Online Change Point Detection of Stochastic Differential Equations

Anonymous Authors[1]

Abstract

Online detection of the change points for evolving stochastic differential equations (SDEs) from streaming observations is of paramount importance yet highly challenging. This is attributed to the inherent stochasticity in the data, compounded by the limitations imposed in an online setting, wherein training is conducted at each time step utilizing solely limited collected data. In this study, we introduce a method that integrates the Sparse Identification of Nonlinear Stochastic Dynamics (SINSDy) technique with a modified two-phase Follow-The-Regularized-Leader (FTRL)-proximal algorithm. This framework enables extracting low-dimensional features that enhance the accurate detection of change points in evolving SDEs by online identifying the governing equations. Our proposed methodology, termed Online Sparse Identification of Nonlinear Stochastic Dynamics (Online-SINSDy), not only demonstrates competitive performance in the identification of SDEs but also surpasses existing online change point detection methods and exhibits broader applicability. We show the efficacy of our framework using the dataset produced by representative real-world systems, such as the collective dynamics of the cichlid fish, the rolling bearing vibration data, and the electroencephalogram dataset.

1. Introduction

Data-driven change point detection of complex dynamical systems has drawn increasing attention in various disciplines, including neuroscience (Cribben et al., 2013), epidemiology (Yu et al., 2013), economics (Bodenham & Adams, 2017), and ecology (Francesco-Ficetola & Denoël, 2009). The ability to accurately detect the critical transitions in data patterns contributes significantly to facilitating informed decision-making. The current mainstream change point detection methods can be broadly categorized into two types. One involves direct analysis of the data, while the other is based on identifying the underlying differential equations from data. Indeed, the data-driven identification of differential equations has emerged as a prominent topic across various domains (Brunton et al., 2016; Rudy et al., 2017; Boninsegna et al., 2018; Champion et al., 2019). Successfully identifying dynamical equations contributes to acquiring the mechanisms of underlying systems and assists in pinpointing change point locations.

Over the past few decades, various techniques have been proposed for the discovery of deterministic differential equations from data (Voss et al., 1999; Daniels & Nemenman, 2015; Lusch et al., 2018), with sparse identification of nonlinear dynamics (SINDy) (Brunton et al., 2016) emerging as one of the prominent methods. Given a set of observation data and a pre-selected library of basis functions, SINDy can automatically select the terms that best align with the underlying deterministic differential equations. However, in many real-world scenarios, complex systems are influenced by random noise, necessitating the use of stochastic differential equations (SDEs) for accurate modeling and analysis of these systems (Carpenter et al., 2020; Jhawar & Guttal, 2020; Jhawar et al., 2020). Therefore, several stochastic variants of SINDy-based methods have been introduced to unveil the underlying SDEs from data (Boninsegna et al., 2018; Wang et al., 2022; Huang et al., 2022), which utilize the Kramers-Moyal (KM) formulae (Risken & Risken, 1996) to estimate the drift and diffusion values and identify the drift and diffusion terms respectively. While effective in capturing the underlying SDEs, these methods are offline algorithms that require substantial amounts of data, rendering them unsuitable for streaming data. Recently, the development of O-SINDy addressed this limitation (Li et al., 2023b). It constructs a sparse regression problem at each time with only one sample and utilizes the follow-the-regularized-leader (FTRL)-proximal method (McMahan, 2011) to solve these sparse regression problems in an online fashion, allowing for sequential processing of samples in streaming data.

Besides, many methods and theories have been proposed to detect abrupt variations in time series data, which can be roughly divided into offline and online methods. The

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

offline algorithms analyze the complete dataset collectively, retrospectively examining the data to recognize when and where the change occurred (Aminikhanghahi & Cook, 2017; Truong et al., 2020; Yu et al., 2023). However, in real-world scenarios, what we often acquire is sequentially arriving streaming data, rendering these offline algorithms no longer applicable. Online change point detection algorithms can be further broadly categorized into methods based on statistical features and prediction errors. Bayesian Online Change-point Detection (BOCD) is a representative online statistical method that performs the detection by computing the probability distribution of the running length since the last change point (Adams & MacKay, 2007). Another typical category of statistical methods detects changes utilizing kernel statistics (Flynn & Yoo, 2019; Arlot et al., 2019; Wei & Xie, 2022; Ferrari et al., 2023; Li et al., 2023a) or the likelihood ratio of the probability distribution (Kawahara & Sugiyama, 2009; Liu et al., 2013) in sliding windows before and after the change point. On the other hand, O-SINDy and Temporal Change-point Detection (TCD) detect the occurrences of change points by measuring the sudden increase of the prediction loss (Hou et al., 2022). Specifically, O-SINDy calculates the predicted values by integrating the identified governing equations, while TCD generates prediction series by the Randomly Distributed Embedding framework (Ma et al., 2018). However, these prediction error-based methods are often not applicable to SDEs because we cannot precisely predict the specific state at the next moment. Additionally, many dynamical phenomena in real-world systems are **noise-induced** (Carpenter et al., 2020; Jhawar et al., 2020), rendering these methods designed for deterministic differential equations often ineffective.

Despite the methods mentioned above, to the best of our knowledge, there are currently limited approaches for simultaneous online identification and change point detection of SDEs. Therefore, in this article, we extend O-SINDy to the stochastic version by combining the stochastic-SINDy with a modified two-phase FTRL-proximal methodology. Additionally, we propose a novel method for detecting the change points in SDEs using the weight dynamics obtained during the identification process. Our proposal framework, named Online Sparse Identification of Nonlinear Stochastic Dynamics (Online-SINSDy), can simultaneously discover the governing SDEs and extract the low-dimensional (low-d) features that can be used to detect the change points from evolving SDEs in an online fashion. The advantages of the proposed framework are summarized as follows.

- Online-SINSDy can precisely identify the underlying SDEs using **limited streaming data** available during each training iteration. In contrast, most existing studies necessitate substantial data sets for concurrent training.
- The variations in weights acquired by Online-SINSDy during training serve as a natural indicator, enabling pinpointing the change points of the underlying SDEs. Its detection performance surpasses that of the baseline methods across numerous **real-world systems**.
- Online-SINSDy exhibits strong robustness to noise, achieving small detection errors and high detection accuracy in identifying system change points across various noise levels.
- Online-SINSDy can concurrently capture the changes in both the **deterministic** and **stochastic** parts of the underlying SDEs. Hence, the applicability of our method is more extensive.

2. Methods

Suppose we receive a few numbers of streaming time series from the same underlying system via a finite number of sensors. Let $\boldsymbol{x}^k(t_l) \in \mathbb{R}^d$ denote the d-d state of the kth time series at time t_l, where $k = 1, 2, \ldots, M$ represent different trajectories collected by different sensors, $t_1 < t_2 < \cdots < t_N$ are discretely sampled times. Since the time series arrive continuously as streams, it is assumed that due to the limited memory capacity, only data within a narrow time window of duration T_{tr} preceding the current time can be stored and utilized for training. Let $\boldsymbol{X}(t_l) \in \mathbb{R}^{MT_{\text{tr}} \times d}$ denote the state matrix, encompassing all available data applicable for training purposes at time t_l.

In this work, we consider dynamics driven by an Ito SDE:

$$\mathrm{d}\boldsymbol{x}(t) = \boldsymbol{f}[\boldsymbol{x}(t)]\mathrm{d}t + \boldsymbol{H}[\boldsymbol{x}(t)]\mathrm{d}\boldsymbol{W}(t),$$

where $\boldsymbol{f}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d$ is the drift function, $\boldsymbol{H}(\cdot) : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is the diffusion function, and $\boldsymbol{W}(t)$ is a d-d standard Brown motion. We further define the positive-definite covariance matrix of the diffusion as $\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{H}^\top(\boldsymbol{x})\boldsymbol{H}(\boldsymbol{x})$.

2.1. Regression problems based on the KM formulae

In most scenarios, we cannot know the specific forms of the drift and diffusion terms. However, based on the KM formulae, the drift and diffusion values can be estimated as

$$\begin{aligned} f_i(\boldsymbol{x}) &= \lim_{\Delta t\to 0} \mathbb{E}\left\{ \frac{1}{\Delta t}[x_i(t+\Delta t) - x_i(t)] | \boldsymbol{x}(t) = \boldsymbol{x} \right\}, \\ G_{ij}(\boldsymbol{x}) &= \lim_{\Delta t\to 0} \mathbb{E}\left\{ \frac{1}{\Delta t}[x_i(t+\Delta t) - x_i(t)][x_j(t+\Delta t) \right. \\ &\qquad \left. - x_j(t)] | \boldsymbol{x}(t) = \boldsymbol{x} \right\}, \end{aligned} \tag{1}$$

where x_i represents the i-th component of the d-d state of the time series, $f_i(\cdot) : \mathbb{R}^d \to \mathbb{R}$ denotes the i-th component of the d-d vector-valued drift function $\boldsymbol{f}(\cdot)$ and $G_{ij}(\cdot)$:

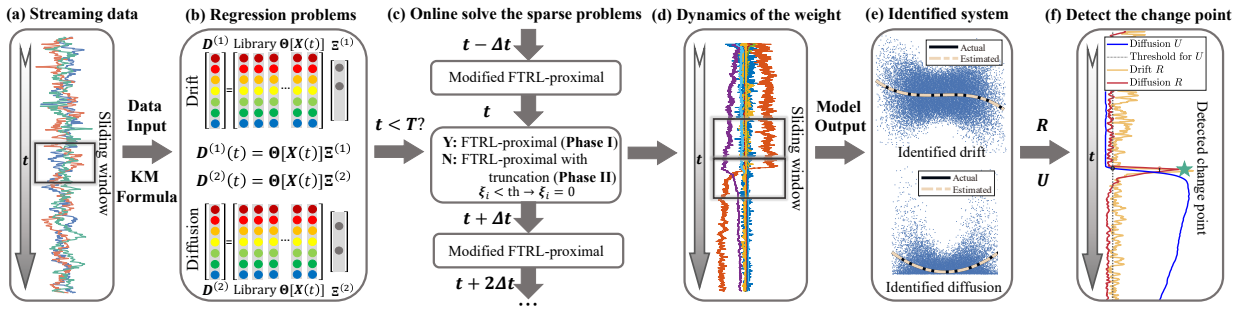

Figure 1. Schematic diagram for Online-SINSDy.

$\mathbb{R}^d \to \mathbb{R}$ refers to the component located in the i-th row and j-th column of the matrix-valued diffusion function $\boldsymbol{G}(\cdot)$.

A conventional technique for estimating KM coefficients is the binning method (Boninsegna et al., 2018). We first partition the observed states into Q equally spaced intervals. Then the states within each bin are used to estimate the drift and diffusion values at the midpoint of each interval. The details of the binning method can be found in Appendix A. However, in scenarios where there is a limited amount of training data, we can directly utilize all available data to calculate the linear and quadratic variation at each point as

$$\begin{aligned}\boldsymbol{D}_i^{(1)}(t_l) &= \{\frac{1}{\Delta t}[\boldsymbol{x}_i^q(t_{l-p+1}) - \boldsymbol{x}_i^q(t_{l-p})]\}_{\substack{q=1,2,\dots,M\\ p=1,2,\dots,T_{\mathrm{tr}}}},\\ \boldsymbol{D}_{ij}^{(2)}(t_l) &= \{\frac{1}{\Delta t}[\boldsymbol{x}_i^q(t_{l-p+1}) - \boldsymbol{x}_i^q(t_{l-p})]\\ &\quad [\boldsymbol{x}_j^q(t_{l-p+1}) - \boldsymbol{x}_j^q(t_{l-p})]\}_{\substack{q=1,2,\dots,M\\ p=1,2,\dots,T_{\mathrm{tr}}}},\end{aligned} \tag{2}$$

where $\boldsymbol{D}_i^{(1)}$ represents the estimated values of the i-th component of the d-d drift vectors at different states, $\boldsymbol{D}_{ij}^{(2)}$ refers to the estimated values of the component located in the i-th row and j-th column of the $(d \times d)$-d matrix-valued diffusion matrices at different states, and $x_i^q(t_l)$ denotes the i-th component of the d-d state at time t_l in the q-th trajectory. In this paper, the time stamps are assumed to be uniformly sampled and the time interval is denoted as Δt, i.e. $\forall l : t_{l+1} - t_l = \Delta t$. We construct the regression problems without the need for bin averaging (Huang et al., 2022).

We then construct a candidate library $\boldsymbol{\Theta}[\boldsymbol{X}(t_l)] \in \mathbb{R}^{MT_{\mathrm{tr}} \times P}$ consisting of P pre-selected basis functions. For example, the P-order polynomial basis

$$\boldsymbol{\Theta}[\boldsymbol{X}(t_l)] = \begin{bmatrix} | & | & \cdots & | \\ 1 & \boldsymbol{X}(t_l) & \cdots & \boldsymbol{X}^P(t_l) \\ | & | & \cdots & | \end{bmatrix}^\top,$$

where the high-order terms of $\boldsymbol{X}$ include all mixed-terms, for example, when $P = 2$ and $d = 2$, the library is $\boldsymbol{\Theta}(\boldsymbol{X}) = [1, x_1.x_2, x_1^2, x_1x_2, x_2^2]$. Without loss of generality, we focus on the case where the observed states contain only one space dimension ($d = 1$), allowing us to omit all subscripts in Eq. (2). The method we propose below can be easily generalized to the multivariate case.

Suppose the basis functions in the library encompass all the function terms present in the underlying SDE. We can determine the weight of basis functions corresponding to the drift $\boldsymbol{\Xi}^{(1)}$ and the diffusion $\boldsymbol{\Xi}^{(2)}$ at t_l by solving the following regression problems:

$$\begin{aligned}\boldsymbol{D}^{(1)}(t_l) &= \boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(1)},\\ \boldsymbol{D}^{(2)}(t_l) &= \boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(2)}.\end{aligned} \tag{3}$$

2.2. Modified two-phase FTRL-proximal algorithm

An essential observation lies in the fact that numerous practical biophysical SDEs exhibit sparsity in their drift and diffusion expressions, characterized by the presence of only a few terms within the space spanned by basis functions. Additionally, when dealing with streaming data, it is imperative to iteratively optimize the weight vectors over time. Here, we propose a modified two-phase FTRL-proximal algorithm, capable of solving the regression problems online with high accuracy while ensuring the sparsity of the solution.

We define the regression loss of Eq. (3) at time t_l as

$$\begin{aligned}L_{t_l}^{(1)}(\boldsymbol{\Xi}^{(1)}) &= \frac{1}{MT_w}\|\boldsymbol{D}^{(1)}(t_l) - \boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(1)}\|_2^2,\\ L_{t_l}^{(2)}(\boldsymbol{\Xi}^{(2)}) &= \frac{1}{MT_w}\|\boldsymbol{D}^{(2)}(t_l) - \boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(2)}\|_2^2,\end{aligned}$$

where $\boldsymbol{D}^{(1)}(t_l)$ and $\boldsymbol{D}^{(2)}(t_l)$ are the MT_{tr}-d vectors of the estimated target values of the drift and diffusion terms at different states calculated by Eq. (2).

The FTRL-proximal algorithm constructs the following unconstrained optimization problems to address the online

sparse regression tasks presented in Eq. (3):

$$\begin{aligned}\boldsymbol{\Xi}(t_{l+1}) = \arg\min_{\boldsymbol{\Xi}} \sum_{i=1}^{l} \nabla L_{t_i}[\boldsymbol{\Xi}(t_i)]^\top \boldsymbol{\Xi} + \lambda_1 \|\boldsymbol{\Xi}\|_1 \\ + \frac{1}{2}\lambda_2 \|\boldsymbol{\Xi}\|_2^2 + \frac{1}{2}\sum_{i=1}^{l} \sigma_i \|\boldsymbol{\Xi} - \boldsymbol{\Xi}(t_i)\|_2^2,\end{aligned} \tag{4}$$

where $\boldsymbol{\Xi}(t_l)$ can denotes either the weight vector of drift $\boldsymbol{\Xi}^{(1)}$ or the weight vector of diffusion $\boldsymbol{\Xi}^{(2)}$ at time t_l of the training phase, $\lambda_1 > 0$ and $\lambda_2 > 0$ are the coefficients of the L_1 and L_2 regularization, whereas $\|\boldsymbol{\Xi} - \boldsymbol{\Xi}(t_i)\|_2^2$ denotes the *proximal* term that center additional regularization at the current weight, which can prevent dramatic change of the weight vector during the training process, $\sigma_i > 0$ serves as the learning rate. In fact, the joint action of the first term and the *proximal* term is equivalent to training the weight vectors using online gradient descent method (McMahan, 2011). We provide a detailed explanation of this part in Appendix E.

Note that we can obtain the following closed-form solution of Eq. (4):

$$\xi_j(t_{l+1}) = \begin{cases} 0, & \text{if } |\zeta_j(t_l)| < \lambda_1, \\ -\frac{\zeta_j(t_l) - \mathrm{sgn}[\zeta_j(t_l)]\lambda_1}{\lambda_2 + \sum_{i=1}^{l}\sigma_i}, & \text{if } |\zeta_j(t_l)| \geq \lambda_1, \end{cases}$$

where $\xi_j(t_l)$ denotes the jth component of $\boldsymbol{\Xi}(t_l)$, $\zeta_j(t_l)$ represents the jth component of $\boldsymbol{\zeta}(t_l) = \sum_{i=1}^{l}\{\nabla L_{t_i}[\boldsymbol{\Xi}(t_i)] - \sigma_i \boldsymbol{\Xi}(t_i)\}$, and $\mathrm{sgn}(\cdot)$ is the sign function. The detailed derivation process is provided in Appendix B.

While the above algorithm incorporates regularization terms to promote sparsity, the learned function expressions often still contain some minor non-zero erroneous terms. Previous approaches such as O-SINDy address this issue by applying a truncation operation after the entire training process, forcing the parameters below a threshold to zero. However, for the task of identifying SDEs, we observe that weak sparsity induced by the regularization terms in Eq. (4) can lead to convergence towards a false surrogate function expression, resulting in the identification task failure.

To overcome this challenge, we propose a two-phase training method. In Phase I, we update the weight vectors based on the original FTRL-proximal algorithm. In Phase II, a truncation operation is performed after each weight update, any scalar component below a pre-selected threshold is set to zero. These components are then excluded from further updates during the subsequent training process. This approach ensures the sparsity of the parameters and disrupts the local stable state of the weight vectors induced by interference terms, facilitating convergence to the authentic function expressions for the drift and diffusion terms.

2.3. Change point detection based on the weight vectors

During the identification process, we dynamically adjust the weight vectors using streaming data, thereby obtaining the optimal solution to the optimization problem Eq. (4) at each moment. During the training process, if the underlying SDE remains unchanged, the weight vectors will converge after a transition period and show minor fluctuations around their true values. However, when the dynamics experience specific changes at a particular time step, the weight vectors will undergo more substantial variations and gradually converge toward a new stable state corresponding to the altered SDE. Thus, detecting change points can be achieved by identifying abrupt alterations in the weight vectors.

Nevertheless, the occurrence of rare events in the evolution of SDEs can significantly impact the single-step changes of the weight vector, leading to erroneous change point detection. To address these problems, a sliding window of length T_{cpd} is introduced. Change points are detected by quantifying the variation of the average weight vectors between two adjacent time windows. Specifically, we first calculate the average weight vector in $[t_{l-T_{\mathrm{cpd}}}, t_l)$:

$$\overline{\boldsymbol{\Xi}}(t_{l-T_{\mathrm{cpd}}}, t_l) = \frac{1}{T_{\mathrm{cpd}}} \sum_{i=l-T_{\mathrm{cpd}}}^{l-1} \boldsymbol{\Xi}(t_i).$$

Then, we define the relative change intensity between two consecutive average weight vectors at the time t_l as

$$R(t_l) = \frac{\|\overline{\boldsymbol{\Xi}}(t_{l-T_{\mathrm{cpd}}}, t_l) - \overline{\boldsymbol{\Xi}}(t_l, t_{l+T_{\mathrm{cpd}}})\|_2}{\|\overline{\boldsymbol{\Xi}}(t_{l-T_{\mathrm{cpd}}}, t_l)\|_2 + \|\overline{\boldsymbol{\Xi}}(t_l, t_{l+T_{\mathrm{cpd}}})\|_2},$$

which serves as an indicator for identifying the change points. If the SDE remains unchanged, R will fluctuate around a relatively low value. However, when a change point falls within the sliding window, R will experience a significant increase. As the sliding window moves past the change point, R will initially ascend and subsequently descend, exhibiting an unimodal pattern. When the intersection of the two adjacent sliding windows is close to the change point, the change intensity of the average weight vector between the preceding and subsequent windows will be maximized and the value of R will reach its peak. Consequently, we can pinpoint the most likely change point locations by identifying the maximum value of R in the unimodal pattern.

During the online detection process, we still need a method to automatically detect when the indicator starts to exhibit significant changes, alerting us to pay attention to the potential occurrence of a change point during that period. Here, we employ the cumulative sum (CUSUM) control chart to detect the moment when there is an incremental change in the mean of the indicator (Page, 1954). We denote this moment as t_{CU}, and the indicator value at this moment as

R_{CU}. Starting from t_{CU}, we record the process of the indicator values rising and then falling until it returns to R_{CU} after a short time. We identify the time corresponding to the maximum value of the indicator during this period as the determined change point moment. Given a sequence of the indicator $R(t_1), R(t_2), \cdots, R(t_n)$, we first estimate the initial average m_R and standard deviation σ_R from the first k samples of the indicator. Then we can recursively define the upper cumulative process sums using:

$$U(t_l) = \begin{cases} 0, & \text{if } l = 1, \\ \max[0, U(t_{l-1}) + R(t_l) & \\ \quad -m_R - \frac{1}{2}h\sigma_R], & \text{if } l > 1, \end{cases} \tag{5}$$

where h denotes the number of standard deviations from the target mean that makes a shift detectable. The indicator violates the CUSUM criterion for the first time at t_{CU} if it obeys $U(t_{\mathrm{CU}}) > \gamma\sigma_R$ and $U(t_l) \leq \gamma\sigma_R (\forall t_l < t_{\mathrm{CU}})$, where the control limit γ is an artificially selected threshold.

Fig. 1 illustrates the complete schematic diagram of Online-SINSDy. The pseudocode of our method is provided in Appendix C for detailed reference.

3. Results

In this section, we validate the effectiveness of our Online-SINSDy algorithm for both identification and change point detection tasks using various synthetic data and real-world complex systems with randomness.

For the identification task, we validate the outstanding performance of Online-SINSDy on a synthetic double-well system, a recorded dataset of collective dynamics of the cichlid fish, and a rolling bearing vibration dataset. The experimental results demonstrate that our method can accurately identify the underlying SDEs even in real-world scenarios with significant observation noise and missing data. Due to limitations in space, we present the system identification results in Appendix D.

For the change point detection task, to validate the effectiveness of our method, we extensively test it to detect the splicing points of two synthetic systems with identical steady-state distribution, the moment when the group size of the cichlid fish experiences a change, the occurrence of the bearing fault, and the occurrence of epilepsy. To evaluate the robustness of our method, we also introduce varying levels of Gaussian random noise into the observational data.

To further evaluate the effectiveness and robustness of our method, we compare it with several baseline methods. Specifically, we compare our model with the following online change point detection methods: (a) Bayesian Online Changepoint Detection (BOCD); (b) Online Kernel Cumulative Sum (KCUSUM) (Flynn & Yoo, 2019); (c) Online Density Ratio Estimation (DRE) (Kawahara & Sugiyama, 2009); (d) No-prior-knowledge Exponential Weighted Moving Average algorithm (NEWMA) (Keriven et al., 2020). Table 1 demonstrates the diverse metrics that different methods utilize for accomplishing the detection task.

3.1. Detect the splicing points of synthetic systems

First, we consider splicing trajectories of two different synthetic systems whose dynamics are in the form of the following two SDEs:

$$\mathrm{d}x = (x - x^3)\mathrm{d}t + \sqrt{(2 + 2x^2)}\mathrm{d}W, \tag{6}$$

$$\mathrm{d}x = -x\mathrm{d}t + \sqrt{2}\mathrm{d}W. \tag{7}$$

Although these two SDEs are completely different, the steady-state distribution of x in Eq. (6) and Eq. (7) is **identical** (Nabeel et al., 2022), as is shown by the black dashed line in Fig. 2(c). Thus, it is quite challenging to detect the time points when the underlying SDE translates from one to another according solely to the statistical features.

Here we generate $M = 100$ trajectories, each containing $N_{\mathrm{tr}} = 1 \times 10^5$ recorded data points with the time interval $\Delta t = 0.01$. The first half of the time series is generated by Eq. (6), whereas the second half follows Eq. (7). The library is set as $\mathbf{\Theta}(x) = [1, x, \ldots, x^9]$. During the training process, we take the sliding window length $T_{\mathrm{tr}} = 10$ with stride step $\Delta T = 10$. Phase II begins when $t > 200$ with the threshold for the drift and diffusion takes $\mathrm{th}^{(1)} = 0.1$ and $\mathrm{th}^{(2)} = 0.01$ respectively. The sliding window for change point detection is set as $T_{\mathrm{cpd}} = 100$. To calculate the upper CUSUM, in all following experiments, we utilize the first 1000 recorded indicators in Phase II to estimate the average m_R and standard deviation σ_R and set $h = 1$ in Eq. (5).

For the sake of convenience, we utilize detection accuracy and detection error to assess the performance of our method in change-point detection tasks. The detection error ε refers to the time difference between the actual occurrence time of a change point T_{true} and the time at which we detect the occurrence of that change point T_{detect}, i.e. $\varepsilon = |T_{\mathrm{true}} - T_{\mathrm{detect}}|$. The detection accuracy pertains to the probability of the algorithm identifying change points with the detection error below an acceptable threshold $\varepsilon_{\mathrm{th}}$. In real-world online change point detection tasks, the objective is to detect the change points promptly following their occurrence. If the detection error ε is too large, it often loses its significance. Hence, we establish a maximum allowable threshold for the error and the detection with error below this threshold is deemed accurate, i.e. $\varepsilon < \varepsilon_{\mathrm{th}}$. We note that many articles directly employ detection delay (the positive part of the stopping time minus the true change-point time) to measure the effectiveness of the detection. In Appendix F, we also include statistical figures corresponding to the detection delay under different noise levels.

Table 1. Comparison of different change point detection methods with diverse metrics they rely on for accomplishing the detection task.

Method	Statistics	Prediction	Identification (Deterministic)	Identification (Stochastic)
O-SINDy (Li et al., 2023b)	×	✓	✓	×
TCD (Hou et al., 2022)	×	✓	×	×
RC-TCD (Li et al., 2023a)	×	✓	×	×
BOCD (Adams & MacKay, 2007)	✓	×	×	×
DRE (Kawahara & Sugiyama, 2009)	✓	×	×	×
NEWMA (Keriven et al., 2020)	✓	×	×	×
KCUSUM (Flynn & Yoo, 2019)	✓	×	×	×
Online-SINSDy (ours)	×	×	✓	✓

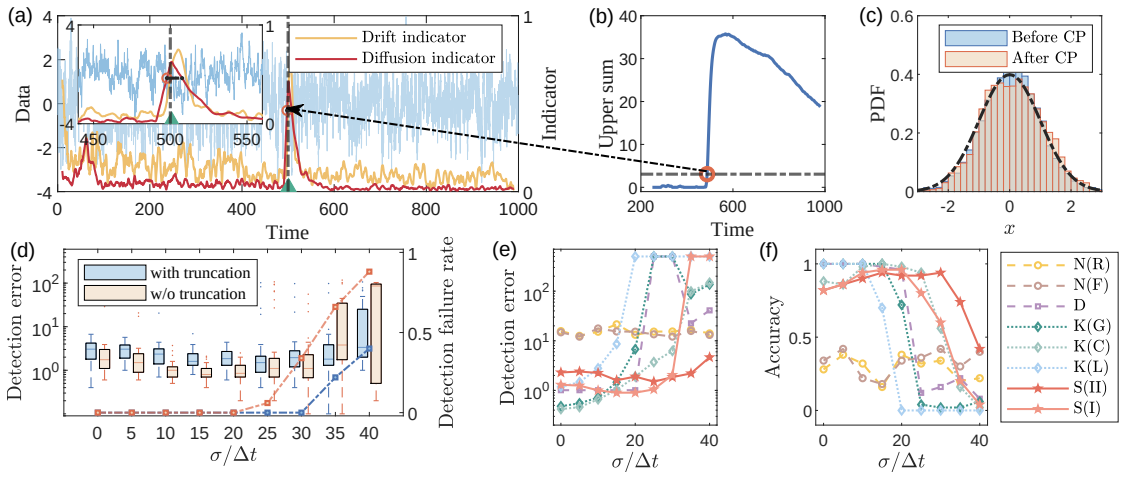

Figure 2. Change point detection results of the synthetic systems with identical steady-state distribution. (a) The streaming data and Online-SINSDy indicators. (b) The upper CUSUM indicator with the control limit $\gamma = 500$. (c) Two separate statistical histograms plotted using data before and after the splicing point, with a black dashed line representing their shared probability density. (d) Box plot of detection errors of Online-SINSDy under different levels of noise. (e-f) The detection errors and accuracy under different noise levels obtained by Online-SINSDy with (S(II)) and without (S(I)) truncation in Phase II, NEWMA with Random Fourier Features (N(R)) and Fastfood approximation (N(F)), DRE (D), KCUSUM with Gaussian (K(G)), Cauchy (K(C)) and Laplace (K(L)) kernels.

Fig. 2(a-b) illustrates the relative change intensity indicator R and the CUSUM indicator U during the training process. As shown in Fig. 2(d), our framework can sensitively detect the change points under varying levels of the noise standard deviation σ. For relatively low noise intensities, Online-SINSDy can accurately detect the existence of the change points and determine their positions with small detection errors, regardless of whether truncation is performed in Phase II. As the noise intensity increases, Online-SINSDy with truncation demonstrates remarkable robustness to the noise, while the algorithm without truncation rapidly loses its detection capability, particularly when $\sigma > 20\Delta t$.

To comprehensively assess the efficacy of our change point detection framework, we conduct a comparative analysis, pitting Online-SINSDy against various baseline methods across diverse noise levels. Each method is subjected to 50 repetitions under each noise intensity, and the median within each group is selected to represent its detection error. Additionally, we define detection accuracy as the proportion of experiments where the detection error was less than 10.

Due to the absence of notable changes in the probability distribution of the data before and after the change point, BOCD fails to detect the presence of the change points. Illustrated in Fig. 2(e-d), KCUSUM with distinct kernels and DRE exhibit smaller detection errors and higher detection accuracy under low noise intensities. However, their robustness to noise is poor, swiftly diminishing their detection capability as noise levels rise. Conversely, NEWMA maintains relatively stable detection errors and accuracy as noise intensity increases. Nonetheless, their detection errors consistently remain at a high level, and the detection accuracy persistently hovers at a low level. In comparison, Online-SINSDy showcases robustness to noise in terms of both detection errors and accuracy, surpassing other baseline algorithms, particularly in scenarios characterized by heightened noise intensity.

3.2. Detect the change of the fish populations

Our framework can also detect the change points of **real-world systems** with randomness, which is quite challenging because of the strong observation noise, the inherent dynamical noise, and the existence of missing data in real scenarios. Here we focus on the dynamics of collective alignment in groups of the cichlid fish *Etroplus suratensis* with different group sizes. Previous work has confirmed that the schooling of fish, which means highly polarized and coherent motion, is a **noise-induced** effect, and the mesoscopic dynamics can be fitted as an SDE (Jhawar et al., 2020):

$$\mathrm{d}\boldsymbol{m} = -\gamma_1 \boldsymbol{m}\mathrm{d}t + \left[\frac{\gamma_2(1-|\boldsymbol{m}|^2)+\gamma_1}{N_f}\right]^{1/2} \mathbb{1}\cdot \mathrm{d}\boldsymbol{W}, \tag{8}$$

where $\gamma_1 \approx -0.1$ and $\gamma_2 \approx 4.0$ are constant parameters, N_f denotes the group size, $\mathbb{1}$ is the 2-d identity matrix, and $\boldsymbol{m} = [m_x, m_y]^\top$ is the 2-d group polarization calculated by

$$\boldsymbol{m}(t) = \frac{1}{N_f}\sum_{i=1}^{N_f} \frac{\boldsymbol{v}_i(t)}{|\boldsymbol{v}_i(t)|}, \tag{9}$$

which represents the collective fish motion direction and the degree of alignment. $\boldsymbol{v}_i(t)$ in Eq. (9) is the 2-d velocity of the ith fish at time t.

Eq. (8) indicates that the diverse dynamical behaviors exhibited by fish with different group sizes are induced by varying intensities of intrinsic noise. Consequently, we can utilize the data of $\boldsymbol{m}(t)$ to discern instances when there are changes in the group size. The dataset was constructed by concatenating the trajectories of $\boldsymbol{m}(t)$ corresponding to $N_f = 15$ and $N_f = 60$, as is shown in Fig. 3(a). The total length of the concatenated data is 4800s with the sampling interval $\Delta t = 0.12$s. The library is set as $\boldsymbol{\Theta}(x) = [1, x, \ldots, x^9]$. During the training process, we take the window length $T_{\mathrm{tr}} = 100$ with the stride step $\Delta T = 10$. Phase II begins at $t = 960$s with the truncation threshold $\mathrm{th}^{(1)} = \mathrm{th}^{(2)} = 0.01$. Notably, all data associated with the splicing points is excluded during the training process. The sliding window for change point detection is set as $T_{\mathrm{cpd}} = 40$.

We find that the dynamics associated with the drift indicators obtained through Online-SINSDy do not manifest a discernible alteration after the change point. This observation suggests that the change in the population does not influence the drift term, aligning with Eq. (8). Consequently, we turn to the indicators corresponding to the two diagonal elements of the diffusion matrix, denoted as G_{xx} and G_{yy}, to identify the change points. Illustrated in Fig. 3(d-e), both indicators promptly identify the moment when changes occur in the population, with the detection error and accuracy displaying robustness across varying noise levels. However, eliminating the truncation in Phase II leads to a notable increase in both detection error and failure rate when the noise intensity $\sigma > 0.5\Delta t$.

According to the results in Fig. 3(b-c), it is challenging for conventional methods to detect the changes in the population. We find that BOCD, DRE, and KCUSUM with Cauchy kernel are unable to identify the changes. NEWMA and KCUSUM with Gaussian and Laplace kernels take a considerable amount of time after the changes have occurred to recognize the variations in the system. In contrast, our approach exhibits significantly lower detection errors and substantially higher detection accuracy compared to the baseline models across diverse noise intensities. Here, we define the detection accuracy as the proportion with a detection error less than 120s in 50 repetitions.

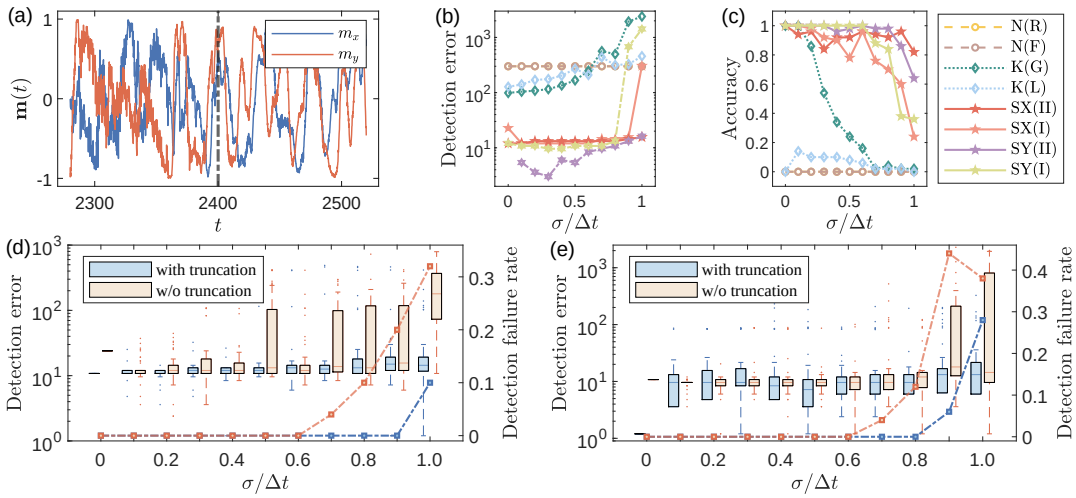

Figure 3. Change point detection results of the fish schooling dataset. (a) Concatenated trajectories of $N_f = 15$ and $N_f = 60$, with 2400s as the splicing point. (b-c) The detection errors and accuracy under different noise levels obtained by different methods. SX and SY represent the Online-SINSDy detection results corresponding to G_{xx} and G_{yy} respectively. (d-e) Box plot of detection errors using G_{xx} indicators (d) and G_{yy} indicators (e) respectively.

3.3. Detect the occurrence of the bearing fault

In manufacturing systems, a bearing fault can lead to a reduction in machine life, a decline in the quality of workpieces, and pose safety risks. Consequently, the diagnosis of bearing faults has emerged as a prominent subject within the engineering community (Safizadeh & Latifi, 2014; Shao et al., 2015; Yuan et al., 2020). Here, we employ Online-SINSDy to identify the occurrence of bearing faults. We generate the dataset by concatenating the trajectory of the normal bearing and that of the fault bearing with a fault diameter of 14 mils at five randomly selected time points to simulate scenarios where the bearings experience faults at different operating conditions. Each trajectory contains around 7×10^5 recorded data with time interval $\Delta t = 1/48,000$s. The data comes from the Case Western Reserve University bearing dataset.

Here, the library is set as $\mathbf{\Theta}(x) = [1, x, x^2, x^3, e^x, e^{2x}, e^{3x}]$. We take the training window length $T_{\text{tr}} = 1,000$ with the stride step $\Delta T = 100$. Phase II begins after observing 20% of the total data with the truncation threshold $\text{th}^{(1)} = 0.1$ and $\text{th}^{(2)} = 0.01$. The sliding window for change point detection is set as $T_{\text{cpd}} = 200$.

During the detection process, we select the point that is detected earlier between the drift and the diffusion indicators as the identified change point. The results are shown in Fig. 4, our method achieves the minimum detection errors compared to the baseline models in various scenarios, except for Case 3, where the detection error of NEWMA with Fourier random features (0.029s) slightly outperforms our approach (0.033s). This indicates that our method can more accurately pinpoint the occurrence of bearing faults, enabling corresponding countermeasures.

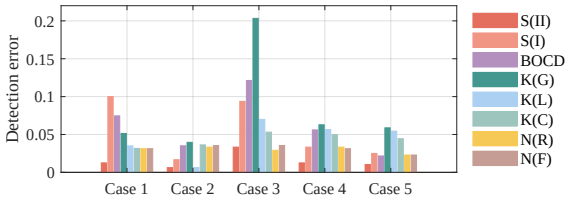

Figure 4. Detection performances in the bearing data.

3.4. Detect the occurrence of the epilepsy

Individuals afflicted with epilepsy experience recurrent seizures that occur at unpredictable times. Detecting the epileptic seizures utilizing the streaming electroencephalogram (EEG) data enables timely intervention, thereby mitigating potentially severe consequences for the patients. Here, we detect the occurrence of epilepsy utilizing the pediatric EEG data (Shoeb, 2009). Most of the recorded EEG signals contain 23 channels with the sampling frequency $\frac{1}{\Delta t} = 256$Hz.

To assess the efficacy of our approach in this task, we randomly select five patients and employ their EEG data to pinpoint the specific times of epileptic seizures. To make full use of data, we apply our method to detect change points for each of the 23 channels and select the time corresponding to the earliest detected change point as the predicted time for the onset of epileptic seizures. The library is set as $\mathbf{\Theta}(x) = [1, x, \ldots, x^9]$. During training, we take the time window $T_{\text{tr}} = 1,000$ with the stride step $\Delta T = 100$. After

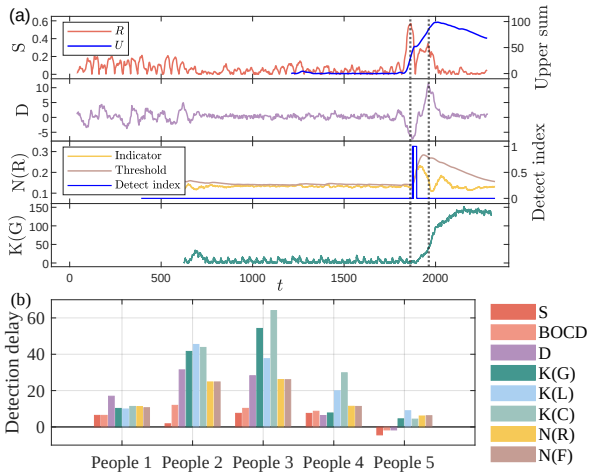

Figure 5. Detection performances in EEG data. (a) Different detection indicators for People 1, including Online-SINSDy, DRE, NEWMA, and KCUSUM. The onset and offset positions of epilepsy are indicated by the black dashed lines. (b) Response time delays using different methods in the data of 5 patients.

observing 20% of the total recorded data, we perform the truncation with $\mathrm{th}^{(1)} = \mathrm{th}^{(2)} = 0.001$. The sliding window for detection is set as $T_{\mathrm{cpd}} = 100$.

Indicators of different methods for People 1 are depicted in Fig. 5(a). The first figure illustrates that our indicators consistently maintain a low level when epileptic seizures are not occurring. However, at the onset and conclusion of epileptic seizures, the indicators experience a significant increase, resulting in a distinctive bimodal pattern. The two peaks precisely align with the onset and conclusion times of epileptic seizures. In comparison, DRE can also identify both the onset and conclusion time points, but with a greater detection delay. BOCD, KCUSUM, and NEWMA can only detect the onset point of epileptic seizures.

As illustrated in Fig. 5(b), our framework shows better detection performance. For People 1-4, the detection delay of our method is lower than that of the baseline models. For People 5, we observe that Online-SINSDy, BOCD, and DRE can provide early warnings before the epileptic seizures, while other methods can only detect changes after the onset of seizures. Furthermore, our approach provides earlier warnings of epilepsy compared with BOCD and DRE.

4. Discussion

In summary, this paper introduces a novel framework, Online-SINSDy, designed for the online change point detection of SDEs. Specifically, Online-SINSDy identifies the drift and diffusion terms of SDEs in an online fashion and utilizes the variations in the identified weight vectors at each moment as indicators to achieve precise online detection of change points. In comparison to conventional methods relying on the identification of deterministic differential equations, our approach marks a pioneering focus on the inherent dynamical noise within complex systems. We model the system as an SDE, thereby facilitating the concurrent detection of variations in both the deterministic and stochastic parts. The experimental results demonstrate that our framework has an exceptional performance in both identification and change point detection tasks. Notably, Online-SINSDy consistently outperforms benchmark models, exhibiting lower detection errors and higher accuracy across various real-world systems.

It is noteworthy that our method does not necessitate a substantial alteration in any statistic of the system. Hence, it exhibits broader applicability compared to statistically-based methods. Additionally, we acknowledge that certain changes observed in real systems can be induced by noise, as exemplified by the collective dynamics of the cichlid fish. In such scenarios, prediction error-based methods assuming deterministic differential equations, such as O-SINDy and TCD, fail to detect the occurrence of these changes. In contrast, our method can concurrently capture the changes in both the deterministic and stochastic parts of the underlying SDE. This broader capability enhances its applicability compared to prediction error-based methods.

In conclusion, the Online-SINSDy framework demonstrates significant potential in the online identification and change point detection of SDEs. Experimental validations on a diverse set of real-world datasets substantiate the effectiveness, wide applicability, and excellent robustness of our approach.

5. Impact Statements

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Adams, R. P. and MacKay, D. J. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

Aminikhanghahi, S. and Cook, D. J. A survey of methods for time series change point detection. *Knowledge and Information Systems*, 51(2):339–367, 2017.

Arlot, S., Celisse, A., and Harchaoui, Z. A kernel multiple change-point algorithm via model selection. *Journal of Machine Learning Research*, 20(162), 2019.

Bodenham, D. A. and Adams, N. M. Continuous moni-

toring for changepoints in data streams using adaptive estimation. *Statistics and Computing*, 27:1257–1270, 2017.

Boninsegna, L., Nüske, F., and Clementi, C. Sparse learning of stochastic dynamical equations. *The Journal of Chemical Physics*, 148(24), 2018.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

Carpenter, S. R., Arani, B. M., Hanson, P. C., Scheffer, M., Stanley, E. H., and Van-Nes, E. Stochastic dynamics of cyanobacteria in long-term high-frequency observations of a eutrophic lake. *Limnology and Oceanography Letters*, 5(5):331–336, 2020.

Champion, K., Lusch, B., Kutz, J. N., and Brunton, S. L. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

Cribben, I., Wager, T. D., and Lindquist, M. A. Detecting functional connectivity change points for single-subject fmri data. *Frontiers in Computational Neuroscience*, 7: 143, 2013.

Daniels, B. C. and Nemenman, I. Automated adaptive inference of phenomenological dynamical models. *Nature Communications*, 6(1):8133, 2015.

Ferrari, A., Richard, C., Bourrier, A., and Bouchikhi, I. Online change-point detection with kernels. *Pattern Recognition*, 133:109022, 2023.

Flynn, T. and Yoo, S. Change detection with the kernel cumulative sum algorithm. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 6092–6099, 2019.

Francesco-Ficetola, G. and Denoël, M. Ecological thresholds: an assessment of methods to identify abrupt changes in species–habitat relationships. *Ecography*, 32(6):1075–1084, 2009.

Hou, J. W., Ma, H. F., He, D., Sun, J., Nie, Q., and Lin, W. Harvesting random embedding for high-frequency change-point detection in temporal complex systems. *National Science Review*, 9(4):nwab228, 2022.

Huang, Y., Mabrouk, Y., Gompper, G., and Sabass, B. Sparse inference and active learning of stochastic differential equations from data. *Scientific Reports*, 12(1): 21691, 2022.

Jhawar, J. and Guttal, V. Noise-induced effects in collective dynamics and inferring local interactions from data. *Philosophical Transactions of the Royal Society B*, 375 (1807):20190381, 2020.

Jhawar, J., Morris, R. G., Amith-Kumar, U. R., Danny-Raj, M., Rogers, T., Rajendran, H., and Guttal, V. Noise-induced schooling of fish. *Nature Physics*, 16(4):488–493, 2020.

Kawahara, Y. and Sugiyama, M. Change-point detection in time-series data by direct density-ratio estimation. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 389–400. Society for Industrial and Applied Mathematics, 2009.

Keriven, N., Garreau, D., and Poli, I. Newma: a new method for scalable model-free online change-point detection. *IEEE Transactions on Signal Processing*, 68:3515–3528, 2020.

Li, X., Zhu, Q., Zhao, C., Qian, X., Zhang, X., Duan, X., and Lin, W. Tipping point detection using reservoir computing. *Research*, 6:0174, 2023a.

Li, Y., Wu, K., and Liu, J. Discover governing differential equations from evolving systems. *Physical Review Research*, 5(2):023126, 2023b.

Liu, S., Yamada, M., Collier, N., and Sugiyama, M. Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–8, 2013.

Lusch, B., Kutz, J. N., and Brunton, S. L. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, 2018.

Ma, H., Leng, S., Aihara, K., Lin, W., and Chen, L. Randomly distributed embedding making short-term high-dimensional data predictable. *Proceedings of the National Academy of Sciences*, 115(43):E9994–E10002, 2018.

McMahan, B. Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 525–533, Cambridge, MA, 2011.

McMahan, H., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., Chikkerur, S., Liu, D., Wattenberg, M., Hrafnkelsson, A., Boulos, T., and Kubica, J. Ad click prediction: a view from the trenches. In Langley, P. (ed.), *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1222–1230, Chicago, Illinois, USA, 2013.

Nabeel, A., A.Karichannavar, Palathingal, S., Jhawar, J., and Guttal, V. Pydaddy: A python package for discovering stochastic dynamical equations from timeseries data. *arXiv preprint arXiv:2205.02645*, 2022.

Page, E. S. Continuous inspection schemes. *Biometrika*, 41 (1/2):100–115, 1954.

Risken, H. and Risken, H. (eds.). *Fokker-Planck Equation*. Springer Berlin Heidelberg, 1996.

Rudy, S. H., Brunton, S. L., Proctor, J. L., and Kutz, J. N. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

Safizadeh, M. and Latifi, S. K. Using multi-sensor data fusion for vibration fault diagnosis of rolling element bearings by accelerometer and load cell. *Information Fusion*, 18:1–8, 2014.

Shao, H., Jiang, H., Zhang, X., and Niu, M. Rolling bearing fault diagnosis using an optimization deep belief network. *Measurement Science and Technology*, 26(11):115002, 2015.

Shoeb, A. H. *Application of machine learning to epileptic seizure onset detection and treatment*. PhD thesis, Massachusetts Institute of Technology, 2009.

Truong, C., Oudre, L., and Vayatis, N. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

Voss, H. U., Kolodner, P., Abel, M., and Kurths, J. Amplitude equations from spatiotemporal binary-fluid convection data. *Physical Review Letters*, 83(17):3422, 1999.

Wang, Y., Fang, H., Jin, J., Ma, G., He, X., Dai, X., Yue, Z., Cheng, C., Zhang, H., Pu, D., Wu, D., Yuan, Y., Goncalves, J., Kurths, J., and Ding, H. Data-driven discovery of stochastic differential equations. *Engineering*, 17:244–252, 2022.

Wei, S. and Xie, Y. Online kernel cusum for change-point detection. *arXiv preprint arXiv:2211.15070*, 2022.

Yu, X., Baron, M., and Choudhary, P. K. Change-point detection in binomial thinning processes, with applications in epidemiology. *Sequential Analysis*, 32(3):350–367, 2013.

Yu, Y., Padilla, O. H. M., Wang, D., and Rinaldo, A. A note on online change point detection. *Sequential Analysis*, 42 (4):438–471, 2023.

Yuan, Y., Ma, G., Cheng, C., Zhou, B., Zhao, H., Zhang, H. T., and Ding, H. A general end-to-end diagnosis framework for manufacturing systems. *National Science Review*, 7(2):418–429, 2020.

A. The binning method for estimating the drift and diffusion values

As mentioned in the main text, a conventional technique for estimating KM coefficients involves discretizing the observed data into spatial intervals, a method often referred to as binning (Boninsegna et al., 2018). In this study, we concentrate on the univariate scenario ($d = 1$). Initially, we partition the MT_{tr} observed states into Q equally spaced bins. The states within each bin are then averaged as

$$\boldsymbol{X}(t_l) = \{\boldsymbol{x}^i(t_{l-j+1})\}_{\substack{i=1,2,\ldots,M \\ j=1,2,\ldots,T_{\mathrm{tr}}}} \longmapsto \{\overline{x}_q, \omega_q\}_{q=1,2,\ldots,Q} = \overline{\boldsymbol{X}}(t_l),$$

where $\overline{x}_q$ indicates the averaged value of the data in the qth bin and ω_q indicates the fraction of the data in the qth bin. Then we can estimate the expectations in Eq. (1) at positions $\{\overline{x}_q\}_{q=1,2,\ldots,Q}$ by calculating the average linear and quadratic variation of the states in each bin, i,e.

$$\boldsymbol{D}^{(1)}(t_l) = \left\{ \frac{\boldsymbol{x}^i(t_{l-j+1}) - \boldsymbol{x}^i(t_{l-j})}{\Delta t} \right\}_{\substack{i=1,2,\ldots,M \\ j=1,2,\ldots,T_{\mathrm{tr}}}} \longmapsto \{\overline{f}_q\}_{q=1,2,\ldots,Q} = \overline{\boldsymbol{D}}^{(1)}(t_l),$$

$$\boldsymbol{D}^{(2)}(t_l) = \left\{ \frac{(\boldsymbol{x}^i(t_{l-j+1}) - \boldsymbol{x}^i(t_{l-j}))^2}{\Delta t} \right\}_{\substack{i=1,2,\ldots,M \\ j=1,2,\ldots,T_{\mathrm{tr}}}} \longmapsto \{\overline{G}_q\}_{q=1,2,\ldots,Q} = \overline{\boldsymbol{D}}^{(2)}(t_l),$$

where $\overline{f}_q$ and $\overline{G}_q$ indicate the averaged value of the linear and quadratic variation of the states in the qth bin respectively.

Next, we can utilize the bin-wise averaged data to construct a matrix based on the candidate library $\boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)] \in \mathbb{R}^{Q\times P}$ consisting of P pre-selected basis functions. For example, the P-order polynomial basis

$$\boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)] = \begin{bmatrix} | & | & \cdots & | \\ 1 & \overline{\boldsymbol{X}}(t_l) & \cdots & \overline{\boldsymbol{X}}^P(t_l) \\ | & | & \cdots & | \end{bmatrix}^\top,$$

where the higher polynomials are denoted as $\overline{\boldsymbol{X}}^2(t_l), \cdots, \overline{\boldsymbol{X}}^P(t_l)$. If the basis functions in the candidate library encompass all the function terms present in the underlying SDE, we can determine the weight vectors corresponding to the drift term $\boldsymbol{\Xi^{(1)}}$ and the diffusion term $\boldsymbol{\Xi^{(2)}}$ by solving the following regression problems:

$$\begin{aligned} \overline{\boldsymbol{D}}^{(1)}(t_l) &= \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(1)}, \\ \overline{\boldsymbol{D}}^{(2)}(t_l) &= \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(2)}. \end{aligned} \tag{10}$$

Due to the variations in the data distribution across different space intervals, data in some bins are inevitably quite sparse, which will lead to substantial errors when estimating the drift and diffusion values in these bins using the binning method. Thus, some works combine the binning method with the filtering process, which involves eliminating bins with data quantities falling below a predetermined threshold, to improve the identification of the SDEs.

In scenarios where there is a limited amount of training data, we can directly utilize all available data to construct the regression task as Eq. (3) without the need for bin averaging.

It is important to emphasize that although bin-wise averaging is not required, grouping the data into distinct bins remains essential for executing the filtering procedure. For all the experiments described in this paper, we excluded all bins with data volume below the average to ensure the robustness of our algorithm. Through experimental validation, we confirm that the solution derived from Eq. (3) yields identification accuracy comparable to, or even superior to, that achieved by the average binning method.

To employ the FTRL-proximal algorithm for solving Eq. (10), we define the regression loss of Eq. (10) at the time t_l as

$$\begin{aligned} L_{t_l}^{(1)}(\boldsymbol{\Xi}^{(1)}) &= \{\overline{\boldsymbol{D}}^{(1)}(t_l) - \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(1)}\}^\top \boldsymbol{\Omega} \{\overline{\boldsymbol{D}}^{(1)}(t_l) - \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(1)}\}, \\ L_{t_l}^{(2)}(\boldsymbol{\Xi}^{(2)}) &= \{\overline{\boldsymbol{D}}^{(2)}(t_l) - \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(2)}\}^\top \boldsymbol{\Omega} \{\overline{\boldsymbol{D}}^{(2)}(t_l) - \boldsymbol{\Theta}[\overline{\boldsymbol{X}}(t_l)]\boldsymbol{\Xi}^{(2)})\}, \end{aligned}$$

where the weight matrix $\boldsymbol{\Omega}$ is defined as

$$\boldsymbol{\Omega} = \mathrm{diag}(\omega_1, \ldots, \omega_Q).$$

B. The detailed derivation of the closed-form solution of the FTRL-proximal algorithm

In the following, our focus remains on the univariate scenario ($d = 1$). Besides, we assume that the stride step $\Delta T = 1$ during the training process. The derivation process presented below is informed by the prior work of McMahan (2011; 2013) and Li (2023b).

Considering Eq. (4) as the regression loss, we can compute its corresponding gradient as

$$\begin{aligned}\nabla L_{t_l}^{(1)} &= \frac{1}{MT_w}\boldsymbol{\Theta}[\boldsymbol{X}(t_l)]^\top\{\boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(1)} - \boldsymbol{D}^{(1)}(t_l)\},\\ \nabla L_{t_l}^{(2)} &= \frac{1}{MT_w}\boldsymbol{\Theta}[\boldsymbol{X}(t_l)]^\top\{\boldsymbol{\Theta}[\boldsymbol{X}(t_l)]\boldsymbol{\Xi}^{(2)} - \boldsymbol{D}^{(2)}(t_l)\}.\end{aligned} \tag{11}$$

The FTRL-proximal algorithm constructs the following unconstrained optimization problems to address the online sparse regression tasks presented in Eq. (3), relying on the gradient described in Eq. (11):

$$\boldsymbol{\Xi}(t_{l+1}) = \arg\min_{\boldsymbol{\Xi}} \sum_{i=1}^{l} \nabla L_{t_i}[\boldsymbol{\Xi}(t_i)]^\top\boldsymbol{\Xi} + \lambda_1\|\boldsymbol{\Xi}\|_1 + \frac{1}{2}\lambda_2\|\boldsymbol{\Xi}\|_2^2 + \frac{1}{2}\sum_{i=1}^{l}\sigma_i\|\boldsymbol{\Xi} - \boldsymbol{\Xi}(t_i)\|_2^2, \tag{12}$$

where $\boldsymbol{\Xi}(t_l)$ denotes the weight vector of drift or diffusion at the time t_l of the training phase, $\lambda_1 > 0$ and $\lambda_2 > 0$ are the coefficients of the *origin-centered* L_1 and L_2 regularization respectively, whereas $\|\boldsymbol{\Xi} - \boldsymbol{\Xi}(t_i)\|_2^2$ denotes the *proximal* term that center additional regularization at the current weight rather than the origin one, which can prevent dramatic change of the weight vector during the training process, and $\sigma_i > 0$ serves as the learning rate that decays over time.

We expand the *proximal* term and delete the constant term, then Eq. (12) can be rewritten as

$$\boldsymbol{\Xi}(t_{l+1}) = \arg\min_{\boldsymbol{\Xi}} \boldsymbol{\zeta}(t_l)^\top\boldsymbol{\Xi} + \lambda_1\|\boldsymbol{\Xi}\|_1 + \frac{1}{2}(\lambda_2 + \sum_{i=1}^{l}\sigma_i)\|\boldsymbol{\Xi}\|_2^2, \tag{13}$$

where

$$\boldsymbol{\zeta}(t_l) = \sum_{i=1}^{l}\{\nabla L_{t_i}[\boldsymbol{\Xi}(t_i)] - \sigma_i\boldsymbol{\Xi}(t_i)\}.$$

Note that we can decompose Eq. (13) into the following unconstrained optimization problems for each scalar component of the weight vector:

$$\xi_j(t_{l+1}) = \arg\min_{\xi_j} \zeta_j(t_l)\xi_j + \lambda_1|\xi_j| + \frac{1}{2}(\lambda_2 + \sum_{i=1}^{l}\sigma_i)\xi_j^2, \tag{14}$$

where $\xi_j(t_l)$ denotes the jth component of $\boldsymbol{\Xi}(t_l)$, $\zeta_j(t_l)$ represents the jth component of $\boldsymbol{\zeta}(t_l)$. Since $|\xi_j|$ is not differentiable at $\xi_j = 0$, we define its subgradient as

$$\nabla|\xi_j| = \begin{cases} \tau = -1, & \text{if } \xi_j < 0,\\ -1 < \tau < 1, & \text{if } \xi_j = 0,\\ \tau = 1, & \text{if } \xi_j > 0.\end{cases}$$

Then we can differentiate the right-hand side of Eq. (14), substitute the subgradient, and set the derivative equal to zero, leading to the following expression:

$$\zeta_j(t_l) + \lambda_1\tau + (\lambda_2 + \sum_{i=1}^{l}\sigma_i)\xi_j = 0. \tag{15}$$

Now, we can categorize and discuss the different possible values of $\zeta_j(t_l)$ as follows.

1. $|\zeta_j(t_l)| \leq \lambda_1$,

(a) $\xi_j > 0, \tau = 1, \zeta_j(t_l) + \lambda_1 + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j > 0$. Eq. (15) does not hold.

(b) $\xi_j = 0, \tau = -\frac{\zeta_j(t_l)}{\lambda_1} \in (-1, 1)$. Eq. (15) holds.

(c) $\xi_j < 0, \tau = -1, \zeta_j(t_l) - \lambda_1 + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j < 0$. Eq. (15) does not hold.

2. $\zeta_j(t_l) < -\lambda_1$,

(a) $\xi_j > 0, \tau = 1, \zeta_j(t_l) + \lambda_1 + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j = 0$ has a solution $\xi_j = -[\zeta_j(t_l) + \lambda_1](\lambda_2 + \sum_{i=1}^{l} \sigma_i)^{-1}$.

(b) $\xi_j = 0, \tau \in (-1, 1), \zeta_j(t_l) + \lambda_1\tau < 0$. Eq. (15) does not hold.

(c) $\xi_j < 0, \tau = -1, \zeta_j(t_l) + \lambda_1\tau + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j < 0$. Eq. (15) does not hold.

3. $\zeta_j(t_l) > \lambda_1$,

(a) $\xi_j > 0, \tau = 1, \zeta_j(t_l) + \lambda_1 + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j > 0$. Eq. (15) does not hold.

(b) $\xi_j = 0, \tau \in (-1, 1), \zeta_j(t_l) + \lambda_1\tau > 0$. Eq. (15) does not hold.

(c) $\xi_j < 0, \tau = -1, \zeta_j(t_l) - \lambda_1 + (\lambda_2 + \sum_{i=1}^{l} \sigma_i)\xi_j = 0$ has a solution $\xi_j = -[\zeta_j(t_l) - \lambda_1](\lambda_2 + \sum_{i=1}^{l} \sigma_i)^{-1}$.

To sum up, we can obtain the following closed-form solution of Eq. (15):

$$\xi_j(t_{l+1}) = \begin{cases} 0, & \text{if } |\zeta_j(t_l)| < \lambda_1, \\ -\frac{\zeta_j(t_l) - \mathrm{sgn}[\zeta_j(t_l)]\lambda_1}{\lambda_2 + \sum_{i=1}^{l} \sigma_i}, & \text{if } |\zeta_j(t_l)| \geq \lambda_1, \end{cases} \tag{16}$$

where $\mathrm{sgn}(\cdot)$ is the sign function.

Inspired by the learning rate setting in the prior work (McMahan et al., 2013; Li et al., 2023b), we set the value of the learning rate σ_i as

$$\sigma_i = \frac{1}{\eta_i} - \frac{1}{\eta_{i-1}},$$

and

$$\eta_i = \frac{\alpha}{\beta + \sqrt{\sum_{k=1}^{i} \nabla L_{t_k}[\xi_j(t_k)]^2}}, \tag{17}$$

where $\alpha > 0$ and $\beta > 0$ are adjustable hyperparameters, $\nabla L_{t_k}[\xi_j(t_k)]$ denotes the partial derivative of the regression loss at time t_k with respect to $\xi_j(t_k)$. Substituting Eq. (17) into Eq. (16), each scalar component of the weight vector adheres to the following iterative update formula at each training step:

$$\xi_j(t_{l+1}) = \begin{cases} 0, & \text{if } |\zeta_j(t_l)| < \lambda_1, \\ -\frac{\zeta_j(t_l) - \mathrm{sgn}[\zeta_j(t_l)]\lambda_1}{\lambda_2 + \frac{\beta + \sqrt{\sum_{k=1}^{i} \nabla L_{t_k}[\xi_j(t_k)]^2}}{\alpha}}, & \text{if } |\zeta_j(t_l)| \geq \lambda_1. \end{cases} \tag{18}$$

The selection of hyperparameters for the FTRL-proximal algorithm in each experiment is presented in Table 2.

Table 2. Hyperparameters for the FTRL-proximal algorithm on various data sets.

Data set	λ_1	λ_2	α	β
Double-well system	10	0.01	1	1
Concatenated synthetic system	0.1	0.01	1	0.1
Fish schooling dataset	0.1	0.1	0.1	50
Rolling bearing vibration dataset	0.1	0.1	10	0.1
EEG dataset	0.1	0.01	0.1	0.1

C. Pseudocode for Online-SINSDy

In this section, we present the pseudocode of Online-SINSDy for the identification and change point detection tasks. To facilitate a clearer exposition, we divide the pseudocode into two parts. Algorithm 1 provides the pseudocode for utilizing Online-SINSDy to obtain the dynamics of the drift and the diffusion weights and perform system identification, while Algorithm 2 presents the pseudocode for change point detection utilizing the dynamics of the weights obtained in Algorithm 1.

It is noteworthy that although the pseudocode is presented in two parts, system identification and change point detection are carried out simultaneously. For the sake of brevity, we only consider the case where the state dimension $d = 1$ in this context. However, the algorithms presented in both pseudocodes can be readily extended to the multivariate case. Moreover, we assume that there is only one change point in the time series data in Algorithm 2. If there are multiple change points in the streaming data, restarting Algorithm 2 after detecting one change point allows for the continued detection of subsequent change points.

Algorithm 1 Online-SINSDy algorithm for system identification

Input: $\boldsymbol{X}(t)$: the streaming data; $\alpha, \beta, \lambda_1, \lambda_2$: the hyperparameters for FTRL-proximal; T_{II}: the beginning time for Phase II; $\mathrm{th}^{(1)}, \mathrm{th}^{(2)}$: the truncation threshold for drift and diffusion in Phase II; T_{tr}: the sliding window length for training; ΔT: the stride step for training.
Output: dynamics of the identified drift weight $\boldsymbol{\Xi}^{(1)}(t)$; dynamics of the identified diffusion weight $\boldsymbol{\Xi}^{(2)}(t)$.
Assume the dimension of the data $d = 1$. Denote N as the total length of the streaming data, M as the number of trajectories, P as the number of basis functions in the library, $\xi_j^{(1)}(t)$ and $\xi_j^{(2)}(t)$ as the jth component of the identified drift and diffusion weight vectors at time t respectively.
Initialize $\zeta(t_{1-\Delta T}) = 0, \xi_j^{(1)}(t_{1-\Delta T}) = \xi_j^{(2)}(t_{1-\Delta T}) = 0, \kappa_j^{(1)} = \kappa_j^{(2)} = 0$ $(j = 1, 2, \cdots, P)$.
for time index l **in** $T_{\mathrm{tr}} : \Delta T : N$ **do**
  Receive the collected data $\mathbf{X}(t_l) = [x^1(t_l), x^2(t_l), \cdots, x^M(t_l), x^1(t_{l-1}), \cdots, x^M(t_{l-T_{\mathrm{tr}}+1})]$.
  Calculate the linear and quadratic variation $\boldsymbol{D}^{(1)}(t_l), \boldsymbol{D}^{(2)}(t_l)$.
  Calculate the library matrix $\boldsymbol{\Theta}[\boldsymbol{X}(t_l)]$.
  if $t_l < T_{\mathrm{II}}$ **then**
    $I^{(1)} = I^{(2)} = \{1, 2, \cdots, P\}$.
  else
    $I^{(1)} = \{i \mid \xi_i^{(1)}(t_{l-\Delta T}) > \mathrm{th}^{(1)}\}$, $I^{(2)} = \{i \mid \xi_i^{(2)}(t_{l-\Delta T}) > \mathrm{th}^{(2)}\}$.
    $\xi_j^{(1)}(t_l) \leftarrow 0, \forall j \notin I^{(1)}$; $\xi_j^{(2)}(t_l) \leftarrow 0, \forall j \notin I^{(2)}$.
  end if
  for each component k **in** $I^{(1)}$ **do**
    Calculate $\xi_k^{(1)}(t_l)$ using Eq. (18).
    Calculate the gradient of the regression loss $\nabla L_{t_l}^{(1)}[\xi_k^{(1)}(t_l)]$ using Eq. (11).
    Calculate $\sigma_k^{(1)}(t_l) = \left\{ \sqrt{\kappa_k^{(1)} + \nabla L_{t_l}^{(1)}[\xi_k^{(1)}(t_l)]^2} - \sqrt{\kappa_k^{(1)}} \right\} / \alpha$.
    Calculate $\zeta_k^{(1)}(t_l) = \zeta_k^{(1)}(t_{l-\Delta T}) + \nabla L_{t_l}^{(1)}[\xi_k^{(1)}(t_l)] - \sigma_k^{(1)}(t_l)\xi_j^{(1)}(t_l)$.
    Update $\kappa_k^{(1)} \leftarrow \kappa_k^{(1)} + \nabla L_{t_l}^{(1)}[\xi_k^{(1)}(t_l)]^2$.
  end for
  for each component k **in** $I^{(2)}$ **do**
    Calculate $\xi_k^{(2)}(t_l)$ using Eq. (18).
    Calculate the gradient of the regression loss $\nabla L_{t_l}^{(2)}[\xi_k^{(2)}(t_l)]$ using Eq. (11).
    Calculate $\sigma_k^{(2)}(t_l) = \left\{ \sqrt{\kappa_k^{(2)} + \nabla L_{t_l}^{(2)}[\xi_k^{(2)}(t_l)]^2} - \sqrt{\kappa_k^{(2)}} \right\} / \alpha$.
    Calculate $\zeta_k^{(2)}(t_l) = \zeta_k^{(2)}(t_{l-\Delta T}) + \nabla L_{t_l}^{(2)}[\xi_k^{(2)}(t_l)] - \sigma_k^{(2)}(t_l)\xi_j^{(2)}(t_l)$.
    Update $\kappa_k^{(2)} \leftarrow \kappa_k^{(2)} + \nabla L_{t_l}^{(2)}[\xi_k^{(2)}(t_l)]^2$.
  end for
end for

Algorithm 2 Online-SINSDy algorithm for change point detection

Input: $\boldsymbol{\Xi}(t)$: the streaming weight vectors of drift or diffusion obtained by Algorithm 1; T_{cpd}: the sliding window length for change point detection; h, γ: the hyperparameters for CUSUM; k: number of indicators in Phase II used to estimate the initial average m_R and standard deviation σ_R.
Output: detected change point T_d.
Initialize $U(t_{k-1}) = 0, t_{\mathrm{CU}} = +\infty, R_{\mathrm{CU}} = +\infty, T_d = 0$.
Assume the dimension of the data $d = 1$ and only one change point exists in the time series data. For conciseness, denote the time series of the drift or diffusion weight vectors obtained by Algorithm 1 from the beginning of Phase II as $\boldsymbol{\Xi}(t_1), \boldsymbol{\Xi}(t_2), \cdots, \boldsymbol{\Xi}(t_{N_d})$ in sequence.
for time index l **in** $2T_{\mathrm{cpd}} + 1 : 1 : N_d$ **do**
  Calculate $\overline{\boldsymbol{\Xi}}(t_{l-T_{\mathrm{cpd}}}, t_l), \overline{\boldsymbol{\Xi}}(t_{l-2T_{\mathrm{cpd}}}, t_{l-T_{\mathrm{cpd}}})$.
  Calculate $R(t_{l-T_{\mathrm{cpd}}})$.
  if $l = k + T_{\mathrm{cpd}} - 1$ **then**
    Calculate $m_R = \frac{1}{k} \sum_{i=T_{\mathrm{cpd}}}^{l} R(t_i)$.
    Calculate $\sigma_R = \sqrt{\frac{1}{k-1} \sum_{i=T_{\mathrm{cpd}}}^{l} [R(t_i) - m_R]^2}$.
  end if
  if $l > k + T_{\mathrm{cpd}} - 1$ **then**
    Calculate $U(t_{l-T_{\mathrm{cpd}}}) = \max[0, U(t_{l-T_{\mathrm{cpd}}-1}) + R(t_{l-T_{\mathrm{cpd}}}) - m_R - \frac{1}{2} h \sigma_R]$.
    if $U(t_{l-T_{\mathrm{cpd}}}) > \gamma \sigma_R$ and $U(t_i) \leq \gamma \sigma_R$ $(\forall i < l - T_{\mathrm{cpd}})$ **then**
      Update $t_{\mathrm{CU}} \leftarrow t_{l-T_{\mathrm{cpd}}}$.
      Update $R_{\mathrm{CU}} \leftarrow R(t_{l-T_{\mathrm{cpd}}})$.
    end if
  end if
  if $t_{l-T_{\mathrm{cpd}}} \geq t_{\mathrm{CU}}$ and $R_{l-T_{\mathrm{cpd}}} \geq R_{\mathrm{CU}}$ **then**
    Update $T_d \leftarrow \mathrm{argmax}_{t \in \{T_d, t_{l-T_{\mathrm{cpd}}}\}} R(t)$.
  end if
end for

c

D. Results of online identification of SDEs

In this section, we validate the effectiveness of our Online-SINSDy algorithm in the identification task, which relies on the KM formulae and the modified FTRL-proximal algorithm, using both synthetic data and real-world systems. We also experimentally demonstrate the necessity and effectiveness of our two-phase training method.

D.1. The double-well system

We first illustrate the online data-driven discovery of SDEs by the example of a 1-d double-well system. The drift term and the diffusion term are given as

$$\begin{aligned} f(x) &= -3x^3 + 2x, \\ G(x) &= \frac{1}{4}. \end{aligned} \tag{19}$$

We generate $M = 100$ independent trajectories by integrating Eq. (1) with the Euler-Maruyama method, each trajectory contains $N = 10^5$ recorded states with time intervals $\Delta t = 0.01$. The basis functions of the library consist of polynomials of the state up to the order of 9, i.e. $\boldsymbol{\Theta}(x) = [1, x, \ldots, x^9]$. We take the taining time window as $T_{\mathrm{tr}} = 10$ with the stride step $\Delta T = 10$. In the following experiments, we repeatedly generate 50 sets of data and construct the expressions of the drift function and the diffusion function of each dataset to verify the robustness of our algorithm. We use the mean value of the weight vectors as the identified coefficients of the function expressions and use the standard deviation to reflect the stability of the identification.

As is shown in Fig. 6(b), Online-SINSDy can accurately recover the analytical expressions for the drift and diffusion

functions with quite small standard deviation. Fig. 6(c) demonstrates that Online-SINSDy with truncation in Phase II can accurately discover the drift function of the double-well system as long as the threshold $\mathrm{th}^{(1)} \geq 0.4$, whereas training without truncation fails to screen the terms appearing in the drift function correctly.

We show the dynamics of all the drift coefficients during the training process in Fig. 6(d) to further explore the reason for the failure of the training without truncation in Phase II. As is shown in the left figure, if the truncation operation of Phase II is not performed during the training process, the weight vector will converge and stabilize at a false surrogate stable state, resulting in the failure of the identification. However, if we conduct the truncation process in Phase II after the stability is achieved in Phase I, the coefficients will escape from this false equilibrium and converge toward their true values. As depicted in the right figure, after $5,000$ training steps, we apply the truncation operation, where the non-zero coefficients of the interfering terms (x^8, x^9) below the threshold $\mathrm{th}^{(1)} = 0.5$ are immediately set to zero. The truncation operation disturbs the previous stable state, allowing the actual terms (x, x^3) in the drift function to further approach their true values, while other interfering terms (x^5) whose coefficients are greater than the threshold gradually decrease until they ultimately become smaller than the threshold and are set to zero. At this point, we have filtered out all genuine terms from the library, and subsequent training will rapidly converge the coefficients of these terms towards their true values.

Actually, if we have more accurate prior knowledge about the range of the coefficients, we can use it to accelerate the convergence of the algorithm. As is shown in Fig. 6(e), we set a relative error threshold $\kappa_{\mathrm{error}} = 0.1$ and record the number of training steps in Phase II required for the coefficients to converge within κ_{error} under different truncation thresholds. As the truncation threshold increases, the convergence rate increases as well.

To evaluate the robustness of our Online-SINSDy algorithm against noisy perturbations, we also introduce the Gaussian white noise with different standard deviation σ into the observational data. Fig. 6(f) illustrates that when $\sigma < \Delta t$, the relative error of the drift coefficients remains at a low level; when $\Delta t \leq \sigma < 2\Delta t$, the relative error experiences a steady increase, but Online-SINSDy still accurately identifies the terms appearing in the drift function; when $\sigma \geq 2\Delta t$, the inference accuracy begins to decrease. The inference accuracy remains 100% for the diffusion function with the noise intensity $0 \leq \sigma \leq 3\Delta t$, in spite of a gradual growth of the relative error. Thus, the algorithm we proposed is robust against the noise to a certain extent.

Note that all the experiments above are conducted both with and without bin-wise averaging of the data, and we find little difference between them.

D.2. The fish schooling dataset

Online-SINSDy can be applied to discover the governing SDEs of many real-world systems with the collected streaming data, which is quite challenging because of the strong observation noise and the existence of missing data points in real scenarios. Here we focus on the dynamics of collective alignment in groups of the cichlid fish *Etroplus suratensis* with different group sizes. We have mentioned in the main text that the highly polarized and coherent motion of the cichlid fish is a finite-size noise-induced effect, and the mesoscopic dynamics can be fitted as Eq. (8), where $\boldsymbol{m}$ defined by Eq. (9) is a 2-d order parameter vector called group polarization, which represents the collective fish motion direction and the degree of alignment.

For Eq. (8), the deterministic dynamics of the fish ($N_f \to \infty$) promote the system to produce an isotropic motion pattern ($|\boldsymbol{m}| = 0$), while the existence of the intrinsic noise, which is caused by the probabilistic interactions between individuals, lead to the noise-introduced schooling behavior ($|\boldsymbol{m}| \lesssim 1$). Moreover, the intensity of the intrinsic noise that introduces the aligned motion is inversely proportional to the group size, meaning that the smaller the group size, the more likelihood of such highly polarized and coherent motion, as is shown in Fig. 7(a-c).

We here focus on three scenarios from small to medium group sizes with $N_f = 15$,$N_f = 30$, and $N_f = 60$ respectively. The dataset consists of four trajectories for $N_f = 15$ and $N_f = 30$ and three trajectories for $N_f = 60$. Each trajectory contains around 2.5×10^4 recorded data of the group polarization $\boldsymbol{m}$ with uniform time interval $\Delta t = 0.12s$ and includes many missing data points. To estimate the KM coefficients, we take $40\Delta t$ as the time interval to calculate the drift value and Δt to calculate the diffusion value, which is the same as prior work (Jhawar et al., 2020). The library is set as $\boldsymbol{\Theta}(x) = [1, x, \ldots, x^9]$. We take the training window length $T_{\mathrm{tr}} = 12$s and the stride step length $\Delta T = 12$s. The first half of the trajectories is used for Phase I training, and the second half is used for Phase II training with truncation. The truncation threshold for both drift and diffusion are set as $\mathrm{th}^{(1)} = \mathrm{th}^{(2)} = 0.01$.

The experimental results are shown in Fig. 7(d-g), we calculate a series of accurate estimates of the drift values and the

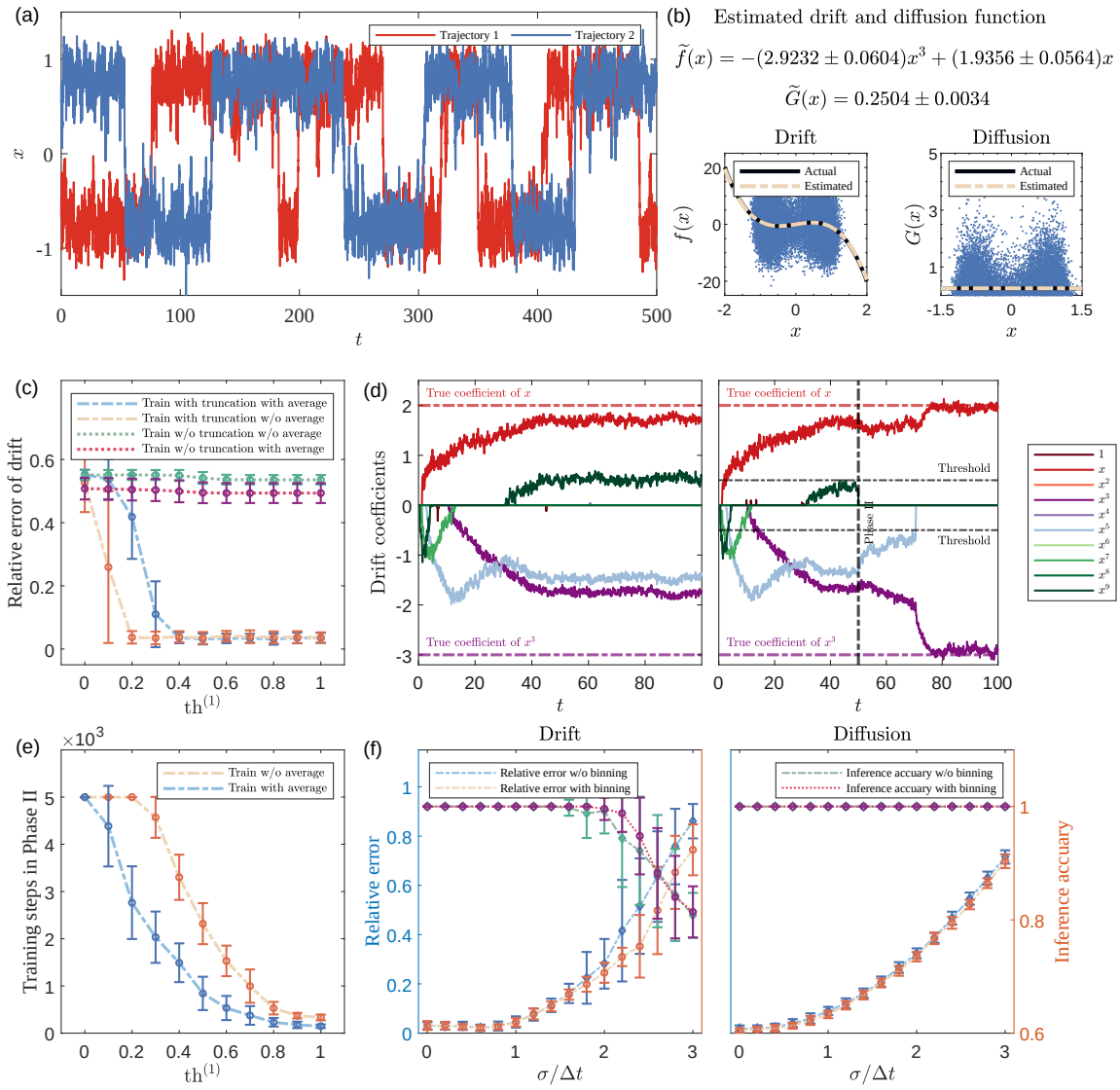

Figure 6. Identification results of the double-well system described by Eq. (19). (a) Two trajectories of the double-well system. (b) The estimated drift and diffusion functions using Online-SINSDy. The blue dots represent the linear and quadratic variation at each point calculated during the training process. (c) The relative error between the identified drift function and the true drift function. (d) The dynamics of the drift coefficients during the training process with (right) or without (left) the truncation operation in Phase II. (e) The number of training steps in Phase II required for the coefficients to converge within a small neighborhood around the true coefficients under different truncation thresholds. (f) The relative error and inference accuracy of the drift (left) and diffusion (right) terms under different noise levels.

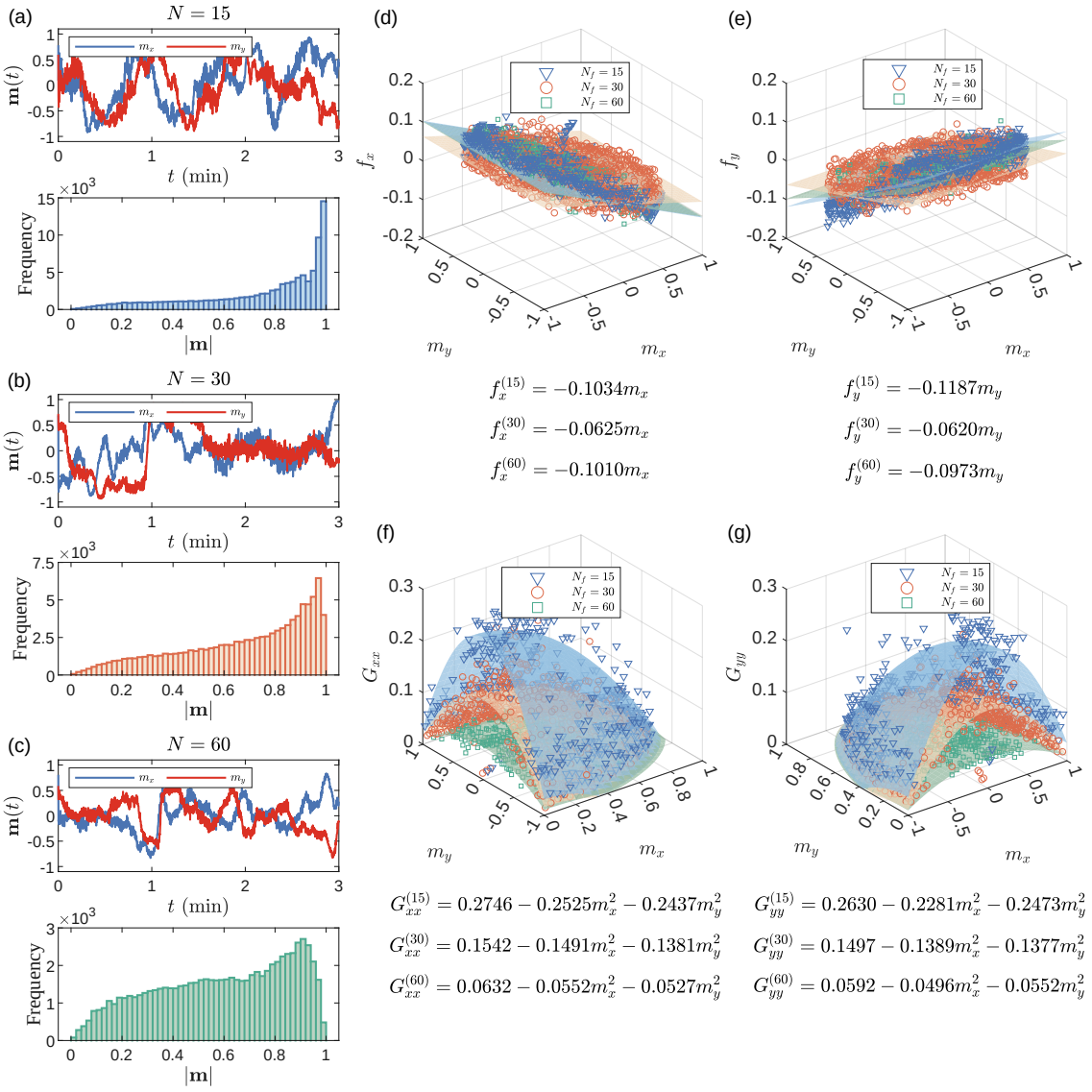

Figure 7. Identification results of the fish schooling dataset. (a-c) Collective dynamical trajectories and statistical histograms of group polarization for the cichlid fish with different group sizes. (d-e) The identified drift functions f_x and f_y for different group sizes. (f-g) The identified diffusion functions G_{xx} and G_{yy} for different group sizes.

diffusion values by bin-wise averaging the whole dataset and find that the surface corresponding to the identified drift and diffusion functions can fit these values well. In fact, the Online-SINSDy algorithm successfully identifies the correct sparse functional terms from the redundant basis functions and ultimately constructs accurate and concise function expressions of the underlying SDEs for different values of N_f. From the discovered SDEs, we can not only identify the deterministic behavior of the fish but also gain insight into the inverse relationship between the noise intensity and the group size. Moreover, we can also recover $\hat{\gamma}_2 = 3.8$ from the coefficients of the identified SDEs, which is consistent with the true value $\gamma_2 \approx 4.0$.

D.3. The rolling bearing vibration dataset

We also use Online-SINSDy to discover the governing equations of the rolling bearing vibration dynamics from the Case Western Reserve University (CWRU) bearing dataset.

We here focus on the trajectories of a normal fan-end bearing and a faulted fan-end bearing with a ball fault diameter of 14 mils, which contain around 4.8×10^5 and 3.8×10^5 recorded data points respectively with the same time interval $\Delta t = 1/48,000$. In each case, only one recorded trajectory is available for training. The library is set as $\mathbf{\Theta}(x) = [1, x, x^2, x^3, e^x, e^{2x}, e^{3x}]$. We take the training window length $T_{\text{tr}} = 1/48$s with stride step length $\Delta T = 1/480$s. After Phase I training with the first 60% data, Phase II training begins with truncation threshold $\text{th}^{(1)} = 0.3$ and $\text{th}^{(2)} = 0.01$.

Although only one trajectory is available for training in each case, which is challenging for the online identification task, our algorithm still provides reasonable and concise functional expressions for the bearing vibration dynamics, as is shown in Table 3. Due to the availability of ample data in the trajectories, we employ the bin-wise averaging method to obtain precise estimates of the drift and diffusion terms. These estimates are subsequently used as reference points to evaluate the accuracy of the identification. The mean square error between the estimates derived from bin-wise averaging and the corresponding identified drift and diffusion terms after training is then calculated. As depicted in Fig. 8, the SDEs identified by Online-SINSDy effectively captured the authentic dynamics of the bearing from the recorded data points.

Table 3. Identification results of the rolling bearing vibration dataset.

Data set	Drift function	Drift MSE	Diffusion function	Diffusion MSE
Normal	$0.9134 - 5.2558x - 4.6694x^2 - 0.6426e^{3x}$	0.0019	$3.7311x^2 + 0.6454e^x$	0.0003
Faulted	$1.9914 + 11.4064x^2 + 0.6755e^x - 0.7715e^{2x} - 1.7427e^{3x}$	0.0013	$-4.4265x + 14.3748x^2 + 0.9046e^{3x}$	0.0109

E. The equivalence of the FTRL-proximal algorithm and the online gradient descent method

The parameter update formula in online gradient descent (OGD) algorithms is given by:

$$\mathbf{\Xi}(t_{l+1}) = \mathbf{\Xi}(t_l) - \eta_l \nabla L_{t_l}[\mathbf{\Xi}(t_l)],$$

where η_l is the learning rate at t_l and L represents the loss function.

If we ignore the L_1 and L_2 regularization terms that are used to induce sparsity of the weight vectors, the Eq.(3) takes the following form:

$$\mathbf{\Xi}(t_{l+1}) = \arg\min_{\mathbf{\Xi}} \sum_{\mathrm{i}=1}^{\mathrm{l}} \nabla \mathrm{L}_{\mathrm{t_i}}[\mathbf{\Xi}(\mathrm{t_i})]^\top \mathbf{\Xi} + \frac{1}{2} \sum_{\mathrm{i}=1}^{\mathrm{l}} \sigma_{\mathrm{i}} \|\mathbf{\Xi} - \mathbf{\Xi}(\mathrm{t_i})\|_2^2,$$

Taking the derivative of the right-hand side and setting it to zero, we obtain:

$$\mathbf{\Xi}(t_{l+1}) = \frac{\sum_{i=1}^{l} \sigma_i \mathbf{\Xi}(t_i)}{\sum_{i=1}^{l} \sigma_i} - \eta_l \sum_{i=1}^{l} \nabla L_{t_i}[\mathbf{\Xi}(t_i)],$$

where $\eta_l = 1/\sum_{i=1}^{l} \sigma_i$ can be regarded as the learning rate. It is evident that the parameter update form induced by our objective function closely resembles that of OGD. In the parameter update formula of OGD, we replace the weight vectors at t_l with the weighted average of all previous weight vectors and utilize the cumulative gradient in place of the instantaneous

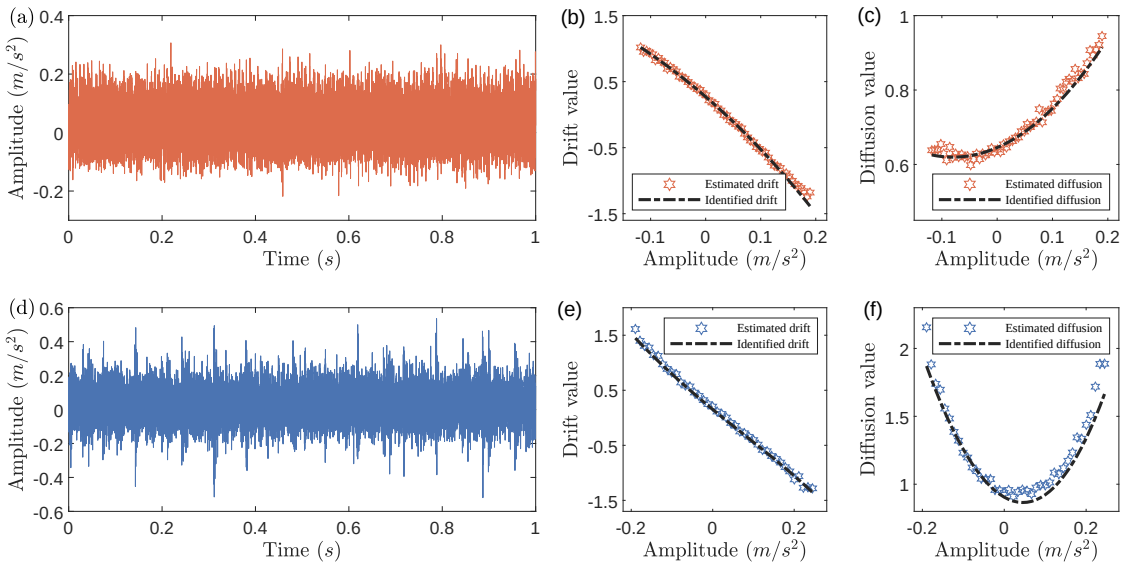

Figure 8. Identification results of the rolling bearing vibration dataset. (a) The trajectory of a normal fan-end bearing. (b-c) The estimated drift and diffusion values and the identified drift and diffusion functions of the normal fan-end bearing. The estimated values are calculated by the bin-wise averaging method using the whole dataset. (d) The trajectory of a faulted fan-end bearing with a ball fault diameter of 14 mils. (e-f) The estimated drift and diffusion values and the identified drift and diffusion functions of the faulted fan-end bearing.

gradient. Indeed, employing the weighted average weight vectors and cumulative gradient can enhance the stability of the OGD algorithm, reducing the impact of rare events in the dataset on training.

In summary, the existence of the first term and the proximal term in Eq.(3) enables the solution to be interpreted as an OGD of the weight vectors.

F. The detection delay of Online-SINSDy on sythetic and real-world datasets.

We note that many articles directly employ detection delay (the positive part of the stopping time minus the true change-point time) to measure the effectiveness of the detection. Here, we include statistical figures corresponding to the detection delay under different noise levels for both the sythetic dataset (Fig. 9) and the fish schooling dataset (Fig. 10) in Section 3.

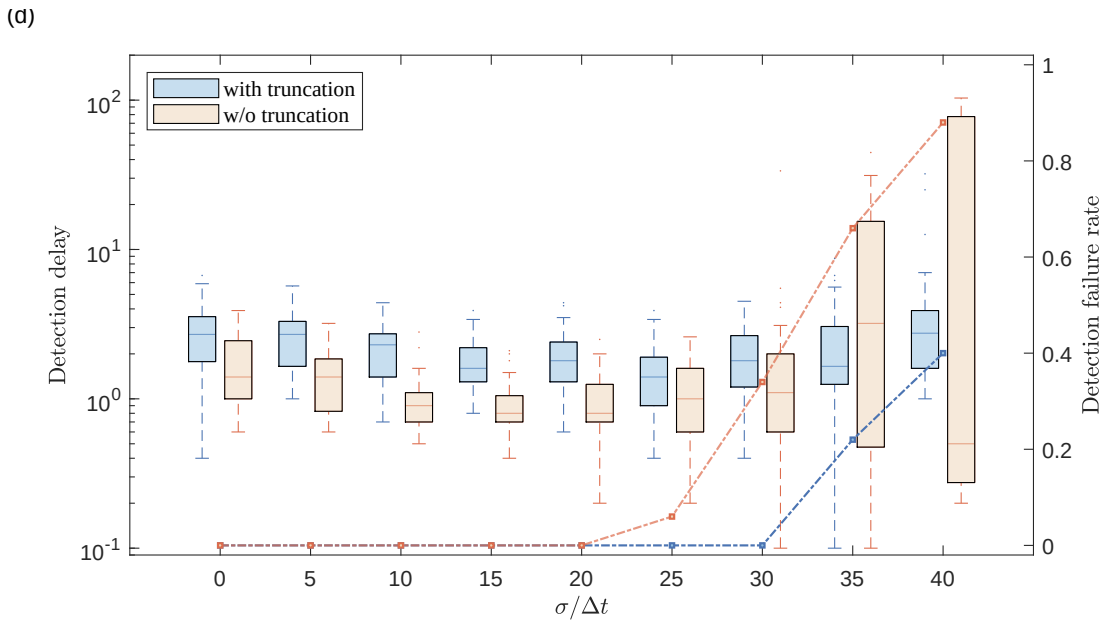

Figure 9. Box plot of the detection delay of Online-SINSDy on the sythetic dataset under different levels of noise.

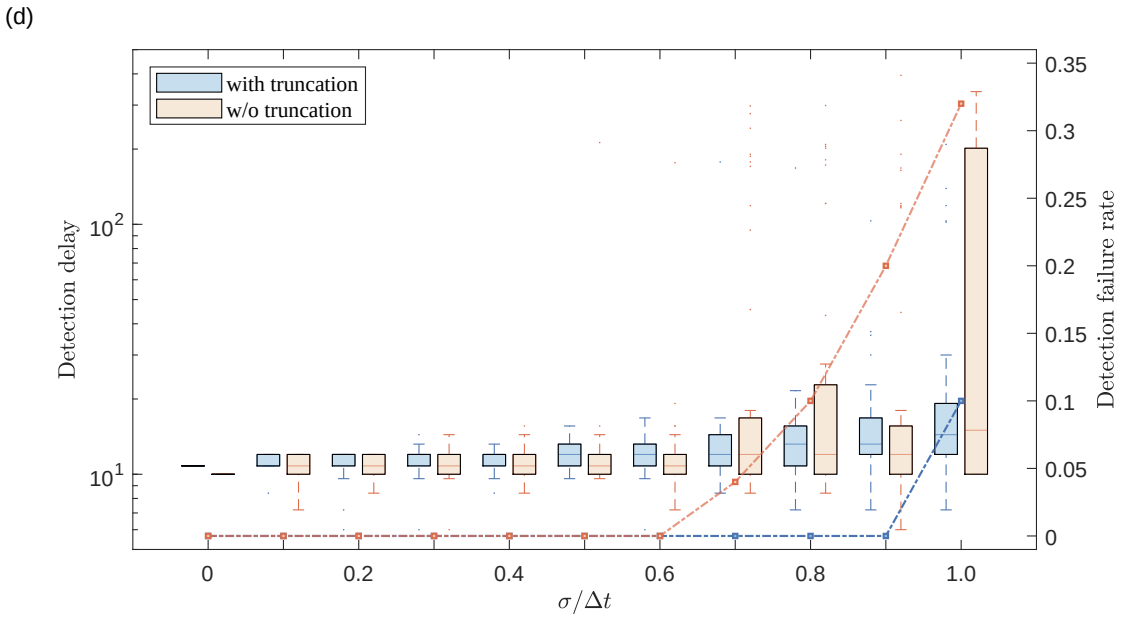

Figure 10. Box plot of the detection delay of Online-SINSDy on the fish schooling dataset using G_{xx} indicator under different levels of noise.