Modeling Latent Non-Linear Dynamical System over Time Series

Ren Fujiwara, Yasuko Matsubara, Yasushi Sakurai

SANKEN, Osaka University
r-fujiwr88@sanken.osaka-u.ac.jp, yasuko@sanken.osaka-u.ac.jp, yasushi@sanken.osaka-u.ac.jp

Abstract

We study the problem of modeling a non-linear dynamical system when given a time series by deriving equations directly from the data. Despite the fact that time series data are given as input, models for dynamics and estimation algorithms that incorporate long-term temporal dependencies are largely absent from existing studies. In this paper, we introduce a latent state to allow time-dependent modeling and formulate this problem as a dynamics estimation problem in latent states. We face multiple technical challenges, including (1) modeling latent non-linear dynamics and (2) solving circular dependencies caused by the presence of latent states. To tackle these challenging problems, we propose a new method, **La**tent **No**n-**L**inear **e**quation **m**odeling (LaNoLem), that can model a latent non-linear dynamical system and a novel alternating minimization algorithm for effectively estimating latent states and model parameters. In addition, we introduce criteria to control model complexity without human intervention. Compared with the state-of-the-art model, LaNoLem achieves competitive performance for estimating dynamics while outperforming other methods in prediction.

Code — https://github.com/renfujiwara/LaNoLem
Extended version — https://arxiv.org/abs/2412.08114

1 Introduction

The past few years have seen a surge in methods for modeling dynamics as mathematical expressions from observed data, encompassing both advancements in traditional evolutionary algorithms and novel machine learning-based techniques (La Cava et al. 2018; Udrescu and Tegmark 2020; Burlacu, Kronberger, and Kommenda 2020; Landajuela et al. 2022; Shojaee et al. 2023). In particular, the sparse identification of non-linear dynamics (SINDy) framework (Brunton, Proctor, and Kutz 2016) has emerged as a leading approach for parsimonious modeling, and its variants have been proposed over a number of years (Boninsegna, Nüske, and Clementi 2018; Champion et al. 2020; Kaptanoglu et al. 2021; Bertsimas and Gurnee 2023).

Despite these advances, limited research considers the temporal dependencies between data, even though many data are given as time series data. The sequential nature of the data can help us address various challenges. For example, distinguishing between noise and non-linearity is challenging when modeling data as non-linear dynamics from noisy data. However, it is easily identifiable by employing a fitting algorithm based on the time dependencies because non-linearity is time-dependent while noise is independent at each time point. In addition, the sequential nature of the data helps stabilize future predictions. Initial states are critical for prediction in non-linear models, and unstable initial states reduce prediction accuracy. This can also be resolved by tracking state transitions in the model and estimating initial states based on them. In time series analysis, such problems are treated as modeling dynamics in the latent state, i.e., modeling dynamics where noise in the data is removed or compressed to a dimension lower than that of the data. Intuitively, we wish to solve the following problem:

Given a time series, how can we estimate the latent states and dynamics with mathematical expressions?

This problem presents us with the following two challenges: (1) how to formulate a model in latent states and (2) how to estimate model parameters and latent states simultaneously. A crucial insight into many dynamical systems of interest is that the function representing the transition consists of only a few but various types of terms (Brunton, Proctor, and Kutz 2016). In other words, transitions among latent states must be represented by the non-linear dynamical system consisting of various terms of the latent states, and the estimation algorithm must incorporate a sparsity of these terms. The presence of latent states also causes the following circular dependency in the algorithm: good latent states require well-estimated models, and the latent states must be properly estimated to find a suitable model. Non-linear state transitions and parameter sparsity further complicate this circular dependency. Specifically, non-linear state transitions introduce challenges in estimating latent states, while sparsity introduces challenges in estimating parameters.

In this paper, we tackle the above challenging problem and propose a method called **La**tent **No**n-**L**inear **e**quation **m**odeling (LaNoLem). LaNoLem can estimate the latent states and dynamics with mathematical expressions from observed data. Our model has two components: (a) a non-linear dynamical system consisting of various terms of latent states and (b) criteria based on the minimum description length (MDL) principle to determine our model complexity.

To overcome the challenges caused by this model (i.e., non-linear state transitions and parameter sparsity), we develop a novel minimization algorithm that alternately estimates latent states and model parameters. In particular, our algorithm can ensure a fast convergence rate for estimating sparse parameters.

In summary, our contributions are as follows:

- We propose a brief model representing latent non-linear dynamics and criteria for evaluating the trade-off between the accuracy and complexity of the model.
- We develop an efficient algorithm for estimating latent states and model parameters simultaneously, considering non-linear state transitions and sparsity in the parameters.
- Compared to state-of-the-art methods on 71 chaotic benchmark datasets, our model achieves competitive performance for estimating dynamics while consistently outperforming state-of-the-art in prediction tasks.

2 Proposed Model

Latent non-linear dynamical system. In our model, we capture the latent dynamics of time series involving non-linear phenomena. We begin with the assumption that there are two classes of values:

$\mathbf{s}(t)$: Latent states, i.e., k-dimensional latent states at time point t, ($\mathbf{s}(t) = \{s_1(t), ...s_k(t)\}$).

$\mathbf{y}(t)$: Estimated values, i.e., d-dimensional values that can be observed at time point t, ($\mathbf{y}(t) = \{y_1(t), ...y_d(t)\}$).

Here, we can only observe the estimated values $\mathbf{y}(t)$ at time point t. $\mathbf{s}(t)$ is a hidden vector that evolves over time as a dynamical system. Consequently, we consider dynamical systems described by the following equations:

$$\mathbf{s}(t+1) = \mathbf{A}\mathbf{s}(t) + \mathbf{F}\boldsymbol{\phi}(\mathbf{s}(t), d_\phi) + \mathbf{b},$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{s}(t) + \mathbf{u}.$$

$\mathbf{A}$ and $\mathbf{F}$ describe the dynamics of latent states $\mathbf{s}(t)$, which capture linear and non-linear dynamics. $\mathbf{C}$ shows the observation projection, which generates the estimated value $\mathbf{y}(t)$ at each time point t. We also refer to $\mathbf{b}$ as state offsets and $\mathbf{u}$ as observation offsets. $\boldsymbol{\phi}(\mathbf{s}(t), d_\phi)$ indicates non-linearities in the latent state space. In summary, we have the following:

Definition 1. *(Full parameter set of LaNoLem: θ) Let θ be a complete set of LaNoLem parameters, namely, $\theta = \{\mathbf{A}, \mathbf{F}, \mathbf{b}, \mathbf{C}, \mathbf{u}\}$.*

The non-linearities $\boldsymbol{\phi}(\mathbf{s}(t), d_\phi)$ require adaption to various dynamics. Therefore, in our model, we describe $\boldsymbol{\phi}(\mathbf{s}(t), d_\phi)$ as follows:

$$\boldsymbol{\phi}(\mathbf{s}(t), d_\phi) = [\mathbf{s}^{*2}(t), ..., \mathbf{s}^{*d_\phi}(t)].$$

Here, d_ϕ represents the order of LaNoLem, and higher polynomials are denoted as $*2$, $*3$, ..., $*d_\phi$, where $*2$ denotes the quadratic non-linearities in the state $\mathbf{s}(t) = \{s_1(t), ..., s_k(t)\}$]:

$$\mathbf{s}^{*2}(t) = [s_1^2(t), s_1(t)s_2(t), ..., s_k^2(t)].$$

The non-linearities $\boldsymbol{\phi}(\mathbf{s}(t), d_\phi)$ indicate that our model can have many terms and make the model more complex than necessary. However, for many dynamical systems of interest, the function consists of only a few but various types of terms. Our method represents this by making $\{\mathbf{A}, \mathbf{F}\}$ sparse. Consequently, our goal can be defined as follows.

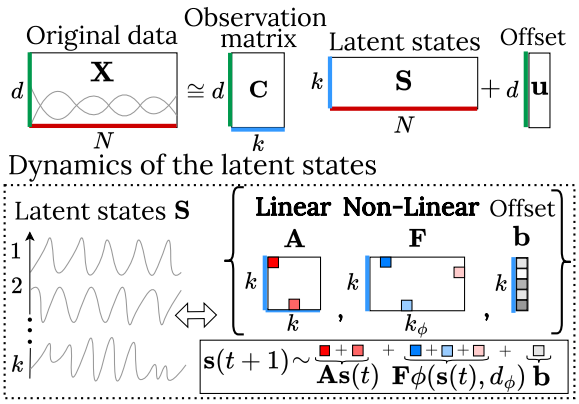

Figure 1: Overview of LaNoLem: Given a data sequence $\mathbf{X}$, we extract latent states represented by a non-linear dynamical system, where parameters for dynamics, i.e., $\mathbf{A}$ and $\mathbf{F}$, are sparse.

Problem 1. *Given a time series $\mathbf{X} \in \mathbb{R}^{N \times d}$, estimate latent states and parameter set, i,e.,*

- ***estimate** latent states $\{\mathbf{s}(t)\}_{t=1}^N$.*
- ***identify** full parameter set θ, s.t., $\{\mathbf{A}, \mathbf{F}\}$ are sparse.*

Criteria for controlling model complexity. In the LaNoLem model, deciding model complexity (i.e., the order of LaNoLem d_ϕ) is an important problem in terms of accurate modeling. However, obtaining any error is also unreliable because it is necessary to consider the sparsity of the model, i.e., that $\mathbf{A}$ and $\mathbf{F}$ are sparse matrices, to evaluate the true complexity of the proposed method. We use the minimum description length (MDL) principle (Grünwald 2007) to deal with this problem. Although important, the MDL principle is a model selection criterion based on lossless compression principles, which does not directly address our problem. Thus, we define a new coding scheme for the summarization of a given model. We introduce an intuitive coding scheme, which is based on lossless compression principles. In short, the goodness of the model can be roughly described as $Cost_T = Cost(M) + Cost(X|M)$, where $Cost(M)$ shows the cost of describing the model M, and $Cost(X|M)$ represents the cost of describing the data $\mathbf{X}$ given the model M.

Fig. 1 shows an overview of our LaNoLem model for time series $\mathbf{X}$. We aim to estimate all these parameters, assuming that only a few important terms govern the dynamics. However, optimizing all the parameters is extremely expensive and depends on the noise. When this is the case, how can we formulate the parameter estimation? We provide the answer in the next section.

Algorithm 1: *Optimization algorithm* ($\mathbf{X}$)

Input: data $\mathbf{X} \in \mathbb{R}^{N \times d}$
Output: (a) estimated latent states $\{\hat{\mathbf{s}}(t)\}_{t=1}^{N}$
(b) parameter set $\theta = \{\mathbf{A}, \mathbf{F}, \mathbf{b}, \mathbf{C}, \mathbf{u}\}$
repeat
  /* Inference */
  Estimate latent space $\{\hat{\mathbf{s}}(t)\}_{t=1}^{N}$ // Eq. (3)-(11)
  Estimate the moments set $\mathbf{M}$ for Learning // Eq. (12)-(17)
  /* Learning */
  θ^{new} = Learning($\{\hat{\mathbf{s}}(t)\}_{t=1}^{N}, \mathbf{M}, \theta$)
until convergence;
return $\{\{\hat{\mathbf{s}}(t)\}_{t=1}^{N}, \theta\}$

3 Optimization Algorithm

Given a data $\mathbf{X}$, we propose an algorithm to estimate:

- the latent states $\hat{\mathbf{s}}(t) = \mathbb{E}[\mathbf{s}(t)], (t = 1, ..., N)$;
- the governing parameter set $\theta = \{\mathbf{A}, \mathbf{F}, \mathbf{b}, \mathbf{C}, \mathbf{u}\}$;

The goal of estimation is achieved by minimizing the negative likelihood of observed data. However, it is difficult to directly minimize the negative likelihood of incomplete data; we minimize the expected negative log-likelihood of the observation sequence. Our overall optimization problem is

$$\arg\min_{\mathbf{S},\theta} -Q(\mathbf{X}, \mathbf{S}, \theta) + r(\mathbf{A}, \mathbf{F}). \tag{1}$$

$\mathbf{S}$ is a set of latent states, i.e., $\mathbf{S} = \{\mathbf{s}(t)\}_{t=1}^{N}$. $r(\cdot)$ is the regularizer for penalizing non-zero solutions. In this paper, we use the elastic-net regularizer (Zou and Hastie 2005) as $r(\cdot)$. However, since the difference operation is expressed as $s(t+h) = s(t) + \Delta s(t)$ (Kelley and Peterson 2001), we must regularize that penalizes solutions that are not 1 in the diagonal component of $\mathbf{A}$. Therefore, we use the regularizer to differentiate the identity matrix I, i.e., $r(\mathbf{\Theta}) = \frac{1}{2}\lambda_2||\mathbf{\Theta} - \mathbf{I}||_F^2 + \lambda_1||\mathbf{\Theta} - \mathbf{I}||_1$. $||\mathbf{\Theta}||_F$ represents the Frobenius norm of matrix $\mathbf{\Theta}$ and $||\mathbf{\Theta}||_1$ is the l_1 norm on every element of matrix $\mathbf{\Theta}$. Additionally, $Q(\mathbf{X}, \mathbf{S}, \theta)$ is as follows,

$$\begin{aligned} Q(\mathbf{X}, \mathbf{S}, \theta) = \mathbb{E}[&-\sum_{t=1}^{N} D(\mathbf{x}(t), \mathbf{C}\mathbf{s}(t) + \mathbf{u}, \mathbf{R}) - \frac{N \log|\mathbf{R}|}{2} \\ &- \sum_{t=1}^{N-1} D(\mathbf{s}(t+1), \mathbf{A}\mathbf{s}(t) + \mathbf{F}\phi(\mathbf{s}(t), d_\phi) + \mathbf{b}, \mathbf{\Gamma}) \\ &- \frac{(N-1)\log|\mathbf{\Gamma}|}{2}], \end{aligned} \tag{2}$$

where D is the square of the Mahalanobis distance $D(x, y, \Sigma) = (x - y)^T \Sigma^{-1} (x - y)$. $\mathbf{\Gamma}/\mathbf{R}$ represents the covariance of Gaussian noises in states/observations.

We must estimate latent states appropriately to find a suitable parameter set for $\mathbf{X}$. Simultaneously, a good latent state requires a well-estimated model. We escape this circular dependency by applying a novel alternating minimization.

Algorithm 1 provides an overview of *Optimization algorithm*. Specifically, we propose an iterative method that alternately employs the following two sub-algorithms (Inference and Learning) until the cost function reaches a minimum value. Next, we describe each sub-algorithm in detail.

Inference

The goal of Inference is achieved by finding the marginal distributions for the latent variables conditional on the observation sequence. These inference problems can be solved efficiently using the sum-product message passing (Pearl 1982). In a linear setting, we can efficiently solve these inference problems with the Kalman filter in the context of LDS (Kalman 1963). However, introducing transition models that depart from the linear Gaussian model leads to an intractable inference problem. Thus, we introduce one widely used approach to realize a Gaussian approximation by linearizing around the mean of the predicted distribution with a matrix of partial derivatives (the Jacobian), which gives rise to an extended Kalman filter (Zarchan 2005). In Inference, we use the Jacobian $\mathbf{J}_{\mathbf{s}(t)} = \mathbf{A} + \frac{\partial \mathbf{F}\boldsymbol{\phi}(\mathbf{s}(t), d_\phi)}{\partial \mathbf{s}(t)}$. This yields the following forward passing of the belief, as in LDS:

$$\hat{\boldsymbol{\mu}}(t) = \mathbf{A}\boldsymbol{\mu}(t-1) + \mathbf{F}\boldsymbol{\phi}(\boldsymbol{\mu}(t-1), d_\phi) + \mathbf{b}, \tag{3}$$

$$\hat{\mathbf{P}}(t) = \mathbf{J}_{\boldsymbol{\mu}(t-1)}\mathbf{P}(t-1)\mathbf{J}_{\boldsymbol{\mu}(t-1)}^T + \mathbf{\Gamma}, \tag{4}$$

$$\mathbf{U}(t) = \mathbf{C}\hat{\mathbf{P}}(t)\mathbf{C}^T + \mathbf{R}, \tag{5}$$

$$\mathbf{K}(t) = \hat{\mathbf{P}}(t)\mathbf{C}^T\mathbf{U}^{-1}(t), \tag{6}$$

$$\boldsymbol{\mu}(t) = \hat{\boldsymbol{\mu}}(t) + \mathbf{K}(t)(\mathbf{x}(t) - \mathbf{C}\hat{\boldsymbol{\mu}}(t)), \tag{7}$$

$$\mathbf{P}(t) = (\mathbf{I} - \mathbf{K}(t)\mathbf{C})\hat{\mathbf{P}}(t). \tag{8}$$

We also obtain the backward passing equations:

$$\mathbf{V}(t) = \mathbf{P}(t)\mathbf{J}_{\boldsymbol{\mu}(t)}^T\hat{\mathbf{P}}^{-1}(t+1), \tag{9}$$

$$\hat{\mathbf{s}}(t) = \boldsymbol{\mu}(t) + \mathbf{V}(t)(\hat{\mathbf{s}}(t+1) - \hat{\boldsymbol{\mu}}(t+1)), \tag{10}$$

$$\mathbf{W}(t) = \mathbf{P}(t) + \mathbf{V}(t)(\mathbf{W}(t+1) - \hat{\mathbf{P}}(t+1))\mathbf{V}^T(t). \tag{11}$$

In this way, the optimal latent states $\{\hat{\mathbf{s}}(t)\}_{t=1}^{N}$ are estimated. For Learning, we require the following moments:

$$\mathbb{E}[\mathbf{s}(t)] = \hat{\mathbf{s}}(t), \tag{12}$$

$$\mathbb{E}[\mathbf{s}(t)\mathbf{s}^T(t)] = \mathbf{W}(t) + \hat{\mathbf{s}}(t)\hat{\mathbf{s}}^T(t), \tag{13}$$

$$\mathbb{E}[\mathbf{s}(t+1)\mathbf{s}^T(t)] = \mathbf{W}(t+1)\mathbf{V}^T(t) + \hat{\mathbf{s}}(t+1)\hat{\mathbf{s}}^T(t). \tag{14}$$

To give our problem a Learning-friendly form, we introduce a concatenated vector $\mathbf{s}_\phi(t) = [\mathbf{s}(t), \phi(\mathbf{s}(t), d_\phi)]$ and concatenated matrix $\mathbf{\Theta}_s = [\mathbf{A}, \mathbf{F}]$. We also require $\mathbb{E}[\mathbf{s}_\phi(t)]$. Here, we introduce the following moment generating function (Feller 1991).

Definition 2 (Moment generating function:$M_X(i)$)**.** *Given a vector μ and nonnegative definite matrix Σ, the moment generating function of a random variable $X \sim \mathcal{N}(\mu, \Sigma)$ is a function $M_X(i)$ defined as*

$$M_X(i) = \mathbb{E}[e^{i^T X}] (= e^{i^T \mu + \frac{1}{2} i^T \Sigma i}).$$

Remark. $M_X(i)$ is the exponential generating function of the moments of the probability distribution:

$$\mathbb{E}[X^n] = \left.\frac{d^n M_X}{di^n}\right|_{i=0}. \tag{15}$$

Since $\mathbf{s}_\phi(t)$ is a higher polynomial of $\mathbf{s}(t)$, the moment $\mathbb{E}[\mathbf{s}_\phi(t)]$ can be regarded as a higher-order moment of a multivariate Gaussian distribution, and we can generate these values with Eq. (15).

A large state noise changes the system, i.e., a different system incorporating the noise should be estimated if the state noise is large. Therefore, we can assume that the high-order state noise is sufficiently small without loss of generality. We use the following approximated moments:

$$\mathbb{E}[\mathbf{s}_\phi(t)\mathbf{s}_\phi^T(t)] = \mathbb{E}[\mathbf{s}_\phi(t)]\mathbb{E}[\mathbf{s}_\phi(t)]^T, \tag{16}$$

$$\mathbb{E}[\mathbf{s}(t+1)\mathbf{s}_\phi^T(t)] = \mathbb{E}[\mathbf{s}(t+1)]\mathbb{E}[\mathbf{s}_\phi(t)]^T. \tag{17}$$

Learning

Once we have the estimated latent states, we can run the algorithm to determine the parameter set θ. The model dynamics parameters $\mathbf{A}$ and $\mathbf{F}$ should include only a few important terms, i.e., they should be sparse matrices. However, simultaneously estimating these sparse matrices along with other parameters presents a considerable challenge. Therefore, we split parameter set θ into two subsets, i.e., $\theta_s = \{\mathbf{A}, \mathbf{F}\}$ and $\theta_{ns} = \theta \setminus \theta_s = \{\mathbf{b}, \mathbf{C}, \mathbf{u}\}$, each of which corresponds to a sparse/non-sparse parameter set, and try to fit the parameter sets separately. Algorithm 2 shows the Learning optimization process. In Learning, a set of moments (Eq. (12) - Eq. (17)) is defined as a moment set $\mathbf{M}$ as follows, which is used to estimate model parameters.

Definition 3 (Moment set: $\mathbf{M}$)**.** *Let* $\mathbf{M}$ *be a set of moments,*

$$\begin{aligned}\mathbf{M} =&\{\mathbb{E}[\mathbf{s}(t)], \mathbb{E}[\mathbf{s}(t)\mathbf{s}^T(t)], \mathbb{E}[\mathbf{s}(t+1)\mathbf{s}^T(t)],\\ &\mathbb{E}[\mathbf{s}_\phi(t)], \mathbb{E}[\mathbf{s}_\phi(t)\mathbf{s}_\phi^T(t)], \mathbb{E}[\mathbf{s}(t+1)\mathbf{s}_\phi^T(t)]\}_{t=1}^N.\end{aligned} \tag{18}$$

Fitting sparse parameter set. In each iteration, we need to minimize Eq. (1) with respect to $\mathbf{\Theta}_s$, which is equivalent to minimizing the following function $f(\mathbf{\Theta}_s)$:

$$\begin{aligned}f(\mathbf{\Theta}_s) = \mathbb{E}[&\sum_{t=1}^{N-1} D(\mathbf{s}(t+1), \mathbf{\Theta}_s\mathbf{s}_\phi(t) + \mathbf{b}, \mathbf{\Gamma})]\\ &+\frac{1}{2}\lambda_2||\mathbf{\Theta}_s - \mathbf{I}||_F^2 + \lambda_1||\mathbf{\Theta}_s - \mathbf{I}||_1.\end{aligned} \tag{19}$$

As we can see, $f(\mathbf{\Theta}_s)$ is convex but non-differentiable, and we can easily decompose $f(\mathbf{\Theta}_s)$ into two parts: $f(\mathbf{\Theta}_s) = g(\mathbf{\Theta}_s) + h(\mathbf{\Theta}_s)$, as shown in Eq. (19). Since $g(\mathbf{\Theta}_s)(= \mathbb{E}[\sum_{t=1}^{N-1} D(\mathbf{s}(t+1), \mathbf{\Theta}_s\mathbf{s}_\phi(t) + \mathbf{b}, \mathbf{\Gamma})] + \frac{1}{2}\lambda_2||\mathbf{\Theta}_s - \mathbf{I}||_F^2)$ is differentiable, we can adopt the generalized gradient descent algorithm to minimize $f(\mathbf{\Theta}_s)$. The update rule is: $\mathbf{\Theta}_s(i+1) = \text{prox}_{\alpha\lambda}(\mathbf{\Theta}_s(i) - \alpha\nabla g(\mathbf{\Theta}_s(i)) - \mathbf{I})$ where α is the step size at iteration i and the proximal function $\text{prox}_{\alpha\lambda}(\mathbf{\Theta}_s)$ is defined as the soft-threshold proximal operator $Th_{\alpha\lambda}(\beta)$, which has the following closed-form solution:

$$Th_{\alpha\lambda_1}(\beta) = \begin{cases}\beta - \alpha\lambda_1 & \text{if } \beta < \alpha\lambda_1,\\ 0 & \text{if } -\alpha\lambda_1 \le \beta \le \alpha\lambda_1,\\ \beta + \alpha\lambda_1 & \text{if } \beta < -\alpha\lambda_1.\end{cases}$$

Algorithm 2: Learning $(\{\hat{\mathbf{s}}(t)\}_{t=1}^N, \mathbf{M}, \theta)$

Input: (a) estimated latent states $\{\hat{\mathbf{s}}(t)\}_{t=1}^N$
(b) moments set $\mathbf{M}$
(c) parameter set $\theta = \{\theta_s, \theta_{ns}\}$
Output: new parameter set $\theta^{new} = \{\theta_s^{new}, \theta_{ns}^{new}\}$
$\mathbf{\Theta}_s = [\mathbf{A}, \mathbf{F}]$
/* Compute the fixed stepsize α */
$\alpha = 1/(\|\mathbf{\Gamma}^{-1}\|_F \cdot \|\sum_{t=1}^{N-1} \mathbb{E}[\mathbf{s}_\phi(t)\mathbf{s}_\phi^T(t)]\|_F + \lambda_2)$
repeat
  Compute gradient $\nabla g(\mathbf{\Theta}_s)$ // Eq. (20)
  $\mathbf{\Theta}_s = Th_{\alpha\lambda}(\mathbf{\Theta}_s - \alpha\nabla g(\mathbf{\Theta}_s))$
until convergence
$\theta_s^{new} \leftarrow \mathbf{\Theta}_s$
/* Fit other parameters (See details in Appendix D)*/
$\theta_{ns}^{new} = \arg\min_{\theta_{ns}} Q(\mathbf{X}, \theta_s^{new}, \theta_{ns})$
return $\{\theta_s^{new}, \theta_{ns}^{new}\}$

Also, the gradient $\nabla g(\mathbf{\Theta}_s)$ is as follows:

$$\begin{aligned}\nabla g(\mathbf{\Theta}_s) =&\mathbf{\Gamma}^{-1}(\sum_{t=1}^{N-1} \mathbf{\Theta}_s\mathbb{E}[\mathbf{s}_\phi(t)\mathbf{s}_\phi^T(t)] + \mathbf{b}\mathbb{E}[\mathbf{s}_\phi^T(t)]\\ &- \mathbb{E}[\mathbf{s}(t+1)\mathbf{s}_\phi^T(t)]) + \lambda_2(\mathbf{\Theta}_s - \mathbf{I}).\end{aligned} \tag{20}$$

There remains the question of how to determine the step size α. The appropriate step size is important in the gradient method. However, it is difficult to determine the step size in advance e.g., by a grid search, since each Inference updates the latent states. The appropriate step size in a linear setting has been discussed in (Liu and Hauskrecht 2015), and we can extend this to our case. Proposition 1 allows us to select the step size while assuring its fast convergence rate.

Proposition 1. *Generalized gradient descent with a fixed step size* $\alpha \le 1/(\|\mathbf{\Gamma}^{-1}\|_F \cdot \|\sum_{t=1}^{N-1} \mathbb{E}[\mathbf{s}_\phi(t)\mathbf{s}_\phi^T(t)]\|_F + \lambda_2)$ *for minimizing has a convergence rate* $O(1/i)$*, where* i *is the number of iterations.*

Proof. Please see Appendix C. □

Fitting non-sparse parameter set. To estimate the non-sparse parameter, using the derivatives of (1) with respect to each of the components of θ_{ns} and setting them at zero yields (i.e., $\frac{dQ}{d\theta_{ns}} = 0$), and here we omit the details provided in Appendix D.

4 Experimental Results

In this section, we run experiments on synthetic data where there are ground truth systems to evaluate the accuracy and robustness of our method. We compare LaNoLem to three state-of-the-art baselines to measure accuracy and robustness, and we show the importance of latent states for them.

Experimental Setup

Datasets. We use synthetic data obtained from dysts database (Gilpin 2021), which provides data, equations, and dynamical properties for chaotic systems exhibiting strange attractors and coming from disparate scientific fields. In

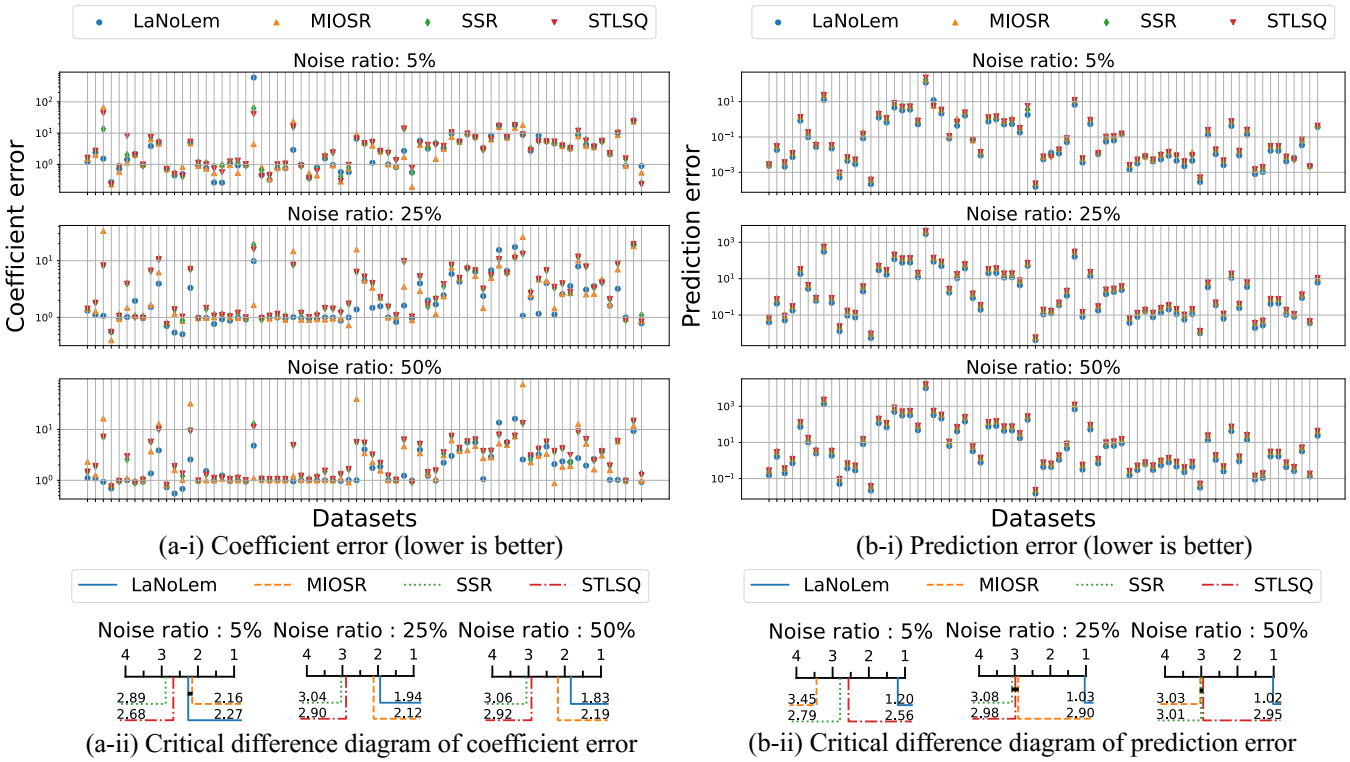

Figure 2: Accuracy and robustness for dysts dataset: (a) LaNoLem achieves competitive performance for coefficient error. (b) LaNoLem also consistently achieved the lowest prediction error.

our experiments, we consider 71 systems representing ordinary differential equations (ODEs) with polynomial nonlinearities that have no more than four degrees. Most importantly, this allows us to use the true governing equations to evaluate the model performance.

Evaluation Metrics. We use normalized coefficient error[1], which simply measures the normalized Euclidean distance between the true and the learned coefficients. The coefficients are parameters for representing dynamics (e.g., $\mathbf{A}$ and $\mathbf{F}$ are the coefficients in our method). The lower the coefficient error, the higher the modeling accuracy. We also use mean squared error (MSE) as the prediction error.

Baseline Methods. We compared our approach with the following methods: MIOSR (Bertsimas and Gurnee 2023), SSR (Boninsegna, Nüske, and Clementi 2018), and STLSQ (Brunton, Proctor, and Kutz 2016), which are optimization algorithms in the SINDy framework.

The experimental settings are detailed in Appendix E, which contains a detailed description of the experimental conditions and hyperparameters used in our study.

Accuracy and Robustness

We discuss the accuracy of LaNoLem in estimating the nonlinear dynamics of data and prediction. In this experiment, we answer the following questions: How accurately and robustly does our method estimate a system and generate future values from noisy data, namely detect the appropriate coefficients (i.e., $\mathbf{A}$ and $\mathbf{F}$) of the governing equation and predict one step ahead? We vary the noise ratio[2], each with additive Gaussian noise, when varying random seeds in each experiment. We evaluate the quality of LaNoLem in estimating the coefficients and prediction when we set the noise ratio at 5%, 25%, and 50%. Fig. 2 shows the summary of the average errors and the critical difference diagram for estimating and predicting each noise ratio. The critical difference diagrams are based on the Wilcoxon-Holm method (Ismail Fawaz et al. 2019), where methods that are not connected by a bold line are significantly different as regards average rank. Table 1 also shows the win count in the

Measures	**Coefficient error**			**Prediction error**		
Noise ratio	5%	25%	50%	5%	25%	50%
STLSQ	6	4	3	4	0	0
SSR	6	5	0	0	0	0
MIOSR	**33**	25	26	1	0	0
LaNoLem	26	**37**	**42**	**66**	**71**	**71**

Table 1: 1st Count in dysts dataset (higher is better).

[1]Coefficient error $= \frac{\|\text{True coefficients} - \text{Learned coefficients}\|_2}{\|\text{True coefficients}\|_2}$

[2]This represents the ratio of additive Gaussian noise scaled to a certain percentage of the ℓ_2 norm of each dataset, i.e., noise ratio (%) $= \frac{\ell_2 \text{ norm of noise}}{\ell_2 \text{ norm of data}} * 100$.

71 datasets. The results presented in the figures and tables demonstrate the effectiveness of LaNoLem. At a low noise ratio (5%), MIOSR is competitive with the LaNoLem in coefficient error. Fig. 2 (a-ii) also demonstrate that LaNoLem and MIOSR aren't significantly different because they are connected by a bold line at the low noise. However, as the noise level increases, the accuracy of the baselines decreases significantly, while the accuracy of LaNoLem decreases less than with the other methods. Our method also consistently achieves the best prediction error. LaNoLem achieved low coefficient error because it can filter out background noise when estimating latent states and systems using the temporal dependency of the data. On the other hand, other methods treat time series data as split samples and cannot successfully separate noise and non-linearity. As a result, they overfit the noise and reduce the accuracy. Furthermore, because the introduction of latent states provides appropriate initial conditions, LaNoLem also provides stable prediction, which contributes significantly to consistent and highly accurate prediction. We also show the numerical results, including statistical information (i.e., means and standard deviations), in Appendix G.

Ablation study. In Appendix F, we prepare a limited version of the proposed method and compare it with the original LaNoLem for a more detailed evaluation.

5 Case Studies

In this section, we describe the performance of LaNoLem using synthetic and real datasets. This section is designed to answer the following questions:

***Q1. Effectiveness*:** What are the advantages of introducing non-linear equations in latent states?

***Q2. Practicality*:** How well does LaNoLem obtain meaningful insights from real-world datasets?

Through these questions, we show how latent states can be useful in various problems. Note that existing methods without latent states, such as the SINDy framework, can't be applied directly to scenarios in these questions.

Q1. Effectiveness Here, we discuss the effectiveness of LaNoLem in interpolating missing values, a typical issue addressed in dynamical systems in latent states (Cai et al. 2015; Li et al. 2009). We show the advantages of describing the latent state with non-linear equations by comparison with rLDS (Liu and Hauskrecht 2015), a linear dynamic system that introduces regularization. Fig. 3 shows results for the Halvorsen attractor (Sprott 2010) when we set the noise ratio at 50%. Fig. 3 (i) shows the results for missing value interpolation using the LaNoLem and rLDS for noisy data. The intervals to be interpolated are each represented by a solid colored line, and the given input, except for the complementary interval, is represented by a gray line. Also, the solid orange line is noise-free (i.e., ground truth). LaNoLem (solid green line) completes a trajectory similar to the ground truth, whereas rLDS shows a completely different trajectory (solid red line). rLDS and LaNoLem introduce latent states, but rLDS is incapable of handling non-linear dynamics.

Our method can also effectively estimate dynamics even when the given data contains missing values. Each row

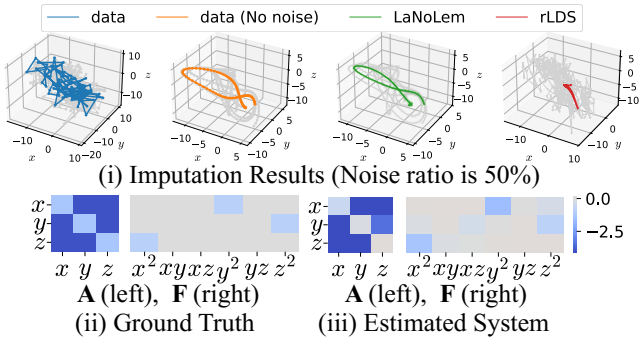

Figure 3: Effectiveness of LaNoLem: LaNoLem can accurately estimate the system and interpolate missing trajectories even when those are noisy and missing data.

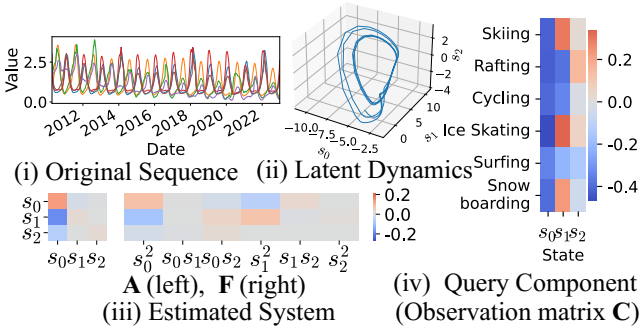

Figure 4: Practicality of LaNoLem: LaNoLem can estimate (ii) latent dynamics, (iii) the system, and (iv) query components in (i) Google Trends data, which consist of the search volumes for six outdoor-related keywords.

of the heat maps in Fig. 3 (ii) and (iii) corresponds to $\frac{dx}{dt}$, $\frac{dy}{dt}$, and $\frac{dz}{dt}$. As shown in Fig. 3 (ii), the dynamics of Halvorsen have linear terms $\{x, y, z\}$ and non-linear terms $\{x^2, y^2, z^2\}$. For example, the first row of the heatmap corresponds to $\frac{dx}{dt}$. This heatmap shows that $\frac{dx}{dt}$ contains the terms $\{x, y, z, y^2\}$. The same is true for the other rows. All these results show that LaNoLem can estimate accurate and meaningful dynamics.

Q2. Practicality In this section, we present each example using the following real dataset.

Web activity data: The dataset contains web-search counts related to outdoor query sets collected over twelve years from 2010 to 2022.

Here, we show what features LaNoLem can extract from real-world data sets. LaNoLem allows flexible modeling by introducing latent states. As a result, our method separates noise and non-linearity well. In addition, our method can estimate the dynamics in low-dimensional latent states (i.e., $k < d$) that existing studies cannot estimate. Modeling in low-dimensional latent states is important because it can be applied to various time series analyses (e.g., forecasting (Baier, Aspandi, and Staab 2023; Matsubara and Sakurai 2019)). Fig. 4 shows our results regarding search volume data. Fig. 4 (i) shows the original data. LaNoLem

can discover (ii) the latent states and (iii) the system, with which it factorizes the data into (iv) query factors, where six queries are represented as three latent states. We can interpret the circumstances of web activities from the connection between the latent states and query factors. For example, skiing, ice skating, and snowboarding are each assigned a positive weight with respect to state s_1, while cycling, surfing, and rafting are each assigned a negative weight. This agrees with our intuition: skiing, ice skating, and snowboarding are winter activities, while the others are summer activities. Furthermore, Fig 4 (ii) shows the estimated latent state. This is generated from the system in Fig 4 (iii), and the trajectory seems to have a limit cycle. This indicates the latent periodicity of each activity.

6 Related Work

In this section, we briefly describe investigations related to this research. Table 2 shows the relative advantages of our method, and only LaNoLem meets all the requirements. We separate the details of previous studies into two categories: modeling non-linear dynamics and time series analysis.

Modeling non-linear dynamics. Estimating mathematical expressions from data represents a significant challenge in a wide range of diverse areas of science and engineering. A seminal work, SINDy, has been published on sparse regression methods for system identification (Brunton, Proctor, and Kutz 2016). In that study, the model was estimated using sequentially thresholded least squares (STLSQ). More recently, representative algorithms for the SINDy framework, namely stepwise sparse regression (SSR) (Boninsegna, Nüske, and Clementi 2018) and mixed-integer optimized sparse regression (MIOSR) (Bertsimas and Gurnee 2023) have been proposed. These methods are effective because they provide interpretable models (systems) for various types of data. However, none of these methods are intended for estimating latent states and dynamics; therefore, they have lower model estimation for noisy data. Moreover, extracting interpretable components from the data and representing their dynamics is impossible. Symbolic regression (SR) can identify the tractable formula, i.e., $y = f(x)$, that best fits a given data pair (x, y) (Shojaee et al. 2023; Landajuela et al. 2022; Burlacu, Kronberger, and Kommenda 2020; La Cava et al. 2018; Udrescu and Tegmark 2020). These methods can accurately estimate the non-linear relationship between given variables. However, these methods cannot extract latent states and estimate their latent dynamics from multiple time series. Moreover, they need target variables (i.e., derivatives) when modeling dynamics such as ODEs. Unlike our methods and SINDy frameworks, these methods must address noise amplification when calculating the derivative value from the noisy data. Therefore, these methods cannot be applied directly to the problem of estimating models from time series, as discussed in this study.

Time series analysis. Linear dynamical systems (LDS) (Kalman 1963) utilize latent space to capture temporal dependencies and learn latent dynamics. They have been applied to various analytical tasks, including forecasting, clustering, and pattern mining (Li et al. 2009; Li, Prakash, and Faloutsos 2010; Cai et al. 2015; Dabrowski et al. 2018; Hairi, Tong, and Ying 2019; Chen and Poor 2022). Switching linear dynamical systems (SLDS) (Pavlovic, Rehg, and MacCormick 2000; Fox et al. 2008) enhance this framework by incorporating both discrete states that represent different motion modes and continuous states that describe the dynamics of each mode. This allows the representation of more complex time series. The regularized LDS model proposed in (Liu and Hauskrecht 2015) aims to learn an LDS model with a low-rank transition matrix from a limited number of sequences. Compared with regular LDS models, regularized LDS is able to find the essential dimensions of hidden states and prevent overfitting problems in advance. While effective, these approaches typically assume state transitions governed by linear equations. Estimating non-linear dynamical systems has also been attracting the attention of many researchers (Ghahramani and Roweis 1998; Kowshik et al. 2021; Sattar and Oymak 2022; Foster, Sarkar, and Rakhlin 2020; Kakade et al. 2011). However, none of these non-linear models focus on modeling non-linear dynamics as mathematical expressions.

	LDS/++	SR/++	SINDy/++	**LaNoLem**
Non-linear equation	-	✓	✓	✓
Sparse Term	-	some	✓	✓
Robustness to differentiation	-	-	✓	✓
Modeling latent dynamics	✓	-	-	✓

Table 2: Capabilities of approaches.

7 Conclusion

This paper presented LaNoLem, an intuitive method for estimating both latent states and non-linear dynamics. Our main idea is to introduce latent states to allow the estimation of dynamics that consider the temporal dependencies of the data. The important point is that the latent non-linear dynamical system and various dynamical systems should consist of fewer terms than the space of possible functions. Therefore, we propose a non-linear dynamical system consisting of latent states and various non-linear terms of states. We also propose new criteria to control its complexity. Finally, we develop a new alternating minimization algorithm to estimate our model effectively. We also demonstrated the practicality and effectiveness of LaNoLem on various types of time series. We can highlight the following key results:

- We have introduced an effective model that represents latent non-linear dynamics and criteria for evaluating the trade-offs between model accuracy and complexity.
- We presented an efficient algorithm capable of simultaneously estimating latent states and model parameters, considering non-linear state transitions and sparsity in the parameters.
- When benchmarked against state-of-the-art baselines, our model demonstrates competitive performance in estimating dynamics and maintains the leading performance in prediction tasks.

Acknowledgements

This work was partly supported by JST CREST JPMJCR23M3, JSPS KAKENHI Grant-in-Aid for Scientific Research Number JP24KJ1618.

References

Baier, A.; Aspandi, D.; and Staab, S. 2023. ReLiNet: Stable and Explainable Multistep Prediction with Recurrent Linear Parameter Varying Networks. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, 3461–3469.

Bertsimas, D.; and Gurnee, W. 2023. Learning sparse nonlinear dynamics via mixed-integer optimization. *Nonlinear Dynamics*, 111(7): 6585–6604.

Boninsegna, L.; Nüske, F.; and Clementi, C. 2018. Sparse learning of stochastic dynamical equations. *The Journal of Chemical Physics*, 148(24): 241723.

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

Burlacu, B.; Kronberger, G.; and Kommenda, M. 2020. Operon C++: An Efficient Genetic Programming Framework for Symbolic Regression. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, GECCO '20, 1562–1570.

Cai, Y.; Tong, H.; Fan, W.; and Ji, P. 2015. Fast Mining of a Network of Coevolving Time Series. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, 298–306. SIAM.

Champion, K.; Zheng, P.; Aravkin, A. Y.; Brunton, S. L.; and Kutz, J. N. 2020. A unified sparse optimization framework to learn parsimonious physics-informed models from data. *IEEE Access*, 8: 169259–169271.

Chen, Y.; and Poor, H. V. 2022. Learning mixtures of linear dynamical systems. In *International Conference on Machine Learning*, 3507–3557. PMLR.

Dabrowski, J. J.; Rahman, A.; George, A.; Arnold, S.; and McCulloch, J. 2018. State Space Models for Forecasting Water Quality Variables: An Application in Aquaculture Prawn Farming. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 177–185. ACM.

Feller, W. 1991. *An introduction to probability theory and its applications, Volume 2*, volume 81. John Wiley & Sons.

Foster, D.; Sarkar, T.; and Rakhlin, A. 2020. Learning nonlinear dynamical systems from a single trajectory. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120, 851–861.

Fox, E. B.; Sudderth, E. B.; Jordan, M. I.; and Willsky, A. S. 2008. Nonparametric Bayesian Learning of Switching Linear Dynamical Systems. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*, 457–464.

Ghahramani, Z.; and Roweis, S. 1998. Learning nonlinear dynamical systems using an EM algorithm. *Advances in neural information processing systems*, 11.

Gilpin, W. 2021. Chaos as an interpretable benchmark for forecasting and data-driven modelling. In Vanschoren, J.; and Yeung, S., eds., *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

Grünwald, P. D. 2007. *The Minimum Description Length Principle*. The MIT Press.

Hairi; Tong, H.; and Ying, L. 2019. NetDyna: Mining Networked Coevolving Time Series with Missing Values. In *2019 IEEE International Conference on Big Data (IEEE BigData)*, 503–512. IEEE.

Ismail Fawaz, H.; Forestier, G.; Weber, J.; Idoumghar, L.; and Muller, P.-A. 2019. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4): 917–963.

Kakade, S. M.; Kanade, V.; Shamir, O.; and Kalai, A. 2011. Efficient Learning of Generalized Linear and Single Index Models with Isotonic Regression. In *Advances in Neural Information Processing Systems*, volume 24.

Kalman, R. E. 1963. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics, Series A: Control*, 1(2): 152–192.

Kaptanoglu, A. A.; Callaham, J. L.; Aravkin, A.; Hansen, C. J.; and Brunton, S. L. 2021. Promoting global stability in data-driven models of quadratic nonlinear dynamics. *Physical Review Fluids*, 6(9): 094401.

Kelley, W. G.; and Peterson, A. C. 2001. *Difference equations: an introduction with applications*. Academic press.

Kowshik, S.; Nagaraj, D.; Jain, P.; and Netrapalli, P. 2021. Near-optimal offline and streaming algorithms for learning non-linear dynamical systems. *Advances in Neural Information Processing Systems*, 34: 8518–8531.

La Cava, W.; Singh, T. R.; Taggart, J.; Suri, S.; and Moore, J. H. 2018. Learning concise representations for regression by evolving networks of trees. In *International Conference on Learning Representations*.

Landajuela, M.; Lee, C. S.; Yang, J.; Glatt, R.; Santiago, C. P.; Aravena, I.; Mundhenk, T.; Mulcahy, G.; and Petersen, B. K. 2022. A Unified Framework for Deep Symbolic Regression. In *Advances in Neural Information Processing Systems*, volume 35, 33985–33998.

Li, L.; McCann, J.; Pollard, N. S.; and Faloutsos, C. 2009. DynaMMo: mining and summarization of coevolving sequences with missing values. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 507–516. ACM.

Li, L.; Prakash, B. A.; and Faloutsos, C. 2010. Parsimonious Linear Fingerprinting for Time Series. *Proc. VLDB Endow.*, 3(1): 385–396.

Liu, Z.; and Hauskrecht, M. 2015. A Regularized Linear Dynamical System Framework for Multivariate Time Series Analysis. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 1798–1804. AAAI Press.

Matsubara, Y.; and Sakurai, Y. 2019. Dynamic Modeling and Forecasting of Time-evolving Data Streams. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 458–468. ACM.

Pavlovic, V.; Rehg, J. M.; and MacCormick, J. 2000. Learning Switching Linear Models of Human Motion. In *Advances in Neural Information Processing Systems*, volume 13, 981–987. MIT Press.

Pearl, J. 1982. Reverend bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, 133–136.

Sattar, Y.; and Oymak, S. 2022. Non-asymptotic and Accurate Learning of Nonlinear Dynamical Systems. *Journal of Machine Learning Research*, 23(140): 1–49.

Shojaee, P.; Meidani, K.; Barati Farimani, A.; and Reddy, C. 2023. Transformer-based Planning for Symbolic Regression. In *Advances in Neural Information Processing Systems*, volume 36, 45907–45919.

Sprott, J. C. 2010. *Elegant chaos: algebraically simple chaotic flows*. World Scientific.

Udrescu, S.-M.; and Tegmark, M. 2020. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16): eaay2631.

Zarchan, P. 2005. *Progress in astronautics and aeronautics: fundamentals of Kalman filtering: a practical approach*, volume 208. Aiaa.

Zou, H.; and Hastie, T. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2): 301–320.