Enhancing Time Series Anomaly Detection with Residuals Stationarity Intervention on State-Space Models

Zhanwen Xin[a,*], James-A. Goulet[a]

[a]Department of Civil, Geologic and Mining Engineering, Polytechnique Montreal, Montreal, H3T 1J4, Quebec, Canada

ARTICLE INFO

Keywords:
Structural health monitoring
Time series anomaly detection
State-space models
Intervention models
Dam
Infrastructure
Bayesian methods

ABSTRACT

Across the world, nations face challenges with maintaining aging infrastructure with limited resources. In order to address a part of this challenge with structural health monitoring (SHM), we need to enable the online detection of anomalies while limiting the number of false alarms; furthermore, we need to do it at scale on a large number of structures. State-space models (SSM) are prime candidates to address this challenge as they enable to separate structural responses from environmental effects in an online manner. Various detection mechanisms, such as the switching Kalman filter and machine learning approaches, have been integrated with SSMs for identifying abnormal behaviours in time series. However, these methods face two main limitations: (1) the residual component characterizing the prediction errors may become non-stationary and mistakenly explain baseline shifts, thereby reducing anomaly detectability; and (2) following a detection, they may require months if not years of observations, to adapt to a new stationary regime. To address these challenges, we propose the *Residuals Stationarity Intervention* (RSI), which enhances the anomaly detection capacity by monitoring the stationarity of residuals and enforcing it through interventions predicted by a Bayesian neural network. Experiments on synthetic and real time series recorded on dams demonstrate that RSI outperforms existing approaches, achieving higher detection probabilities, shorter detection times, and faster adaptation, while maintaining a low false alarm rate.

1. Introduction

A key challenge in Structural Health Monitoring (SHM) is online anomaly detection [1], that is, detecting changes in time series while data is continuously collected. Existing methods used for tackling this challenge typically rely either on a moving window [2], which risks discarding valuable past information, or on retraining with the entire historical dataset as new data arrive [3], which becomes computationally challenging as decades of data accumulate and models grow in complexity. Consequently, such approaches are restricted to critical structures or for the short-term monitoring of extreme events [4, 5], which are not representative of the subtle irreversible changes associated with early signs of degradation. Despite these limitations, long-term SHM is needed in many practical applications [6] and its usage at scale across infrastructure networks remains an open problem [7].

Anomalies can alter time-series in various ways [8]. While the detection of obvious pattern shifts and outliers have been addressed with various machine learning methods [9, 10, 11], subtle irreversible changes in the response of structure that gradually accumulate over the long term have seldom been investigated. Such changes associated with the progressive degradation of infrastructure [12, 13] are notoriously difficult to detect in their early stages, when models often mix them up with model prediction errors.

State-space models (SSM), an inherently online method, address these challenges by recursively assimilating new data while providing an interpretable data decomposition into physically meaningful components [14], where the time correlated prediction errors are modelled by a residual component [15]. SSMs perform well with stationary or trend-stationary baselines [14, 16], but their predictive capacities degrade when the baselines become non-stationary due to the presence of anomalies. The Switching Kalman Filter (SKF) enables detecting anomalies by quantifying the probability that a regime switch occurs from a stationary SSM to a non-stationary one [17]. In practice, the SKF may fail to capture regime switches when the residual component mistakenly explains baseline shifts [18]. Another limitation of the SKF is that defining the configuration of the non-stationary SSM requires some prior knowledge about future anomalies. Moreover, introducing multiple non-stationary models to cover a broader range of cases induces a quadratic increase in the computational cost. After a change is detected, the SKF typically requires a period, ranging from several weeks up to years, in order to adapt its baseline to a new stationary regime, during which false or missed alarms may occur.

In this paper, we present a new method for applying interventions on a SSM in order to maintain the stationarity of its residuals. In this approach, referred to as *Residuals Stationarity Intervention* (RSI), interventions are predicted by a Bayesian neural network (BNN) and are only applied to SSMs if the probability of having an anomaly exceeds a threshold. This probability is quantified jointly from the data likelihood, along with the likelihood that the residual term remains stationary. The interventions correct the models' baselines and enforce the stationarity of residuals, allowing SSMs to adapt more quickly to new regimes. In addition to the interventions, the BNN predicts the amount of time that has elapsed between the start of the anomaly and the time when

*Corresponding author
zhanwen.xin@etud.polymtl.ca (Z. Xin); james.goulet@polymtl.ca (James-A. Goulet)

the intervention has been triggered. This allows accounting for the inevitable delays in the detection by intervening in the SSM multiple steps ahead, which further quickens the adaptation to a new stationary regime.

This paper is organized as follows: Section 2 reviews existing time-series anomaly detection methods. Section 3 provides background on the Kalman filter and the approach to train Bayesian neural networks. Section 4 presents the details of the RSI method. Finally, Section 5 presents a comparative performance evaluation of RSI against existing anomaly detection approaches using both synthetic and real time series.

2. Related works

Time-series anomaly detection (TSAD) aims to identify unexpected patterns in temporal data that deviate from a normal behaviour. Depending on whether a predictive model is involved, existing TSAD approaches can be categorized as *model-based* or *model-free*. Model-based methods detect anomalies by quantifying the inconsistency between observations and the behaviour predicted by an underlying model [17, 19, 20, 21, 22, 23, 24], whereas model-free methods infer anomalies directly from intrinsic features present in the data without relying on predictive models [25, 26, 27]. Vuong conducted a comprehensive comparison of both paradigms, showing that two model-based methods, the Switching Kalman filter (SKF) and Prophet, achieved the strongest performance, followed by two model-free methods, Matrix Profile and k-nearest neighbours (kNN) [28]. In this section, we will first review the model-free methods, followed by the model-based ones, and finally discuss their operational schemes.

Model-free anomaly detection typically measures the distance between a target window of data points and a subset of the remaining time series, identifying samples that deviate significantly as anomalies. kNN [26] quantifies this distance by averaging the k-nearest distances, whereas Matrix Profile [25] considers only the nearest distance, making it a special case of kNN with $k = 1$. These methods are effective for detecting abnormal patterns in stationary time series, but often fail in trend-stationary ones, whose baseline is persistently deviating and increases the distances. This issue can be mitigated through preprocessing steps such as normalization or decomposition; however, such transformations may distort the original patterns and reduce their capacity of detecting anomalies.

The underlying models in model-based anomaly detection methods range from simple probabilistic distributions [19], to statistical models [17, 21, 22, 20], and neural networks [23, 24]. These methods detect anomalies based on either 1) prediction errors, where observations deviate significantly from model predictions [21, 22, 24]; 2) model selection, by quantifying the likelihood of a set of candidate models describing normal and abnormal behaviours [19, 17]; 3) changes in the hidden state or model parameters, such as shifts in estimated trends [20]. In the following, we review three representative model-based anomaly detection methods: *Bayesian online changepoint detection (BOCD)* [19], *Prophet* [20] and *Switching Kalman filter (SKF)* [17].

BOCD detects anomalies by estimating the posterior distribution over the *run length*, i.e., the number of time steps since the parameters of the underlying probabilistic distribution last changed. This method performs well on time series following simple dynamics governed by a fixed probabilistic distribution, but requires coupling with other models for complex ones [29]. Moreover, estimating the posterior distribution of run lengths across all past time steps can be computationally expensive for long time series.

Prophet fits time series using a parametric regression model that incorporates trend, seasonality, and holiday effects. It partitions the time series into segments and estimates a separate trend for each. Anomalies are flagged when the change in trend between consecutive segments exceeds a threshold.

SKF quantifies the probabilities of stationary and non-stationary SSMs, and flags an anomaly when the probability of the non-stationary SSM exceeds a threshold. To address its limitations that are discussed in Section 1, reinforcement [30] and imitation learning [31] have been used with SKF to exploit drift information in trend estimates. However, the reliance of these methods on high-quality anomaly labels limits their applicability to real time series. The SKF's anomaly detectability has also been improved by constraining its residual component to prevent it from mistakenly explaining baseline shifts [18].

Among the methods reviewed above, only SKF and BOCD are inherently online anomaly detection methods, while the others are offline. Offline model-free methods can be adapted to online operation without increasing computational cost by only computing distances with respect to past subsequences instead of the entire time series. Although offline model-based methods can also be operated in an online manner, for example, by retraining models as new data arrive, such adaptation quickly becomes computationally expensive as data accumulate and therefore does not scale well.

3. Background methods

This section covers the background material associated with the Kalman filter and Bayesian neural networks.

3.1. Kalman filter

The Kalman filter is an algorithm for online state estimation of a dynamic system from noisy data, without requiring storage of the full observation history [32]. For a sequence of observations up to time t, $\boldsymbol{y}_{1:t} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t\}$, the Kalman filter produces the prior and posterior for the current hidden state estimates, $\boldsymbol{X}_{t|t-1} \sim \mathcal{N}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$ and $\boldsymbol{X}_{t|t} \sim \mathcal{N}(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})$, as well as the observation prediction, $\boldsymbol{Y}_{t|t-1} \sim \mathcal{N}(\boldsymbol{\mu}^{\boldsymbol{Y}}_{t|t-1}, \boldsymbol{\Sigma}^{\boldsymbol{Y}}_{t|t-1})$, through a *prediction step* and an *update step*. Here, $\boldsymbol{\mu}_{t|t-1} = \mathbb{E}[\boldsymbol{X}_t \mid \boldsymbol{y}_{1:t-1}]$ and $\boldsymbol{\mu}_{t|t} = \mathbb{E}[\boldsymbol{X}_t \mid \boldsymbol{y}_{1:t}]$ denote the prior and posterior means of the hidden state, while $\boldsymbol{\Sigma}_{t|t-1} = \mathbb{V}[\boldsymbol{X}_t \mid \boldsymbol{y}_{1:t-1}]$ and $\boldsymbol{\Sigma}_{t|t} = \mathbb{V}[\boldsymbol{X}_t \mid \boldsymbol{y}_{1:t}]$ are the corresponding covariance matrices, $\boldsymbol{\mu}^{\boldsymbol{Y}}_{t|t-1} = \mathbb{E}[\boldsymbol{Y}_t \mid \boldsymbol{y}_{1:t-1}]$

and $\Sigma^{Y}_{t|t-1} = \mathbb{V}[Y_t \mid y_{1:t-1}]$ are the mean and covariance matrix of the observation prediction. We adopt a compact subscript notation throughout this paper, e.g., $t|t-1$ and $t|t$, where the first index denotes the time step of the estimate and the second index denotes the time step up to which observations are used.

The prediction step of a basic Kalman filter takes as input the posterior Gaussian distribution of the hidden states from the previous time step, $X_{t-1|t-1} \sim \mathcal{N}(x_{t-1}; \mu_{t-1|t-1}, \Sigma_{t-1|t-1})$, and outputs the prior state estimates $X_{t|t-1}$ as well as the observation prediction $Y_{t|t-1}$. These are obtained through a transition model,

$$x_t = \mathbf{A}x_{t-1} + w_t, \quad W \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \tag{1}$$

and an observation model, which projects the prior hidden states onto the observation space,

$$y_t = \mathbf{F}x_t + v_t, \quad V \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \tag{2}$$

where $\mathbf{A}$ is a transition matrix, w_t is a Gaussian transition error with zero mean and covariance $\mathbf{Q}$, $\mathbf{F}$ is an observation matrix and v_t is a Gaussian observation error with zero mean and covariance $\mathbf{R}$.

The update step estimates the posterior distribution of the hidden states by combining the prior with the current observation y_t,

$$\mu^{X}_{t|t} = \mu^{X}_{t|t-1} + \Sigma^{XY}_{t|t-1}(\Sigma^{Y}_{t|t-1})^{-1}(y_t - \mu^{Y}_{t|t-1}), \tag{3}$$

$$\Sigma^{X}_{t|t} = \Sigma^{X}_{t|t-1} - \Sigma^{XY}_{t|t-1}(\Sigma^{Y}_{t|t-1})^{-1}(\Sigma^{XY}_{t|t-1})^{\intercal}, \tag{4}$$

where the cross-covariance matrix is given by $\Sigma^{XY}_{t|t-1} = \Sigma^{X}_{t|t-1}\mathbf{F}^{\intercal}$. The posterior distribution $X_{t|t} \sim \mathcal{N}(x_{t|t}; \mu_{t|t}, \Sigma_{t|t})$ can then be carried forward to the next time step $t+1$ as new observations arrive, allowing the Kalman filter to operate recursively.

3.1.1. Hidden state decomposition

When the dynamic system in the Kalman filter is expressed as a state-space model (SSM), its components can be modularly specified by adjusting the hidden state vector x and the model matrices, which enable the decomposition of a time series into interpretable quantities, such as: 1) baseline components which describe the underlying tendency (e.g., level and trend) of a time series after excluding short-term fluctuations and reversible patterns, 2) periodic components capturing structured temporal dependencies, and 3) residual components which account for the remaining errors, expressed as the discrepancy between the observation and the prediction from the other components.

The formulations of the common components in each category is reviewed as follow:

Baselines. For a trend-stationary time series, a local trend component (LT) can be incorporated into the SSM to model both the level and its rate of change over time. The corresponding hidden state vector, as well as the transition and observation matrices, are formulated as

$$x^{\mathrm{LT}} = \begin{bmatrix} x^{\mathrm{LL}} \\ x^{\mathrm{LT}} \end{bmatrix}, \ \mathbf{A}^{\mathrm{LT}} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{Q}^{\mathrm{LT}} = (\sigma^{\mathrm{LT}}) \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix}, \ \mathbf{F}^{\mathrm{LT}} = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Periodicity. Periodic phenomena can be modelled using a Fourier representation in the harmonic case, or kernel regression [16] in the non-harmonic case. More recently, state-of-the-art approaches leverage an analytically tractable Bayesian Long Short-Term Memory (LSTM) model [33], which is non-deterministic and compatible with the SSM framework. This formulation can capture complex periodicity and nonlinear temporal dependencies without requiring feature engineering. The corresponding hidden state vector, along with the transition and observation matrices, is defined as

$$x^{\mathrm{LSTM}} = \begin{bmatrix} z^0 \end{bmatrix}, \ \mathbf{A}^{\mathrm{LSTM}} = \begin{bmatrix} 1 \end{bmatrix},$$
$$\mathbf{Q}^{\mathrm{LSTM}} = \begin{bmatrix} \sigma^{\mathrm{LSTM}} \end{bmatrix}, \ \mathbf{F}^{\mathrm{LSTM}} = \begin{bmatrix} 1 \end{bmatrix},$$

where the pattern hidden state z^0 is predicted not through the transition model in eq. 1, but instead by a TAGI-LSTM network, summarized as

$$z^0 = \text{TAGI-LSTM}(i_t, h_{t-1}, c_{t-1}, \theta_{t-1}),$$

where i_t denotes the input covariates, h_{t-1} and c_{t-1} represent the TAGI-LSTM hidden states and cell states, and θ_{t-1} denotes the network parameters from the previous time step $t-1$.

Residuals. Model errors are typically correlated across time steps but are assumed to be stationary in the long term. A common residual component is the first-order autoregressive (AR) model, formulated as

$$x^{\mathrm{AR}} = \begin{bmatrix} x^{\mathrm{AR}} \end{bmatrix}, \ \mathbf{A}^{\mathrm{AR}} = \begin{bmatrix} \phi^{\mathrm{AR}} \end{bmatrix},$$
$$\mathbf{Q}^{\mathrm{AR}} = \begin{bmatrix} \sigma^{\mathrm{AR}} \end{bmatrix}, \ \mathbf{F}^{\mathrm{AR}} = \begin{bmatrix} 1 \end{bmatrix},$$

where the autoregression coefficient ϕ^{AR}, which describes the dependency between consecutive time steps, is constrained to the interval $[0, 1)$ to ensure stationarity of the process errors, and σ^{AR} denotes the standard deviation of the process error. These two hyperparameters can be augmented into the hidden state vector and inferred jointly with x^{AR} using eq. 3 and 4, a method referred to in the literature as *online AR* [15, 34].

The components described above can be assembled into a single SSM as

$$x = \begin{bmatrix} x^{\mathrm{LT}} \\ x^{\mathrm{LSTM}} \\ x^{\mathrm{AR}} \end{bmatrix}, \ \mathbf{A} = \begin{bmatrix} \mathbf{A}^{\mathrm{LT}} & 0 & 0 \\ 0 & \mathbf{A}^{\mathrm{LSTM}} & 0 \\ 0 & 0 & \mathbf{A}^{\mathrm{AR}} \end{bmatrix},$$
$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}^{\mathrm{LT}} & 0 & 0 \\ 0 & \mathbf{Q}^{\mathrm{LSTM}} & 0 \\ 0 & 0 & \mathbf{Q}^{\mathrm{AR}} \end{bmatrix}, \ \mathbf{F} = \begin{bmatrix} \mathbf{F}^{\mathrm{LT}}, \mathbf{F}^{\mathrm{LSTM}}, \mathbf{F}^{\mathrm{AR}} \end{bmatrix}.$$

A synthetic example of the decomposition of an 8-year time series with weekly intervals using these three components is shown in Figure 1. The normalized data (red) exhibits both seasonal and yearly periodic patterns, a constant zero-valued trend, and autoregressive residuals. The LSTM is trained on 30% of the data and validated on the subsequent 10%. For the remaining portion of the time series, the LSTM network is fixed within the SSM and provides one-step-ahead predictions at each time step. The prior distributions of the hidden states over time (blue) illustrate that each component captures distinct features of the time series. By projecting all the hidden states onto the observation predictions (green), the prediction results align well with the data.

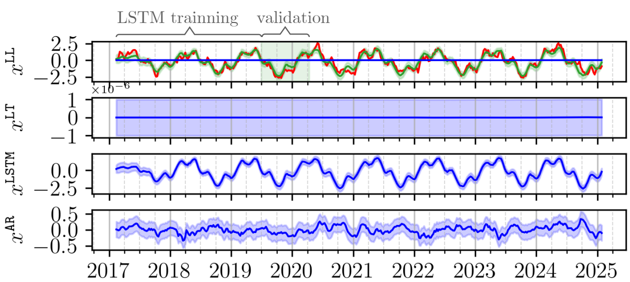


Figure 1: Decomposition of a synthetic time series (red) into local trend, LSTM, and autoregressive components. The prior distributions of the hidden states at different time steps are shown in blue, and the predicted observations are shown in green.

3.1.2. *Delays in the residual hidden states*

When the time series is trend-stationary, the Kalman filter attributes parts of the signal to each component with stationary residuals. By contrast, when the time series has a non-stationary trend, the reviewed baseline component cannot adapt without an acceleration hidden state to model the change in trend. In such cases, the drift in the level under the new trend is absorbed by the residual component, which is typically more flexible than the others. The residual component cannot immediately capture the new trend due to

1) the impact of a small change in trend being negligible at the beginning, when the anomaly is developing,

2) the LSTM's ability to accommodate minor to moderate baseline changes in its predictions before it eventually fails to generalize,

3) the transition from $x_{t-1}^{\mathtt{AR}}$ to $x_t^{\mathtt{AR}}$ being limited by the autoregressive coefficient $\phi^{\mathtt{AR}} \in [0, 1)$, where smaller values of $\phi^{\mathtt{AR}}$ require longer adaptation time to the drift.

An example of such a *residual adaptation process* is shown in Figure 2, where the zero trend in the time series shown in Figure 1 shifts to a positive trend after the red dashed line. This change in trend has a negligible impact on the time series during the first six months, when the prior estimates of the trend and the LSTM remain almost identical to those in the stationary period before January 2021. A small drift appears in the residual beginning in June 2021 but soon returns to the stationary range by December. The drift is not distinguishable from the stationary period until May 2022.

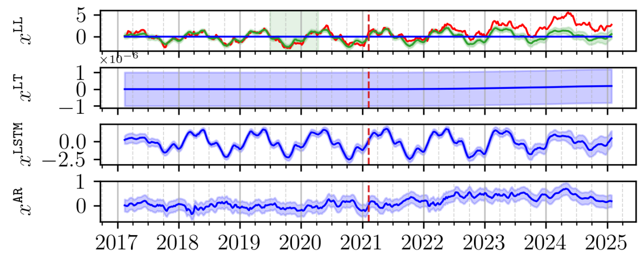


Figure 2: Delay in the residual estimation when the time series exhibits a non-stationary trend.

This example illustrates the three main reasons listed above that account for the delay in residual drift in a non-stationary time series. To mitigate such delays, we propose a two-step detection agent and a neural network–based intervention model, which will be further discussed in Section 4.

3.2. Tractable approximate Gaussian inference

Tractable Approximate Gaussian Inference (TAGI) is a Bayesian approach for updating neural network parameters that are modelled as Gaussian distributions, rather than as deterministic values updated via backpropagation [35]. This method has been shown to quantify the epistemic uncertainty in the network parameters, to model heteroscedastic aleatory uncertainty [36], and to propagate these uncertainties to obtain Gaussian outputs with associated variances in the forward path. Backward inference in TAGI is carried out recursively in a layer-wise manner, with the network's inputs, hidden states, parameters, and outputs all described as Gaussian random variables: $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{X}}, \boldsymbol{\Sigma}_{\boldsymbol{X}})$, $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{Z}}, \boldsymbol{\Sigma}_{\boldsymbol{Z}})$, $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$ and $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{Y}}, \boldsymbol{\Sigma}_{\boldsymbol{Y}})$.

When incorporating a neural network into a SSM, either as a component such as an LSTM [33] or as the intervention model proposed in this paper, it is important that its output be represented as a Gaussian variable, which allows the Kalman filter to operate on it. TAGI, whose effectiveness has been demonstrated in various machine learning tasks (e.g., adversarial attack generation, optimization, and reinforcement learning) [37], provides a suitable approach for training the intervention model in this study.

4. Methodology

The autoregressive (AR) residual component in a SSM is typically more flexible than the baseline component, and often captures a part of the drift caused by anomalies in time series. In this section, we leverage such drifts in the residuals in order to detect anomalies. The probability of anomaly is estimated by combining the likelihood of the data along with the probability that the residual remains stationary. When the probability of anomaly exceeds a threshold, an intervention model predicts the changes required for the hidden states to adapt to a new stationary regime. Section 4.1 presents the procedure to isolate the non-stationary drift from the stationary residuals. Section 4.2 presents how to quantify the probability of anomaly, and Section 4.3 describes the intervention model relying on a Bayesian neural network.

4.1. Residual drift model

As presented in Section 3.1.2, the $\mathtt{AR}$ component captures stationary residuals as well as drifts caused by anomalies. The residual's drift can be isolated from its stationary part using an auxiliary SSM consisting of a local trend and an autoregressive component ($\mathcal{C}$), referred to as the *drift model* (d), $\mathcal{M}^{\mathrm{d}} = \{\mathcal{C}^{\mathtt{LTd}},\ \mathcal{C}^{\mathtt{ARd}}\}$. The drift model further decomposes the base SSM $\mathtt{AR}$'s posterior hidden state $X^{\mathtt{AR}}_{t|t} \sim \mathcal{N}(\mu^{\mathtt{AR}}_{t|t}, (\sigma^{\mathtt{AR}}_{t|t})^2)$ into the state vector $\boldsymbol{x}^{\mathrm{d}} = [x^{\mathtt{LLd}}, x^{\mathtt{LTd}}, x^{\mathtt{ARd}}]^{\intercal}$. The transition and observation models of the drift model are

$$
\begin{aligned}
\boldsymbol{x}^{\mathrm{d}}_t &= \mathbf{A}^{\mathrm{d}}\boldsymbol{x}^{\mathrm{d}}_{t-1} + \boldsymbol{w}^{\mathrm{d}}_t, \quad \boldsymbol{W}^{\mathrm{d}} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}^{\mathrm{d}}),\\
\hat{x}^{\mathtt{AR}}_t &= \mathbf{F}^{\mathrm{d}}\boldsymbol{x}^{\mathrm{d}}_t + \boldsymbol{v}^{\mathrm{d}}_t, \quad \boldsymbol{V}^{\mathrm{d}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}^{\mathrm{d}}),
\end{aligned}
$$

where $\boldsymbol{x}^{\mathrm{d}}_t$ is a realization of the prior hidden-states $\boldsymbol{X}^{\mathrm{d}}_{t|t-1} \sim \mathcal{N}(\boldsymbol{\mu}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t-1}, \boldsymbol{\Sigma}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t-1})$, and $\hat{x}^{\mathtt{AR}}_t$ is a realization of the prior predictive for the $\mathtt{AR}$ of the base SSM, i.e., $\hat{X}^{\mathtt{AR}}_{t|t-1} \sim \mathcal{N}(\hat{\mu}^{\mathtt{AR}}_{t|t-1}, (\hat{\sigma}^{\mathtt{AR}}_{t|t-1})^2)$. The $\mathbf{A}^{\mathrm{d}}$, $\mathbf{Q}^{\mathrm{d}}$, $\mathbf{F}^{\mathrm{d}}$, $\mathbf{R}^{\mathrm{d}}$ matrices are detailed in Appendix A. The posterior hidden states in the drift model are obtained using the RTS smoother equations [38] following

$$
\begin{aligned}
\boldsymbol{\mu}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t} &= \boldsymbol{\mu}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t-1} + \mathbf{J}_t(\mu^{\mathtt{AR}}_{t|t} - \hat{\mu}^{\mathtt{AR}}_{t|t-1}),\\
\boldsymbol{\Sigma}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t} &= \boldsymbol{\Sigma}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t-1} + \mathbf{J}_t\left((\sigma^{\mathtt{AR}}_{t|t})^2 - (\hat{\sigma}^{\mathtt{AR}}_{t|t-1})^2\right)(\mathbf{J}_t)^{\intercal},\\
\mathbf{J}_t &= \boldsymbol{\Sigma}^{\boldsymbol{X}^{\mathrm{d}}}_{t|t-1}\mathbf{F}^{\intercal}(\hat{\sigma}^{\mathtt{AR}}_{t|t-1})^{-2}.
\end{aligned}
$$

Returning to the example shown in Figure 2, the drift can now be isolated by the $x^{\mathtt{LLd}}$ and $x^{\mathtt{LTd}}$ hidden states as shown in Figure 3, where the posterior hidden states from the drift model are plotted in orange. The $\mathtt{LTd}$ estimates provide a clean state representation while remaining sensitive to drifts caused by anomalies, offering a more distinguishable separation from the stationary regime than the base SSM residuals $x^{\mathtt{AR}}_{t|t}$. In Section 4.2, we will rely on the $x^{\mathtt{LTd}}_{t|t}$ to quantify the probability that the residual remains stationary, and in Section 4.3 to learn the intervention required on the hidden states of the base SSM.

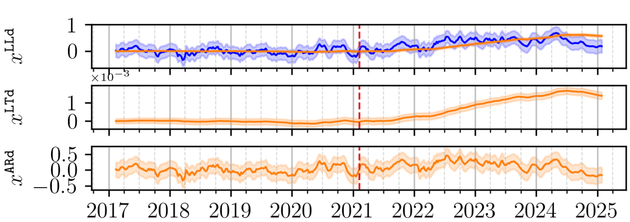


Figure 3: Residual decomposition with the drift model. The posterior hidden states from the drift model are plotted in orange. The residuals plotted with $x^{\mathtt{LLd}}$ in the first subfigure are estimated using the base SSM in the example shown in Figure 2.

4.2. Probability of anomaly

In this section, we present how to identify anomalies by combining the probability that the residual remains stationary along with the data likelihood. It concludes with a discussion explaining why alternative methods, such as supervised- or reinforcement-based methods, have difficulties coping with the detection of slowly-developing anomalies.

The probability density describing stationary residuals is quantified using a set of synthetic realization of $\mu^{\mathtt{LTd}}_{t|t}$, the expected value for the trend of the drift model, generated under a stationary condition. We rely on synthetic data because the real training and validation sets typically only consist in short stationary segments which limits their ability to generalize to unforeseen cases. To address this issue, we use the base SSM to generate $\mathtt{N}$ synthetic time series of length $\mathtt{L}$, $\{\boldsymbol{y}^1_{1:\mathtt{L}}, \boldsymbol{y}^2_{1:\mathtt{L}}, ..., \boldsymbol{y}^{\mathtt{N}}_{1:\mathtt{L}}\}$. Both the base and drift SSMs are then applied on these time series, along with the real training and validation sets, to collect realizations of $\mu^{\mathtt{LTd}}_{t|t}$. The expected values for the trend hidden state of the drift model are then used to estimate the parameters of a Gaussian distribution, $\mathcal{N}(\mu^{\mathtt{SR}}, (\sigma^{\mathtt{SR}})^2)$, describing the *stationary residual* ($\mathtt{SR}$).

We consider two possible control actions, $\mathcal{A} = \{a_0, a_1\}$, applied to the posterior state, $\boldsymbol{X}_{t-1|t-1}$, of the base model before applying the Kalman filter at each time step t. The action a_0 corresponds to "*do nothing*", leaving $\boldsymbol{X}_{t-1|t-1}$ unchanged, while a_1 corresponds to "*intervene*" on the base and drift models so that an unexpected drift wrongfully captured by the $\mathtt{AR}$ is reassigned to the baseline component. Under a_1, the posterior states $\boldsymbol{X}_{t-1|t-1}$ are modified by transferring the local level and trend components from the drift model to the base model, according to

$$
\begin{aligned}
\mu^{\mathtt{LL}}_{t-1|t-1} &= \mu^{\mathtt{LL}}_{t-1|t-1} + \mu^{\mathtt{LLd}}_{t-1|t-1}, &(5)\\
\mu^{\mathtt{LT}}_{t-1|t-1} &= \mu^{\mathtt{LT}}_{t-1|t-1} + \mu^{\mathtt{LTd}}_{t-1|t-1}, &(6)\\
(\sigma^{\mathtt{LL}}_{t-1|t-1})^2 &= (\sigma^{\mathtt{LL}}_{t-1|t-1})^2 + (\sigma^{\mathtt{LLd}}_{t-1|t-1})^2, &(7)\\
(\sigma^{\mathtt{LT}}_{t-1|t-1})^2 &= (\sigma^{\mathtt{LT}}_{t-1|t-1})^2 + (\sigma^{\mathtt{LTd}}_{t-1|t-1})^2, &(8)\\
\mu^{\mathtt{LLd}}_{t-1|t-1} &= 0, &(9)\\
\mu^{\mathtt{LTd}}_{t-1|t-1} &= \mu^{\mathtt{SR}}. &(10)
\end{aligned}
$$

Depending on the action selected, the Kalman filter begins with one of two possible hidden states $\boldsymbol{X}_{t-1|t-1}$, resulting in different observation predictions $\boldsymbol{Y}_{t|t-1}$ and posterior drift estimates $X^{\mathtt{LTd}}_{t|t}$. Given an action a_i with $i \in \{0, 1\}$, the data likelihood is

$$
f(\boldsymbol{y}_t|a_i, \boldsymbol{y}_{1:t-1}) = \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\mu}^{\boldsymbol{Y}|a_i}_{t|t-1}, \boldsymbol{\Sigma}^{\boldsymbol{Y}|a_i}_{t|t-1}),
$$

and the likelihood that the expected value for the trend of the drift model is a realization from the $\mathtt{SR}$ distribution is

$$
f(\mu^{\mathtt{LTd}}_{t|t}) = \mathcal{N}(\mu^{\mathtt{LTd}}_{t|t}; \mu^{\mathtt{SR}}, (\sigma^{\mathtt{SR}})^2).
$$

By assuming that the observation and the stationary residual prior predictives are independent, and having the prior probability of intervention $p(a_1)$, the joint likelihood of observing data $\boldsymbol{y}_t$, having a stationary residual, and taking action a_i is computed following

$$f(\boldsymbol{y}_t, \mu^{\mathtt{LTd}}_{t|t}, a_i|\boldsymbol{y}_{1:t-1}) = f(\boldsymbol{y}_t|a_i, \boldsymbol{y}_{1:t-1})f(\mu^{\mathtt{LTd}}_{t|t})p(a_i).$$

The probability of anomaly is quantified as the amount of the probability that a_1 is required to maintain a stationary residual along with predictive likelihood of $\boldsymbol{y}_t$, normalized over the two possible actions,

$$\begin{aligned} p_{\mathtt{anm}} = p(a_1|\boldsymbol{y}_{1:t}, \mu^{\mathtt{LTd}}_{t|t}) &= \frac{f(\boldsymbol{y}_t, \mu^{\mathtt{LTd}}_{t|t}, a_1|\boldsymbol{y}_{1:t-1})}{f(\boldsymbol{y}_t, \mu^{\mathtt{LTd}}_{t|t}|\boldsymbol{y}_{1:t-1})} \\ &= \frac{f(\boldsymbol{y}_t, \mu^{\mathtt{LTd}}_{t|t}, a_1|\boldsymbol{y}_{1:t-1})}{\sum_i f(\boldsymbol{y}_t, \mu^{\mathtt{LTd}}_{t|t}, a_i|\boldsymbol{y}_{1:t-1})}. \end{aligned}$$

A $p_{\mathtt{anm}}$ value above a specified threshold triggers an alarm for a potential anomaly. The method for estimating that threshold is further described in Appendix C.

Alternative anomaly detection methods, such as supervised classification and reinforcement learning, could also be incorporated as a decision layer for SSMs. However, any method relying on a drift residual is affected by the delayed adaptation described in Section 3.1.2, where residuals falsely appear to be stationary when the magnitude of the anomaly is small, in comparison to them. In a supervised classification setting, such residuals labeled as abnormal are mixed in the training set with residuals from stationary regimes that are labeled as normal. Assigning two different labels to similar stationary residual inputs leads to ambiguous predictions. In a reinforcement learning framework, agents encounter stationary states both during stationary regimes and at the early stage of an anomaly development when the magnitude is still lower than the residuals. This leads them to associate two conflicting optimal actions with the same state. Therefore, both approaches are prone to false alarms and unreliable detection during the adaptation period. In contrast, using the anomaly probability proposed in this section provides a simple tunable threshold that can distinguish normal from abnormal states.

4.3. Intervention model

The intervention defined in equations 5-10 assigns the drift wrongfully captured by the residual to the baseline component of the base SSM. This intervention is typically insufficient for allowing the base SSM to fully adapt to a new trend, because the residual only captures a part of the drift caused by anomalies.

In this section, we propose using a Bayesian neural network (BNN) to predict a target intervention that compensates for the baseline drift caused by anomalies. The BNN is trained using tractable approximate Gaussian inference (TAGI) method introduced in Section 3.2. The training set, $\mathcal{D} = \{\tau_1, \tau_2, ...\}$, is collected through the simulation of synthetic anomalies, where we use the base SSM to generate a set of $\mathtt{N}$ synthetic stationary time series of length $\mathtt{L}$, and apply each time an anomaly with random magnitude and starting time, yielding $\{\boldsymbol{y}^{\mathtt{anm1}}_{1:\mathtt{L}}, \boldsymbol{y}^{\mathtt{anm2}}_{1:\mathtt{L}}, ..., \boldsymbol{y}^{\mathtt{anm\ N}}_{1:\mathtt{L}}\}$. These time series are then processed with the base and drift SSMs in order to generate the training labels of input states and output labels for the BNN.

With $\mathtt{q} \in \mathbb{Z}_{\geq 0}$ being a non negative integer, we start to collect training samples from the time step $2^{\mathtt{q}} + 1$ in all synthetic time series with anomalies. For each time step t in the $\mathtt{n}^{\text{th}}$ time series, $\boldsymbol{y}^{\mathtt{anm\ n}}_{2^{\mathtt{q}}+1:\mathtt{L}}$, one training sample τ_j is collected as an input–target pair. The input consists of the current expected value for $X^{\mathtt{LTd}}_{t|t}$ together with its estimates over a logarithmically spaced time history, according to

$$\begin{aligned} \text{input}_j = \mu^{\mathtt{LTd}}_{\boldsymbol{\mathcal{H}}} = \{&\mu^{\mathtt{LTd}}_{t|t},\ \mu^{\mathtt{LTd}}_{t-1|t-1},\ \mu^{\mathtt{LTd}}_{t-2|t-2}, \\ &\mu^{\mathtt{LTd}}_{t-4|t-4}, ...,\ \mu^{\mathtt{LTd}}_{t-2^{\mathtt{q}}|t-2^{\mathtt{q}}}\}. \end{aligned}$$

The target that need to be learnt by the BNN is defined as

$$\text{target}_j = \begin{cases} \{0, 0, 0\}, & \text{if } t < t^{\mathtt{anm}}_0 \\ \{x^{\mathtt{LL,anm}}_t, x^{\mathtt{LT,anm}}_t, d^{\mathtt{anm}}_t\}, & \text{otherwise} \end{cases},$$

where $x^{\mathtt{LL,anm}}_t$ and $x^{\mathtt{LT,anm}}_t$ are the level and trend of the synthetic anomaly ($\mathtt{anm\ n}$) at the time t, $t^{\mathtt{anm}}_0$ denotes the anomaly starting time, and $d^{\mathtt{anm}}_t = t - t^{\mathtt{anm}}_0$ is the anomaly duration between the current time and its introduction.

The BNN trained on the dataset $\mathcal{D}$ can generalize to unforeseen anomalies. For future data, it takes as input the historical expected values ($\mu^{\mathtt{LTd}}_{\boldsymbol{\mathcal{H}}}$) for the trend of the drift model, and predicts the Gaussian distributions for the anomaly features, $\{\hat{X}^{\mathtt{LL,anm}}_t, \hat{X}^{\mathtt{LT,anm}}_t, \hat{D}^{\mathtt{anm}}_t\}$, where

$$\begin{aligned} \hat{X}^{\mathtt{LL,anm}}_t &\sim \mathcal{N}(\hat{\mu}^{\mathtt{LL,anm}}_t, (\hat{\sigma}^{\mathtt{LL,anm}}_t)^2), \\ \hat{X}^{\mathtt{LT,anm}}_t &\sim \mathcal{N}(\hat{\mu}^{\mathtt{LT,anm}}_t, (\hat{\sigma}^{\mathtt{LT,anm}}_t)^2), \text{ and} \\ \hat{D}^{\mathtt{anm}}_t &\sim \mathcal{N}(\hat{\mu}^{D,\mathtt{anm}}_t, (\hat{\sigma}^{D,\mathtt{anm}}_t)^2). \end{aligned}$$

The predictions $\hat{X}^{\mathtt{LL,anm}}_t$ and $\hat{X}^{\mathtt{LT,anm}}_t$ could directly be used to intervene on the posterior hidden states from $t-1$ as in equations 5-10, allowing the Kalman filter to adapt to a new regime from time t. Nonetheless, we can obtain a more effective intervention by applying the intervention at the time of the anomaly initiation, as quantified by $\hat{\mu}^{D,\mathtt{anm}}_t$, so that by the current time t, the SSMs have already adapted and can better capture the new baselines. In this case, the SSMs processing is sent back to the anomaly starting time predicted by the network, $\hat{t}^{\mathtt{anm}}_0 = t - \lfloor \hat{\mu}^{D,\mathtt{anm}}_t \rceil$, with the hidden state posteriors modified by the intervention according to

$$\begin{aligned} \mu^{\mathtt{LT}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0} &= \mu^{\mathtt{LT}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0} + \hat{\mu}^{\mathtt{LT,anm}}_t, \\ (\sigma^{\mathtt{LT}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0})^2 &= (\sigma^{\mathtt{LT}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0})^2 + (\hat{\sigma}^{\mathtt{LT,anm}}_t)^2, \\ (\sigma^{\mathtt{LL}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0})^2 &= (\sigma^{\mathtt{LL}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0})^2 + (\hat{\sigma}^{\mathtt{LL,anm}}_t)^2, \\ \mu^{\mathtt{LLd}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0} &= 0, \\ \mu^{\mathtt{LTd}}_{\hat{t}^{\mathtt{anm}}_0|\hat{t}^{\mathtt{anm}}_0} &= \mu^{\mathtt{SR}}. \end{aligned}$$

With the methods proposed in this section, the Kalman filter at time step t is processed as follow: First, we quantify the probability of anomaly $p_{\mathtt{anm}}$ as described in Section 4.2. If $p_{\mathtt{anm}}$ is below a specified threshold, the base and drift SSMs proceed with the Kalman filter following Sections 3.1 and 4.1. Otherwise, we apply the intervention $\hat{\mu}_t^{D,\mathtt{anm}}$ steps prior to the detection time, according to the BNN predictions obtained following Section 4.3. The SSMs processing is then sent back to the time step $\hat{t}_0^{\mathtt{anm}}$ with the intervention applied on the posteriors, and onto which the Kalman filter is reapplied recursively until the current time t corresponding to the anomaly detection time.

5. Experiments

In this section, we compare the Residuals Stationarity Intervention (RSI) method that has been implemented in the open-source library Canari [39], with three other anomaly detection approaches, i.e., Switching Kalman Filter (SKF), Matrix Profile, and Prophet. Section 5.1 presents the comparison on synthetic time series, and their performance on real time series is evaluated through a verification and validation process in Section 5.2.

5.1. Synthetic time series

This section compares the performance of four detection methods on synthetic time series. Section 5.1.1 presents two synthetic time series with different residual complexities, along with the anomaly generation procedure, and Section 5.1.2 probabilistically compares the anomaly detection performance of the four methods.

5.1.1. Description of dataset

Using SSMs, we generate two stationary synthetic time series of length $\mathtt{T} = 12$ years with weekly intervals, as illustrated in Figure 4. Both SSMs share identical baseline and periodic components, while differing only in their residual components. The baseline component consists of a zero-valued level and trend throughout the entire time series. The periodic component is a superposition of two Fourier forms with periods of 52 and 13 weeks. Their residuals are generated with autoregressive ($\mathtt{AR}$) components having the same process error but different autocorrelation coefficients: $\phi^{\mathtt{AR}} = 0.5$ for the simple time series and $\phi^{\mathtt{AR}} = 0.9$ for the complex one. The autocorrelation coefficient determines the complexity of the time series, as larger values of $\phi^{\mathtt{AR}}$ amplify short-term fluctuations, which may be misinterpreted by detection models as baseline shifts, thereby causing false alarms. The model matrices for the two generative SSMs are provided in Appendix B.

The two SSMs are then used to generate test sets consisting in different stationary time series realizations, to which synthetic anomalies are added for the probabilistic quantification of anomaly detectability. Each test series contains a single synthetic anomaly that shifts the zero-valued trend to a new constant magnitude, selected from $\{0.01, 0.02, 0.03, ..., 0.1, 0.2, 0.3, ..., 1\}$ unit/year in its normalized space. For each magnitude, we test 10 different anomalies, each at a randomly set starting time between 40% of the series length and three years before the end. The

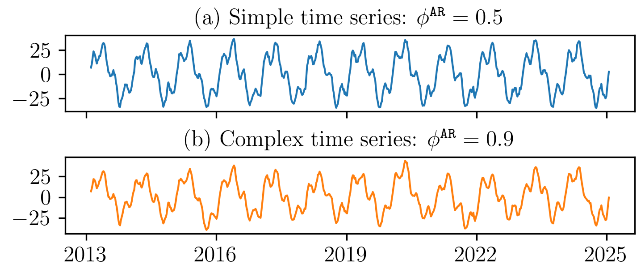


Figure 4: A simple and a complex synthetic time series generated using 2 different SSMs. The baselines and periodic components for them are the same. The residuals of the simple time series are generated with $\phi^{\mathtt{AR}} = 0.5$, and those of the complex one are generated with $\phi^{\mathtt{AR}} = 0.9$.

first 40% of the data is not involved in the evaluation and kept identical across all time series. Figure 5 illustrates one anomaly example for each magnitude, with its starting time marked by a dashed line.

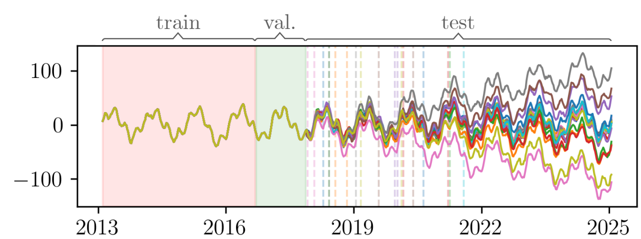


Figure 5: Examples of synthetic anomalies in different time series realizations generated by the SSM with $\phi^{\mathtt{AR}} = 0.9$. The starting times of anomalies with varying magnitudes are indicated by the vertical dashed lines. The trainning, validation (val.), and test sets are marked on top of the time series.

5.1.2. Results

The four detection models (i.e., RSI, SKF, Matrix Profile, and Prophet) are configured following the experiment setup specified in Appendix C. For SKF and RSI, the SSMs rely on the training and validation sets as illustrated in Figure 5. For Matrix Profile, the stationary subsequence spaces are constructed using the combined training and validation sets. For Prophet, the model is retrained at each time step using all past observations.

The anomaly thresholds for all four methods are calibrated on the entire synthetic stationary time series presented in Figure 4. The false alarm rates on the test set are reported in Table 1. All detection models exhibit false alarm rates close to zero on the simple time series owing to the low variability of its residuals. In contrast, only Prophet maintains a zero false alarm rate on the complex time series, while RSI and SKF each trigger approximately one false alarm every 10 years. Matrix Profile, however, produces more than eight false alarms/10 years due to its limited generalizability to unseen data outside the stationary subsequence space and its poor adaptation following detection.

We exclude time series with false alarms from the comparison of detection time (Δ_t), detection probability

Table 1
False alarms rate on the test set (#/10 years) of four detection models on two datasets.

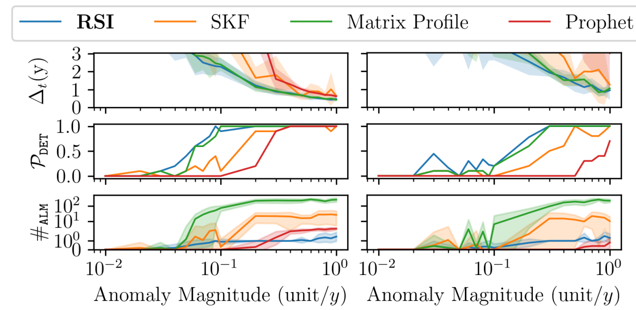
Model	Simple time series	Complex time series
RSI	0.02	1.26
SKF	0.17	1.03
Matrix Profile	0	8.65
Prophet	0	0

($\mathcal{P}_{\mathtt{DET}}$), and the number of alarms ($\#_{\mathtt{ALM}}$) that are presented in Figure 6. Detection time is defined as the difference between the first time when an alarm is triggered and the anomaly starting time, with a maximum detection time set to three years. The expected value, along with the standard deviation for the detection time for each anomaly magnitude is plotted in the first row of Figure 6. The detection probability reported in the second row is computed as the ratio between the number of anomalies detected within 3 years and the total number of test series under a same anomaly magnitude. We also count the number of alarms triggered on each time series to evaluate how well each model adapts to the new trend induced by anomalies. Because each time series in the test sets contains only one anomaly, all the alarms after the first are regarded as redundant ones required for the adaptation to new regimes. The expected values and standard deviations for the number of alarms are reported in the third row.

All four models detect small-magnitude anomalies with shorter detection time in the simple time series compared to the complex one. Our method and Matrix Profile achieve the best performance in both detection time and detection probability across both simple and complex time series, followed by SKF and Prophet. Matrix Profile poorly adapts to new regimes, continuously triggering alarms after the first detection. This occurs because the level of all the subsequences $\boldsymbol{s}_t$ deviates from the zero-valued level in the stationary subsequence space $S^{\mathtt{MP}}$ once an anomaly is introduced. While normalizing subsequences can reduce repeated alarms in Matrix Profile, it severely degrades the performance in terms of detection time and probability, performing even worse than Prophet. In terms of computational cost, Prophet is the most demanding, requiring approximately 4 hours to complete the test, followed by RSI at 18 minutes, and both SKF and Matrix Profile at about 1 minute each, on an M1 CPU with 32 GB of memory. Overall, our method demonstrates a superior anomaly detectability and adaptability to new regimes compared to the other methods on this synthetic dataset, with a moderate computational cost.

5.2. Real time series

In this section, we compare the performance of four detection methods on real time series collected on dams in Canada. Section 5.2.1 presents the real time series and describes the detrending process, in which baseline shifts are removed to obtain stationary time series for the purpose of comparison. Section 5.2.2 probabilistically compares the four methods by quantifying the detectability of anomalies on the detrended time series onto which we add synthetic anomalies.



(a) Simple time series (b) Complex time series

Figure 6: Comparison of four detection models on simple and complex time series with anomalies with respect to detection time (Δ_t), detection probability ($\mathcal{P}_{\mathtt{DET}}$), and number of alarms ($\#_{\mathtt{ALM}}$) across a range of anomaly magnitudes. The anomaly magnitude and the number of alarms ($\#_{\mathtt{ALM}}$) are plotted on a logarithmic scale. Solid lines correspond to the expected value, and the shaded regions indicate the $\pm 1\sigma$ confidence region.

Finally, Section 5.2.3 compares the methods on the original time series.

5.2.1. Description of dataset and detrending process

Figure 7 presents ten normalized time series of crack opening and displacement from a dam, with lengths ranging from 9 to 18 years, weekly data acquisition frequency, and containing missing data. All time series exhibit baseline shifts that may result from factors such as changes in dam behaviour, interventions on the structure, or sensor malfunctions.

Because no true anomaly labels are available, except for the time series #3 in 2010 and 2016 [13], potential anomalies are instead inferred from the changepoints, corresponding to the intersections of manually fitted linear baselines (black dashed lines), which are highlighted as grey shaded areas. Although manually identifying changepoints is feasible when having decades of data at our disposition, it is a notoriously difficult task to do online as data is collected. Among them, time series #1, 2, 4, 5, 9 and 10 show clear baseline shifts, whereas time series #3, 6, 7, and 8 display more subtle changes.

We cannot systematically evaluate the anomaly detectability of the different methods using the raw data presented in Figure 7 because the precise location and the exact number of anomalies present in the real time series are uncertain. To address this limitation, we first compare the methods on detrended time series onto which we add synthetic anomalies. For that purpose, the baselines of each time series is removed using the Prophet library, thereby correcting the time series to a stationary range centered around zero, as shown in Figure 8. The detrended time series do not present clear changepoint like the raw ones, yet short-term reversible fluctuations and variations in periodic patterns remain because of the inherent complexity of real data. Note that because the detrending process relies on the Prophet method, the latter is advantaged by the pre-processing.

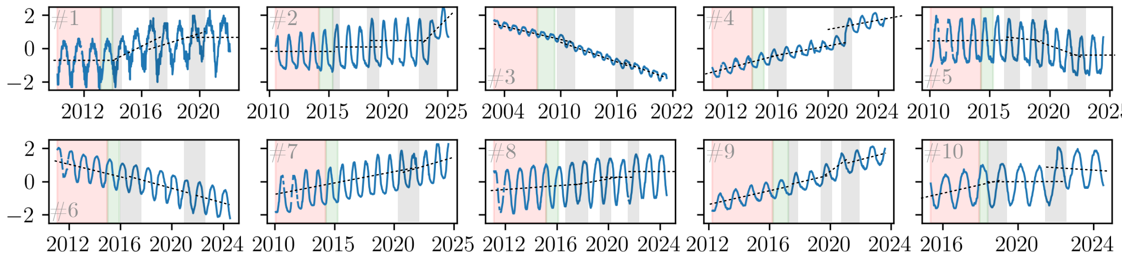


Figure 7: Normalized weekly time series of crack opening and displacement on a dam, with lengths ranging from 9 to 18 years and containing missing data. The red, green, and grey shaded regions correspond to the training sets, validation sets, and manually identified anomalies corresponding to changepoints in the baseline responses.

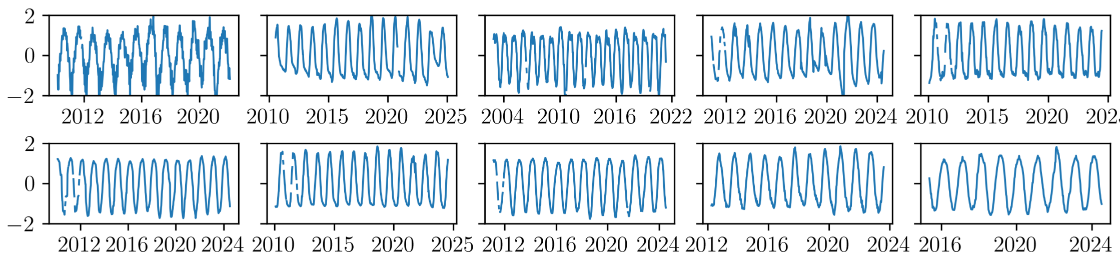


Figure 8: Detrended stationary time series obtained by removing the baselines estimated by Prophet from the real time series.

5.2.2. Verification on detrended time series

The anomaly detection performance of the four models is first verified using synthetic anomalies generated according to the procedure described in Section 5.1.1, that are added to the detrended series. The different methods are configured following the procedure presented in Appendix C, with training and validation sets fixed to 30% and 10% of the detrended time series. The stationary subsequence spaces in Matrix Profile are constructed using the combined training and validation sets. The anomaly thresholds for the different methods are calibrated on the entire detrended time series to prevent false alarms. This ensures a) that all methods operate under the shared assumption that the time series are stationary, and b) that they are maximally sensitive to detect deviations introduced by the synthetic anomalies. Without enforcing the absence of false alarms, it would be impossible to determine whether a detection corresponds to an anomaly or to short-term variations inherent to the real time series.

Figure 9 presents the evaluation metrics (detection time, detection probability and number of alarms) averaged across all ten detrended series. Our method, RSI, achieves the highest detection probability with shorter detection delay for all anomaly magnitudes, followed by Matrix Profile, while Prophet and SKF exhibit a weaker detectability. Although in Section 5.1.2, Matrix Profile matched the detection performance of RSI on synthetic time series, it now underperforms RSI on real data in both detection and adaptation in new regimes, showing longer detection delays, lower detection probabilities, and a large number of redundant alarms. This limitation arises because real time series contain more complex periodic patterns that are difficult to capture with a simple stationary subsequence space. While SKF and Prophet show similar performance in terms of detection time and probability, SKF has the lowest computational cost among the four methods, whereas Prophet is the most computationally demanding. The relative running times of the four methods are consistent with those observed on synthetic time series, as reported in Section 5.1.2.

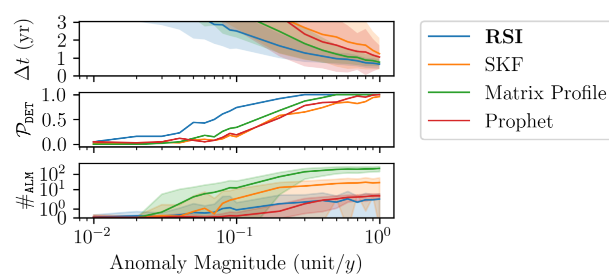


Figure 9: Comparison of four detection models on detrended time series with anomalies, evaluated in terms of detection time (Δ_t), detection probability ($\mathcal{P}_{\text{DET}}$), and number of alarms ($\#_{\text{ALM}}$). The results are averaged across all the abnormal time series derived from the 10 detrended time series.

5.2.3. Validation on real time series

We compare the anomaly detection performance of the four methods on the ten raw time series using the manually defined anomaly labels. The proportions of the training and validation sets, which are used by all four methods, are represented by red and green shaded area plotted in Figure 7, so that they each ends before the first anomaly occurs.

Following the experiment setup in Appendix C, the SSMs for SKF and RSI rely on the training and validation sets. For Matrix Profile, the stationary subsequence spaces are

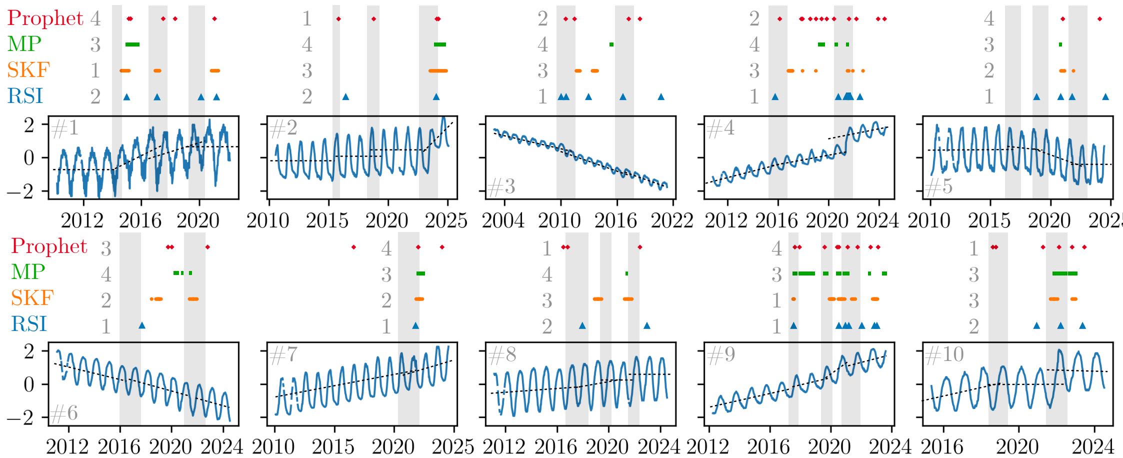


Figure 10: Comparison of four methods against manual identification (grey areas) on real time series. Grey numbers indicate the ranking of their first valid detection. The markers above the time series indicate the alarms triggered by different methods.

constructed using the training sets. All subsequences in Matrix Profile are normalized in this experiment to handle multiple anomalies in real data and to prevent anomaly scores from continuously increasing due to deviations in the levels. For Prophet, the model is retrained at each time step using all past observations. The anomaly thresholds for all four methods are calibrated on the combined training and validation sets.

Figure 10 compares the detections from the four methods within the test sets against the manual labels marked by grey regions. The methods are ranked for each time series according to their first detection, except for time series #7, where the earliest detection by Prophet is considered as a false alarm occurring within the stationary portion of the time series. The average ranking across all ten time series are 1.4, 2.3, 2.6, and 3.5 for RSI, SKF, Prophet, and Matrix Profile, respectively. The rankings on real time series are consistent with the experiments on synthetic and detrended time series from Section 5.1.2 and 5.2.2, validating that: 1) our method detects anomalies with shorter detection delays relative to the other approaches, and 2) SKF and Prophet exhibit comparable detectability. Matrix Profile performs worse than on synthetic and detrended series, largely because the subsequences must be normalized in time series with changing levels and multiple anomalies.

After the first detection, all four methods continue to trigger alarms indicating potential anomalies. RSI tend to provide the smallest number of redundant alarms, e.g., Prophet repeatedly triggers more alarms in time series #4 and #9. However, without true anomaly labels, it is difficult to determine whether these subsequent alarms correspond to genuine anomalies, model adaptation to a new regime, or false positives inherent to the method. The manual labels provided here serve only as a reference baseline and may be interpreted differently by other annotators. This is a limitation of working with real time series whereas an objective comparison is only feasible with experiments on detrended time series like it was done in Section 5.2.2.

6. Conclusions

This study introduces the RSI method that relies on an intervention for SSMs applied to non-stationary time series, enabling the detection of anomalies that shift the data from one stationary regime to another, and facilitating the adaptation to new regimes. The inevitable delay in detecting gradual drifts in SHM data, can be mitigated by applying the intervention at the anomaly's starting time predicted by the invention model. The proposed method outperforms the existing online anomaly detection approaches it was compared with on both synthetic and real time series, showing higher detection probabilities and shorter detection times across anomalies of varying magnitudes. Overall, the results demonstrate that RSI is an effective intervention model for detecting small-magnitude and slow-developing anomalies in infrastructure.

7. Acknowledgement

The authors acknowledge the contribution of Hydro-Québec who provided the datasets and who have funded this research, as well as funding from the Hydro-Québec Research Institute (IREQ) and Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] Q. Zhang, S. Sun, B. Yang, R. Wüchner, L. Pan, and H. Zhu. Real-time structural health monitoring system based on streaming data. *Smart Structures and Systems*, 28(2):275–287, 2021.

[2] C. Wang and G. Liu. From anomaly detection to classification with graph attention and transformer for multivariate time series. *Advanced Engineering Informatics*, 60:102357, 2024.

[3] A. Alshuhail, H. A. Mengash, M. H. Alanazi, M. K. Saeed, M. Ghaleb, M. Al Duhayyim, N. Alhebaishi, and A. Alzahrani. Tinyml-enabled structural health monitoring for real-time anomaly detection in civil infrastructure. *Alexandria Engineering Journal*, 129:1340–1348, 2025.

[4] H. Li, S. Zheng, Y. Shen, M. Han, R. Zhang, and H. Zhao. Hydro-steel structure digital twins: Application in structural health monitoring

and maintenance of large-scale reservoir. *Advanced Engineering Informatics*, 62:102922, 2024.

[5] C. Rainieri, G. Fabbrocino, and E. Cosenza. Near real-time tracking of dynamic properties for standalone structural health monitoring systems. *Mechanical Systems and Signal Processing*, 25(8):3010–3026, 2011.

[6] B. Xu, H. Zhang, C. Gu, Z. Chen, and H. Gu. Multi-target prediction and dynamic interpretation method for displacement of arch dam with cracks based on A-DSRSN and SHAP. *Advanced Engineering Informatics*, 66:103467, 2025.

[7] S. A. K. Fakhri, Z. Hamida, and J.-A. Goulet. Scalable probabilistic deterioration model based on visual inspections and structural attributes from large networks of bridges. *Advanced Engineering Informatics*, 64:103035, 2025.

[8] J. Paparrizos, Y. Kang, P. Boniol, R. S. Tsay, T. Palpanas, and M. J. Franklin. TSB-UAD: An end-to-end benchmark suite for univariate time-series anomaly detection. In *Proceedings of the VLDB Endowment*, 2022.

[9] Q. Meng and S. Zhu. Anomaly detection for construction vibration signals using unsupervised deep learning and cloud computing. *Advanced Engineering Informatics*, 55:101907, 2023.

[10] D. Jana, J. Patil, S. Herkal, S. Nagarajaiah, and L. Duenas-Osorio. CNN and convolutional autoencoder (CAE) based real-time sensor fault detection, localization, and correction. *Mechanical Systems and Signal Processing*, 169:108723, 2022.

[11] J. Zhang, J. Zhang, and Z. Wu. Long-short term memory network-based monitoring data anomaly detection of a long-span suspension bridge. *Sensors*, 22(16), 2022.

[12] G. Valdeyron, P. Losset, S. Dubroca, M. Mariko, and S. Bonelli. Assessment of Pont de Pierre settlements by Bayesian dynamic linear model (BDLM)–comparison with hydrostatic-season-time (HST) model and benefits for the pont de pierre reinforcement works monitoring. In *Proceedings of the 9ICEG, 9th International Congress on Environmental Geotechnics*, 2023.

[13] P. Côté, V. Roy, A. Meynadier, and B. Miquel. Applicability of the Bayesian dynamic linear model method in dam behaviour analysis and comparison with the HST method. In *Proceedings of Dam Safety 2022 - 39th Annual Conference*. Association of State Dam Safety Officials, 2022.

[14] J.-A. Goulet. Bayesian dynamic linear models for structural health monitoring. *Structural Control and Health Monitoring*, 24(12):1545–2263, 2017.

[15] B. Deka, L. H. Nguyen, S. Amiri, and J.-A. Goulet. The Gaussian multiplicative approximation for state-space models. *Structural Control and Health Monitoring*, 29(3):e2904, 2022.

[16] L. H. Nguyen, I. Gaudot, S. Khazaeli, and J.-A. Goulet. A kernel-based method for modeling non-harmonic periodic phenomena in Bayesian dynamic linear models. *Frontiers in Built Environment*, 5:8, 2019.

[17] L. H. Nguyen and J.-A. Goulet. Anomaly detection with the switching kalman filter for structural health monitoring. *Structural Control and Health Monitoring*, 25(4):e2136, 2018.

[18] Z. Xin and J.-A. Goulet. Enhancing structural anomaly detection using a bounded autoregressive component. *Mechanical Systems and Signal Processing*, 212:111279, 2024.

[19] R. P. Adams and D. J. C. MacKay. Bayesian online changepoint detection. Technical report, 2007.

[20] S. J. Taylor and B. Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[21] H. Zare Moayedi and M.A. Masnadi-Shirazi. Arima model for network traffic prediction and anomaly detection. In *2008 International Symposium on Information Technology*, volume 4, pages 1–6, 2008.

[22] M. Gu, J. Fei, and S. Sun. Online anomaly detection with sparse Gaussian processes. *Neurocomputing*, 403:383–399, 2020.

[23] M. Ma, L. Han, and C. Zhou. BTAD: A binary transformer deep neural network model for anomaly detection in multivariate time series data. *Advanced Engineering Informatics*, 56:101949, 2023.

[24] H. He, C. Fan, Y. Lei, B. Li, Z. Chen, and C. Zheng. An unsupervised ensemble learning method for real-time anomaly detections in variable refrigerant flow systems. *Advanced Engineering Informatics*, 68:103648, 2025.

[25] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, and E. 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)*, pages 1317–1322, 2016.

[26] Z. Lei, L. Zhu, Y. Fang, X. Li, and B. Liu. Anomaly detection of bridge health monitoring data based on kNN algorithm. *Journal of Intelligent & Fuzzy Systems*, 39(4):5243–5252, 2020.

[27] J. Singh and D. Singh. A comprehensive review of clustering techniques in artificial intelligence for knowledge discovery: Taxonomy, challenges, applications and future prospects. *Advanced Engineering Informatics*, 62:102799, 2024.

[28] V.-D. Vuong. *Analytically Tractable Bayesian Recurrent Neural Networks with Structural Health Monitoring Applications*. PhD thesis, Polytechnique Montréal, 2024.

[29] Y. Saatçi, R. D. Turner, and C. E. Rasmussen. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 927–934, 2010.

[30] S. Khazaeli, L. H. Nguyen, and J.-A. Goulet. Anomaly detection using state-space models and reinforcement learning. *Structural Control and Health Monitoring*, 28(6):e2720, 2021.

[31] S. Khazaeli and J.-A. Goulet. Damage detection for structural health monitoring using reinforcement and imitation learning. *Structure and Infrastructure Engineering*, pages 1–18, 2024.

[32] G. Welch and G. Bishop. An introduction to the Kalman filter. 1995.

[33] V. D. Vuong, L. H. Nguyen, and J.-A. Goulet. Coupling LSTM neural networks and state-space models through analytically tractable inference. *International Journal of Forecasting*, 41(1):128–140, 2025.

[34] B. Deka and J.-A. Goulet. Approximate Gaussian variance inference for state-space models. *International Journal of Adaptive Control and Signal Processing*, 37(11):2934–2962, 2023.

[35] J.-A. Goulet, L. H. Nguyen, and S. Amiri. Tractable approximate Gaussian inference for Bayesian neural networks. *Journal of Machine Learning Research*, 22(251):1–23, 2021.

[36] B. Deka, L. H. Nguyen, and J.-A. Goulet. Analytically tractable heteroscedastic uncertainty quantification in Bayesian neural networks for regression tasks. *Neurocomputing*, 572:127183, 2024.

[37] L. H. Nguyen and J.-A. Goulet. Analytically tractable hidden-states inference in Bayesian neural networks. *Journal of Machine Learning Research*, 23:1–33, 2022.

[38] H. E. Rauch, C. T. Striebel, and F. Tung. Maximum likelihood estimates of linear dynamic systems. *AIAA journal*, 3(8):1445–1450, 1965.

[39] Bayes-Works. Canari. https://github.com/Bayes-Works/canari, 2025.

[40] Z. Wu, E. Liang, M. Luo, S. Mika, J. E. Gonzalez, and I. Stoica. RLlib flow: Distributed reinforcement learning is a dataflow problem. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

A. Appendix: Drift model matrices

The matrices defining the drift SSM ($\mathcal{M}^{\mathrm{d}}$) are given by

$$\boldsymbol{x}^{\mathrm{d}} = \begin{bmatrix} \boldsymbol{x}^{\mathtt{LT}} \\ \boldsymbol{x}^{\mathtt{AR}} \end{bmatrix},\ \mathbf{A}^{\mathrm{d}} = \begin{bmatrix} \mathbf{A}^{\mathtt{LT}} & 0 \\ 0 & \mathbf{A}^{\mathtt{AR}} \end{bmatrix},$$
$$\mathbf{Q}^{\mathrm{d}} = \begin{bmatrix} \mathbf{Q}^{\mathtt{LT}} & 0 \\ 0 & \mathbf{Q}^{\mathtt{AR}} \end{bmatrix},\ \mathbf{F}^{\mathrm{d}} = \left[\mathbf{F}^{\mathtt{LT}}, \mathbf{F}^{\mathtt{AR}}\right],$$

where the matrices for $\mathtt{LT}$ and $\mathtt{AR}$ are provided in Section 3.1.1. The parameters are $\sigma^{\mathtt{LT}} = 10^{-5}$, $\phi^{\mathtt{AR}}$, while $\sigma^{\mathtt{AR}}$ and Δt are set to the same values as in the base model.

B. Appendix: Model matrices for synthetic time series generation

The synthetic time series used in Section 5.1 is generated by two SSMs comprising a local trend (LT) component, periodic components with periods of 52 weeks (PD(52)) and 13 weeks (PD(13)), and an autoregressive (AR) component. The matrices defining the SSMs are

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}^{\mathtt{LT}} \\ \boldsymbol{x}^{\mathtt{PD(52)}} \\ \boldsymbol{x}^{\mathtt{PD(13)}} \\ \boldsymbol{x}^{\mathtt{AR}} \end{bmatrix}, \ \mathbf{A} = \begin{bmatrix} \mathbf{A}^{\mathtt{LT}} & 0 & 0 & 0 \\ 0 & \mathbf{A}^{\mathtt{PD(52)}} & 0 & 0 \\ 0 & 0 & \mathbf{A}^{\mathtt{PD(13)}} & 0 \\ 0 & 0 & 0 & \mathbf{A}^{\mathtt{AR}} \end{bmatrix},$$
$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}^{\mathtt{LT}} & 0 & 0 & 0 \\ 0 & \mathbf{Q}^{\mathtt{PD(52)}} & 0 & 0 \\ 0 & 0 & \mathbf{Q}^{\mathtt{PD(13)}} & 0 \\ 0 & 0 & 0 & \mathbf{Q}^{\mathtt{AR}} \end{bmatrix},$$
$$\mathbf{F} = \left[\mathbf{F}^{\mathtt{LT}}, \mathbf{F}^{\mathtt{PD(52)}}, \mathbf{F}^{\mathtt{PD(13)}}, \mathbf{F}^{\mathtt{AR}}\right],$$

where the matrices for LT and AR are provided in Section 3.1.1. The matrices for the PD(p) component, with p denoting the period, are defined by

$$\boldsymbol{x}^{\mathtt{PD(p)}} = \begin{bmatrix} x^{\mathtt{PD1}} \\ x^{\mathtt{PD2}} \end{bmatrix}, \ \mathbf{A}^{\mathtt{PD(p)}} = \begin{bmatrix} \cos\omega & \sin\omega \\ \sin\omega & \cos\omega \end{bmatrix},$$
$$\mathbf{Q}^{\mathtt{PD(p)}} = \left(\sigma_w^{\mathtt{PD(p)}}\right)^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{F}^{\mathtt{PD(p)}} = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

where $\omega = \frac{2\pi\cdot\Delta t}{\mathtt{p}}$.

The two SSMs share the same parameters except for $\phi^{\mathtt{AR}}$, which is 0.5 for the simple synthetic time series, and 0.9 for the complex one. The shared parameters are $\sigma^{\mathtt{LT}} = 0$, $\sigma^{\mathtt{PD(52)}} = 0$, $\sigma^{\mathtt{PD(13)}} = 0$, $\sigma^{\mathtt{AR}} = 2$, and $\Delta t = 1$ week. The vectors of expected values for the initial hidden states $\boldsymbol{X}_0^{\mathtt{LT}}$, $\boldsymbol{X}_0^{\mathtt{PD(52)}}$, $\boldsymbol{X}_0^{\mathtt{PD(13)}}$, and $\boldsymbol{X}_0^{\mathtt{AR}}$ are

$$\mathbb{E}_0^{\mathtt{LT}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \ \mathbb{E}_0^{\mathtt{PD(52)}} = \begin{bmatrix} 0 \\ 25 \end{bmatrix},$$
$$\mathbb{E}_0^{\mathtt{PD(13)}} = \begin{bmatrix} 0 \\ 10 \end{bmatrix}, \ \mathbb{E}_0^{\mathtt{AR}} = -0.0621.$$

Their initial covariance matrices are

$$\mathbb{V}_0^{\mathtt{LT}} = \begin{bmatrix} 10^{-12} & 0 \\ 0 & 10^{-12} \end{bmatrix},$$
$$\mathbb{V}_0^{\mathtt{PD(52)}} = \begin{bmatrix} 10^{-12} & 0 \\ 0 & 10^{-12} \end{bmatrix},$$
$$\mathbb{V}_0^{\mathtt{PD(13)}} = \begin{bmatrix} 10^{-12} & 0 \\ 0 & 10^{-12} \end{bmatrix},$$
$$\mathbb{V}_0^{\mathtt{AR}} = 6.36 \times 10^{-5}.$$

C. Appendix: Experiment setup

The four detection models compared in Section 5 (i.e., RSI, SKF, Matrix Profile, and Prophet) quantify an anomaly score η_t^{anm} at each time step t as new observations arrive, and trigger an alarm when η_t^{anm} exceeds a threshold $\hat{\eta}^{\mathrm{anm}}$. The definition of η_t^{anm} differs across methods:

1) As described in Section 4.2 and Section 2, both RSI and SKF quantify the probability of anomaly, $\eta_t^{\mathrm{anm}} = p_t^{\mathrm{anm}} \in [0, 1]$ using a stationary SSM obtained using training and validation sets.

2) Matrix Profile measures the distance between the most recent subsequence of length n, $\boldsymbol{s}_t = \boldsymbol{y}_{t-n:t}$, and a stationary subsequence space $\mathcal{S}^{\mathtt{MP}} = \{\boldsymbol{s}_{n+1}, \boldsymbol{s}_{n+2}, ..., \boldsymbol{s}_{\bar{\mathtt{T}}}\}$, where $\bar{\mathtt{T}} < \mathtt{T}$, and $\mathtt{T}$ denotes the total length of a time series. This distance is used as the anomaly score, i.e., $\eta_t^{\mathrm{anm}} = d(\boldsymbol{s}_t, \mathcal{S}^{\mathtt{MP}})$.

3) Unlike the other three methods, which rely only on part of the time series to construct a stationary underlying model (RSI and SKF) or a stationary representation (Matrix Profile), Prophet must be retrained on all past data. When operated in an online manner, it requires retraining each time a new observation arrives. It fits a regression model with piecewise linear trends to the data up to the current time t, and triggers an alarm when deviations in the estimated trend ($\delta^{\mathtt{trend}}$) exceed a threshold. The anomaly score in Prophet is therefore defined as the maximum trend deviation up to time t, i.e., $\eta_t^{\mathrm{anm}} = \max(\delta_{1:t}^{\mathtt{trend}})$.

The anomaly thresholds $\hat{\eta}^{\mathrm{anm}}$ for all four models are tuned on a time series subset $\boldsymbol{y}_{1:\hat{\mathtt{T}}}$. For RSI, we adopt for all analysis $\hat{\eta}^{\mathrm{anm}} = \max(\max(\eta_{1:\hat{\mathtt{T}}}^{\mathrm{anm}})\cdot 1.1,\ 0.1)$, with a minimum threshold of 0.1 to prevent detections arising from a near-zero anomaly score during the early phase of anomaly development, when the drift is too small to learn an effective intervention. For SKF, we use the Ray [40] optimization package to learn the optimal threshold during SSM training. If the optimized threshold produces false alarms anywhere on time series subset $\boldsymbol{y}_{1:\hat{\mathtt{T}}}$, we conduct a grid search starting from the optimized value, increasing by 0.1 each time within the interval (0, 1) until no false alarm is triggered. For Matrix Profile, we set $\hat{\eta}^{\mathrm{anm}} = \max(\eta_{1:\hat{\mathtt{T}}}^{\mathrm{anm}}) \cdot 1.1$ without a lower bound because its anomaly score lies in the positive real domain. Finally, for Prophet, we determine the threshold through a grid search starting from 0.1 and increasing in increments of 0.1 until no false alarm is triggered on the time series subset $\boldsymbol{y}_{1:\hat{\mathtt{T}}}$. The anomaly thresholds for all four methods across the time series in Section 5 are reported in Table B2.

Table B2
Thresholds of anomaly detection methods across synthetic, detrended, and real time series.

	Anomaly detection methods			
Time series	**RSI**	**SKF**	**Matrix Profile**	**Prophet**
synthetic simple	0.16	0.17	27.14	0.3
synthetic complex	0.1	0.26	56.93	1.1
detrended #1	0.11	0.09	0.26	0.2
detrended #2	0.1	0.12	1.45	1
detrended #3	0.1	0.09	2.84	0.4
detrended #4	0.1	0.42	2.73	0.4
detrended #5	0.1	0.08	2.16	0.5
detrended #6	0.1	0.14	4.25	0.4
detrended #7	0.14	0.15	5.34	0.5
detrended #8	0.1	0.1	0.14	0.5
detrended #9	0.16	0.77	1.44	0.3
detrended #10	0.1	0.1	0.34	0.2
real #1	0.1	0.27	3.37	0.1
real #2	0.1	0.17	1.09	0.4
real #3	0.1	0.02	1.74	0.8
real #4	0.1	0.41	2.27	0.3
real #5	0.28	0.17	1.93	0.3
real #6	0.1	0.02	1.25	0.3
real #7	0.25	0.05	1.17	0.7
real #8	0.1	0.09	1.01	0.4
real #9	0.31	0.07	0.85	0.4
real #10	0.1	0.14	1.13	0.7