures, while practical deployment demands methods that are accurate, efficient, and scalable. PCA offers an efficient and interpretable model of normal variation, but its standard formulation operates on static vector representations without explicitly accounting for temporal structure. We introduce TAP, a Time-series Adaptive PCA framework that addresses this limitation through three novel designs: Multi-Resolution PCA captures anomalies across different durations and temporal resolutions; Cross-Resolution Evidence Consensus identifies anomaly evidence supported by related representations; and Shape-Residual PCA detects deviations from recurrent normal patterns. We further develop efficient covariance compression and batched alignment to support scalable computation. Extensive experiments across 23 public dataset collections demonstrate state-of-the-art overall detection performance, improving upon the second-best method by 17.9% while running $10\times$ faster. On a single CPU core, TAP requires only 2.5 seconds per benchmark series on average and processes approximately 28,000 points per second, making it practical for accurate, large-scale monitoring and diagnostic workloads. The code is available at `https://anonymous.4open.science/r/TAP-TSAD`.

1 INTRODUCTION

Time series are ubiquitous in modern data-driven systems, including industrial monitoring, cloud infrastructure, finance, healthcare, transportation, and scientific observation (Nizam et al., 2022; Kashpruk et al., 2023; Huang et al., 2017; Al-Ghuwairi et al., 2023; Rousseeuw et al., 2019; Chauhan & Vig, 2015; Hadjem et al., 2016; He et al., 2023; Feigelson et al., 2018; Li & Paparrizos, 2026; d'Hondt et al., 2025; Huang et al., 2026; Papadopoulos et al., 2026). In these settings, anomaly detection is critical for identifying rare but consequential events, such as abnormal physiological patterns, equipment degradation, service disruption, or safety-critical failures (Chuah & Fu, 2007; Van Wyk et al., 2019). These events are often sparse, temporally localized, and difficult to anticipate, yet delayed detection can lead to missed interventions or costly downstream failures. Because time series are frequently long, noisy, and continuously collected, anomaly detection methods must be accurate, robust, and efficient enough for large-scale or real-time deployment.

Recent progress in time-series modeling has increasingly emphasized expressive representation learning, including deep neural networks, Transformer architectures (Vaswani et al., 2017), and large-scale time-series foundation models (Liang et al., 2024). However, the practical benefit of increasing model complexity remains unclear. A recent study on time-series anomaly detection (Zhu et al., 2026) found that complex foundation models do not consistently outperform naive one-line baselines based on simple local temporal statistics. Meanwhile, large-scale benchmarks show that classical statistical and machine-learning methods remain highly competitive, and in some cases outperform deep models that require extensive tuning and training (Liu & Paparrizos, 2024; Qiu et al., 2025). These findings suggest that traditional anomaly detection models are not merely weak baselines, and raise a natural question: *have traditional detectors been fully exploited?*

This question is especially important for subsequence-based anomaly detection. Many classical detectors operate on sliding windows, treating each subsequence as a fixed-dimensional vector (Aggarwal, 2016; Yairi et al., 2001; Li et al., 2007; Ramaswamy et al., 2000). However, time-series windows are not ordinary independent samples: adjacent windows overlap heavily, anomalies may occur at different temporal scales, periodic signals may contain phase shifts, and recurrent local shapes may dominate the observed variation. When these properties are ignored, a detector may fail to expose the most informative deviation patterns, even if the underlying scoring model is effective.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Thus, the performance of a classical detector can depend critically on whether its preprocessing and representation pipeline is aligned with the temporal structure of the data.

Motivated by this view, we revisit Principal Component Analysis (PCA) (Maćkiewicz & Ratajczak, 1993; Aggarwal, 2016), a classical linear projection method that characterizes dominant variation patterns and scores deviations from them with little cost, although it has typically been used for feature compression (Yang & Paparrizos, 2025). Figure 1 provides empirical support for this choice: the favorable runtime–accuracy frontier is mainly formed by statistical and traditional machine-learning methods, and Sub-PCA appears as an important elbow that is both highly efficient and competitive in VUS-PR. We therefore propose **T**ime-series **A**daptive **PCA** (TAP), a unified framework that equips PCA with time-series-aware representations of normal behavior and integration of anomaly evidence.

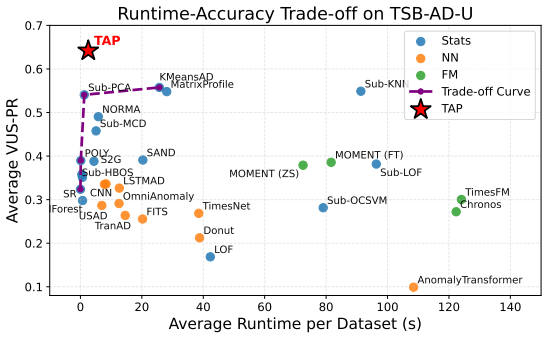


Figure 1: Runtime–accuracy trade-off on a widely used univariate benchmark. Stats, NN, and FM denote statistical/traditional machine-learning methods, neural-network-based methods, and foundation-model-based methods, respectively. The curve shows the Pareto frontier among non-TAP baselines.

TAP first develops Multi-Resolution PCA by expressing window reduction and temporal coarsening as transformations of a shared lag-covariance representation. This construction derives the covariance matrices for an entire window–scale family from a single estimate, instead of independently fitting unrelated PCA models. To distinguish corroborated anomaly evidence from isolated resolution-specific responses, we introduce Cross-Resolution Evidence Consensus (CREC). CREC organizes the resulting scores into a graph whose edges encode neighboring window and scale configurations, and formulates consensus discovery as a maximum-bottleneck path problem. We further develop Shape-Residual PCA, which phase-aligns each subsequence with a recurrent normal prototype and decomposes the residual windows after removing this dominant waveform. The resulting multi-resolution and shape-residual scores capture complementary structural deviations and are fused into the final anomaly score.

We conduct an extensive evaluation across 23 public dataset collections spanning diverse domains and anomaly patterns, complemented by ablation, scalability, and rigorous statistical significance analyses. TAP achieves state-of-the-art overall performance, improving VUS-PR by 17.9% over the second-best method while running $10\times$ faster, with Friedman and Nemenyi tests confirming significant improvements over the strongest baselines. On a single CPU core, TAP requires only 2.5 seconds per benchmark series on average and processes approximately 28,000 points per second. These results demonstrate that explicitly modeling temporal structure can substantially strengthen classical detection methods without sacrificing efficiency.

We first review preliminaries and related work (Section 2), then summarize our main contributions:

- We formulate TAP, a unified PCA framework that models temporal resolution, recurrent morphology, and anomaly-evidence relations in sliding-window time series (Section 3).
- We develop Multi-Resolution PCA, deriving a coupled window–scale covariance family from a shared lag-covariance estimate through nested covariance reduction (Section 3.1).
- We build Cross-Resolution Evidence Consensus to identify jointly supported anomaly evidence and suppress isolated resolution-specific responses (Section 3.1).
- We propose Shape-Residual PCA to expose local waveform deviations and combine it with Multi-Resolution PCA to capture complementary anomaly evidence (Sections 3.2 and 3.3).
- Experiments across 23 public dataset collections demonstrate state-of-the-art overall performance, efficient runtime, and favorable scalability (Section 4).

Finally, we conclude with the implications of our findings (Section 5).

2 Related Work and Preliminaries

In this section, we first introduce the notation used throughout the paper, and then briefly review representative time-series anomaly detection methods, including deep learning, foundation-model-based, and classical statistical or machine-learning-based approaches.

2.1 PRELIMINARY

Time series and windowing. Let $\mathbf{x} = [x_1, \ldots, x_T]^\top \in \mathbb{R}^T$ denote a univariate time series, where $x_t \in \mathbb{R}$ is the observation at timestamp t. Given a window length w and stride s, the i-th sliding window is $\mathbf{W}_i = [x_{1+(i-1)s}, \ldots, x_{1+(i-1)s+w-1}]^\top \in \mathbb{R}^w$, where $i = 1, \ldots, N$ and $N = \lfloor (T-w)/s \rfloor + 1$. Time-series anomaly detection assigns an anomaly score to each timestamp or window, with larger scores indicating stronger deviation from normal temporal patterns.

Subsequence PCA anomaly scoring (Aggarwal, 2016). Given $\mathbf{x} = [x_1, \ldots, x_T]^\top$, subsequence PCA constructs stride-one windows $\mathbf{W}_i = [x_i, x_{i+1}, \ldots, x_{i+w-1}]^\top \in \mathbb{R}^w$ for $i = 1, \ldots, T-w+1$. Let $\boldsymbol{\mu}_w$ be the mean window and $\boldsymbol{\Sigma}_w \in \mathbb{R}^{w \times w}$ be the covariance matrix of the window vectors. PCA decomposes the covariance matrix as: $\boldsymbol{\Sigma}_w = \mathbf{U}_w \boldsymbol{\Lambda}_w \mathbf{U}_w^\top$, where $\mathbf{U}_w = [\mathbf{u}_1, \ldots, \mathbf{u}_w]$ contains orthonormal principal directions and $\boldsymbol{\Lambda}_w = \mathrm{diag}(\lambda_1, \ldots, \lambda_w)$ contains eigenvalues in descending order. For each centered window $\tilde{\mathbf{W}}_i = \mathbf{W}_i - \boldsymbol{\mu}_w$, its projection on the k-th principal direction is $p_{i,k} = \mathbf{u}_k^\top \tilde{\mathbf{W}}_i$. Directions with large eigenvalues capture common temporal variations, while directions with small eigenvalues correspond to rare or residual patterns. Thus, a window with large projection on low-variance directions is more likely to be anomalous.

Following standard PCA-based anomaly detection, let $\rho_k = \lambda_k / \sum_{\ell=1}^{w} \lambda_\ell$ denote the explained variance ratio of the k-th principal direction. The PCA anomaly score is computed as:

$$A_i = \sum_{k=1}^{w} \frac{p_{i,k}^2}{\rho_k + \epsilon},$$

where ϵ is a small constant for numerical stability. This score assigns larger weights to projections on low-variance directions. In this sense, PCA detects anomalies by measuring how strongly a window deviates from the dominant subspace and enters rare residual directions.

2.2 RELATED WORK

Time-series anomaly detection. Time-series anomaly detection (TSAD) aims to identify rare or unexpected temporal behaviors in sequential data (Blázquez-García et al., 2021; Boniol et al., 2024; Liu et al., 2024; Paparrizos et al., 2025). Unlike anomaly detection in independent tabular data, TSAD must account for temporal context, since a point may appear normal marginally but become anomalous when it violates local trends, periodic patterns, temporal dependencies, or multivariate relationships. Existing methods usually operate on raw time points, extracted features, or sliding-window representations, and detect anomalies by measuring prediction error, reconstruction error, distance to normal patterns, density, cluster structure, or nearest-neighbor dissimilarity.

Deep learning and foundation-model-based methods. Deep learning methods have become a major direction in TSAD because of their ability to learn nonlinear temporal representations. Many methods are forecasting- or reconstruction-based: they predict future observations or reconstruct input windows, and use the prediction or reconstruction error as the anomaly score (Wang et al., 2024; Yue et al., 2022). Another line of work designs models specifically for anomaly detection, often assuming that training data are mostly normal (Park & Kang, 2026; Kim et al., 2026) or generating synthetic anomalies (Zhang et al., 2022; Darban et al., 2025) to learn a contrast between normal and abnormal patterns. These methods aim to learn representations of normal temporal dynamics and score new observations by their deviation from the learned normality. More recently, time-series foundation models and large pretrained models have been adapted to anomaly detection (Ekambaram et al., 2025; Lan et al., 2025). While these models provide strong representation capacity, they often require substantial training, adaptation, and tuning, making their practical advantage less clear in lightweight or real-time detection scenarios.

Statistical and traditional machine-learning-based methods. Classical statistical and machine-learning-based methods remain strong competitors in TSAD. Clustering-based methods (Yairi et al., 2001; Boniol et al., 2021b; Li et al., 2021) group sliding windows or extracted features and assign higher anomaly scores to rare clusters or windows far from cluster centroids. Discord-based methods (Yeh et al., 2016) identify subsequences whose nearest non-overlapping neighbors are far away, using the nearest-neighbor distance as a measure of whether a temporal pattern has rarely appeared before. Other traditional methods directly treat windows as fixed-dimensional vectors and apply general anomaly detectors. These approaches are often simpler than deep models, require less training and tuning, and can be more suitable for efficient deployment (Liu & Paparrizos, 2024).

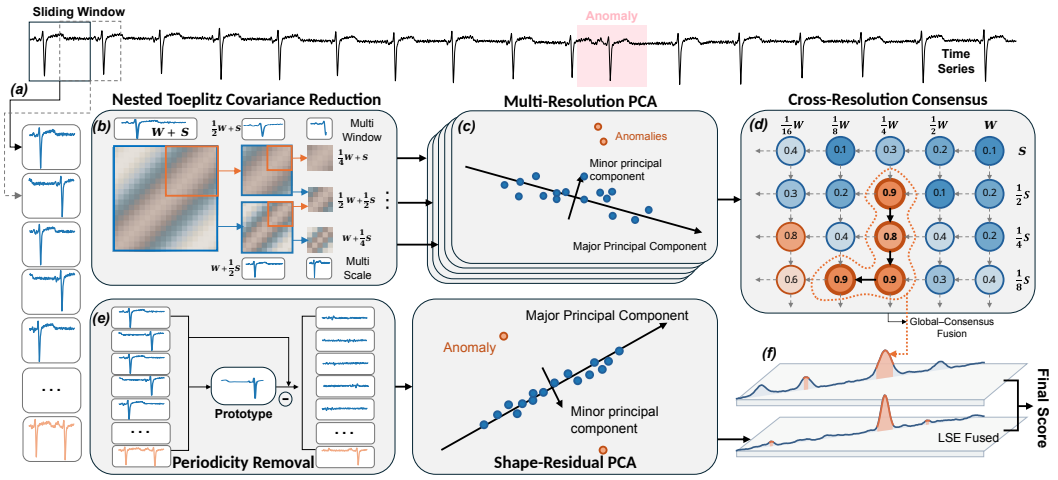


Figure 2: Overview of TAP. (a) Sliding windows are extracted from the input time series. (b) Nested Toeplitz covariance compression efficiently constructs multi-resolution covariance matrices. (c) Multi-Resolution PCA scores anomalies across multiple window sizes and temporal resolutions. (d) Cross-resolution evidence consensus identifies anomaly evidence shared across resolutions. (e) Shape-Residual PCA removes a phase-adjusted recurrent prototype and scores deviations in the residual-window representation. (f) The two branches are combined into the final anomaly score.

2.3 Limitations of Fixed-Window Detection

Despite the strong empirical performance of many subsequence-based anomaly detectors, their effectiveness is often highly sensitive to the choice of window size. When an anomaly is longer than the selected window, the detector may only observe fragmented local patterns and fail to capture the full abnormal event. Conversely, when an anomaly is much shorter than the selected window, its signal can be diluted by surrounding normal observations, making it easy to overlook. Recent work has attempted to mitigate this issue by relaxing the fixed-window assumption. For example, GDFlex (Imamura, 2025) extends classical discords to flexible subsequence lengths and detects anomalies across multiple temporal scales, while MADRID (Lu et al., 2023) addresses the all-discords problem by searching for anomalies of all lengths with a hyper-anytime computation strategy. These methods highlight the importance of moving beyond a single manually selected window length. Motivated by this direction, we aim to develop an anomaly detector that can capture deviations across multiple temporal resolutions while remaining lightweight and efficient.

3 Method

Building on subsequence PCA, TAP redesigns the representation, covariance construction, and score aggregation to sliding-window time series. As illustrated in Figure 2, Multi-Resolution PCA detects deviations across multiple window sizes and temporal scales and uses cross-resolution evidence consensus to identify evidence supported across neighboring resolutions. Shape-Residual PCA instead scores residuals obtained after removing a recurrent normal-shape prototype. We present the univariate formulation below and its channel-wise multivariate extension in Appendix A.2.

3.1 Multi-Resolution PCA (M-R PCA)

The first branch formulates PCA anomaly scoring over a multi-resolution subsequence family. For each window size w and scale level m, we construct a window representation, compute a PCA anomaly score, and map the window-level scores back to timestamps. Combining short and long windows with fine and coarse resolutions allows the detector to capture both local fluctuations and broader abnormal patterns. The resulting score sequences are subsequently organized into a window–scale graph, where neighboring nodes describe closely related views of the same subsequence.

However, a direct implementation is expensive. Running PCA separately for every (w, m) pair requires repeatedly constructing window matrices and computing empirical covariance matrices via $(\mathbf{Z}_w^{(m)})^\top \mathbf{Z}_w^{(m)}$. Since TAP merges scores from multiple resolutions, this naive strategy would make

PCA scoring unnecessarily slow. We therefore design a Toeplitz-nested covariance construction that turns these PCA problems into coupled submatrix-extraction and covariance-compression operations.

Toeplitz covariance from stride-one windowing. For a univariate time series $\mathbf{x} = [x_1, \ldots, x_T]^\top$, the stride-one window matrix of length w is:

$$\mathbf{Z}_w(i,:) = [x_i, x_{i+1}, \ldots, x_{i+w-1}], \qquad i = 1, \ldots, T-w+1.$$

The key observation is that the columns of $\mathbf{Z}_w$ are shifted copies of the same time series. Thus, the covariance between the i-th and j-th window positions depends mainly on their lag $|i-j|$. Let $\hat{\gamma}(\ell)$ denote the empirical autocovariance at lag ℓ. We approximate the window covariance by:

$$(\mathbf{\Sigma}_w)_{ij} = \hat{\gamma}(|i-j|), \qquad 1 \le i, j \le w,$$

or equivalently $\mathbf{\Sigma}_w = \mathrm{Toep}(\hat{\gamma}(0), \hat{\gamma}(1), \ldots, \hat{\gamma}(w-1))$. This avoids recomputing $\mathbf{Z}_w^\top \mathbf{Z}_w$ for every window size: once the autocovariance sequence is computed, the covariance matrix for any w can be obtained by selecting the corresponding leading Toeplitz submatrix.

Nested multi-resolution covariance compression. To capture anomalies at coarser resolutions, we compress each window by averaging adjacent entries. For a window $\mathbf{W}_i$, one-step compression is $\mathbf{W}_i^{(1)} = \mathbf{A}_1 \mathbf{W}_i$, where $\mathbf{A}_1$ averages every adjacent pair. The corresponding covariance is: $\mathbf{\Sigma}^{(1)} = \mathbf{A}_1 \mathbf{\Sigma}^{(0)} \mathbf{A}_1^\top$, so each entry of the compressed covariance is the average of a 2×2 block:

$$\mathbf{\Sigma}_{ij}^{(1)} = \frac{1}{4}\left(\mathbf{\Sigma}_{2i-1,2j-1}^{(0)} + \mathbf{\Sigma}_{2i-1,2j}^{(0)} + \mathbf{\Sigma}_{2i,2j-1}^{(0)} + \mathbf{\Sigma}_{2i,2j}^{(0)}\right).$$

Repeating this operation gives covariance matrices for coarser resolutions without reconstructing compressed window matrices or recomputing covariance from raw data.

In practice, we choose window sizes and scale factors as powers of two. As shown in Figure 2(b), all covariance matrices can then be derived from a single fine-scale covariance matrix, e.g., a 512×512 matrix. Smaller-window covariances are obtained by leading submatrices, and coarser-scale covariances by repeated 2×2 block averaging. Therefore, instead of performing many large matrix multiplications for different window-scale pairs, the proposed branch obtains all remaining PCA covariances through submatrix extraction and nested multi-resolution covariance compression.

Using multiple window sizes and scales broadens the range of detectable patterns, but also increases the chance that noise or incidental variation produces an isolated high response at a particular resolution. Carrying such responses directly into the final score can introduce false positives and reduce detection quality. We therefore introduce cross-resolution evidence consensus (CREC), which uses the structural relationships among resolution-specific scores to identify evidence supported across neighboring window and scale configurations.

Cross-resolution evidence consensus. Each window–scale pair defines a node $v = (w, m)$ in a resolution graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Edges connect nodes that differ by one resolution operation: $(2w, m)$ with (w, m) along the window axis, and $(w, m+1)$ with (w, m) along the scale axis. The graph thereby encodes the construction relationships among the multi-resolution representations.

Let A_t^v denote the timestamp-level PCA score produced by node v. We first standardize each node:

$$Z_t^v = \frac{A_t^v - \mu_v}{\sigma_v + \epsilon},$$

where μ_v and σ_v are the temporal mean and standard deviation of A^v. Because the score distributions can still differ in shape across resolutions, we transform each score sequence into within-node empirical percentiles that quantify cross-resolution agreement. Specifically,

$$Q_t^v = \frac{\mathrm{rank}_{\tau \in \{1,\ldots,T\}}(Z_t^v) - \frac{1}{2}}{T} \in (0,1),$$

where the rank is computed among all timestamps of the same node. Thus, Q_t^v indicates how unusual timestamp t is relative to the typical behavior observed at resolution v. The percentile scores are used to compare support across graph nodes, while the standardized scores Z_t^v retain the evidence magnitude used in the subsequent aggregation.

We construct a collection $\mathcal{P}$ of local paths that follow the edges of $\mathcal{G}$ across neighboring window and scale configurations, with each path containing ℓ nodes. At timestamp t, the consensus strength of a path is determined by its weakest node:

$$C_t(p) = \min_{v \in p} Q_t^v, \qquad p_t^* = \arg\max_{p \in \mathcal{P}} C_t(p).$$

The maximum-bottleneck path p_t^* identifies the window–scale neighborhood whose evidence is most consistently supported at timestamp t. We retain this path as a distinct consensus backbone, while pooling all off-path scores into a single complementary summary. This prevents the consensus evidence from being diluted by numerous resolution-specific responses, while preserving sensitivity to subtle anomalies detected at only one resolution. CREC thereby limits the influence of isolated resolution-specific responses while retaining potentially useful evidence outside the selected path.

3.2 SHAPE-RESIDUAL PCA (S-R PCA)

The second branch targets shape-level deviations that are complementary to M-R PCA. The Toeplitz covariance structure induced by stride-one windowing is closely related to circulant structure, whose eigenvectors are Fourier modes (Bamieh, 2018; Gray, 2006). Thus, the PCA directions often behave similarly to frequency-ordered patterns: dominant components mainly capture smooth and low-frequency variation, while low-variance components tend to emphasize high-frequency residuals (see Appendix B). As a result, M-R PCA is effective for spike-like or noise-like anomalies, which often create pronounced high-frequency residuals, but it may be less sensitive when the abnormality primarily changes the overall waveform shape while preserving a similar local noise level.

To address this limitation, we use a single-prototype residual construction. Rather than clustering windows into multiple shape groups, we assume that a training time series is dominated by one recurrent normal state. Under this assumption, a single representative normal-shape prototype is sufficient to describe the regular periodic pattern, and deviations from this prototype provide a direct signal for shape anomalies. We estimate this prototype using normalized cross-correlation-based phase alignment, a criterion commonly used for shape-based time-series comparison (Paparrizos & Gravano, 2015). While shape-based distances are commonly used to compare or cluster entire time series, our setting is different: we apply shape comparison to densely overlapping sliding windows. Therefore, we first introduce the standard shape-based distance criterion, and then describe how we incorporate it to our single-prototype residual construction for window-level anomaly detection.

Given a sliding window $\mathbf{W}_i$, we first remove its within-window mean, $\breve{\mathbf{W}}_i = \mathbf{W}_i - \bar{W}_i \mathbf{1}$. We only demean each window instead of standardizing it, because amplitude variation may itself be anomalous and should not be normalized away. A global prototype $\mathbf{c}$ is estimated by repeatedly aligning the demeaned windows to the current prototype using normalized cross-correlation and averaging the aligned windows. Let $\mathcal{S}_\tau(\cdot)$ denote a circular shift operator. For each window, the best phase shift is:

$$\tau_i^* = \arg\max_{\tau} \frac{\left\langle \mathcal{S}_\tau(\breve{\mathbf{W}}_i), \mathbf{c} \right\rangle}{\|\breve{\mathbf{W}}_i\|_2 \|\mathbf{c}\|_2 + \epsilon}.$$

After the prototype is learned, we shift it back to the original phase of each window and subtract it from the demeaned window: $\mathbf{r}_i = \breve{\mathbf{W}}_i - \mathcal{S}_{-\tau_i^*}(\mathbf{c})$. The resulting residual $\mathbf{r}_i$ removes the shared normal shape while retaining local deviations.

We score deviations in this residual-window space rather than in the aligned raw-window space. This design is important for stride-one sliding windows. Because neighboring windows overlap heavily, phase alignment can make many adjacent windows nearly identical after shifting. If PCA is applied directly to aligned raw windows, the principal directions may be dominated by duplicated subsequences, alignment artifacts, or the variance of the common periodic waveform itself, rather than by abnormal deviations. In contrast, subtracting the phase-adjusted prototype first removes the dominant normal waveform and leaves a residual representation anchored in the original window phase. Thus, the PCA score is driven by the variation that remains after normal-shape removal, making the score more sensitive to structured shape deviations.

Finally, we form the residual matrix $\mathbf{R} = [\mathbf{r}_1, \ldots, \mathbf{r}_N]^\top$ and compute an S-R PCA score A_i^{shape} using the residual-space deviation score. S-R PCA therefore complements M-R PCA: M-R PCA uses multi-resolution analysis to capture noise fluctuations at different temporal durations and spike-like anomalies, while S-R PCA captures deviations from the dominant normal shape.

Efficient batched alignment. The prototype alignment step requires computing the cross-correlation between each demeaned window and the current prototype over all circular shifts. Let $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ denote the discrete Fourier transform (DFT) and its inverse. In practice, the DFT is computed by the fast Fourier transform (FFT), which takes $O(w \log w)$ time for a length-w sequence. For a demeaned window $\breve{\mathbf{W}}_i$ and prototype $\mathbf{c}^{(q)}$, the circular cross-correlation over all shifts is computed by the convolution theorem as $\mathbf{g}_i^{(q)} = \mathcal{F}^{-1}\left(\mathcal{F}(\breve{\mathbf{W}}_i) \odot \overline{\mathcal{F}(\mathbf{c}^{(q)})}\right)$, where $\odot$ denotes element-wise multiplication and the overline denotes complex conjugation. The best shift is then selected by:

$$\tau_i^{(q)} = \arg\max_{\tau} \frac{g_{i,\tau}^{(q)}}{\|\breve{\mathbf{W}}_i\|_2 \|\mathbf{c}^{(q)}\|_2 + \epsilon}.$$

For all sliding windows, we stack the demeaned windows into $\breve{\mathbf{X}} \in \mathbb{R}^{N \times w}$ and compute the row-wise DFT once, $\widehat{\mathbf{X}} = \mathcal{F}_{\text{row}}(\breve{\mathbf{X}})$. At iteration q, only the prototype transform $\widehat{\mathbf{c}}^{(q)} = \mathcal{F}(\mathbf{c}^{(q)})$ is updated. All window-to-prototype cross-correlations are then obtained simultaneously by:

$$\mathbf{G}^{(q)} = \mathcal{F}_{\text{row}}^{-1}\left(\widehat{\mathbf{X}} \odot \overline{\widehat{\mathbf{c}}^{(q)}}^{\top}\right),$$

and the shifts are selected by row-wise maximization. Thus, the alignment step is implemented as one batched inverse FFT and one row-wise maximum per iteration, avoiding explicit shift enumeration for every window and making S-R PCA practical for dense stride-one windows. The formulation above uses a single recurrent prototype, but it can be extended to non-stationary series by learning separate prototypes for automatically identified contiguous regimes; see Appendix A.3.

3.3 Score Fusion

M-R PCA and S-R PCA capture complementary anomaly evidence from the multi-resolution subsequence and shape-residual representations, respectively. We independently standardize their timestamp-level scores and fuse them using normalized LogSumExp: $A_t^{\text{final}} = \log\left[\frac{1}{2}\left(\exp\left(\widetilde{A}_t^{\text{MR}}\right) + \exp\left(\widetilde{A}_t^{\text{shape}}\right)\right)\right]$. This fusion preserves pronounced branch-specific responses while allowing evidence from both representations to contribute to the final anomaly score.

4 Experiments

In this section, we first describe the experimental setup, and then present the experimental results.

4.1 Experimental setup

Datasets. We conduct our primary experiments on TSB-AD (Liu & Paparrizos, 2024), a large-scale and heterogeneous benchmark spanning web services, medical signals, facility monitoring, sensors, human activity, environmental data, finance, and traffic. The main evaluation focuses on its univariate collections and follows the official data split and evaluation protocol. To broaden the evaluation, the Appendix reports multivariate results on TSB-AD-M and additional univariate and multivariate results on the independent TAB benchmark (Qiu et al., 2025)(Appendix C.2 and C.3).

Baselines. We compare TAP with a broad set of strong baselines, including top-performing methods from existing benchmarks and representative recent algorithms from three major categories. The first group includes classical statistical and traditional machine learning methods, including Matrix Profile (Yeh et al., 2016), KMeansAD (Yairi et al., 2001), Sub-KNN (Ramaswamy et al., 2000), NormA (Boniol et al., 2021a), POLY (Li et al., 2007), SR (Ren et al., 2019), and Sub-PCA (Aggarwal, 2016). The second group includes deep learning methods based on forecasting or reconstruction errors, including KAN-AD (Zhou et al., 2024), PaAno (Park & Kang, 2026), CARLA (Darban et al., 2025), TFAD (Zhang et al., 2022), CNN (Munir et al., 2018), LSTM-AD (Malhotra et al., 2015), DADA (Shentu et al., 2024), and USAD (Audibert et al., 2020). The third group includes recent foundation-model-based or large time-series models, including MOMENT (Goswami et al., 2024), TimesFM (Das et al., 2023), TSPulse (Ekambaram et al., 2025) and Time-RCD (Lan et al., 2025). For subsequence-based baselines, we evaluate both normalized and unnormalized subsequence representations and report the better result for each method. Unless otherwise stated, we use the official implementations and recommended hyperparameters from the benchmark, and follow the PaAno experimental setup for baselines implemented in that study.

Evaluation metrics. Following recent TSAD benchmarks (Liu & Paparrizos, 2024), we report both range-wise and point-wise metrics. Range-wise metrics include Volume Under the Surface Precision-Recall (VUS-PR), Volume Under the Surface Receiver Operating Characteristic (VUS-ROC), and

Table 1: Overall results on TSB-AD-U. Best results are shown in bold and second best are underlined.

	Method	Range-Wise Measure ↑			Point-Wise Measure ↑		
		VUS-PR	VUS-ROC	Range-F1	AUC-PR	AUC-ROC	F1
Stat&ML	SR 2019	0.32	0.81	0.35	0.32	0.74	0.38
	POLY 2007	0.39	0.76	0.35	0.31	0.73	0.37
	NormA 2021a	0.49	0.83	0.49	0.46	0.81	0.49
	Sub-PCA 2016	0.54	0.84	0.49	0.47	0.81	0.52
	Sub-KNN 2000	0.54	0.86	0.47	0.44	0.83	0.51
	Matrix Profile 2016	0.55	0.87	0.47	0.45	0.84	0.50
	KMeansAD 2001	<u>0.56</u>	0.86	0.52	<u>0.50</u>	0.83	<u>0.54</u>
NN	CARLA 2025	0.29	0.71	0.26	0.24	0.68	0.30
	DADA 2024	0.31	0.77	0.31	0.29	0.71	0.38
	LSTM-AD 2015	0.33	0.76	0.34	0.31	0.68	0.37
	CNN 2018	0.34	0.79	0.35	0.33	0.71	0.38
	TFAD 2022	0.35	0.74	0.33	0.25	0.67	0.31
	USAD 2020	0.36	0.71	0.40	0.32	0.66	0.37
	KAN-AD 2024	0.43	0.82	0.43	0.41	0.80	0.44
	PaAno 2026	0.52	<u>0.89</u>	0.48	0.46	<u>0.86</u>	0.51
FM	TimesFM 2023	0.30	0.74	0.34	0.28	0.67	0.34
	MOMENT 2024	0.39	0.76	0.35	0.30	0.69	0.35
	Time-RCD 2025	0.52	0.87	<u>0.52</u>	0.50	0.84	0.53
	TSPulse 2025	0.53	0.89	0.46	0.41	0.85	0.49
	TAP_S-R	0.60	0.87	0.54	0.53	0.84	0.57
	TAP_M-R	0.64	0.92	0.56	0.57	0.89	0.61
	TAP	**0.66**	**0.92**	**0.57**	**0.58**	**0.90**	**0.62**

range-wise F1. Point-wise metrics include AUC-PR, AUC-ROC, and F1. We use VUS-PR (Boniol et al., 2025) as the primary metric because it is threshold-independent, range-aware, and robust to temporal misalignment. Runtime is also reported to assess deployment efficiency.

Implementation details. We follow the official preprocessing and evaluation protocol. Implementation and parameter details are provided in Appendix A.1 and A.4, while the multivariate extension and its results are presented in Appendices A.2 and C.2.

4.2 Overall Performance

Table 1 reports the overall performance of all methods. TAP achieves the best or tied-best result across all six metrics, demonstrating consistent performance under both range-wise and point-wise evaluation. On the primary VUS-PR metric, TAP reaches 0.66, outperforming the strongest baseline, KMeansAD, by 17.9%. These improvements show that the proposed PCA framework remains effective across different evaluation protocols. The branch-level results further illustrate the contributions of the two representations. M-R and S-R PCA achieve VUS-PR scores of 0.64 and 0.60, respectively, both outperforming the strongest baseline, while their fusion reaches 0.66 and matches or improves upon both branches across all metrics. This indicates that multi-resolution subsequence evidence and shape-residual evidence capture complementary anomaly characteristics. Beyond these aggregate gains, Friedman and Nemenyi tests across all 350 datasets confirm that TAP significantly outperforms the five strongest baselines, as detailed in Appendix C.1.

4.3 Ablation Study

To assess the contribution of each component, we conduct an ablation study over the multi-resolution subsequence branch, the shape-residual branch, and their final fusion. Table 2 reports VUS-PR for each variant. Vanilla PCA with a fixed window size and scale achieves 0.521. Removing multi-scale scoring, and therefore retaining only multiple window sizes, yields 0.565, while removing multi-window scoring and retaining only multiple scales yields 0.568. Incorporating both

Table 2: Ablation study of the M-R construction, S-R residualization, and score fusion. Vanilla denotes single-window, single-scale PCA, and CREC denotes cross-resolution evidence consensus. w/o CREC replaces CREC with mean aggregation over the anomaly scores.

	Multi-Resolution					Shape		Fusion	
Metric	Vanilla	w/o MS	w/o MW	w/o CREC	Full	w/o Res.	w/ Res.	Mean	LSE
VUS-PR ↑	0.521	0.565	0.568	0.617	**0.645**	0.527	**0.602**	0.644	**0.659**

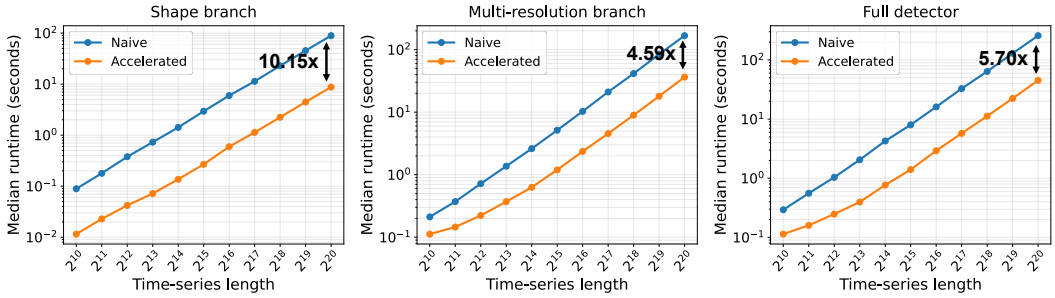


Figure 3: Runtime scalability of the proposed acceleration strategy for time-series lengths from 2^{10} to 2^{20}. The accelerated implementation consistently reduces runtime for both branches.

resolution axes improves the score to 0.617 without CREC. The complete M-R branch further increases VUS-PR to 0.645, demonstrating that cross-resolution consensus helps identify consistently supported anomaly evidence and reduces isolated responses from individual resolutions.

For the shape branch, PCA on aligned windows gives 0.527, whereas residual-space scoring reaches 0.602. Removing the recurrent normal-shape prototype suppresses shared variation and focuses PCA on abnormal deviations. Finally, mean fusion achieves a VUS-PR of 0.644, while LogSumExp reaches 0.659, showing that emphasizing strong evidence while retaining both branches improves fusion. Qualitative analyses of CREC and branch complementarity, together with CREC path-length sensitivity, are provided in Appendices C.5 and C.6, respectively.

4.4 Scalability

All runtime experiments are conducted on the same Ubuntu 22.04.3 server with two AMD EPYC 7713 CPUs and two NVIDIA A100 GPUs. TAP and all baselines are implemented in Python 3.11 with PyTorch 2.3.0 (Paszke et al., 2019) and scikit-learn 1.3.2 (Pedregosa et al., 2011). For fair CPU comparison, all methods are restricted to one CPU core. Reported runtimes exclude data loading and include model fitting, inference, and anomaly-score generation.

To verify the scalability and computational benefit of the proposed acceleration strategy, we compare the accelerated implementation with the original window-first implementation. As shown in Figure 3, acceleration consistently reduces runtime as the time-series length increases, with especially clear gains on long sequences. At length 2^{20}, the accelerated implementation is $10.15\times$ faster for the S-R PCA branch, $4.59\times$ faster for the M-R PCA branch, and $5.70\times$ faster for the full detector. These results show that the Toeplitz-nested covariance construction and batched shape alignment substantially reduce the cost of constructing multi-resolution and residual-space PCA scores.

Second, we compare the average runtime of TAP with top-performing baselines on the benchmark. Table 3 reports the average runtime per time series. Despite computing multiple PCA scores and combining two complementary branches, TAP remains efficient, processing each benchmark time series in 2.5 seconds on average. This makes TAP substantially more scalable than most subsequence-based baselines while maintaining the best overall detection accuracy. Experiments in Appendix C.4 further compare subsequence-based methods under different time-series lengths.

Table 3: Average runtime per time series on TSB-AD-U.

Method	TSPulse	KAN-AD	PaAno	Matrix Profile	KMeans	Sub-KNN	NormA	Sub-PCA	**TAP**
Runtime (s)	431.37	11.70	6.60	28.10	25.71	91.32	5.83	**1.26**	<u>2.52</u>

5 Conclusion

In this work, we introduced TAP, a Time-series Adaptive PCA framework for efficient time-series anomaly detection. Instead of increasing model complexity, TAP turns classical projection-based detection into a time-series-adaptive framework by jointly redesigning subsequence representation, residual modeling, and covariance computation. The multi-resolution PCA branch captures anomalies across different durations and temporal resolutions, while the shape-residual PCA branch provides complementary sensitivity to local shape and pattern changes. The ested covariance construction and batched alignment implementation further make the framework scalable to long time series. Experiments on TSB-AD-U show that TAP achieves the best overall performance across range-wise and point-wise metrics, while maintaining low average runtime. These results suggest that carefully designed classical methods can remain highly competitive for time-series anomaly detection when their preprocessing, representation, and computation are aligned with temporal structure.

AI USE STATEMENT

In this work, generative AI tools were used to assist with writing portions of the software implementation and to improve the grammar, clarity, and readability of the manuscript. They were not used to develop the research methodology, formulate scientific claims, or interpret experimental results. All AI-assisted code was reviewed and tested by the authors, and all AI-assisted writing was carefully reviewed. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The proposed method and experimental protocol are described in Sections 3 and 4. Pseudocode and implementation details are provided in Appendix A, the multivariate extension is described in Appendix A.2, and parameter settings and additional experimental results are reported in Appendix C. An anonymized implementation is available at `https://anonymous.4open.science/r/TAP-TSAD`.

REFERENCES

Charu C Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pp. 1–34. Springer, 2016.

Abdel-Rahman Al-Ghuwairi, Yousef Sharrab, Dimah Al-Fraihat, Majed AlElaimat, Ayoub Alsarhan, and Abdulmohsen Algarni. Intrusion detection in cloud computing based on time series anomalies utilizing machine learning. *Journal of Cloud Computing*, 12(1):127, 2023.

Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3395–3404, 2020.

Bassam Bamieh. Discovering transforms: A tutorial on circulant matrices, circular convolution, and the discrete fourier transform. *arXiv preprint arXiv:1805.05533*, 2018.

Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *ACM computing surveys (CSUR)*, 54(3):1–33, 2021.

Paul Boniol, Michele Linardi, Federico Roncallo, Themis Palpanas, Mohammed Meftah, and Emmanuel Remy. Unsupervised and scalable subsequence anomaly detection in large data series. *The VLDB Journal*, 30(6):909–931, 2021a.

Paul Boniol, John Paparrizos, Themis Palpanas, and Michael J Franklin. Sand: Streaming subsequence anomaly detection. *Proc. VLDB Endow.*, 14(10):1717–1729, 2021b.

Paul Boniol, Qinghua Liu, Mingyi Huang, Themis Palpanas, and John Paparrizos. Dive into time-series anomaly detection: A decade review. *arXiv preprint arXiv:2412.20512*, 2024.

Paul Boniol, Ashwin K Krishna, Marine Bruel, Qinghua Liu, Mingyi Huang, Themis Palpanas, Ruey S Tsay, Aaron Elmore, Michael J Franklin, and John Paparrizos. Vus: effective and efficient accuracy measures for time-series anomaly detection. *The VLDB Journal*, 34(3):32, 2025.

S. Chauhan and L. Vig. Anomaly detection in ECG time signals via deep long short-term memory networks. In *Proceedings of the International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–7, 2015. doi: 10.1109/DSAA.2015.7344872.

Mooi Choo Chuah and Fen Fu. Ecg anomaly detection via time series analysis. In *International Symposium on Parallel and Distributed Processing and Applications*, pp. 123–135. Springer, 2007.

Zahra Zamanzadeh Darban, Geoffrey I Webb, Shirui Pan, Charu C Aggarwal, and Mahsa Salehi. Carla: Self-supervised contrastive representation learning for time series anomaly detection. *Pattern Recognition*, 157:110874, 2025.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

Jens E d'Hondt, Haojun Li, Fan Yang, Odysseas Papapetrou, and John Paparrizos. A structured study of multivariate time-series distance measures. *Proceedings of the ACM on Management of Data*, 3 (3):1–29, 2025.

Vijay Ekambaram, Subodh Kumar, Arindam Jati, Sumanta Mukherjee, Tomoya Sakai, Pankaj Dayama, Wesley M Gifford, and Jayant Kalagnanam. Tspulse: Dual space tiny pre-trained models for rapid time-series analysis. *arXiv preprint arXiv:2505.13033*, 2025.

Determinant EStimator. A fast algorithm for the minimum covariance. *Technometrics*, 41(3):212, 1999.

Kamil Faber, Marcin Pietron, Dominik Zurek, and Roberto Corizzo. xlstmad: A powerful xlstm-based method for anomaly detection. In *2025 IEEE International Conference on Data Mining (ICDM)*, pp. 247–256. IEEE, 2025.

Eric D Feigelson, G Jogesh Babu, and Gabriel A Caceres. Autoregressive times series methods for time domain astronomy. *Frontiers in Physics*, 6:80, 2018.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Robert M Gray. Toeplitz and circulant matrices: A review. 2006.

Medina Hadjem, Farid Naït-Abdesselam, and Ashfaq A. Khokhar. St-segment and t-wave anomalies prediction in an ECG data using rusboost. In *Healthcom*, 2016.

Dan He, Jiwon Kim, Hua Shi, and Boyu Ruan. Autonomous anomaly detection on traffic flow time series with reinforcement learning. *Transportation Research Part C: Emerging Technologies*, 150: 104089, 2023.

Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.

Chengqiang Huang, Geyong Min, Yulei Wu, Yiming Ying, Ke Pei, and Zuochang Xiang. Time series anomaly detection for trustworthy services in cloud computing systems. *IEEE Transactions on Big Data*, 8(1):60–72, 2017.

Mingyi Huang, Qinghua Liu, Paul Boniol, and John Paparrizos. Hydra: A multi-level hierarchy-driven approach for robust anomaly detection in time series. *Proceedings of the ACM on Management of Data*, 4(3 (SIGMOD):1–30, 2026.

Makoto Imamura. Generalized discords for time series anomaly detection with flexible subsequence lengths. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 1013–1024, 2025.

Nataliia Kashpruk, Cezary Piskor-Ignatowicz, and Jerzy Baranowski. Time series prediction in industry 4.0: a comprehensive review and prospects for future advancements. *Applied sciences*, 13 (22):12374, 2023.

HyunGi Kim, Jisoo Mok, Hyungyu Lee, Juhyeon Shin, and Sungroh Yoon. Candi: Curated test-time adaptation for multivariate time-series anomaly detection under distribution shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 15018–15026, 2026.

Tian Lan, Hao Duong Le, Jinbo Li, Wenjun He, Meng Wang, Chenghao Liu, and Chen Zhang. Towards foundation models for zero-shot time series anomaly detection: Leveraging synthetic data and relative context discrepancy. *arXiv preprint arXiv:2509.21190*, 2025.

Haojun Li and John Paparrizos. Mufasa: Fast and accurate multivariate time-series clustering. *Proceedings of the ACM on Management of Data*, 4(3 (SIGMOD):1–29, 2026.

Jinbo Li, Hesam Izakian, Witold Pedrycz, and Iqbal Jamal. Clustering-based anomaly detection in multivariate time series data. *Applied Soft Computing*, 100:106919, 2021.

Zhi Li, Hong Ma, and Yongbing Mei. A unifying method for outlier and change detection from data streams based on local polynomial fitting. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 150–161. Springer, 2007.

Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 6555–6565, 2024.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 37:108231–108261, 2024.

Qinghua Liu, Paul Boniol, Themis Palpanas, and John Paparrizos. Time-series anomaly detection: Overview and new trends. *Proceedings of the VLDB Endowment (PVLDB)*, 17(12):4229–4232, 2024.

Yue Lu, Thirumalai Vinjamoor Akhil Srinivas, Takaaki Nakamura, Makoto Imamura, and Eamonn Keogh. Matrix profile xxx: Madrid: A hyper-anytime and parameter-free algorithm to find time series anomalies of all lengths. In *2023 ieee international conference on data mining (icdm)*, pp. 1199–1204. IEEE, 2023.

Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, Puneet Agarwal, et al. Long short term memory networks for anomaly detection in time series. In *Proceedings*, volume 89, pp. 94, 2015.

Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *IEEE access*, 7:1991–2005, 2018.

Hussain Nizam, Samra Zafar, Zefeng Lv, Fan Wang, and Xiaopeng Hu. Real-time deep anomaly detection framework for multivariate time-series data in industrial iot. *IEEE Sensors Journal*, 22 (23):22836–22849, 2022.

Anastasios Papadopoulos, Apostolos Giannoulidis, Anastasios Gounaris, and John Paparrizos. The power of anomaly detection in predictive maintenance:[experiments & analysis]. *Proceedings of the ACM on Management of Data*, 4(3 (SIGMOD):1–33, 2026.

John Paparrizos and Luis Gravano. k-shape: Efficient and accurate clustering of time series. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pp. 1855–1870, 2015.

John Paparrizos, Paul Boniol, Qinghua Liu, and Themis Palpanas. Advances in time-series anomaly detection: Algorithms, benchmarks, and evaluation measures. In *Proceedings of the 31st ACM SIGKDD conference on knowledge discovery and data mining v. 2*, pp. 6151–6161, 2025.

Jinju Park and Seokho Kang. Paano: Patch-based representation learning for time-series anomaly detection. *arXiv preprint arXiv:2602.01359*, 2026.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen, and Bin Yang. Tab: Unified benchmarking of time series anomaly detection methods, 2025. URL https://arxiv.org/abs/2506.18046.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pp. 427–438, 2000.

Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3009–3017, 2019.

Peter Rousseeuw, Domenico Perrotta, Marco Riani, and Mia Hubert. Robust monitoring of time series with application to fraud detection. *Econometrics and statistics*, 9:108–121, 2019.

Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis*, pp. 4–11, 2014.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

Gideon Schwarz. Estimating the dimension of a model. *The annals of statistics*, pp. 461–464, 1978.

Qichao Shentu, Beibu Li, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Towards a general time series anomaly detector with adaptive bottlenecks and dual adversarial decoders. *arXiv preprint arXiv:2405.15273*, 2024.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2828–2837, 2019.

Franco Van Wyk, Yiyang Wang, Anahita Khojandi, and Neda Masoud. Real-time sensor anomaly detection and identification in automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 21(3):1264–1276, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*, 2024.

Takehisa Yairi, Yoshikiyo Kato, and Koichi Hori. Fault detection by mining association rules from house-keeping data. In *proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, volume 18, pp. 21. Citeseer, 2001.

Fan Yang and John Paparrizos. Spartan: Data-adaptive symbolic time-series approximation. *Proceedings of the ACM on Management of Data*, 3(3):1–30, 2025.

Yi-Ching Yao. Estimating the number of change-points via schwarz'criterion. *Statistics & Probability Letters*, 6(3):181–189, 1988.

Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)*, pp. 1317–1322. Ieee, 2016.

Chin-Chia Michael Yeh, Audrey Der, Uday Singh Saini, Vivian Lai, Yan Zheng, Junpeng Wang, Xin Dai, Zhongfang Zhuang, Yujie Fan, Huiyuan Chen, et al. Matrix profile for anomaly detection on multidimensional time series. In *2024 IEEE International Conference on Data Mining (ICDM)*, pp. 911–916. IEEE, 2024.

Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 8980–8987, 2022.

Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. Tfad: A decomposition time series anomaly detection architecture with time-frequency analysis. In *Proceedings of the 31st ACM international conference on information & knowledge management*, pp. 2497–2507, 2022.

Quan Zhou, Changhua Pei, Fei Sun, Jing Han, Zhengwei Gao, Dan Pei, Haiming Zhang, Gaogang Xie, and Jianhui Li. Kan-ad: Time series anomaly detection with kolmogorov-arnold networks. *arXiv preprint arXiv:2411.00278*, 2024.

Xiaokun Zhu, Louis Carpentier, and Mathias Verbeke. When foundation models are one-liners: Limitations and future directions for time series anomaly detection. In *The Fourteenth International Conference on Learning Representations*, 2026.

A Algorithm and Implementation Details

A.1 Pseudocode of TAP

We provide pseudocode for the core algorithms of TAP, including the overall framework, the multi-resolution PCA branch, and the shape-residual PCA branch.

Algorithm 1 TAP: Time-series Adaptive PCA

Require: Univariate time series $\mathbf{x} \in \mathbb{R}^T$; window-size set $\mathcal{W}$; scale set $\mathcal{M}$; shape period p; CREC path length ℓ
Ensure: Final anomaly score $\mathbf{A}^{\text{final}} \in \mathbb{R}^T$
1: Compute the cross-resolution multi-resolution score $\mathbf{A}^{\text{MR}}$ using Algorithm 2 with $(\mathbf{x}, \mathcal{W}, \mathcal{M}, \ell)$
2: Compute the shape-residual score $\mathbf{A}^{\text{SR}}$ using Algorithm 3 with $(\mathbf{x}, p)$
3: Normalize the two branch-level scores: $\widetilde{\mathbf{A}}^{\text{MR}} = \text{zscore}(\mathbf{A}^{\text{MR}})$ and $\widetilde{\mathbf{A}}^{\text{SR}} = \text{zscore}(\mathbf{A}^{\text{SR}})$
4: Fuse the two branches using normalized LogSumExp:

$$A_t^{\text{final}} = \log\left(\frac{\exp(\widetilde{A}_t^{\text{MR}}) + \exp(\widetilde{A}_t^{\text{SR}})}{2}\right)$$

5: **return** $\mathbf{A}^{\text{final}}$

Algorithm 2 Multi-resolution PCA branch

Require: Time series $\mathbf{x} \in \mathbb{R}^T$; window sizes $\mathcal{W}$; scale levels $\mathcal{M}$; path length $\ell \in \{2, 3, 4\}$
Ensure: Multi-resolution anomaly score $\mathbf{A}^{\text{MR}} \in \mathbb{R}^T$
Set $w_{\max} = \max \mathcal{W}$ and estimate $\hat{\gamma}(0), \ldots, \hat{\gamma}(w_{\max} - 1)$
Construct $\mathbf{\Sigma}_{w_{\max}} = \text{Toep}(\hat{\gamma}(0), \ldots, \hat{\gamma}(w_{\max} - 1))$
Initialize the resolution graph nodes $\mathcal{V} \leftarrow \varnothing$
for each window size $w \in \mathcal{W}$ **do**
    Extract $\mathbf{\Sigma}_w^{(0)}$ as the leading $w \times w$ submatrix of $\mathbf{\Sigma}_{w_{\max}}$
    Construct the stride-one window matrix $\mathbf{Z}_w^{(0)}$ from $\mathbf{x}$
    for each valid scale level $m \in \mathcal{M}$ **do**
        if $m > 0$ **then**
            $\mathbf{Z}_w^{(m)} \leftarrow \text{AvgPool}(\mathbf{Z}_w^{(m-1)})$
            $\mathbf{\Sigma}_w^{(m)} \leftarrow \text{BlockAvg}(\mathbf{\Sigma}_w^{(m-1)})$
        end if
        Fit PCA using $\mathbf{\Sigma}_w^{(m)}$
        Compute window-level anomaly scores from $\mathbf{Z}_w^{(m)}$
        Map the window-level scores to the timestamp-level sequence $\mathbf{A}^{(w,m)}$
        Standardize $\mathbf{A}^{(w,m)}$ to obtain $\mathbf{Z}^{(w,m)}$
        Add node $v = (w, m)$ with score $\mathbf{Z}^v$ to $\mathcal{V}$
    end for
end for
Construct $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by connecting $(2w, m)$ with (w, m) and $(w, m+1)$ with (w, m)
Convert each $\mathbf{Z}^v$ into its within-node empirical-percentile sequence $\mathbf{Q}^v$
Enumerate valid directed paths $\mathcal{P}_\ell$ containing exactly ℓ adjacent nodes
for each timestamp $t = 1, \ldots, T$ **do**
    $p_t^* \leftarrow \arg\max_{p \in \mathcal{P}_\ell} \min_{v \in p} Q_t^v$
    $A_t^{\text{con}} \leftarrow \text{LSE}(\{Z_t^v : v \in p_t^*\})$
    $A_t^{\text{comp}} \leftarrow \text{LSE}(\{Z_t^v : v \in \mathcal{V} \setminus p_t^*\})$
end for
Standardize $\mathbf{A}^{\text{con}}$ and $\mathbf{A}^{\text{comp}}$
$\mathbf{A}^{\text{MR}} \leftarrow \text{LSE}(\text{zscore}(\mathbf{A}^{\text{con}}), \text{zscore}(\mathbf{A}^{\text{comp}}))$
return $\mathbf{A}^{\text{MR}}$

Algorithm 3 Shape-residual PCA branch

Require: Time series $\mathbf{x} \in \mathbb{R}^T$; window length p; alignment iterations Q
Ensure: Timestamp-level shape score $\mathbf{A}^{\text{shape}} \in \mathbb{R}^T$

Construct stride-one windows $\{\mathbf{W}_i\}_{i=1}^N$ of length p
Demean each window: $\breve{\mathbf{W}}_i = \mathbf{W}_i - \bar{W}_i \mathbf{1}$
Stack demeaned windows into $\breve{\mathbf{X}} \in \mathbb{R}^{N \times p}$
Precompute row-wise FFT of all windows: $\widehat{\mathbf{X}} = \mathcal{F}_{\text{row}}(\breve{\mathbf{X}})$
Initialize prototype $\mathbf{c}^{(0)}$ from a demeaned window or the average of demeaned windows
for $q = 0, \ldots, Q-1$ **do**
    Compute prototype FFT: $\widehat{\mathbf{c}}^{(q)} = \mathcal{F}(\mathbf{c}^{(q)})$
    Compute all circular cross-correlations by batched inverse FFT:

$$\mathbf{G}^{(q)} = \mathcal{F}_{\text{row}}^{-1}\left(\widehat{\mathbf{X}} \odot \overline{\widehat{\mathbf{c}}^{(q)}}^{\top}\right)$$

    Normalize correlations: $G_{i,\tau}^{(q)} \leftarrow G_{i,\tau}^{(q)} / (\|\breve{\mathbf{W}}_i\|_2 \|\mathbf{c}^{(q)}\|_2 + \epsilon)$
    Select shifts by row-wise maximization: $\tau_i^{(q)} = \arg\max_\tau G_{i,\tau}^{(q)}$
    Align all windows using vectorized circular indexing: $\breve{\mathbf{W}}_{i,\text{align}}^{(q)} = \mathcal{S}_{\tau_i^{(q)}}(\breve{\mathbf{W}}_i)$
    Update prototype: $\mathbf{c}^{(q+1)} = \frac{1}{N} \sum_{i=1}^N \breve{\mathbf{W}}_{i,\text{align}}^{(q)}$
    Demean prototype: $\mathbf{c}^{(q+1)} \leftarrow \mathbf{c}^{(q+1)} - \bar{c}^{(q+1)} \mathbf{1}$
end for
Set final prototype $\mathbf{c} = \mathbf{c}^{(Q)}$ and final shifts $\tau_i = \tau_i^{(Q-1)}$
Shift prototype back to each window phase: $\mathbf{c}_i^{\text{orig}} = \mathcal{S}_{-\tau_i}(\mathbf{c})$
Construct residual windows: $\mathbf{r}_i = \breve{\mathbf{W}}_i - \mathbf{c}_i^{\text{orig}}$
Stack residuals into $\mathbf{R} \in \mathbb{R}^{N \times p}$
Fit PCA on $\mathbf{R}$ and compute residual-window anomaly scores
Map residual-window scores back to timestamps by averaging scores over covering windows
return $\mathbf{A}^{\text{shape}}$

A.2 Extension to Multivariate Time Series

Although TAP is primarily designed for univariate time series, it can be directly extended to a multivariate input $\mathbf{X} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(D)}] \in \mathbb{R}^{T \times D}$ through channel-independent processing. Each channel is analyzed using the same window sizes, temporal scales, and PCA scoring procedure as in the univariate formulation.

For the M-R branch, let $A_t^{v,d}$ denote the score produced at timestamp t by resolution node $v = (w, m)$ for channel d. We standardize every node score within its channel as $Z_t^{v,d} = \mathrm{zscore}(\mathbf{A}^{v,d})_t$ and aggregate corresponding nodes across channels:

$$\bar{Z}_t^v = \frac{1}{D} \sum_{d=1}^{D} Z_t^{v,d}.$$

The resulting channel-aggregated node scores $\{\bar{Z}_t^v : v \in \mathcal{V}\}$ are then processed by CREC exactly as described in Section 3.1, including percentile-based path selection and the aggregation of consensus and complementary evidence, to obtain the multivariate M-R score $\mathbf{A}^{\mathrm{MR}}$.

For the S-R branch, the dominant period and recurrent shape prototype are estimated independently for each channel, allowing different variables to exhibit different periodic structures. Let $A_t^{\mathrm{shape},d}$ denote the resulting S-R score for channel d. The multivariate shape score is obtained by averaging the standardized channel-wise scores:

$$A_t^{\mathrm{shape}} = \frac{1}{D} \sum_{d=1}^{D} \mathrm{zscore}(\mathbf{A}^{\mathrm{shape},d})_t.$$

Finally, the multivariate M-R and S-R scores are independently standardized and fused using the same normalized LogSumExp operation as in Section 3.3:

$$A_t^{\mathrm{final}} = \log\left[\frac{1}{2}\left(\exp\big(\widetilde{A}_t^{\mathrm{MR}}\big) + \exp\big(\widetilde{A}_t^{\mathrm{shape}}\big)\right)\right].$$

This extension preserves the univariate scoring procedure within each channel while producing a single timestamp-level anomaly score for the multivariate series. It does not explicitly model cross-channel dependencies, which we leave for future work.

A.3 Extension to Multiple Normal Shapes

The S-R branch in the main method models recurrent normal behavior using a single phase-aligned prototype. Although this design is effective in our experiments, it assumes that the time series is dominated by one normal shape. For non-stationary series containing multiple persistent normal states or regime shifts, a single prototype may leave systematic residuals in otherwise normal regions and consequently assign them high anomaly scores.

We therefore extend S-R to support multiple temporally contiguous normal shapes. The extension first extracts phase-invariant shape and amplitude features from cycle-level windows and partitions the time series into contiguous regimes using an unsupervised change-point criterion. We automatically partition the time series into temporally contiguous regimes and select the number of change points using the Bayesian information criterion (BIC) (Schwarz, 1978; Yao, 1988), which balances the fit obtained from additional regimes against the complexity introduced by unnecessary splits. A separate phase-aligned prototype is then estimated for each detected regime, and each window is residualized using the prototype associated with its temporal segment. The residuals are calibrated across regimes and pooled to fit a shared residual PCA, keeping anomaly scores comparable across the complete series. The batched alignment acceleration introduced for S-R remains applicable because alignment is performed independently within each segment. The original full-series batch is simply divided into several smaller batches, without changing the total number of processed windows.

Determining whether a persistent change represents a new normal state or a long-lasting anomaly is fundamentally ambiguous without supervision. Introducing unnecessary prototypes may therefore absorb anomalous patterns into the normal model. For this reason, we retain the single-shape formulation in the main experiments as a conservative default that does not require estimating the

number of normal states. Nevertheless, the multiple-shape extension improves detection on several non-stationary series containing clear normal regime changes. A case study illustrating the benefit of this extension is provided in Appendix C.7.

A.4 Detailed Experimental Settings

Implementation details. All experiments are implemented in Python using standard numerical libraries, and runtime comparisons use the same hardware and threading configuration (Section 4.4). the multi-resolution PCA branch uses window sizes $\{8, 16, 32, 64, 128, 256, 512\}$ with a maximum scaling level of 4 to capture anomalies at different temporal resolutions. For the shape-residual PCA branch, we estimate the period of time series using the first peak of the autocorrelation function, and use this estimated period as the input window size for prototype alignment and residual construction.

TSB-AD (Liu & Paparrizos, 2024). Following the benchmark protocol, we tune baseline methods using the recommended hyperparameter search spaces provided by the benchmark (Tables 4 and 5).

Table 4: Hyperparameter variations of **univariate** detection algorithms used in our experiments. See hyperparameter definitions from TSB-AD (`https://github.com/TheDatumOrg/TSB-AD`).

	Method	Hyperparameter 1	Hyperparameter 2
Stats	Sub-KNN (Ramaswamy et al., 2000)	periodicity: [1, 2, 3]	n_neighbors: [10, 20, 30, 40, 50]
	KMeansAD (Yairi et al., 2001)	n_clusters: [10, 20, 30, 40]	window_size: [10, 20, 30, 40]
	POLY (Li et al., 2007)	periodicity: [1, 2, 3]	power: [1, 2, 3, 4]
	Matrix Profile (Yeh et al., 2016)	periodicity: [1, 2, 3]	None
	Sub-PCA (Aggarwal, 2016)	periodicity: [1, 2, 3]	n_components: [0.25, 0.5, 0.75, None]
	SR (Ren et al., 2019)	periodicity: [1, 2, 3]	None
	NormA (Boniol et al., 2021a)	periodicity: [1, 2, 3]	clustering: [hierarchical, kshape]
NN	LSTM-AD (Malhotra et al., 2015)	win_size: [50, 100, 150]	lr: [0.0004, 0.0008]
	CNN (Munir et al., 2018)	win_size: [50, 100, 150]	num_channel: [[32, 32, 40], [16, 32, 64]]
	USAD (Audibert et al., 2020)	win_size: [5, 50, 100]	lr: [0.001, 0.0001, 1e-05]
	KAN-AD (Zhou et al., 2024)	win_size: [32, 64, 96]	lr: [0.01, 0.001, 0.0001]
	PaAno (Park & Kang, 2026)	patch_size: [32, 64, 96]	lr: [0.001, 0.0001, 1e-05]
	CARLA (Darban et al., 2025)	win_size: [50, 100, 200]	lr: [0.01, 0.001, 0.0001]
	TFAD (Zhang et al., 2022)	win_size: [50, 100, 200]	stride: [1, 3, 6, 12]
	DADA (Shentu et al., 2024)	batch_size: [32, 64, 96]	None
FM	TimesFM (Das et al., 2023)	win_size: [32, 64, 96]	None
	MOMENT (Goswami et al., 2024)	win_size: [64, 128, 256]	None
	Time-RCD (Lan et al., 2025)	win_size: [15000]	None
	TSPulse (Ekambaram et al., 2025)	win_size: [96]; prediction_mode: time	lr: 1e-4
Ours	TAP	path_length: [2, 3, 4]	None

Table 5: Hyperparameter variations of **multivariate** detection algorithms used in our experiments. See hyperparameter definitions from TSB-AD (`https://github.com/TheDatumOrg/TSB-AD`).

	Method	Hyperparameter 1	Hyperparameter 2
Stat&ML	PCA (Aggarwal, 2016)	n_components: [0.25, 0.5, 0.75, None]	None
	KMeansAD (Yairi et al., 2001)	n_clusters: [10, 20, 30, 40]	window_size: [10, 20, 30, 40]
	CBLOF (He et al., 2003)	n_clusters: [4, 8, 16, 32]	alpha: [0.6, 0.7, 0.8, 0.9]
	MCD (EStimator, 1999)	support_fraction: [0.2, 0.4, 0.6, 0.8, None]	None
	OCSVM (Schölkopf et al., 1999)	kernel: [linear, poly, rbf, sigmoid]	nu: [0.1, 0.3, 0.5, 0.7]
	MMPAD (Yeh et al., 2024)	n_dim: [1, 0.1, 0.3, 0.5, 0.7]	n_neighbor: [1, 5, 10, 15]
NN	AutoEncoder (Sakurada & Yairi, 2014)	win_size: [50, 100, 150]	hidden_neurons: [[64, 32], [32, 16], [128, 64]]
	CNN (Munir et al., 2018)	win_size: [50, 100, 150]	num_channel: [[32, 32, 40], [16, 32, 64]]
	LSTM-AD (Malhotra et al., 2015)	win_size: [50, 100, 150]	lr: [0.0004, 0.0008]
	OmniAnomaly (Su et al., 2019)	win_size: [5, 50, 100]	lr: [0.002, 0.0002]
	USAD (Audibert et al., 2020)	win_size: [5, 50, 100]	lr: [0.001, 0.0001, 1e-05]
	DADA (Shentu et al., 2024)	batch_size: [32, 64, 96]	None
	KAN-AD (Zhou et al., 2024)	win_size: [32, 64, 96]	lr: [0.01, 0.001, 0.0001]
	PaAno (Park & Kang, 2026)	patch_size: [32, 64, 96]	lr: [0.001, 0.0001, 1e-05]
	xLSTMAD (Faber et al., 2025)	window_size: [50, 100, 150]	lr: [0.001]; embedding_dim: [20, 40]
FM	Time-RCD (Lan et al., 2025)	win_size: [15000]	None
Ours	TAP	path_length: [2, 3, 4]	None

TAB (Qiu et al., 2025). Given ongoing community discussion regarding the quality of anomaly annotations and the limited difficulty of some datasets in TSB-AD, we conduct a complementary evaluation on the TAB benchmark to assess the robustness of our findings under a different benchmark design. Because TAB does not provide the complete implementation and hyperparameter configurations required to reproduce all baselines under identical settings, we use the results released

on its official leaderboard[1] for comparison. Specifically, we compare TAP with the 6 highest-ranked baselines on the TAB-U and -M leaderboards and report their dataset-level VUS-PR in Tables 7 and 8.

[1] https://decisionintelligence.github.io/OpenTS/leaderboards/

B Additional Analysis of the PCA Structure

B.1 Toeplitz Covariance and Frequency-like PCA Directions

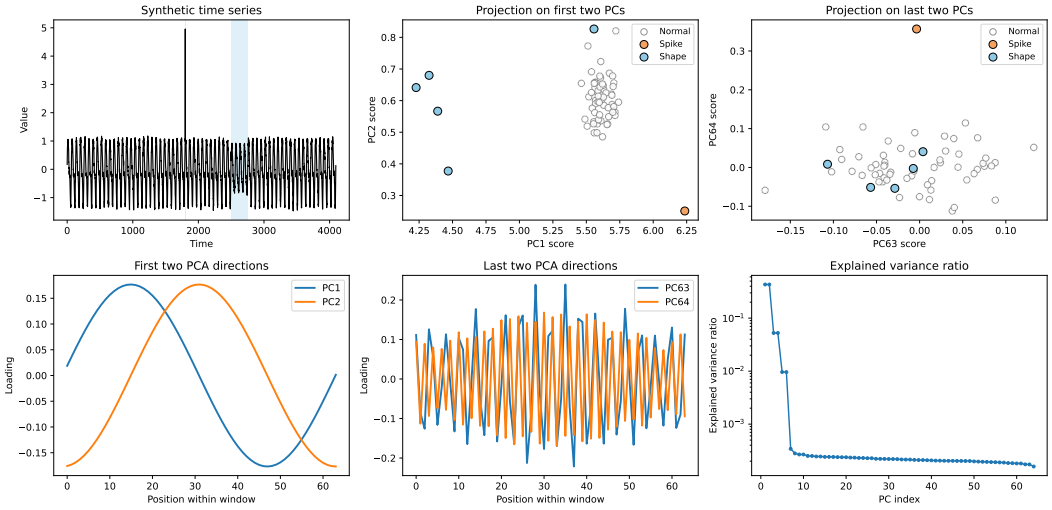


Figure 4: Frequency-like PCA directions induced by Toeplitz covariance from stride-one windows. The leading PCs are smooth low-frequency directions, while the trailing PCs are high-frequency directions. The spike anomaly is isolated in the last-PC space, whereas the shape anomaly is not.

We provide an additional illustration to explain why the multi-resolution PCA branch is particularly effective for spike-like anomalies, but less sensitive to smooth shape changes. For a stride-one window matrix $\mathbf{Z}_w \in \mathbb{R}^{N\times w}$, the i-th column is a shifted copy of the time series, i.e., $\mathbf{Z}_w(:,i) = [x_i,\dots,x_{i+N-1}]^\top$. Therefore, the empirical covariance between the i-th and j-th window positions can be written as

$$(\mathbf{\Sigma}_w)_{ij} = \frac{1}{N-1}\sum_{t=1}^{N}(x_{t+i-1}-\bar{x}_i)(x_{t+j-1}-\bar{x}_j).$$

When the local mean changes slowly, this covariance is mainly determined by the relative lag $|i-j|$. Writing $\hat{\gamma}(\ell)$ as the empirical autocovariance at lag ℓ, we obtain the Toeplitz approximation

$$(\mathbf{\Sigma}_w)_{ij} \approx \hat{\gamma}(|i-j|), \qquad \mathbf{\Sigma}_w \approx \mathrm{Toep}\{\hat{\gamma}(0),\hat{\gamma}(1),\dots,\hat{\gamma}(w-1)\}.$$

This Toeplitz structure provides a frequency-domain interpretation of PCA on stride-one windows. A circulant matrix $\mathbf{C}$ is completely determined by its first row and can be diagonalized by the discrete Fourier transform (DFT) matrix:

$$\mathbf{C} = \mathbf{F}^*\mathbf{\Lambda}\mathbf{F}, \qquad F_{k\ell} = \frac{1}{\sqrt{w}}\exp\left(-2\pi\sqrt{-1}\,k\ell/w\right),$$

where the columns of $\mathbf{F}^*$ are Fourier modes. Toeplitz matrices generated by absolutely summable autocovariance sequences are asymptotically equivalent to suitable circulant matrices (Gray, 2006), and circulant matrices are diagonalized by the DFT (Bamieh, 2018). Thus, although finite Toeplitz matrices are not exactly diagonalized by the DFT, their eigenvectors often behave like frequency-ordered directions: leading PCs correspond to smooth, low-frequency variation, while small-variance PCs correspond to high-frequency residual directions.

This interpretation explains the behavior observed in Figure 4. A localized spike resembles an impulse vector within a window. Since an impulse has non-negligible projection on many Fourier-like directions, including high-frequency ones, it can produce a large PCA anomaly score along small-variance directions:

$$A(\mathbf{z}) = \sum_r \frac{\langle \mathbf{z}-\boldsymbol{\mu}, \mathbf{u}_r\rangle^2}{\lambda_r+\epsilon},$$

where small λ_r amplifies deviations in trailing PCs. Therefore, the spike anomaly is clearly isolated in the last-PC projection. In contrast, a shape anomaly with similar roughness to the normal waveform has most of its energy in low-frequency directions and little energy in the high-frequency residual directions. As a result, it is not well separated by the last PCs. Moreover, if the shape change moves along the dominant smooth subspace, it may not receive a large score even in the first two PCs; in our synthetic example, the shape-anomaly windows obtain lower first-PC-space scores than many normal windows. This highlights a limitation of standard subsequence PCA and motivates the shape-residual PCA branch, which removes the recurrent normal shape before applying PCA. Therefore, the PCA directions of stride-one window covariance often exhibit frequency-like behavior: leading components correspond to smoother low-frequency directions, while small-variance components tend to correspond to high-frequency residual directions.

B.2 Effect of the Accelerated Implementation

We further examine whether the accelerated implementation changes the detection results. Figure 5 compares the VUS-PR obtained by the naive implementation and the accelerated implementation on individual time series. Most points lie close to the diagonal, indicating that the acceleration has little effect on detection accuracy for the majority of cases. Only a small number of time series show noticeable differences.

The main source of this difference comes from the covariance estimation step. In the naive window-based implementation, the covariance between the first and second window positions is computed using the two aligned columns of the window matrix, i.e., using paired samples approximately of the form $(x_1, \ldots, x_{T-w+1})$ and $(x_2, \ldots, x_{T-w+2})$ after window construction. In the accelerated implementation, we estimate the same lag-one covariance directly from the empirical autocovariance of the original time series, using pairs of the form $(x_1, \ldots, x_{T-1})$ and $(x_2, \ldots, x_T)$. Thus, the accelerated version uses a longer lag-based estimate that is shared across all window sizes, while the naive version uses the finite columns induced by each specific window matrix. These two estimates are nearly identical for long and stable sequences, but they can differ slightly near boundaries or in sequences with strong local nonstationarity. This explains why most accuracy values remain unchanged while a few individual cases are affected.

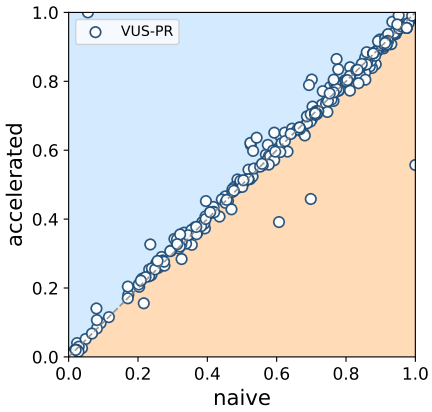


Figure 5: Effect of acceleration on VUS-PR.

Overall, the results suggest that the Toeplitz-based acceleration preserves the detection behavior of the original implementation in most cases, while greatly reducing the computational cost.

C Additional Experimental Results

C.1 Statistical Significance

We compare TAP with the five top-performing baselines using the Friedman test followed by the Nemenyi post-hoc test (Demšar, 2006). Figure 6 reports their average VUS-PR ranks across the 350 datasets comprising the 23 collections in TSB-AD-U. The black line connects methods without a statistically significant difference at $\alpha = 0.05$. TAP achieves the best average rank and is significantly better than all five competing methods.

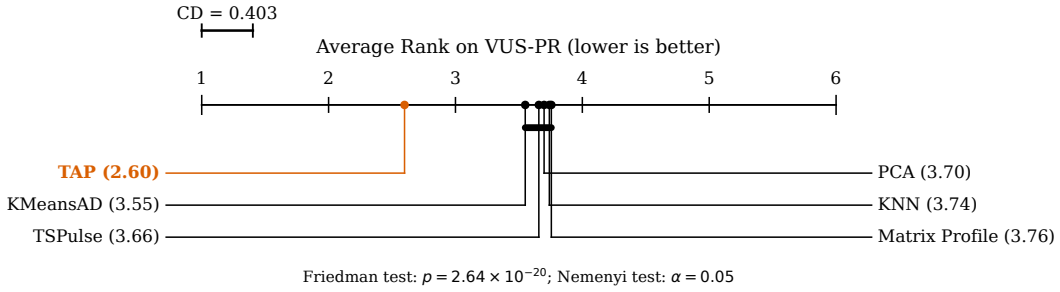


Figure 6: Critical-difference diagram based on average VUS-PR ranks across 350 datasets. Lower ranks indicate better performance.

C.2 Multivariate Results

We further evaluate the multivariate extension of TAP on TSB-AD-M. Table 6 compares TAP with representative statistical, machine-learning, neural-network, and foundation-model detectors. TAP achieves the best overall performance on five of the six metrics, obtaining a VUS-PR of 0.45, VUS-ROC of 0.79, AUC-PR of 0.39, AUC-ROC of 0.76, and point-wise F1 of 0.44. It improves VUS-PR over the second-best method, PaAno, from 0.43 to 0.45, while Time-RCD obtains the highest Range-F1 of 0.43. These results show that the proposed PCA representations and scoring framework extend effectively to multivariate time series and remain competitive across both range-wise and point-wise evaluation measures.

Table 6: Overall results on TSB-AD-M. Best results are shown in bold and second best are underlined.

		Range-Wise Measure ↑			Point-Wise Measure ↑		
	Method	VUS-PR	VUS-ROC	Range-F1	AUC-PR	AUC-ROC	F1
Stat&ML	PCA 2016	0.31	0.74	0.29	0.31	0.70	0.37
	KMeansAD 2001	0.34	0.77	0.37	0.30	0.73	0.35
	CBLOF 2003	0.27	0.70	0.31	0.28	0.67	0.32
	MCD 1999	0.27	0.69	0.20	0.27	0.65	0.33
	OCSVM 1999	0.26	0.67	0.30	0.23	0.61	0.28
	MMPAD 2024	0.35	0.76	0.30	0.31	0.74	0.36
NN	AutoEncoder 2014	0.30	0.69	0.28	0.30	0.67	0.34
	CNN 2018	0.31	0.76	0.37	0.32	0.73	0.37
	LSTM-AD 2015	0.31	0.74	0.38	0.31	0.70	0.36
	OmniAnomaly 2019	0.31	0.69	0.37	0.27	0.65	0.32
	USAD 2020	0.30	0.68	0.37	0.26	0.64	0.31
	DADA 2024	0.31	0.73	0.25	0.31	0.69	0.35
	KAN-AD 2024	0.40	0.76	0.41	0.35	0.73	0.40
	PaAno 2026	0.43	0.79	0.41	0.38	0.76	0.43
	xLSTMAD 2025	0.39	0.78	0.39	0.34	0.74	0.42
FM	Time-RCD 2025	0.32	0.71	**0.43**	0.34	0.72	0.38
TAP		**0.45**	**0.79**	0.40	**0.39**	**0.76**	**0.44**

C.3 SUPPLEMENTARY RESULTS ON TAB

Tables 7 and 8 show that TAP remains competitive under the alternative annotations and dataset composition of TAB. On TAB-U, TAP achieves the highest VUS-PR on 9 of the 14 datasets and ranks among the top two on 12, with particularly strong performance on IOPS, KDD21, MSL, SMAP, and YAHOO. On the more diverse TAB-M collection, TAP ranks first on 5 datasets and among the top two on 12 of the 25 datasets, including the best results on DLR, Exathlon, MITDB, MSL, and SMD. These results provide complementary evidence that the effectiveness of TAP is not specific to the annotations or dataset characteristics of TSB-AD.

Table 7: Dataset-level VUS-PR results on TAB-U. The six baselines are selected from the top-ranked methods on the official leaderboard. Best results are shown in bold and second best are underlined.

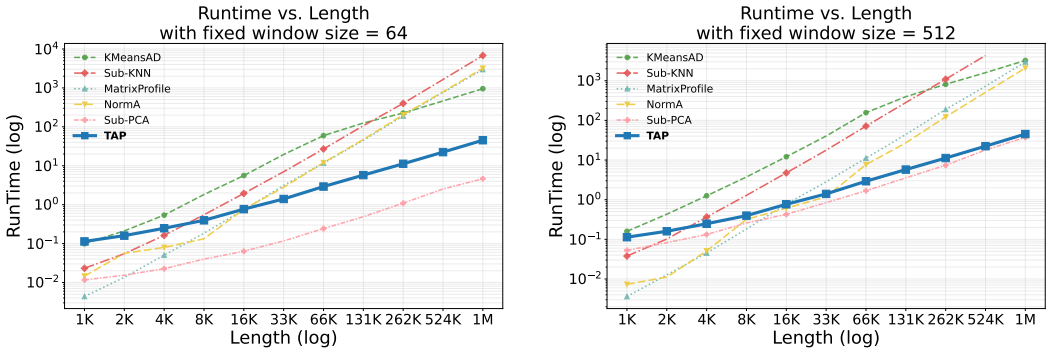
Method	Daphnet	ECG	GAIA	Genesis	GHL	IOPS	KDD21	MGAB	MSL	NAB	SMAP	SMD	SVDB	YAHOO
KMeans	**0.249**	**0.671**	0.193	0.710	**0.016**	0.240	0.267	0.014	0.570	0.563	0.636	0.386	**0.447**	0.109
DWT-MLEAD	0.104	0.262	0.211	0.811	0.005	0.111	0.243	**0.023**	0.613	0.550	0.708	0.408	0.223	0.099
iTransformer	0.131	0.156	0.354	0.362	0.010	0.209	0.055	0.005	0.403	0.336	0.452	0.327	0.109	0.512
SARIMA	0.090	0.146	0.157	0.006	0.011	0.101	0.023	0.003	0.293	0.293	0.181	0.138	0.093	0.247
ARIMA	0.091	0.146	0.153	0.003	0.011	0.103	0.022	0.003	0.297	0.286	0.201	0.134	0.093	0.263
Torsk	0.229	0.327	0.060	0.032	0.013	0.041	0.129	0.004	0.388	0.493	0.313	0.187	0.211	0.051
TAP	0.228	0.569	**0.376**	**0.821**	0.011	**0.312**	**0.509**	0.016	**0.680**	**0.602**	**0.802**	**0.443**	0.417	**0.526**

Table 8: Dataset-level VUS-PR results on TAB-M. The six baselines are selected from the top-ranked methods on the official leaderboard. Best results are shown in bold and second best are underlined.

Method	ASD	CalIt2	CATSV2	CICIDS	DAPHNET	DLR	ECG	Exathlon	GECCO	GHL	GutenTAG	LTDB	MITDB
KMeans	0.318	0.054	0.150	0.001	0.254	0.689	0.323	0.874	0.041	0.020	**0.493**	**0.601**	0.243
KNN	**0.373**	0.101	0.128	0.001	0.203	0.072	0.530	0.822	0.068	0.141	0.320	0.393	0.180
TimesNet	0.281	0.062	0.265	0.002	0.319	0.073	0.421	0.765	**0.297**	0.019	0.238	0.384	0.140
EIF	0.244	**0.114**	0.094	0.002	0.196	0.081	**0.556**	0.752	0.040	0.086	0.134	0.330	0.125
LOF	0.321	0.061	0.106	0.001	0.119	0.012	0.209	0.785	0.058	0.045	0.353	0.375	0.116
OCSVM	0.103	0.109	**0.327**	**0.389**	**0.570**	0.056	0.539	0.609	0.101	**0.157**	0.056	0.327	0.114
TAP	0.354	0.038	0.323	0.001	0.240	**0.756**	0.329	**0.950**	0.059	0.022	0.445	0.590	**0.566**

Method	MSL	NYC	PSM	PUMP	SKAB	SMAP	SMD	SVDB	SWAN	SWAT	TAO	TODS
KMeans	0.217	0.092	0.452	0.257	**0.908**	0.131	0.142	**0.712**	0.455	0.405	0.125	0.178
KNN	0.242	0.051	0.456	0.138	0.847	0.180	0.134	0.167	0.696	0.507	0.062	0.521
TimesNet	0.225	0.053	0.360	0.216	0.665	0.136	0.159	0.192	0.516	0.175	**0.815**	**0.608**
EIF	0.215	0.092	**0.458**	0.188	0.649	0.181	0.110	0.112	**0.709**	0.499	0.487	0.518
LOF	0.181	0.049	0.423	0.127	0.733	**0.191**	0.069	0.090	0.517	0.465	0.068	0.521
OCSVM	0.185	**0.318**	0.370	**0.571**	0.776	0.117	0.081	0.095	0.675	**0.523**	0.128	0.464
TAP	**0.265**	0.232	0.413	0.320	0.636	0.137	**0.174**	0.627	0.434	0.268	0.194	0.473

C.4 Scalability Comparison with Subsequence-based Baselines



(a) Representative setting: $w = 64$ **(b)** Maximum-window setting: $w = 512$

Figure 7: Runtime scalability against subsequence-based baselines under fixed window sizes (a) $w = 64$, a representative empirical choice, and (b) $w = 512$, matching the largest window used by TAP. For TAP, the indicated window size applies only to Shape-Residual PCA; Multi-Resolution PCA retains its full configuration in both settings.

We compare the runtime scalability of TAP with representative subsequence-based baselines under two fixed-window settings. The commonly used empirical choice $w = 64$ represents typical operation, while $w = 512$ matches the largest window considered by TAP and provides a more demanding scalability test. For the baselines and Shape-Residual PCA, the indicated window size is used directly; Multi-Resolution PCA retains its complete set of window sizes and temporal scales in both experiments. Figure 7 shows that TAP maintains approximately linear runtime growth under both settings and scores one million points in 36 seconds, corresponding to approximately 28,000 points per second using a single thread on one CPU core, making it practical for large-scale monitoring and diagnostic workloads. With $w = 64$, most competing methods grow to hundreds or thousands of seconds as the series length increases. This advantage persists at $w = 512$, where TAP remains close to Sub-PCA while the other baselines grow substantially more expensive. Sub-PCA is faster because it evaluates only one fixed-window representation, whereas TAP additionally computes multi-resolution and shape-residual evidence. These results show that the additional temporal representations introduce only moderate overhead while preserving scalability at the largest window setting.

Benefiting from PCA-based scoring, TAP neither repeatedly compares all subsequences nor rebuilds large covariance matrices for each resolution. It estimates the autocovariance using FFT-based computation, constructs the largest required Toeplitz covariance matrix, and performs eigendecomposition on a bounded matrix of size at most 512×512. Anomaly scores are then obtained through linear projection of the window representations. For a series of length T, the main costs are $O(T \log T)$ for autocovariance estimation, $O(w_{\max}^3)$ for eigendecomposition, and $O(T \sum_r d_r k_r)$ for projection, where $w_{\max} \leq 512$ and d_r and k_r denote the dimension and number of retained PCA directions at resolution r. Since these quantities are bounded in our implementation, runtime grows approximately linearly with T, consistent with the empirical results.

Figure 7 shows that TAP has a smaller growth rate on the log-log runtime plot than most subsequence-based baselines, including KMeansAD, Sub-KNN, Matrix Profile, NormA, and Sub-PCA. The advantage becomes increasingly clear on long time series: while several baselines grow rapidly as the sequence length approaches 1M, TAP remains substantially faster due to its bounded PCA decomposition and efficient linear projection. This demonstrates that the proposed design is especially suitable for large-scale time-series anomaly detection.

C.5 Detailed Analysis

Case study of cross-resolution consensus. Figure 8 compares CREC with direct mean aggregation of all multi-resolution scores. For Yahoo-238, the anomaly response is substantially weakened by averaging across resolutions, whereas CREC concentrates the jointly supported evidence around the labeled anomaly. UCR-194 contains several competing local responses, but CREC more clearly distinguishes the labeled anomaly from the background. These examples illustrate that cross-resolution consensus can prevent strongly supported anomaly evidence from being diluted by less informative resolution-specific responses.

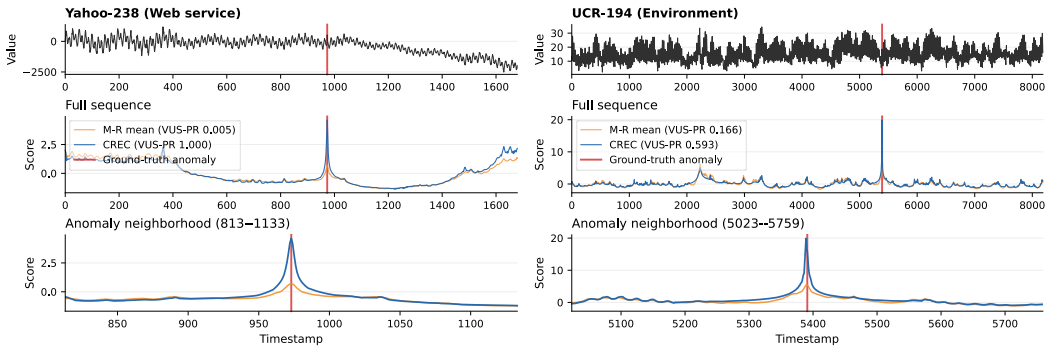


Figure 8: Case studies comparing direct mean aggregation of multi-resolution scores with CREC. The top row shows the input time series, the middle row shows the complete anomaly-score sequences, and the bottom row presents an enlarged view around the labeled anomaly. CREC prevents consistently supported anomaly evidence from being diluted by less informative resolution-specific responses.

Complementary anomaly evidence from M-R and S-R. Figure 9 provides two representative cases to illustrate the complementary behavior of the shape-residual PCA branch and the multi-resolution PCA branch. In the first case, the anomaly mainly appears as a change in local waveform shape. The shape-residual PCA branch successfully assigns high anomaly scores to the abnormal region, while the multi-resolution PCA branch mostly responds to periodic spike-like patterns and is less specific to the labeled shape anomaly. In the second case, the shape change is less pronounced, but the anomalous segment exhibits noticeably stronger noise fluctuation than the normal region. In this setting, the multi-resolution PCA branch produces larger anomaly scores, whereas the shape-residual PCA branch is less sensitive. The scatter plot on the left further shows that the two branches often assign high scores to different time series, indicating that they capture different anomaly evidence and are complementary rather than redundant.

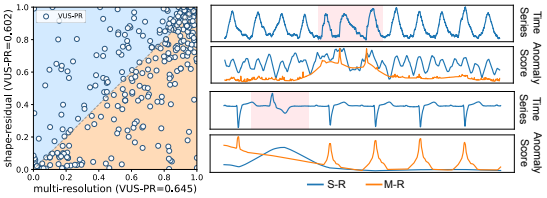


Figure 9: Two branches capture complementary anomaly evidence across different cases.

C.6 Parameter Sensitivity

Figure 10 evaluates CREC using paths containing two, three, or four nodes. Increasing the path length from two to three improves M-R VUS-PR from 0.643 to 0.645 and full-model VUS-PR from 0.655 to 0.659, suggesting that consensus across a broader resolution neighborhood provides more reliable anomaly evidence. Extending the path to four nodes produces a further marginal improvement for M-R to 0.646, while the complete model remains nearly unchanged at 0.659. Moreover, the full model consistently outperforms the M-R branch under every path length, showing that the shape-residual evidence remains complementary across different CREC configurations. Overall, performance varies only slightly across the three settings, indicating that CREC is robust to the exact path length. We use a path length of three in all the other experiments.

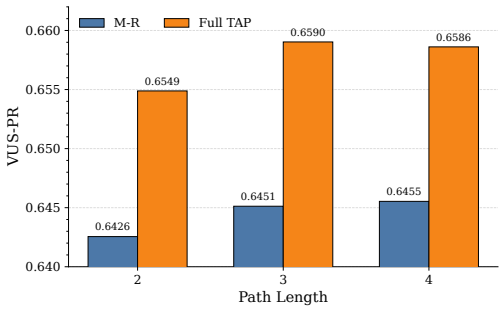


Figure 10: Sensitivity of M-R and full TAP to the CREC path length.

C.7 Multiple-Shape S-R under Regime Shifts

Figure 11 presents examples from Yahoo containing persistent regime shifts. The single-shape model assigns elevated scores to normal patterns that differ from its global prototype. By estimating separate

prototypes for the detected regimes, the multiple-shape extension suppresses these false-positive responses while preserving the labeled anomaly.

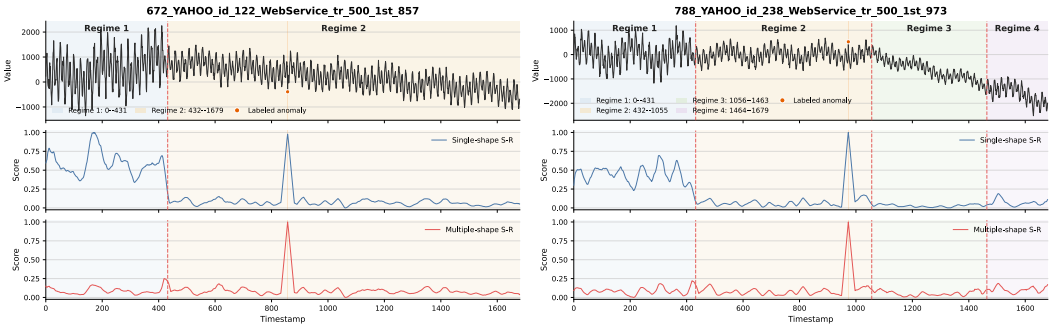


Figure 11: Comparison between single-shape and multiple-shape S-R on a non-stationary Yahoo series. Dashed lines indicate automatically detected regime boundaries, and shaded regions indicate labeled anomalies. Scores are independently normalized to ([0,1]) for visualization.

C.8 Limitations and Future Work

The M-R and S-R branches capture complementary anomaly evidence, but their relative usefulness varies across time series. Our normalized LogSumExp fusion combines both branches without supervision and therefore cannot determine which branch is more informative for a particular input. In comparison, an oracle strategy that uses the ground-truth labels to select the better branch for each time series reaches a VUS-PR of approximately 0.70, remaining above the current fused result. This gap suggests that the individual branches already contain additional useful evidence that is not fully exploited by a fixed unsupervised fusion rule. Developing reliable label-free strategies for estimating branch relevance and adapting the fusion at the series or timestamp level is therefore a promising direction for future work. Moreover, our current multivariate extension processes each channel independently and subsequently aggregates the channel-wise scores. Explicitly modeling cross-channel dependencies may further improve detection in multivariate time series and represents another direction for future work.