Amortized Equation Discovery in Hybrid Dynamical Systems

Yongtuo Liu[1] **Sara Magliacane**[1] **Miltiadis Kofinas**[1] **Efstratios Gavves**[1]

Abstract

Hybrid dynamical systems are prevalent in science and engineering to express complex systems with continuous and discrete states. To learn laws of systems, all previous methods for equation discovery in hybrid systems follow a two-stage paradigm, i.e. they first group time series into small cluster fragments and then discover equations in each fragment separately through methods in non-hybrid systems. Although effective, these methods do not take fully advantage of the commonalities in the shared dynamics of multiple fragments that are driven by the same equations. Besides, the two-stage paradigm breaks the interdependence between categorizing and representing dynamics that jointly form hybrid systems. In this paper, we reformulate the problem and propose an end-to-end learning framework, i.e. Amortized Equation Discovery (AMORE), to jointly categorize modes and discover equations characterizing the dynamics of each mode by all segments of the mode. Experiments on four hybrid and six non-hybrid systems show that our method outperforms previous methods on equation discovery, segmentation, and forecasting.

1. Introduction

Complex systems in science and engineering often exhibit behaviors and patterns that change over time. Hybrid dynamical systems (Van Der Schaft & Schumacher, 2007) characterize these systems by continuous time series which are interleaved with structural changes producing discrete modes. For instance, consider the motions of antelopes in a herd and how these suddenly change in the presence of lions. Hybrid systems are researched widely with applications in epidemiology (Keeling et al., 2001), legged locomotion (Holmes et al., 2006), robotics (Cortes, 2008), the designs of cyber-physical systems (Sanfelice et al., 2016), and systems with interacting objects (Liu et al., 2023).

A major challenge with hybrid dynamical systems is that one cannot know a priori the number of possible modes and when the switching happens within them. The dynamic modes might alternate from one to another constantly and at any time, due to either internal mechanisms or external influences. When modeling generalized time series as hybrid dynamical systems, it is thus crucial that we categorize the complex dynamics into the most likely discrete modes while characterizing the continuous motion dynamics in between.

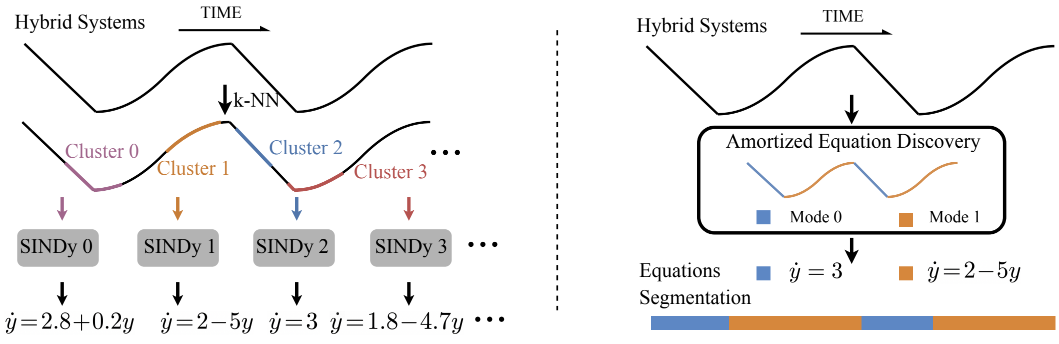
Another challenge with characterizing dynamics in hybrid systems, especially physical ones, is that predictive models are often not interpretable. We are often interested in the underlying laws that govern the dynamics, thus preferring analytic models, usually in the form of closed-form ordinary differential equations. Equation discovery from first principles is a challenging problem in all fields of science. To bypass expensive and cumbersome targeted experimentation, researchers have explored using data-driven methods for equation discovery of systems from observations (Langley, 1981; Lemos et al., 2023). They distill parsimonious equations from data and find that compared with black-box neural networks, learned equations can provide insight into the essential dynamics of systems and tend to generalize better (Lutter et al., 2019; Karniadakis et al., 2021).

Equation discovery for hybrid dynamical systems has been a topic of interest for a long time (Vidal et al., 2003; Ozay et al., 2008; Bako, 2011; Ohlsson & Ljung, 2013). Recently, Mangan et al. (2019); Novelli et al. (2022) proposed methods for equation discovery in non-linear hybrid systems. Both methods consist of two stages: they first group time series fragments into a large number of small cluster fragments and then apply an equation discovery method proposed in non-hybrid systems, e.g. SINDy (Brunton et al., 2016), to discover equations in each fragment separately. The separate multi-stage processing limits the potential performance because it does not leverage the commonalities across fragments from the same mode and splits learning into two separate stages, i.e. categorizing and then representing motion dynamics which jointly form hybrid systems.

In this paper, we reformulate the problem of equation discovery in hybrid dynamical systems and propose a one-

[1] University of Amsterdam. Correspondence to: Yongtuo Liu <y.liu6@uva.nl>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).



(a) Previous methods

(b) Our method (AMORE)

Figure 1. (a) Previous methods for equation discovery in hybrid dynamical systems typically follow a two-stage paradigm, i.e. they first group time series into small cluster fragments and then apply methods proposed in non-hybrid systems, e.g. SINDy (Brunton et al., 2016) to discover equations in each fragment separately. (b) Different from all previous methods, we reformulate the problem and propose a one-stage end-to-end learning framework, Amortized Equation Discovery (a.k.a. AMORE), to jointly categorize hybrid systems into discrete modes and discover equations characterizing motion dynamics of each mode based on all segments belonging to the mode.

stage end-to-end learning framework, Amortized Equation Discovery (a.k.a. AMORE), to jointly categorize motion dynamics and discover equations by modeling categorical modes and mode-switching behaviors within systems. Equations are discovered to characterize the dynamics of each mode based on all segments that are assigned to the mode, by learning combinations of candidate basis functions and encouraging parsimony. To model switching behaviors, inspired by REDSDS (Ansari et al., 2021), we infer latent categorical variables, i.e. mode variables, to categorize motion dynamics into discrete modes and learn probabilistic transition behaviors within them. Equations, mode variables, and mode-switching behaviors are jointly learned in the proposed end-to-end learning framework by maximizing the system observation likelihood. We also consider another challenge in previous methods for equation discovery for hybrid systems: they are limited to single-object scenarios where the dynamics of only one object or objects as a whole are considered. We extend our method to multi-object scenarios, AMORE-MIO, where multiple objects interact with each other and change their dynamics accordingly. Extensive experiments on single- and multi-object hybrid systems demonstrate the superior performance of our method on equation discovery, segmentation, and forecasting. The code and datasets are available at https://github.com/yongtuoliu/Amortized-Equation-Discovery-in-Hybrid-Dynamical-Systems.

2. Related Work

Equation discovery in hybrid dynamical systems. Prior works focus on the simplest hybrid dynamical models, i.e. piece-wise affine systems with linear transition rules (Vidal et al., 2003; Ferrari-Trecate et al., 2003; Roll et al., 2004; Juloski et al., 2005; Paoletti et al., 2007; Ozay et al., 2008; Bako, 2011; Ohlsson & Ljung, 2013). Recently, Mangan et al. (2019); Novelli et al. (2022) relieve these constraints and propose methods for non-linear hybrid systems. Among them, Hybrid-SINDy (Mangan et al., 2019) proposes a two-stage method, i.e. it first uses k-NN to group time series into small cluster fragments and then discovers governing equations separately in each fragment by models proposed in non-hybrid systems, e.g. SINDy (Brunton et al., 2016). Based on Hybrid-SINDy, Novelli et al. (2022) also follows a two-stage paradigm while introducing the number of discontinuities in hybrid systems as a known prior for better performance. Although effective, these two-stage methods learn the mode of each segment individually and do not leverage the commonalities across segments. In this paper, we reformulate the problem and propose an amortized end-to-end learning framework to jointly categorize modes, discover equations, and learn mode-switching behaviors.

Equation discovery in non-hybrid dynamical systems. Many methods have been proposed for equation discovery in non-hybrid dynamical systems. Bongard & Lipson (2007) and Schmidt & Lipson (2009) leverage genetic programming (Koza et al., 1994) to discover nonlinear differential equations from data. SINDy (Brunton et al., 2016) uses symbolic sparse regression on a library of candidate model terms to select the fewest terms required to describe the observed dynamics. Several methods extend this approach to new settings, e.g. identifying partial differential equations (Rudy et al., 2017), considering additional physical constraints (Loiseau & Brunton, 2018), including control signals (Kaiser et al., 2018), and introducing integral terms for denoising (Schaeffer & McCalla, 2017). These methods cannot be directly applied to hybrid systems because they cannot model an unknown number of modes and unknown mode-switching behaviors.

Switching dynamical systems. Switching dynamical systems refer to the same systems as hybrid dynamical systems, but highlight different aspects in the literature. Hybrid systems denote systems with a mixture of continuous and discrete states, while switching dynamical systems highlight the switching behaviors of system states. Many methods focus on switching linear dynamical systems where they set matrix calculations to model linear state transitions (Ackerson & Fu, 1970; Ghahramani & Hinton, 2000; Oh et al., 2005). Recently, switching non-linear dynamical systems model state transitions as neural networks, e.g. SNLDS (Dong et al., 2020), REDSDS (Ansari et al., 2021), and GRASS (Liu et al., 2023). While effective in modeling state-switching behaviors, they cannot discover closed-form equations from data. To categorize dynamics, our method is inspired by previous switching dynamical systems (Dong et al., 2020; Ansari et al., 2021; Liu et al., 2023) to infer latent mode variables. Differently, our method can jointly discover parsimonious closed-form equations to characterize dynamics and infer the values of the mode variables.

3. Equation Discovery in Dynamical Systems

In dynamical systems, the dynamics can be expressed by sets of differential equations in the form:

$$\dot{\mathbf{y}}_t := \frac{d\mathbf{y}_t}{dt} = \mathbf{f}(\mathbf{y}_t). \tag{1}$$

Equation discovery in dynamical systems is the task of learning the function $\mathbf{f} : \mathbb{R}^M \to \mathbb{R}^M$ from time-series observations $\mathbf{y} = \{\mathbf{y}_1, \cdots, \mathbf{y}_T\}$ where each state $\mathbf{y}_t = [y_t^1, \cdots, y_t^M] \in \mathbb{R}^M$. Following SINDy (Brunton et al., 2016), we approximate each dimension $\dot{y}_t^m$ for $m \in [M]$ of $\dot{\mathbf{y}}_t$ in Eq. (1) as

$$\dot{y}_t^m = \frac{dy_t^m}{dt} = f_m(\mathbf{y}_t) \approx \Theta(\mathbf{y}_t)\xi_m, \tag{2}$$

where $\Theta(\mathbf{y}_t) = [\theta_1(\mathbf{y}_t), \theta_2(\mathbf{y}_t), \cdots, \theta_P(\mathbf{y}_t))]$ is a set of candidate basis functions and ξ_m is a sparse vector indicating which of these function terms are active in characterizing the dynamics. We encourage the sparsity of ξ_m based on Occam's razor principle, where the simplest model is likely the correct one. Ideally, we could encourage this principle by minimizing the L_0 norm of the coefficients and solving the following constrained minimization problem

$$\min_{\xi} ||\xi||_0 \quad \text{subject to} \quad ||\Theta(\mathbf{y}_t)\xi - \dot{\mathbf{y}}_t|| \leqslant \epsilon, \tag{3}$$

where ϵ is a hyperparameter representing maximal optimization errors. The L_0 regularization penalizes the number of non-zero entries to encourage sparsity. However, optimization under this penalty is computationally intractable due to the non-differentiability and the combinatorial nature of all possible states. Various continuous relaxation methods are proposed in the literature to handle the optimization problems of L_0 norm, e.g. L_1, L_2, etc. As our focus in this paper is not to design advanced optimization methods, we utilize the simple and effective L_1 norm to optimize Eq. (3).

We implement the coefficients ξ_m as learnable weights in neural networks. We set the polynomial degree as D and use a set of learnable weights $\mathbf{w} = [w_1, \cdots, w_C]$ to model the coefficients of C candidate basis functions. For instance, if the observation $\mathbf{y}_t = [a, b]$ is a two-dimensional vector and we set the polynomial degree D as 2, the candidate basis polynomial functions are $\Theta(\mathbf{y}_t) = [1, a, b, a^2, b^2, ab]$. In this case, $C = 6$ and $\mathbf{w} = [w_1, \cdots, w_6]$.

4. Equation Discovery in Hybrid Systems

Hybrid dynamical systems produce generalized time series with continuous states and discrete events that need to be modeled, featuring multiple modes that represent different types of dynamics. Instead of learning a single equation for each dimension $m \in [M]$, as described in Sec. 3, we learn K sets of equations for each dimension m that represent K different modes in hybrid systems. We first introduce how we model mode-switching behaviors and then introduce our whole framework for equation discovery in hybrid systems.

Mode variables. To model modes and mode-switching behaviors in hybrid systems, inspired by REDSDS (Ansari et al., 2021), we introduce latent categorical variables, i.e. mode variables, to learn categorical distributions of modes and index each set of equations representing each type of dynamics. Specifically, mode variables are discrete variables $\mathbf{z} := \mathbf{z}_{1:T}$, where each $z_t \in \{1, \ldots, K\}$ categorizes the mode at time step $t \in \{1, \ldots, T\}$.

Count variables. Besides mode variables, we also model latent count variables to learn the duration distributions of each mode. Count variables can help us avoid frequent mode switching, thanks to the fact that mode durations typically follow a geometric distribution (Ansari et al., 2021). They are modeled as discrete categorical variables $\mathbf{c} := \mathbf{c}_{1:T}$, where each state $c_t \in \{1, \ldots, d_{\max}\}$ explicitly models the run-length of the currently active mode at time t and $\mathrm{d}_{\max}$ is the maximal number of steps before a mode switch. Count variables c_t are incremented by 1 when the mode $z_t = k$ stays the same at the next time step $z_{t+1} = k$, or they reset to 1 when mode $z_t = k$ switches to another one $z_t \neq k$.

Mode-specific equation discovery. Each mode $k \in [K]$ is assigned its own set of candidate basis functions $\Theta_k(\mathbf{y}_t)$ and learnable coefficient weights $\mathbf{w}_k$, which we will use to discover its equation. For instance, at time t, the mode variable $z_t = k$ indexes the candidate basis function $\Theta_k(\mathbf{y}_t)$ and the learnable weights $\mathbf{w}_k$, which together define the

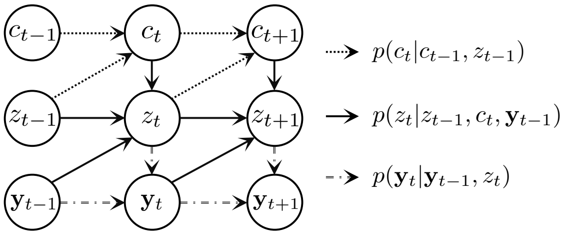

c_{t-1}
c_t
c_{t+1}
z_{t-1}
z_t
z_{t+1}
$\mathbf{y}_{t-1}$
$\mathbf{y}_t$
$\mathbf{y}_{t+1}$
$p(c_t|c_{t-1}, z_{t-1})$
$p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1})$
$p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t)$

Figure 2. Generative model for amortized equation discovery. $p(c_t|c_{t-1}, z_{t-1})$ and $p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1})$ are count and mode transition probabilities, respectively. $p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t)$ denotes the observation transition probability where equations are discovered to characterize the dynamics of each mode.

equation representing the dynamics of mode k. In practice, different modes share the same candidate basis functions, i.e. $\Theta_j(\mathbf{y}_t) = \Theta_k(\mathbf{y}_t), \forall j, k \in \{1, \cdots, K\}$, unless we have some prior knowledge of the hybrid system. However, the learnable coefficient weights of different modes are individual and never shared, i.e. $\mathbf{w}_j \neq \mathbf{w}_k, \forall j, k \in \{1, \cdots, K\}$. We collect all the candidate basis functions in a single vector $\Theta(\mathbf{y}_t) = (\Theta_1(\mathbf{y}_t), \cdots, \Theta_K(\mathbf{y}_t))$ and similarly we collect all learnable coefficient weights $\mathbf{w} = (\mathbf{w}_1, \cdots, \mathbf{w}_K)$.

Generative model for AMORE. Assuming Markovian dynamics, the joint generative probability of hybrid systems in our model is described as

$$p(\mathbf{y}, \mathbf{z}, \mathbf{c}) = \underbrace{p(\mathbf{y}_1|z_1)\, p(z_1)}_{\text{Initial States}} \cdot \prod_{t=2}^{T} \Big[p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t) \\ p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1}) p(c_t|c_{t-1}, z_{t-1}) \Big] \quad (4)$$

In the initial states, count variables are ignored as they are always 1 when starting. $p(z_1)$ is the initial distribution over all possible modes. $p(\mathbf{y}_1|z_1)$ models the initial observation states conditioned on the initial modes. For later time steps $t \geqslant 2$, the count transition probability $p(c_t|c_{t-1}, z_{t-1})$ models how the count variables at time t change over time depending on their previous values and mode variables at time $t-1$. The mode transition probability $p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1})$ models mode-switching behaviors on how modes switch at time t conditioned on the previous mode and observation states at time $t-1$ as well as the updated count state c_t at time t. The observation transition probability $p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t)$ models how the observations at time t are influenced by their previous values at time $t-1$ conditioned on the updated mode variables at time t. Equations are amortized and learned at $p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t)$ by all segments of each mode to characterize mode dynamics. More specifically, conditioned on motion mode $z_t = k$, $p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t)$ first indexes a set of candidate basis functions Θ_k and coefficient weights $\mathbf{w}_k$, which are used together to obtain derivatives $\dot{\mathbf{y}}_{t-1} = \Theta_k \cdot \mathbf{w}_k$ of $\mathbf{y}_{t-1}$ at time $t-1$. With known time intervals Δ_t, we finally achieve $\mathbf{y}_t = \dot{\mathbf{y}}_{t-1} \cdot \Delta_t + \mathbf{y}_{t-1}$ assuming the dynamics do not change much in short time intervals. For inference of the latent mode and count variables, we conduct exact inference similar to the forward-backward algorithm in HMM (Eddy, 1996). The graphical model for the exact inference is the same as the generative model, which is illustrated in Figure 2. The neural network implementations and details of the inference model are in Appendix A.1 and A.2.

Learnable parameters of AMORE are optimized by maximizing the observation likelihood with sparse regularization on coefficient weights $\mathbf{w}$ of candidate basis functions

$$\begin{aligned} \mathcal{L}_{\text{AMORE}} &= -\log \mathrm{p}_\theta(\mathbf{y}) + ||\mathbf{w}||_1 \\ &= -\mathbb{E}_{p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y})} \left[\log p_\theta(\mathbf{y}, \mathbf{z}, \mathbf{c}) \right] + ||\mathbf{w}||_1. \end{aligned} \quad (5)$$

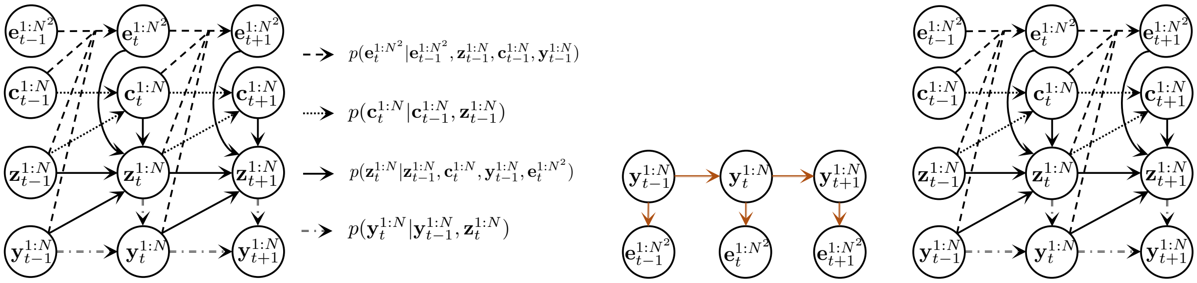
The derivatives of the training objective and further expansions over time are detailed in Appendix A.3.

5. Equation Discovery in Multi-object Hybrid Systems

While equation discovery in hybrid dynamical systems has been researched in single-object scenarios, the more general setting of systems with multiple potentially interacting objects is an unexplored yet natural setting. In this section, we elaborate on how our model can be extended for multi-object scenarios, and present AMORtized Equation discovery in MultI-Object hybrid systems, a.k.a. AMORE-MIO. We first introduce how our method models interactions and then introduce the whole framework of AMORE-MIO.

Edge variables. Assume that N objects and K motion modes exist in multi-object hybrid systems. Inspired by Kipf et al. (2018); Liu et al. (2023), we model interactions between objects by latent edge variables $\mathbf{e} = \mathbf{e}_{1:T}^{1:N^2} = \{e_t^1, \cdots, e_t^{N^2}\}_{t=1}^T$ including self-loop, thus totally N^2 for N objects. For each pair of objects, interactions $e_t^{m \to n}$ model whether object m interacts with object n at time t. The edge variables are modeled in a latent temporal graph $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, where edges $e_t^{m \to n} \in \mathcal{E}_t$ and nodes $\mathcal{V}_t$ summarize states of objects. For instance, $\mathbf{v}_t^m = \{z_t^m, c_t^m, \mathbf{y}_t^m\}$ for $\mathbf{v}_t^m \in \mathcal{V}_t$ defines one graph node summarizing states of object m at time t which includes observation $\{\mathbf{y}_t^m\}$ and latent states $\{z_t^m, c_t^m\}$. Edge $e_t^{m \to n}$ signals interaction relationships between node $\mathbf{v}_t^m$ and node $\mathbf{v}_t^n$ in graph $\mathcal{G}_t$.

Object-shared and mode-specific equation discovery. We set the number of all possible motion modes as K in multi-object hybrid dynamical systems. The K motion modes are shared across N objects, which are implemented by K sets of candidate basis functions $\Theta(\mathbf{y}_t) = (\Theta_1(\mathbf{y}_t), \cdots, \Theta_K(\mathbf{y}_t))$ and learnable coefficient weights $\mathbf{w} = (\mathbf{w}_1, \cdots, \mathbf{w}_K)$. Each mode $k \in [K]$ has its own


$\mathbf{e}_{t-1}^{1:N^2}$
$\mathbf{e}_{t}^{1:N^2}$
$\mathbf{e}_{t+1}^{1:N^2}$
$\mathbf{c}_{t-1}^{1:N}$
$\mathbf{c}_{t}^{1:N}$
$\mathbf{c}_{t+1}^{1:N}$
$\mathbf{z}_{t-1}^{1:N}$
$\mathbf{z}_{t}^{1:N}$
$\mathbf{z}_{t+1}^{1:N}$
$\mathbf{y}_{t-1}^{1:N}$
$\mathbf{y}_{t}^{1:N}$
$\mathbf{y}_{t+1}^{1:N}$
$p(\mathbf{e}_t^{1:N^2}|\mathbf{e}_{t-1}^{1:N^2}, \mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N})$
$p(\mathbf{c}_t^{1:N}|\mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N})$
$p(\mathbf{z}_t^{1:N}|\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t}^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{e}_{t}^{1:N^2})$
$p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_{t}^{1:N})$

(a) Generative Model

(b) Inference Model

Figure 3. (a) Generative model of AMORE-MIO. $p(\mathbf{e}_t^{1:N^2}|\mathbf{e}_{t-1}^{1:N^2}, \mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N})$, $p(\mathbf{c}_t^{1:N}|\mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N})$, $p(\mathbf{z}_t^{1:N}|\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t}^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{e}_{t}^{1:N^2})$, and $p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_{t}^{1:N})$ denotes the edge, count, mode, and observation transition probabilities, respectively. Equations are modeled at $p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_{t}^{1:N})$ which characterize object-shared and mode-specific dynamics. (b) Inference model of AMORE-MIO. Left: posterior approximate inference of edge variables $\mathbf{e}_t^{1:N^2}$. Right: Exact inference of discrete mode and count variables $\mathbf{z}_t^{1:N}$ and $\mathbf{c}_t^{1:N}$ based on observations $\mathbf{y}_t^{1:N}$ and the approximate edge variables $\mathbf{e}_t^{1:N^2}$. Orange arrows denote the approximate inference flow.

$\Theta_k(\mathbf{y}_t)$ and $\mathbf{w}_k$ as in Sec. 4. Thus there are K sets of learnable weights for learning dynamics of K modes across N objects. Both the time and space complexity of AMORE-MIO regarding learnable weights of basis functions is $\mathcal{O}(K)$. For instance, the mode variable $z_t^n = k$ of the object n at time t indexes $\Theta_k(\mathbf{y}_t)$ and $\mathbf{w}_k$ which together form equations to represent the dynamics of mode k. Different from single-object scenarios, the mode-switching behaviors of each object are not only influenced by their own evolving nature but also by external influences of potentially interacting objects. We model the influences of interactions on the mode-switching behaviors between objects, which are detailed in the following generative model of AMORE-MIO.

Generative model for amortized equation discovery in multi-object settings. Assuming Markovian dynamics, the joint generative probability of multi-object hybrid systems in AMORE-MIO is calculated as

$$p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c}) = \underbrace{p(\mathbf{y}_1^{1:N}|\mathbf{z}_1^{1:N})p(\mathbf{z}_1^{1:N})}_{\text{Initial States}} \cdot$$
$$\prod_{t=2}^{T}\Big[p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_t^{1:N})p(\mathbf{z}_t^{1:N}|\mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{e}_t^{1:N^2})$$
$$p(\mathbf{c}_t^{1:N}|\mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N})p(\mathbf{e}_t^{1:N^2}|\mathbf{e}_{t-1}^{1:N^2}, \mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N})\Big] \quad (6)$$

where the initial states and count transition probability are defined as in Eq. 4 of single-object scenarios but with n objects. For later time steps $t \geqslant 2$, the edge transition probability $p(\mathbf{e}_t^{1:N^2}|\mathbf{e}_{t-1}^{1:N^2}, \mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N})$ models how the edge variables evolve depending on all the states at the previous time step. We model the influences of interactions on the mode transition probability $p(\mathbf{z}_t^{1:N}|\mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{e}_t^{1:N^2})$, which characterizes the mode-switching behaviors of multi-object hybrid dynamical systems. Based on the updated modes of each object, the observation transition probability $p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_t^{1:N})$ can be factorized over objects $\prod_{n=1}^{N} p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n)$ where equations of each mode are amortized and learned by all segments from all objects belonging to the same mode. The further expansion over objects of the joint generative probability is in Appendix B.1. For inference of latent mode, count, and edge variables, we conduct posterior approximate inference for edge variables $q_{\phi_e}(\mathbf{e}|\mathbf{y})$ conditioned on observations $\mathbf{y}$, and then conduct exact inference of mode and count variables $p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y}, \tilde{\mathbf{e}})$ conditioned on observations $\mathbf{y}$ and the approximate edge variables $\tilde{\mathbf{e}} \sim q_{\phi_e}(\mathbf{e}|\mathbf{y})$. The generative and inference models of AMORE-MIO are illustrated in Fig. 3. Learnable parameters of AMORE-MIO are optimized by maximizing the evidence lower bound with sparse regularization on the learnable coefficient weights

$$\mathcal{L}_{\text{ AMORE-MIO}} = -\log p_\theta(\mathbf{y}) +$$
$$D_{KL}\left[q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y}) \,\|\, p_\theta(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})\right] + ||\mathbf{w}||_1 \quad (7)$$

Neural network implementations, the derivations and the detailed inference model are in Appendix B.2, B.4. and B.3.

6. Experiments

We extensively validate our method on 10 dynamical systems. Specifically, we validate on single-object scenarios using the Mass-spring Hopper dataset, and the Susceptible, Infected and Recovered (SIR) disease dataset from Hybrid-SINDy (Mangan et al., 2019). We validate on multi-object scenarios using the ODE-driven particle dataset and Salsa-dancing dataset from GRASS (Liu et al., 2023). Further, we test the robustness of our methods on non-hybrid systems using datasets of the Coupled linear, Cubic oscillator, Lorenz' 63, Hopf bifurcation, Seklov glycolysis, and Duffing oscillator from Course & Nair (2023). Detailed settings of the datasets are in Appendix C.1.

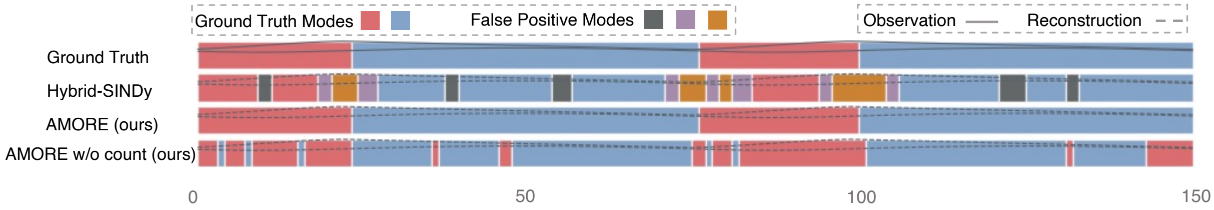


Figure 4. Qualitative time series segmentation results of AMORE compared to Hybrid-SINDy (Brunton et al., 2016) on the Mass-spring Hopper dataset. For Hybrid-SINDy, we aggregate the discovered equations with the same number of coefficients as one mode. We can see that with joint learning of modes and equations, AMORE can categorize the exact number of modes and achieve superior segmentation results with fewer switching errors.

Table 1. Segmentation results on Mass-spring Hopper dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.426	0.383	0.705	0.691
AMORE (ours)	**0.928**	**0.967**	**0.991**	**0.993**

Table 2. Forecasting results of Location/Velocity on the Mass-spring Hopper dataset.

Method	NMAE ↓	NRMSE ↓
LLMTime	0.113 / 0.305	0.417 / 0.454
TimeGPT	0.092 / 0.217	0.322 / 0.340
SVI	0.068 / 0.075	0.148 / 0.262
Hybrid-SINDy	0.240 / 0.314	0.336 / 0.372
AMORE (ours)	**0.008 / 0.039**	**0.026 / 0.059**

Table 3. Segmentation results on the SIR disease dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.296	0.283	0.538	0.519
AMORE (ours)	**0.475**	**0.483**	**0.731**	**0.735**

Table 4. Forecasting results of Susceptible/Infected on the SIR disease dataset.

Method	NMAE ↓	NRMSE ↓
LLMTime	0.352 / 0.396	0.481 / 0.523
TimeGPT	0.301 / 0.347	0.403 / 0.452
SVI	0.257 / 0.273	0.355 / 0.401
Hybrid-SINDy	0.316 / 0.363	0.414 / 0.453
AMORE (ours)	**0.088 / 0.113**	**0.142 / 0.181**

Implementation Details. We train all datasets with a fixed batch size of 40 for 20,000 training steps. We use the Adam optimizer with 10^{-5} weight-decay and clip gradients norm to 10. The learning rate is warmed up linearly from 5×10^{-5} to 2×10^{-4} for the first 2,000 steps, and then decays following a cosine manner with a rate of 0.99. Each experiment is running on one Nvidia GeForce RTX 3090 GPU. d_{min} and d_{max} of the count variables are simply set as 20 and 50, respectively for all datasets. The number of edge types L is set as 2, containing one no-interaction type and one with-interaction type. More details are in Appendix C.2.

Evaluation metrics. For evaluation of discovered equations, following Course & Nair (2023), we use the reconstruction error between the discovered coefficients of equations and ground truth, i.e. $\text{RER} = \frac{1}{T}\sum_{t=1}^{T}(||\mathbf{w}_t - \xi_t||_2 \,/\, ||\xi_t||_2)$ where $\mathbf{w}_t$ and ξ_t are the learned and ground-truth coefficients at time t. For evaluation of segmentation, following Ansari et al. (2021), we use frame-wise segmentation accuracy, i.e. Accuracy and F_1 after matching the labels using the Hungarian algorithm (Kuhn, 1955), Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) to measure similarities. For evaluation of forecasting, we use Normalized Mean Absolute Error (NMAE) and Normalized Root Mean Squared Error (NRMSE). We conducted each experiment with 5 random seeds. We report the average score of each experiment in the main paper and put the statistics (error bars) in the appendix due to limited space.

Baselines. Hybrid-SINDy (Mangan et al., 2019) uses a two-stage paradigm and cannot perform forecasting, thus we compare with it on discovered equations and segmentation. To compare with Hybrid-SINDy on forecasting, we continue the value of the last observable time point as forecasting results of Hybrid-SINDy. For forecasting, we compare with other three recent representative methods, i.e. SVI (Course & Nair, 2023) which is designed for equation discovery in non-hybrid systems and can perform forecasting, LLMTime (Gruver et al., 2023) which utilize pre-trained large language models (LLM) to do forecasting, and TimeGPT (Garza & Mergenthaler-Canseco, 2023) which is the first foundation model for time series. GRASS (Liu et al., 2023) does not discover equations, but models multi-object switching dynamical systems, so it is used for comparison in multi-object systems.

6.1. Single-object Dynamical Systems

6.1.1. Mass-spring Hopper

In the mass-spring hopper system, a mass and spring connect and hop on the ground with two modes, i.e. flight and compression. Details of the dataset are in Appendix C.1.1. Comparison results of time series segmentation on the dataset are in Table 1. We can see that AMORE can achieve significant and consistent performance improvements across all metrics. AMORE categorizes exactly two modes from the system and discovers equations for each mode

$$\begin{cases} \dot{l} = v \;\text{ and }\; \dot{v} = 11.03 - 10.08l \\ \dot{l} = v \;\text{ and }\; \dot{v} = -1 \end{cases}$$

which are nearly identical to the ground truth in Eq. (8). In Hybrid-SINDy, equations are discovered in each cluster fragment, thus producing a massive number of equations. To quantitatively compare discovered equations, we compute RER for Hybrid-SINDy and AMORE which are $7.5e^{-3}$ and $2.4e^{-4}$, respectively.

Qualitative segmentation results of Hybrid-SINDy and AMORE are shown in Fig. 4. Thanks to the amortized joint learning of modes and equations, AMORE can categorize the exact number of modes, achieve superior segmentation results, and discover high-quality equations. In these experiments, the maximal number of possible modes K is set as 3 in our model. After learning, our model chooses 2 modes to be enough to categorize and express the dynamics of the specific hybrid systems. Note that the number of discovered equations in Hybrid-SINDy is the same as the number of time points, which is much larger than the fixed number of modes, e.g. $K = 3$ in our model. To visualize discovered modes of Hybrid-SINDy, we aggregate the discovered equations with the same type of function terms as one mode, thus appearing more than 3 modes in the system. Besides, "AMORE w/o count" represents our model without setting count variables. We can see that count variables can help AMORE learn fewer false-positive mode-switching behaviors. More quantitative ablation studies on count variables are in Appendix C.4.3.

We summarize time series forecasting results on the Mass-spring Hopper dataset in Table 2. We can see that our method significantly outperforms SVI which is designed for non-hybrid systems, and LLMTime as well as TimeGPT which utilizes pre-trained large models for forecasting, thanks to the proposed joint learning framework originally designed for hybrid systems.

6.1.2. SIR Disease Dataset

The Susceptible, Infected and Recovered (SIR) disease model is an epidemiological model used to understand the spread of infectious diseases. Numbers of susceptible, infected, and recovered individuals are involved in model dynamics where some external events describe the modes, e.g. school in session or not. Detailed settings for this dataset are in Appendix C.1.2.

We summarize the segmentation and forecasting results on the dataset in Tables 3 and 4. We can observe similar findings as in the Mass-spring Hopper dataset. AMORE can achieve consistently higher segmentation accuracy and lower forecasting errors across all metrics compared to Hybrid-SINDy, SVI, LLMTime, and TimeGPT. AMORE categorizes exactly two modes from the system and discovers equations for each mode

$$\begin{cases} \dot{S} = 2.74 - 0.0172\, IS - 0.0024\, S,\ \dot{I} = 0.0171\, IS - 0.2\, I \\ \dot{S} = 2.74 - 0.0057\, IS - 0.0021\, S,\ \dot{I} = 0.0051\, IS - 0.2\, I \end{cases}$$

which are nearly exact to the ground truth in Eq. (9). Quantitative comparisons of the discovered equations are calculated by RER where Hybrid-SINDy and AMORE are $3.4e^{-3}$ and $1.8e^{-4}$, respectively. We can see that AMORE can discover high-quality equations, and achieve superior segmentation and forecasting results thanks to the proposed joint learning framework designed for equation discovery in hybrid dynamical systems.

Table 5. Forecasting results on non-hybrid dynamical systems. Results are shown in $\log_{10}(\mathrm{NRMSE})$ where lower is better.

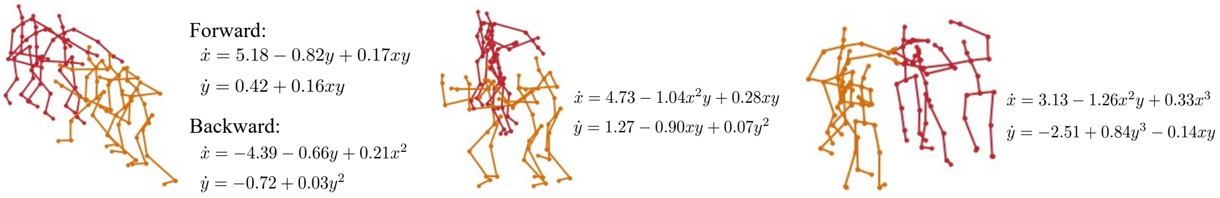
System	LLMTime	SVI	AMORE (ours)
Coupled linear	-0.39	-1.13	**-1.18**
Cubic oscillator	-0.45	-1.02	**-1.06**
Lorenz'63	-0.41	**-1.27**	-1.23
Hopf bifurcation	-0.32	-0.94	**-1.03**
Selkov glycolysis	-0.68	**-1.55**	-1.49
Duffing oscillator	-0.53	-1.12	**-1.17**

6.1.3. Non-hybrid Dynamical Systems

In some cases, we have prior knowledge of the dynamical systems whether they are hybrid or not. To answer the question of whether our method, which is originally designed for hybrid systems, can still perform well if we know the systems are non-hybrid in advance, we conduct experiments on six non-hybrid physical systems (Course & Nair, 2023), including Coupled linear, Cubic oscillator, Lorenz'63, Hopf bifurcation, Selkov glycolysis, and Duffing oscillator. Detailed settings of the datasets are in Appendix C.1.3. As we have the prior, we set the maximal possible number of modes in AMORE as 1 for all physical systems. Following Course & Nair (2023), we summarize the forecasting results in Table 5. We can see that although our model is not specialized for non-hybrid systems, AMORE can still achieve better forecasting results on 4 out of 6 non-hybrid physical systems, which verifies the robustness of our model to non-hybrid dynamical systems.

6.2. Multi-object Hybrid Dynamical Systems

Equation discovery in multi-object hybrid dynamical systems is an unexplored but more general setting. In this section, we verify the effectiveness of the multi-object vari-


Forward:
$\dot{x} = 5.18 - 0.82y + 0.17xy$
$\dot{y} = 0.42 + 0.16xy$
Backward:
$\dot{x} = -4.39 - 0.66y + 0.21x^2$
$\dot{y} = -0.72 + 0.03y^2$
$\dot{x} = 4.73 - 1.04x^2y + 0.28xy$
$\dot{y} = 1.27 - 0.90xy + 0.07y^2$
$\dot{x} = 3.13 - 1.26x^2y + 0.33x^3$
$\dot{y} = -2.51 + 0.84y^3 - 0.14xy$

(a) Forward and Backward (b) Clockwise (c) Counter Clockwise

Figure 5. Discovered equations on the Salsa-dancing dataset. Locations (x, y) of the hip joints are used as observations.

Table 6. Segmentation results on ODE-driven Particle Dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.205	0.192	0.414	0.407
GRASS	0.445	0.437	0.732	0.726
AMORE (ours)	0.418	0.405	0.692	0.684
AMORE-MIO (ours)	**0.453**	**0.442**	**0.741**	**0.735**

Table 7. Forecasting results of in terms of NMAE / NRMSE on ODE-driven Particle dataset.

Method	One-step	Multi-step
LLMTime	0.335 / 0.438	0.370 / 0.473
TimeGPT	0.351 / 0.445	0.392 / 0.490
SVI	0.319 / 0.432	0.346 / 0.465
Hybrid-SINDy	0.340 / 0.431	0.372 / 0.487
GRASS	0.151 / 0.224	0.193 / 0.270
AMORE (ours)	0.184 / 0.265	0.217 / 0.302
AMORE-MIO (ours)	**0.146 / 0.217**	**0.186 / 0.259**

Table 8. Segmentation results on the Salsa-dancing dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.102	0.097	0.325	0.309
GRASS	0.173	0.177	0.579	0.526
AMORE (ours)	0.167	0.173	0.565	0.518
AMORE-MIO (ours)	**0.179**	**0.182**	**0.583**	**0.531**

Table 9. Forecasting results in terms of NMAE / NRMSE on the Salsa-dancing dataset.

Method	One-step	Multi-step
LLMTime	0.402 / 0.452	0.449 / 0.480
TimeGPT	0.341 / 0.417	0.394 / 0.446
SVI	0.384 / 0.441	0.423 / 0.465
Hybrid-SINDy	0.362 / 0.405	0.416 / 0.433
GRASS	0.285 / 0.344	0.313 / 0.359
AMORE (ours)	0.291 / 0.361	0.334 / 0.373
AMORE-MIO (ours)	**0.272 / 0.335**	**0.301 / 0.352**

ant of our method, i.e. AMORE-MIO, on two multi-object datasets (Liu et al., 2023), i.e. the ODE-driven Particle dataset and the Salsa-dancing dataset.

6.2.1. ODE-driven Particle Dataset

In ODE-driven particle systems, trajectories of particles are driven by Ordinary Differential Equations where particles switch their driven equations/modes when they collide with each other. Detailed settings of the ODE-driven Particle dataset are in Appendix C.1.4. We summarize the segmentation results on the dataset in Table 6. We can see that our methods including both AMORE and AMORE-MIO achieve better time series segmentation results compared to Hybrid-SINDy and GRASS. Besides, AMORE-MIO can outperform AMORE consistently across all metrics. AMORE-MIO categorizes 4 modes from the system and the discovered equations for each mode are

$$\begin{cases} \dot{x} = 1.08x - 0.92xy; \ \dot{y} = -0.93y + 1.11xy \\ \dot{x} = -0.17x^3 + 2.00y^3; \ \dot{y} = -2.13x^3 - 0.06y^3 \\ \dot{x} = 0; \ \dot{y} = 2.00 \\ \dot{x} = 0; \ \dot{y} = -2.00 \end{cases}$$

which share the same number of coefficients and similar values as the ground truth in Eq. (10). RER of discovered equations by Hybrid-SINDy, AMORE, and AMORE-MIO are $2.7e^{-2}$, $6.1e^{-3}$, and $4.3e^{-3}$, respectively, which shows that as a multi-object extension of AMORE, AMORE-MIO consistently outperforms AMORE and Hybrid-SINDy for equation discovery and mode categorization in multi-object hybrid systems thanks to the specially-designed interaction modeling of AMORE-MIO. We further show the forecasting results in Table 7. We can see that AMORE-MIO consistently achieves the lowest forecasting errors for both one-step and multi-step predictions. Compared with GRASS, AMORE-MIO can obtain better results thanks to the introduced equation priors on the latent motion dynamics.

6.2.2. Salsa-dancing Dataset

The Salsa-dancing dataset contains four modes, i.e. "moving forward", "moving backward", "clockwise turning", and "counter-clockwise turning". Details of the dataset are in Appendix C.1.5. We summarize the segmentation and forecasting results on the Salsa-dancing dataset in Table 8 and Table 9. We observe similar findings in this real-world video dataset, as with the ODE-driven particle dataset. AMORE-

Table 10. Analyses on robustness to different orders of polynomial as candidate basis functions on Mass-spring Hopper dataset.

Polynomial order	2		3		5	
	NMI↑	RER↓	NMI↑	RER↓	NMI↑	RER↓
Hybrid-SINDy	0.426	$7.5e^{-3}$	0.384	$8.1e^{-3}$	0.316	$9.7e^{-3}$
AMORE (ours)	**0.934**	$\mathbf{2.1e^{-4}}$	**0.936**	$\mathbf{2.3e^{-4}}$	**0.933**	$\mathbf{2.8e^{-4}}$

Table 11. Analyses on robustness to different maximal numbers of predefined modes on Mass-spring Hopper dataset.

Number of modes	3		5		10	
	NMI↑	RER↓	NMI↑	RER↓	NMI↑	RER↓
AMORE (ours)	0.934	$2.1e^{-4}$	0.932	$2.0e^{-4}$	0.937	$2.1e^{-4}$

MIO achieves significantly higher segmentation accuracies compared to Hybrid-SINDy and AMORE. Different from previous datasets, the salsa-dancing system is not generated synthetically by equations while results show that structural learning in the form of equations still benefits forecasting compared to purely autoregressive data-driven methods, i.e. LLMTime, SVI, and GRASS. Qualitative results of the discovered equations on the dancing dataset are in Figure 5.

6.3. Ablation Studies

Sensitivity to order of polynomial functions. To test the sensitivity of our method to different orders of polynomials as candidate basis functions, we conduct experiments on the Mass-spring Hopper dataset by changing the order of polynomial functions to 2, 3, and 5. We present results in Table 10. We observe that AMORE consistently outperforms Hybrid-SINDy, while AMORE is not sensitive to the polynomial orders compared to Hybrid-SINDy.

Sensitivity to number of dynamic modes We test the robustness of our method to different maximum numbers of modes, that is 3, 5, and 10, while the true number is 2 on the Mass-spring Hopper dataset. The results of segmentation and discovered equations are in Table 11. We can see that AMORE is impervious to this misspecification, which indicates that we can set a large number of possible modes while AMORE can still learn those needed.

Sensitivity to more complex dynamical systems We originally followed the setup of Hybrid-SINDy, where all of the dynamics can be approximated by polynomial basis functions. However, our model is not limited to these functions. To show results on more complex dynamical systems, we conduct experiments on a synthetic dataset where two modes are driven by $\dot{x} = x + x^2 + cos(x)$ and $\dot{x} = x + e^x$, respectively. We set the basis functions as polynomials order 3 together with $\{cos(x), sin(x), e^x\}$. The discovered equations by our model are

$$\begin{cases} \dot{x} = 0.97x + 1.02x^2 + 1.08cos(x) \\ \dot{x} = 0.05 + 1.12x + 0.96e^x \end{cases}$$

When we set the basis functions as polynomials order 3 without $\{cos(x), sin(x), e^x\}$. The discovered equations by our model are $\dot{x} = 0.92 + x + 0.76x^2$ and $\dot{x} = 1.26 + 1.31x + 0.83x^2 + 0.34x^3$, respectively. We can see that our model can be extended to equation discovery with more complex basis functions. When the candidate basis functions are limited to polynomial functions, our model can discover approximated ones with more terms, complexity, and errors.

Table 12. Comparisons on model complexity regarding the numbers of learnable parameters.

Method	Number of parameters
Hybrid-SINDy	0
AMORE (ours)	2,240
AMORE-MIO (ours)	2,512
GRASS	4,628
SVI	2,826
LLMTime	175 billion (GPT-3)

Model complexity analysis The numbers of parameters used in baseline methods are summarized in Table 12. As Hybrid-SINDy is not a deep learning method and does not use neural networks, it does not involve learnable parameters and does not need much data for the training of any little parameters the model has. This comes at the expense that Hybrid-SINDy tends to not generalize beyond simple dynamical settings, as shown in our experiments in the main paper. When given a complex dynamical setting with sufficient data, AMORE and AMORE-MIO perform better and have slightly fewer parameters than the other deep learning-based approaches, except LLMTime.

7. Conclusion and Future work

In this paper, we reformulate the problem of equation discovery in hybrid dynamical systems and propose an end-to-end learning framework, i.e. Amortized Equation Discovery (AMORE) to jointly categorize motion dynamics and discover equations by modeling categorical modes and mode-switching behaviors. Besides, we extend our method to multi-object scenarios, i.e. AMORE-MIO, which is unexplored by previous methods and a more natural setting. Extensive experiments on 10 hybrid and non-hybrid systems demonstrate the effectiveness of our method. Future work can include equation discovery with partial known knowledge, equation discovery from videos of hybrid systems, and more complex candidate basis functions.

Impact Statement

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

Acknowledgements

This work is financially supported by NWO TIMING VI.Vidi.193.129. We also thank SURF for the support in using the National Supercomputer Snellius.

References

Ackerson, G. and Fu, K. On state estimation in switching environments. *IEEE transactions on automatic control*, 15(1):10–17, 1970.

Ansari, A. F., Benidis, K., Kurle, R., Turkmen, A. C., Soh, H., Smola, A. J., Wang, B., and Januschowski, T. Deep explicit duration switching models for time series. *Advances in Neural Information Processing Systems*, 34: 29949–29961, 2021.

Bako, L. Identification of switched linear systems via sparse optimization. *Automatica*, 47(4):668–677, 2011.

Bongard, J. and Lipson, H. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.

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.

Cortes, J. Discontinuous dynamical systems. *IEEE Control systems magazine*, 28(3):36–73, 2008.

Course, K. and Nair, P. B. State estimation of a physical system with unknown governing equations. *Nature*, 622 (7982):261–267, 2023.

Dong, Z., Seybold, B., Murphy, K., and Bui, H. Collapsed amortized variational inference for switching nonlinear dynamical systems. In *International Conference on Machine Learning*, pp. 2638–2647. PMLR, 2020.

Eddy, S. R. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365, 1996.

Ferrari-Trecate, G., Muselli, M., Liberati, D., and Morari, M. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

Garza, A. and Mergenthaler-Canseco, M. Timegpt-1. *arXiv preprint arXiv:2310.03589*, 2023.

Ghahramani, Z. and Hinton, G. E. Variational learning for switching state-space models. *Neural computation*, 12 (4):831–864, 2000.

Gruver, N., Finzi, M., Qiu, S., and Wilson, A. G. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*, 2023.

Holmes, P., Full, R. J., Koditschek, D., and Guckenheimer, J. The dynamics of legged locomotion: Models, analyses, and challenges. *SIAM review*, 48(2):207–304, 2006.

Juloski, A. L., Weiland, S., and Heemels, W. M. H. A bayesian approach to identification of hybrid systems. *IEEE Transactions on Automatic Control*, 50(10):1520–1533, 2005.

Kaiser, E., Kutz, J. N., and Brunton, S. L. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A*, 474(2219):20180335, 2018.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Keeling, M. J., Rohani, P., and Grenfell, B. T. Seasonally forced disease dynamics explored as switching between attractors. *Physica D: Nonlinear Phenomena*, 148(3-4): 317–335, 2001.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., and Zemel, R. Neural relational inference for interacting systems. In *International conference on machine learning*, pp. 2688–2697. PMLR, 2018.

Koza, J. R. et al. *Genetic programming II*, volume 17. MIT press Cambridge, 1994.

Kuhn, H. W. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Langley, P. Data-driven discovery of physical laws. *Cognitive Science*, 5(1):31–54, 1981.

Lemos, P., Jeffrey, N., Cranmer, M., Ho, S., and Battaglia, P. Rediscovering orbital mechanics with machine learning. *Machine Learning: Science and Technology*, 4(4):045002, 2023.

Liu, Y., Magliacane, S., Kofinas, M., and Gavves, E. Graph switching dynamical systems. *arXiv preprint arXiv:2306.00370*, 2023.

Loiseau, J.-C. and Brunton, S. L. Constrained sparse galerkin regression. *Journal of Fluid Mechanics*, 838: 42–67, 2018.

Lutter, M., Ritter, C., and Peters, J. Deep lagrangian networks: Using physics as model prior for deep learning. *arXiv preprint arXiv:1907.04490*, 2019.

Mangan, N. M., Askham, T., Brunton, S. L., Kutz, J. N., and Proctor, J. L. Model selection for hybrid dynamical systems via sparse regression. *Proceedings of the Royal Society A*, 475(2223):20180534, 2019.

McMahon, A., Robb, N. C., et al. Reinfection with sars-cov-2: Discrete sir (susceptible, infected, recovered) modeling using empirical infection data. *JMIR public health and surveillance*, 6(4):e21168, 2020.

Novelli, N., Lenci, S., and Belardinelli, P. Boosting the model discovery of hybrid dynamical systems in an informed sparse regression approach. *Journal of Computational and Nonlinear Dynamics*, 17(5):051007, 2022.

Oh, S. M., Ranganathan, A., Rehg, J. M., and Dellaert, F. A variational inference method for switching linear dynamic systems. *TR GIT-GVU-05-16*, 2005.

Ohlsson, H. and Ljung, L. Identification of switched linear regression models using sum-of-norms regularization. *Automatica*, 49(4):1045–1050, 2013.

Ozay, N., Sznaier, M., Lagoa, C., and Camps, O. A sparsification approach to set membership identification of a class of affine hybrid systems. In *2008 47th IEEE Conference on Decision and Control*, pp. 123–130. IEEE, 2008.

Paoletti, S., Juloski, A. L., Ferrari-Trecate, G., and Vidal, R. Identification of hybrid systems a tutorial. *European journal of control*, 13(2-3):242–260, 2007.

Roll, J., Bemporad, A., and Ljung, L. Identification of piecewise affine systems via mixed-integer programming. *Automatica*, 40(1):37–50, 2004.

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.

Sanfelice, R. G. et al. Analysis and design of cyber-physical systems. a hybrid control systems approach. In *Cyber-physical systems: From theory to practice*, pp. 1–29. CRC Press Boca Raton, FL, USA, 2016.

Schaeffer, H. and McCalla, S. G. Sparse model selection via integral terms. *Physical Review E*, 96(2):023302, 2017.

Schmidt, M. and Lipson, H. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

Toda, A. A. Susceptible-infected-recovered (sir) dynamics of covid-19 and economic impact. *arXiv preprint arXiv:2003.11221*, 2020.

Van Der Schaft, A. J. and Schumacher, H. *An introduction to hybrid dynamical systems*, volume 251. springer, 2007.

Vidal, R., Soatto, S., Ma, Y., and Sastry, S. An algebraic geometric approach to the identification of a class of linear hybrid systems. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, volume 1, pp. 167–172. IEEE, 2003.

Appendix

A. More Details of AMORE

A.1. Neural Network Implementation

We use neural networks to model the joint generative probabilities of hybrid systems in our model, i.e. Eq. (4). For the initial states, we model the initial prior distributions as:

$$p(z_1) = \mathrm{Cat}(\mathrm{z}_1; \boldsymbol{\pi}),$$
$$p(\mathbf{y}_1|z_1) = \mathcal{N}(\mathbf{y}_1; \boldsymbol{\mu}_{z_1}; \boldsymbol{\Sigma}_{z_1}),$$

where Cat and $\mathcal{N}$ denote categorical and multivariate Gaussian distributions, respectively. We set the prior distribution of $p(z_1)$ as uniform to encourage diversity.

Count variables and count transition probability. To implement the count variables, we set a categorical distribution over $\{d_{\min}, \cdots, d_{\max}\}$ for each mode, where $d_{\min}$ and $d_{\max}$ are the minimal and maximal numbers of time steps before making a mode switch. The count transition probability $p(c_t|c_{t-1}, z_{t-1})$ is modeled as a learnable matrix $\mathbf{P} \in \mathbb{R}^{K \times (d_{\max} - d_{\min} + 1)}$, which is fixed across all time steps. Each term $\rho_k(c)$ in $\mathbf{P}$ represents the probability of the k-th mode switching to another mode when its current count is c. The probability of a count increment at count c for mode k can be calculated as

$$\mu_k(c) = 1 - \frac{\rho_k(c)}{\sum_{d=c}^{d_{\max}} \rho_k(d)}.$$

The count transition probability is thus defined as

$$p(c_t|c_{t-1}, z_{t-1} = k) = \begin{cases} \mu_k(c_{t-1}) & \text{if } c_t = c_{t-1} + 1 \\ 1 - \mu_k(c_{t-1}) & \text{if } c_t = 1 \end{cases}.$$

Mode variables and mode transition probability. Since the mode variables z_t take one out of K possible values, we model them as categorical variables, parameterized by mode transition matrix $\mathbf{T}_t$ at timestep t. The mode transition probability is modeled as

$$p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1}) = \begin{cases} \delta_{z_t = z_{t-1}} & \text{if } c_t > 1 \\ \mathrm{Cat}(z_t; \mathbf{T}_t) & \text{if } c_t = 1 \end{cases},$$

where we resample the modes or preserve them depending on whether count variables are reset to 1 or not. We model the parameters $\mathbf{T}_t$ of the categorical distributions with a neural network, i.e. a simple MLP, $\mathbf{T}_t = f_z(\mathbf{y}_{t-1})$ that takes as input the observations. The network returns a $K \times K$ transition matrix per time step t, where rows correspond to past modes z_{t-1} and columns current modes z_t. Each term $\tau_t^{j,k}$ in $\mathbf{T}_t$ represents the probability of mode j switching to mode k at timestep t. To satisfy the positivity $\tau_t^{j,k} > 0,\ \forall j, k = 1, \cdots, K$ and ℓ_1 constraints $\sum_k \tau_t^{j,k} = 1,\ \forall j = 1, ..., K$, we apply a tempered softmax after f_z, i.e. $\mathcal{S}_{\tau_z} \circ f_z(\cdot)$.

A.2. Inference Model of AMORE

We perform conditionally exact inference for the two discrete latent variables, i.e. modes $\mathbf{z}_{1:T}$ and counts $\mathbf{c}_{1:T}$, similar to the forward-backward procedure for HMM (Eddy, 1996). Conditioned on observations $\mathbf{y}_{1:T}$, the posterior joint distribution $p_\theta(\mathbf{z}_{1:T}, \mathbf{c}_{1:T}|\mathbf{y}_{1:T})$ is calculated by modifying the forward-backward recursions to handle the joint hierarchical latent variables. Specifically, the forward α_t and backward β_t parts are defined as

$$\alpha_t(z_t, c_t) = p(z_t, c_t, \mathbf{y}_{1:t}),$$
$$\beta_t(z_t, c_t) = p(\mathbf{y}_{t+1:T}|\mathbf{y}_t, z_t, c_t).$$

Specifically, the posterior joint probability of mode and count variables $\mathbf{z}$, $\mathbf{c}$ conditioned on observations $\mathbf{y}$ is calculated as

$$\begin{aligned} p(z_t, c_t|\mathbf{y}_{1:T}) &\propto p(z_t, c_t, \mathbf{y}_{1:T}) \\ &= \underbrace{p(z_t, c_t, \mathbf{y}_{1:t})}_{Forward} \underbrace{p(\mathbf{y}_{t+1:T}|\mathbf{y}_t, z_t, c_t)}_{Backward} \\ &= \alpha_t(z_t, c_t) \cdot \beta_t(z_t, c_t). \end{aligned}$$

The derivatives of the forward section $\alpha_t(z_t, c_t)$ are

$$\begin{aligned} \alpha_1(z_1, c_1) &= p(z_1, c_1, \mathbf{y}_1) \\ &= \delta_{c_1=1} p(z_1) p(\mathbf{y}_1|z_1), \\ \underline{\alpha_t(z_t, c_t)} &= p(z_t, c_t, \mathbf{y}_{1:t}) \\ &= \sum_{z_{t-1}, c_{t-1}} p(z_t, c_t, \mathbf{y}_{1:t}, z_{t-1}, c_{t-1}) \\ &= \sum_{z_{t-1}, c_{t-1}} p(z_{t-1}, c_{t-1}, \mathbf{y}_{1:t-1}) p(c_t|c_{t-1}, z_{t-1}) p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1}) p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t) \\ &= p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t) \sum_{z_{t-1}, c_{t-1}} \underline{\alpha_{t-1}(z_{t-1}, c_{t-1})} p(c_t|c_{t-1}, z_{t-1}) p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1}) \\ &= p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t) \Bigg[\delta_{c_t=1} \sum_{z_{t-1}} p(z_t|z_{t-1}, c_t, \mathbf{y}_{t-1}) \sum_{c_{t-1}} (1 - \mu_{z_{t-1}(c_{t-1})}) \alpha_{t-1}(z_{t-1}, c_{t-1}) \\ &+ \delta_{\substack{z_{t-1}=z_t \\ c_t>1 \\ c_{t-1}=c_t-1}} \mu_{z_{t-1}}(c_{t-1}) \alpha_{t-1}(z_{t-1}, c_{t-1}) \Bigg], \end{aligned}$$

where $\alpha_t(z_t, c_t)$ can be expressed by $\alpha_{t-1}(z_{t-1}, c_{t-1})$ recursively with states transitions.

The derivatives of the backward section $\beta_t(z_t, c_t)$ are

$$\begin{aligned} \beta_T(z_T, c_T) &= 1 \\ \underline{\beta_t(z_t, c_t)} &= p(\mathbf{y}_{t+1:T}|\mathbf{y}_t, z_t, c_t) \\ &= \sum_{z_{t+1}, c_{t+1}} p(\mathbf{y}_{t+1:T}, z_{t+1}, c_{t+1}|\mathbf{y}_t, z_t, c_t) \\ &= \sum_{z_{t+1}, c_{t+1}} p(c_{t+1}|c_t, z_t) p(z_{t+1}|z_t, c_t, \mathbf{y}_t) p(\mathbf{y}_{t+1}|\mathbf{y}_t, z_{t+1}) p(\mathbf{y}_{t+2:T}|\mathbf{y}_{t+1}, z_{t+1}, c_{t+1}) \\ &= \sum_{z_{t+1}, c_{t+1}} p(c_{t+1}|c_t, z_t) p(z_{t+1}|z_t, c_{t+1}, \mathbf{y}_t) p(\mathbf{y}_{t+1}|\mathbf{y}_t, z_{t+1}) \underline{\beta_{t+1}(z_{t+1}, c_{t+1})} \\ &= \delta_{\substack{c_{t+1}=1 \\ c_t \geqslant d_{\min}}} (1 - \mu_{z_t}(c_t)) \sum_{z_{t+1}} p(z_{t+1}|z_t, c_{t+1}, \mathbf{y}_t) p(\mathbf{y}_{t+1}|\mathbf{y}_t, z_{t+1}) \beta_{t+1}(z_{t+1}, c_{t+1}) \\ &+ \delta_{\substack{c_{t+1}=c_t+1 \\ z_{t+1}=z_t}} \mu_{z_t}(c_t) p(\mathbf{y}_{t+1}|\mathbf{y}_t, z_{t+1}) \beta_{t+1}(z_{t+1}, c_{t+1}), \end{aligned}$$

where $\beta_t(z_t, c_t)$ can be computed via $\beta_{t+1}(z_{t+1}, c_{t+1})$ recursively with states transitions.

A.3. Derivation of Optimization Objective

The optimization objective of our model is to maximize the observation likelihood $\log p(\mathbf{y})$ with sparse regularization on coefficients of candidate basis functions, where the observation likelihood $\log p(\mathbf{y})$ can be calculated as

$$\begin{aligned} \log \mathrm{p}(\mathbf{y}) &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})} [\log p(\mathbf{y})] \\ &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})} [\log p(\mathbf{y}, \mathbf{z}, \mathbf{c})] - \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})} [\log p(\mathbf{z}, \mathbf{c}|\mathbf{y})] \\ &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})} [\log p(\mathbf{y}, \mathbf{z}, \mathbf{c})], \end{aligned}$$

where $\mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})}\left[\log p(\mathbf{z},\mathbf{c}|\mathbf{y})\right]$ is calculated as

$$\mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})}\left[\log p(\mathbf{z},\mathbf{c}|\mathbf{y})\right] = \int p(\mathbf{z},\mathbf{c}|\mathbf{y}) \frac{\log p(\mathbf{z},\mathbf{c}|\mathbf{y})}{p(\mathbf{z},\mathbf{c}|\mathbf{y})} d(\mathbf{z},\mathbf{c}) = \int \log p(\mathbf{z},\mathbf{c}|\mathbf{y}) d(\mathbf{z},\mathbf{c}) = 1 = 0.$$

Following Markovian property, we expand $\log p(\mathbf{y},\mathbf{z},\mathbf{c})$ over time and calculate it as

$$\begin{aligned}\log p(\mathbf{y},\mathbf{z},\mathbf{c}) &= \log p(\mathbf{y}_{1:T},\mathbf{z}_{1:T},\mathbf{c}_{1:T}) \\ &= \log[p(\mathbf{y}_1|z_1)p(z_1)] + \sum_{t=2}^{T} \log[p(\mathbf{y}_t|\mathbf{y}_{t-1},z_t)p(z_t|z_{t-1},c_t,\mathbf{y}_{t-1})p(c_t|c_{t-1},z_{t-1})]\,.\end{aligned}$$

Finally, combined with expectations, $\log p(\mathbf{y})$ can be calculated as

$$\begin{aligned}\log p(\mathbf{y}) &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y})}\left[\log p(\mathbf{y},\mathbf{z},\mathbf{c})\right], \\ &= \mathbb{E}_{p(\mathbf{z}_{1:T},\mathbf{c}_{1:T}|\mathbf{y}_{1:T})}\left[\log p(\mathbf{y}_{1:T},\mathbf{z}_{1:T},\mathbf{c}_{1:T})\right] \\ &= \sum_k p(z_1 = k|\mathbf{y}_{1:T}) \log\left[p(\mathbf{y}_1|z_1)p(z_1 = k)\right] \\ &\quad + \sum_{t=2}^{T} \sum_{k,j,u,v} \xi(k,j,u,v)\ \log[p(\mathbf{y}_t|\mathbf{y}_{t-1},z_t = k)p(z_t\!=\!k|z_{t-1}\!=\!j,c_t\!=\!v,\mathbf{y}_{t-1})p(c_t\!=\!v|c_{t-1}\!=\!u,z_{t-1}\!=\!j)] \\ &= \sum_k \gamma(k)\ \log[B_1(k)\cdot\pi(k)] \\ &\quad + \sum_{t=2}^{T} \sum_{k,j,u,v} \xi(k,j,u,v)\ \log[B_t(k)\cdot A_t(k,j,v)\cdot C_t(j,u,v)]\end{aligned}$$

where $\pi(k)$, $\gamma(k)$, $\xi(k,j,u,v)$, $B_t(k)$, $A_t(k,j,v)$, and $C_t(j,u,v)$ are defined as

$$\begin{aligned}\pi(k) &= p(z_1 = k), \\ \gamma(k) &= p(z_1 = k|\mathbf{y}_{1:T}), \\ \xi(k,j,u,v) &= p(z_t\!=\!k, z_{t-1}\!=\!j, c_t\!=\!v, c_{t-1}\!=\!u|\mathbf{y}_{1:T}), \\ B_t(k) &= p(\mathbf{y}_t|\mathbf{y}_{t-1}, z_t = k), \\ A_t(k,j,v) &= p(z_t\!=\!k|z_{t-1}\!=\!j, c_t\!=\!v, \mathbf{y}_{t-1}), \\ C_t(j,u,v) &= p(c_t\!=\!v|c_{t-1}\!=\!u, z_{t-1}\!=\!j).\end{aligned}$$

$\pi(\mathbf{k})$ is the initial discrete mode probability. $B_t(k)$ is the continuous state transition probability conditioned on different types of discrete modes k. $A_t(k,j,v)$ is the discrete mode transition probability. $C_t(j,u,v)$ is the mode duration count transition probability. Besides, $\gamma(k) = p(z_1 = k|\mathbf{y}_{1:T})$ and $\xi(k,j,u,v) = p(z_t\!=\!k, z_{t-1}\!=\!j, c_t\!=\!v, c_{t-1}\!=\!u|\mathbf{y}_{1:T})$ can be calculated similarly to the forward and backward algorithm in HMMs (Eddy, 1996) which is detailed in Appendix A.2.

B. More Details of AMORE-MIO

B.1. Expansion of Generative Model over Objects

The joint generative probability of AMORE-MIO for multi-object hybrid systems is expanded over objects as

$$\begin{aligned}p(\mathbf{y},\mathbf{z},\mathbf{c},\mathbf{e}) = \underbrace{\prod_{n=1}^{N} p(\mathbf{y}_1^n|z_1^n)\cdot\prod_{n=1}^{N} p(z_1^n)\cdot\prod_{n=1}^{N}\prod_{m=1}^{N} p(e_1^{m\to n})}_{\text{Initial states}}\cdot\prod_{t=2}^{T}\Bigg[\prod_{n=1}^{N} p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n)\cdot\prod_{n=1}^{N} p(c_t^n|c_{t-1}^n, z_{t-1}^n)\cdot \\ \prod_{n=1}^{N}\sum_{m=1}^{N} p(z_t^n|z_{t-1}^m, c_t^n, e_t^{m\to n}, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n)\cdot\prod_{n=1}^{N}\sum_{m=1}^{N} p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n)\Bigg],\end{aligned}$$

where in the initial states, we model for each object n an initial mode and observation distributions, i.e. $p(z_1^n)$ and $p(\mathbf{y}_1^n|z_1^n)$. For each pair of interactions, $p(e_1^{m\to n})$ models the initial edge distribution. For later time steps $t \geqslant 2$, $p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n)$ models the edge variable transition probability conditioned on node states $\{\mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n\}$ in graph $\mathcal{G}_t$. $p(z_t^n|z_{t-1}^m, c_t^n, e_t^{m\to n}, \mathbf{y}_{t-1}^{m,n})$ models how the modes of objects are affected by the modes of all other objects, conditioned on count variables c_t^n, edge variables $e_t^{m\to n}$, and observations $\{\mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n\}$. $p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n)$ and $p(c_t^n|c_{t-1}^n, z_{t-1}^n)$ model for each object an observation transition probability and count variable transition probability.

B.2. Neural Network Implementation

Implementations of $p(\mathbf{y}_1^n|z_1^n)$, $p(z_1^n)$, $p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n)$, and $p(c_t^n|c_{t-1}^n, z_{t-1}^n)$ in multi-object scenarios are the same as those in single-object scenarios. Next, we elaborate on how we implement the other terms, i.e. $p(e_1^{m\to n})$, $p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n)$, and $p(z_t^n|z_{t-1}^m, c_t^n, e_t^{m\to n}, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n)$.

Edge variables and edge transition probability. We implement the edge variable $\mathbf{e}$ as a categorical distribution over $\{1, \cdots, L\}$ for L possible interaction types including a *no-interaction* type. We set the prior distribution to be higher for *no-interaction* edges in $p(e_1^{m\to n})$ to encourage sparse graphs. The edge transition probability is modeled as

$$p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n) = \mathrm{Cat}(e_t^{m\to n}; \mathcal{S}_{\tau_e}(f_e(e_{t-1}^{m\to n}, \mathbf{v}_{t-1}^m, \mathbf{v}_{t-1}^n))),$$

where The neural network f_e takes $e_{t-1}^{m\to n}$, $\mathbf{v}_{t-1}^m$, and $\mathbf{v}_{t-1}^n$ as input and outputs the probabilities of all possible edge types at time step t, which are further post-processed by a tempered softmax function $\mathcal{S}_{\tau_e}$ with temperature τ_e to ensure normalization. In practice, the edge transition network f_e is a single hidden layer MLP.

Extension of mode transition probability. After getting $e_t^{m\to n}$ by the edge transition probability, we show how $e_t^{m\to n}$ affects the mod-switching behaviors. We model the mode transition probability in multi-object hybrid systems as

$$p(z_t^n|z_{t-1}^m, c_t^n, e_t^{m\to n}, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n) = \begin{cases} \delta_{z_t^n = z_{t-1}^n} & \text{if } c_t^n > 1 \\ \mathrm{Cat}(z_t^n; \mathcal{S}_{\tau_z}(\sum_l e_{t,l}^{m\to n} f_l(\mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n)) & \text{if } c_t^n = 1 \end{cases},$$

where δ and $\mathcal{S}_{\tau_z}$ are a Kronecker function and a tempered softmax function. $e_{t,l}^{m\to n}$ denotes the probability of each edge type l. We set a neural network f_l for each edge type l (totally L) to model different interaction effects, which are normalized by $e_{t,l}^{m\to n}$ to aggregate effects from all the interaction types.

B.3. Inference Model of AMORE-MIO

Approximate inference of edge variables. We use a graph neural network $f_{\phi_e}(\mathbf{y})$ to conduct approximate inference of edge variables $\mathbf{e}$, i.e. $q_{\phi_e}(\mathbf{e}|\mathbf{y})$. The node embeddings in the latent graph $\mathcal{G}_t$ are the observations $\mathbf{y}$, and the edge embeddings are calculated by two rounds of message-passing

$$\begin{aligned} \mathbf{h}_n^1 &= f_{\phi_e}^{\mathrm{emb}}(\mathbf{y}_t^n), \\ v \to e\colon \mathbf{h}_{m\to n}^1 &= f_{\phi_z}^{e,1}([\mathbf{h}_m^1, \mathbf{h}_n^1]), \\ e \to v\colon \mathbf{h}_n^2 &= f_{\phi_e}^{v,1}(\sum_{m=1}^{N} \mathbf{h}_{m\to n}^1), \\ v \to e\colon \mathbf{h}_{m\to n}^2 &= f_{\phi_e}^{e,2}([\mathbf{h}_m^2, \mathbf{h}_n^2]), \end{aligned}$$

where $\mathbf{h}_{m\to n}^2$ is further processed by a tempered Gumbel softmax $\mathrm{softmax}((\mathbf{h}_{m\to n}^2 + \mathbf{g})/\tau)$ to achieve $q_{\phi_e}(\mathbf{e}|\mathbf{y})$, to be more specific $q_{\phi_e}(e_t^{m\to n}|\mathbf{y}_t^m, \mathbf{y}_t^n)$. Here, we use continuous relaxation and reparameterization of discrete distributions for gradient backpropagation (Kipf et al., 2018). $\mathbf{g}$ is a vector sampled from a $\mathrm{Gumbel}(0, 1)$ distribution and the softmax temperature τ controls relaxation smoothness.

Exact inference of mode and count variables. Given the approximate edge variables $\tilde{\mathbf{e}} \sim q_{\phi_e}(\mathbf{e}|\mathbf{y})$, we do exact inference of the mode and count variables $p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y}, \tilde{\mathbf{e}})$. Similar to the single-object scenarios, the conditional joint distribution is calculated by modifying the forward-backward algorithm. Specifically, the forward α_t and backward β_t are calculated as

$$\begin{aligned}
\alpha_t(\mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}) &= p(\mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{y}_{1:t}^{1:N}, \mathbf{e}_{1:t}^{1:N^2}), \\
\beta_t(\mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}) &= p(\mathbf{y}_{t+1:T}^{1:N} | \mathbf{y}_t^{1:N}, \mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{e}_t^{1:N^2}).
\end{aligned}$$

Specifically, the joint probability of mode and count variables $\mathbf{z}$, $\mathbf{c}$ conditioned on observations $\mathbf{y}$ and approximate edge variables $\mathbf{e}$ is calculated as

$$\begin{aligned}
p(\mathbf{z}_t, \mathbf{c}_t | \mathbf{y}_{1:T}, \mathbf{e}_{1:T}) \propto p(\mathbf{z}_t, \mathbf{c}_t, \mathbf{y}_{1:T}, \mathbf{e}_{1:T}) \\
&= \underbrace{p(\mathbf{z}_t, \mathbf{c}_t, \mathbf{y}_{1:t}, \mathbf{e}_{1:t})}_{Forward} \underbrace{p(\mathbf{y}_{t+1:T}, \mathbf{e}_{t+1:T} | \mathbf{y}_t, \mathbf{z}_t, \mathbf{c}_t)}_{Backward} \\
&= \alpha_t(\mathbf{z}_t, \mathbf{c}_t) \cdot \beta_t(\mathbf{z}_t, \mathbf{c}_t).
\end{aligned}$$

The derivatives of the forward section $\alpha_t(\mathbf{z}_t, \mathbf{c}_t)$ is calculated as:

$$\begin{aligned}
\alpha_1(\mathbf{z}_1, \mathbf{c}_1) &= p(\mathbf{z}_1, \mathbf{c}_1, \mathbf{y}_1, \mathbf{e}_1) \\
&= p(\mathbf{z}_1^{1:N}, \mathbf{c}_1^{1:N}, \mathbf{y}_1^{1:N}, \mathbf{e}_1^{1:N^2}) \\
&= \delta_{\mathbf{c}_1^{1:N}=1} p(\mathbf{z}_1^{1:N}) p(\mathbf{e}_1^{1:N^2}) p(\mathbf{y}_1^{1:N} | \mathbf{z}_1^{1:N}) \\
&= \delta_{\mathbf{c}_1^{1:N}=1} p(\mathbf{z}_1^{1:N}) p(\mathbf{e}_1^{1:N^2}) \prod_{n=1}^{N} p(\mathbf{y}_1^n | \mathbf{z}_1^n) \\
\underline{\alpha_t(\mathbf{z}_t, \mathbf{c}_t)} &= p(\mathbf{z}_t, \mathbf{c}_t, \mathbf{y}_{1:t}, \mathbf{e}_{1:t}) \\
&= p(\mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{y}_{1:t}^{1:N}, \mathbf{e}_{1:t}^{1:N^2}) \\
&= \sum_{\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}} p(\mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{y}_{1:t}^{1:N}, \mathbf{e}_{1:t}^{1:N^2}, \mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}) \\
&= \sum_{\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}} \Bigg[p(\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}, \mathbf{y}_{1:t-1}^{1:N}, \mathbf{e}_{1:t-1}^{1:N^2}) p(\mathbf{y}_t^{1:N} | \mathbf{y}_{t-1}^{1:N}, \mathbf{z}_t^{1:N}) p(\mathbf{z}_t^{1:N} | \mathbf{z}_{t-1}^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{e}_t^{1:N^2}) \\
&\qquad \cdot p(\mathbf{c}_t^{1:N} | \mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N}) p(\mathbf{e}_t^{1:N^2} | \mathbf{e}_{t-1}^{1:N^2}, \mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N}) \Bigg] \\
&= \sum_{\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_{t-1}^{1:N}} \Bigg[\underline{\alpha_{t-1}(\mathbf{z}_{t-1}, \mathbf{c}_{t-1})} \cdot \prod_{n=1}^{N} p(\mathbf{y}_t^n | \mathbf{y}_{t-1}^n, z_t^n) \cdot \prod_{n=1}^{N} \prod_{m=1}^{N} p(z_t^n | z_{t-1}^m, c_t^n, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n, e_t^{m \to n}) \cdot \\
&\qquad \cdot \prod_{n=1}^{N} p(c_t^n | c_{t-1}^n, z_{t-1}^n) \prod_{n=1}^{N} \prod_{m=1}^{N} p(e_t^{m \to n} | e_{t-1}^{m \to n}, z_{t-1}^m, z_{t-1}^n, c_{t-1}^m, c_{t-1}^n, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n) \Bigg],
\end{aligned}$$

where $\alpha_t(\mathbf{z}_t, \mathbf{c}_t)$ are calculated by $\alpha_{t-1}(\mathbf{z}_{t-1}, \mathbf{c}_{t-1})$ recursively with states transitions.

The derivatives of the backward section $\beta_t(\mathbf{z}_t, \mathbf{c}_t)$ is calculated as

$$
\begin{aligned}
\beta_T(\mathbf{z}_T, \mathbf{c}_T) &= 1 \\
\underline{\beta_t(\mathbf{z}_t, \mathbf{c}_t)} &= p(\mathbf{y}_{t+1:T}, \mathbf{e}_{t+1:T} | \mathbf{y}_t, \mathbf{z}_t, \mathbf{c}_t) \\
&= p(\mathbf{y}_{t+1:T}^{1:N}, \mathbf{e}_{t+1:T}^{1:N^2} | \mathbf{y}_t^{1:N}, \mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}) \\
&= \sum_{\mathbf{z}_{t+1}^{1:N}, \mathbf{c}_{t+1}^{1:N}} p(\mathbf{y}_{t+1:T}^{1:N}, \mathbf{e}_{t+1:T}^{1:N^2}, \mathbf{z}_{t+1}^{1:N}, \mathbf{c}_{t+1}^{1:N} | \mathbf{y}_t^{1:N}, \mathbf{z}_t^{1:N}, \mathbf{c}_t^{1:N}) \\
&= \sum_{\mathbf{z}_{t+1}^{1:N}, \mathbf{c}_{t+1}^{1:N}} \Bigg[p(\mathbf{y}_{t+1}^{1:N} | \mathbf{y}_t^{1:N}, \mathbf{z}_{t+1}^{1:N}) p(\mathbf{z}_{t+1}^{1:N} | \mathbf{z}_t^{1:N}, \mathbf{c}_{t+1}^{1:N}, \mathbf{y}_t^{1:N}, \mathbf{e}_{t+1}^{1:N^2}) \\
&\cdot p(\mathbf{c}_{t+1}^{1:N} | \mathbf{c}_t^{1:N}, \mathbf{z}_t^{1:N}) p(\mathbf{e}_{t+1}^{1:N^2} | \mathbf{e}_t^{1:N^2}, \mathbf{z}_t^{1:N}, \mathbf{y}_t^{1:N}) p(\mathbf{y}_{t+2:T}^{1:N}, \mathbf{e}_{t+2:T}^{1:N^2} | \mathbf{y}_{t+1}^{1:N}, \mathbf{z}_{t+1}^{1:N}, \mathbf{c}_{t+1}^{1:N}) \Bigg] \\
&= \sum_{\mathbf{z}_{t+1}^{1:N}, \mathbf{c}_{t+1}^{1:N}} \Bigg[\prod_{n=1}^{N} p(\mathbf{y}_{t+1}^n | \mathbf{y}_t^n, z_{t+1}^n) \cdot \prod_{n=1}^{N} \prod_{m=1}^{N} p(z_{t+1}^n | z_t^m, c_{t+1}^n, \mathbf{y}_t^{m,n}, e_{t+1}^{m \to n}) \\
&\cdot \prod_{n=1}^{N} p(c_{t+1}^n | c_t^n, z_t^n) \cdot \prod_{n=1}^{N} \prod_{m=1}^{N} p(e_{t+1}^{m \to n} | e_t^{m \to n}, z_t^m, z_t^n, c_t^m, c_t^n, \mathbf{y}_t^m, \mathbf{y}_t^n) \; \underline{\beta_{t+1}(\mathbf{z}_{t+1}, \mathbf{c}_{t+1})} \Bigg],
\end{aligned}
$$

where $\beta_t(\mathbf{z}_t, \mathbf{c}_t)$ is computed via $\beta_{t+1}(\mathbf{z}_{t+1}, \mathbf{c}_{t+1})$ recursively by state transitions.

B.4. Derivation of Optimization Objective

Learnable parameters of our model are optimized by maximizing the evidence lower bound (ELBO) with sparse regularization on coefficients of candidate basis functions where the derivatives of ELBO are as follows. For brevity, $\mathbf{y}$, $\mathbf{z}$, $\mathbf{c}$, and $\mathbf{e}$ represents $\mathbf{y}_{1:T}^{1:N}$, $\mathbf{z}_{1:T}^{1:N}$, $\mathbf{c}_{1:T}^{1:N}$, and $\mathbf{e}_{1:T}^{1:N^2}$ respectively. N is the number of objects. T is the number of time steps.

$$
\begin{aligned}
ELBO &= \log p_\theta(\mathbf{y}) - D_{KL}\left[q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y}) \,\|\, p_\theta(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})\right] \\
&= \int q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y}) \log p_\theta(\mathbf{y}) \, d(\mathbf{z}, \mathbf{c}, \mathbf{e}) - \int q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y}) \log \frac{q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})}{p_\theta(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})} \, d(\mathbf{z}, \mathbf{c}, \mathbf{e}) \\
&= \int q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y}) \left[\log p_\theta(\mathbf{z}, \mathbf{c}, \mathbf{e}, \mathbf{y}) - \log q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})\right] \, d(\mathbf{z}, \mathbf{c}, \mathbf{e}) \\
&= \mathbb{E}_{q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})} \left[\log p_\theta(\mathbf{z}, \mathbf{c}, \mathbf{e}, \mathbf{y}) - \log q_\phi(\mathbf{z}, \mathbf{c}, \mathbf{e}|\mathbf{y})\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y}) p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e})} \left[\log p_\theta(\mathbf{y}, \mathbf{e}) p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e}) - \log q_\phi(\mathbf{e}|\mathbf{y}) p_\theta(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e})\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \left[\log p_\theta(\mathbf{y}, \mathbf{e}) - \log q_\phi(\mathbf{e}|\mathbf{y})\right] \\
&= \mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \left[\log p_\theta(\mathbf{y}, \mathbf{e})\right] + H(q_\phi(\mathbf{e}|\mathbf{y})),
\end{aligned}
$$

where $\log p_\theta(\mathbf{y}, \mathbf{e})$ is a joint likelihood, and $H(q_\phi(\mathbf{e}|\mathbf{y}))$ is a conditional entropy for the approximate posterior of edge variable $\mathbf{e}$.

B.4.1. Training of ELBO

We use the mini-batch stochastic gradient descent algorithm for training of ELBO. The gradients with respect to θ or ϕ in ELBO are calculated as

$$
\begin{aligned}
\nabla_\theta ELBO &= \nabla_\theta \left[\mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \log p_\theta(\mathbf{y}, \mathbf{e})\right] = \mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \nabla_\theta \log p_\theta(\mathbf{y}, \mathbf{e}), \\
\nabla_\phi ELBO &= \nabla_\phi \left[\mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \log p_\theta(\mathbf{y}, \mathbf{e}) + H(q_\phi(\mathbf{e}|\mathbf{y}))\right] \\
&= \nabla_\phi \left[\mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \log p_\theta(\mathbf{y}, \mathbf{e})\right] + \nabla_\phi H(q_\phi(\mathbf{e}|\mathbf{y})) \\
&= \mathbb{E}_{\epsilon \sim \mathcal{N}} \left[\nabla_\phi \log p_\theta(\mathbf{e}, \mathbf{y}_\phi(\mathbf{e}, \epsilon))\right] + \nabla_\phi H(q_\phi(\mathbf{e}|\mathbf{y})),
\end{aligned}
$$

where we use the reparameterization trick (Kingma & Welling, 2013) to calculate the gradients of $\nabla_\phi \left[\mathbb{E}_{q_\phi(\mathbf{e}|\mathbf{y})} \log p_\theta(\mathbf{y}, \mathbf{e})\right]$. $\nabla_\phi H(q_\phi(\mathbf{e}|\mathbf{y}))$ is an entropy loss. Among the derivative terms, the challenging part is the gradients of joint probability

$\nabla_\theta \log p_\theta(\mathbf{y}, \mathbf{e})$, which is calculated as

$$\begin{aligned}
\nabla \log p(\mathbf{y}, \mathbf{e}) &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{y}, \mathbf{e})\right] \\
&= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c})\right] - \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e})\right] \\
&= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c})\right] - \int p(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e}) \frac{\nabla \log p(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e})}{p(\mathbf{z}, \mathbf{c}|\mathbf{y}, \mathbf{e})} d(\mathbf{z}, \mathbf{c}) \\
&= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c})\right],
\end{aligned}$$

Following the Markovian property, we unfold the joint likelihood $p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c})$ over time as:

$$\begin{aligned}
&\nabla \log p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c}) \\
&= \nabla \log p(\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2}, \mathbf{z}_{1:T}^{1:N}, \mathbf{c}_{1:T}^{1:N}) \\
&= \nabla \log\big[p(\mathbf{y}_1^{1:N}|\mathbf{z}_1^{1:N})p(\mathbf{z}_1^{1:N})\big] + \sum_{t=2}^{T} \nabla \log\Big[p(\mathbf{y}_t^{1:N}|\mathbf{y}_{t-1}^{1:N}, \mathbf{z}_t^{1:N})p(\mathbf{z}_t^{1:N}|\mathbf{z}_{t-1}^{1:N}, \mathbf{c}_t^{1:N}, \mathbf{y}_{t-1}^{1:N}, \mathbf{e}_t^{1:N^2})\cdot \\
&\quad p(\mathbf{c}_t^{1:N}|\mathbf{c}_{t-1}^{1:N}, \mathbf{z}_{t-1}^{1:N})p(\mathbf{e}_t^{1:N^2}|\mathbf{e}_{t-1}^{1:N^2}, \mathbf{z}_{t-1}^{1:N}, \mathbf{y}_{t-1}^{1:N})\Big] \\
&= \nabla \log\left[\prod_{n=1}^{N} p(\mathbf{y}_1^n|z_1^n) \cdot \prod_{n=1}^{N} p(z_1^n)\right] + \sum_{t=2}^{T} \nabla \log\Bigg[\prod_{n=1}^{N} p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n) \cdot \prod_{n=1}^{N}\prod_{m=1}^{N} p(z_t^n|z_{t-1}^m, c_t^n, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n, e_t^{m\to n})\cdot \\
&\quad \prod_{n=1}^{N} p(c_t^n|c_{t-1}^n, z_{t-1}^n) \cdot \prod_{n=1}^{N}\prod_{m=1}^{N} (e_t^{m\to n}|e_{t-1}^{m\to n}, z_{t-1}^m, z_{t-1}^n, c_{t-1}^m, c_{t-1}^n, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n)\Bigg],
\end{aligned}$$

where edge variables evolve based on all previous states of both objects. We model the influences of interactions between each pair of objects by $p(z_t^n|z_{t-1}^m, c_t^n, \mathbf{y}_{t-1}^m, \mathbf{y}_{t-1}^n, e_t^{m\to n})$ without instantaneous dependences. Combining with expectation, $\nabla \log p(\mathbf{y}, \mathbf{e})$ is finally calculated as

$$\begin{aligned}
\nabla \log p(\mathbf{y}, \mathbf{e}) &= \mathbb{E}_{p(\mathbf{z},\mathbf{c}|\mathbf{y},\mathbf{e})}\left[\nabla \log p(\mathbf{y}, \mathbf{e}, \mathbf{z}, \mathbf{c})\right] \\
&= \mathbb{E}_{p(\mathbf{z}_{1:T}^{1:N}, \mathbf{c}_{1:T}^{1:N}|\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2})}\left[\nabla \log p(\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2}, \mathbf{z}_{1:T}^{1:N}, \mathbf{c}_{1:T}^{1:N})\right] \\
&= \sum_{\mathbf{k}} p(\mathbf{z}_1^{1:N} = \mathbf{k}|\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2}) \nabla \log\left[\prod_{n=1}^{N} p(\mathbf{y}_1^n|\mathbf{z}_1^n = k^n) \cdot p(\mathbf{z}_1^{1:N} = \mathbf{k})\right] \\
&\quad + \sum_{t=2}^{T} \sum_{\mathbf{k},\mathbf{j},\mathbf{u},\mathbf{v}} \xi(\mathbf{k}, \mathbf{j}, \mathbf{u}, \mathbf{v}) \nabla \log\Bigg[\prod_{n=1}^{N}\prod_{m=1}^{N} p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{z}_t^{m,n} = \mathbf{k}^{m,n}, \mathbf{y}_t^{m,n}) \cdot \prod_{n=1}^{N} p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n = k^n) \\
&\quad \cdot \prod_{n=1}^{N}\prod_{m=1}^{N} p(z_t^n = k^n|z_{t-1}^m = j^m, c_t^n = v^n, \mathbf{y}_{t-1}^{m,n}, e_{t-1}^{m\to n}) \cdot \prod_{n=1}^{N} p(c_t^n = v^n|c_{t-1}^n = u^n, z_{t-1}^n = j^n)\Bigg] \\
&= \sum_{\mathbf{k}} \gamma(\mathbf{k}) \nabla \log[B_1(k^n) \cdot \pi(\mathbf{k})] \\
&\quad + \sum_{t=2}^{T} \sum_{\mathbf{k},\mathbf{j},\mathbf{u},\mathbf{v}} \xi(\mathbf{k}, \mathbf{j}, \mathbf{u}, \mathbf{v}) \nabla \log[B_t(\mathbf{k}) \cdot E_t(\mathbf{k}) \cdot A_t(\mathbf{k}, \mathbf{j}, \mathbf{v}) \cdot C_t(\mathbf{u}, \mathbf{v}, \mathbf{j})],
\end{aligned}$$

where $\pi(\mathbf{k})$, $\gamma(\mathbf{k})$, $\xi(\mathbf{k},\mathbf{j},\mathbf{u},\mathbf{v})$, $B_t(\mathbf{k})$, $E_t(\mathbf{k})$, $A_t(\mathbf{k},\mathbf{j},\mathbf{v})$, and $C_t(\mathbf{u},\mathbf{v},\mathbf{j})$ are defined as

$$\begin{aligned}
\pi(\mathbf{k}) &= p(\mathbf{z}_1^{1:N} = \mathbf{k}),\\
\gamma(\mathbf{k}) &= p(\mathbf{z}_1^{1:N} = \mathbf{k}|\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2}),\\
\xi(\mathbf{k},\mathbf{j},\mathbf{u},\mathbf{v}) &= p(\mathbf{z}_t^{1:N}{=}\mathbf{k}, \mathbf{z}_{t-1}^{1:N}{=}\mathbf{j}, \mathbf{c}_t^{1:N}{=}\mathbf{v}, \mathbf{c}_{t-1}^{1:N}{=}\mathbf{u}|\mathbf{y}_{1:T}^{1:N}, \mathbf{e}_{1:T}^{1:N^2}),\\
B_t(\mathbf{k}) &= \prod_{n=1}^{N} p(\mathbf{y}_t^n|\mathbf{y}_{t-1}^n, z_t^n = k^n),\\
E_t(\mathbf{k}) &= \prod_{n=1}^{N}\prod_{m=1}^{N} p(e_t^{m\to n}|e_{t-1}^{m\to n}, \mathbf{z}_t^{m,n} = \mathbf{k}^{m,n}, \mathbf{y}_t^{m,n}),\\
A_t(\mathbf{k},\mathbf{j},\mathbf{v}) &= \prod_{n=1}^{N}\prod_{m=1}^{N} p(z_t^n{=}k^n|z_{t-1}^m{=}j^m, c_t^n{=}v^n, \mathbf{y}_{t-1}^{m,n}, e_{t-1}^{m\to n}),\\
C_t(\mathbf{u},\mathbf{v},\mathbf{j}) &= \prod_{n=1}^{N} p(c_t^n{=}v^n|c_{t-1}^n{=}u^n, z_{t-1}^n{=}j^n),
\end{aligned}$$

Among these, $\pi(\mathbf{k})$ is the initial joint discrete mode probability. $B_t(\mathbf{k})$ is the observation transition probability conditioned on motion modes $\mathbf{k}$. $E_t(\mathbf{k})$ is the discrete edge transition probability. $A_t(\mathbf{k},\mathbf{j},\mathbf{v})$ is the discrete motion mode transition probability. $C_t(\mathbf{u},\mathbf{v},\mathbf{j})$ is the mode count transition probability. Besides, $\gamma(\mathbf{k})$ and $\xi(\mathbf{k},\mathbf{j},\mathbf{u},\mathbf{v})$ are conditional posterior distributions, which can be calculated by the forward-backward algorithm in Appendix B.3.

C. More Experiments

C.1. Details of Datasets

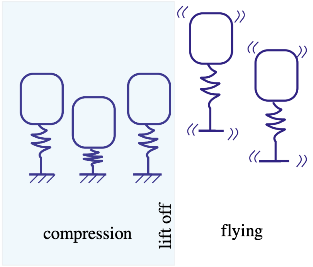


Figure 6. An illustration of Mass-spring hopper system (Brunton et al., 2016).

C.1.1. Mass-spring hopper

Figure 6 shows an illustration of a Mass-spring system that contains two motion modes, i.e. flying and compression. A minimal model of the Mass-spring hopper system is defined as

$$m\ddot{x} = \begin{cases} -k(x-x_0) - mg, & x \leqslant x_0 \\ -mg, & x > x_0 \end{cases},$$

where k, m, and g are the spring constant, mass, and gravity, respectively. x_0 is the unstretched spring length, which defines the flying $x > x_0$ and compression $x \leqslant x_0$ modes. After scaling by $\kappa = kx_0/mg$, the equations above becomes

$$\ddot{y} = \begin{cases} 1 - \kappa(y-1), & y \leqslant 1 \\ -1, & y > 1 \end{cases}.$$

Following Hybrid-SINDy (Mangan et al., 2019), we set $\kappa = 10$ for data generation. Denoting y as l and $\dot{y}$ as v, thus the target closed-form ordinary differential equations are

$$\begin{cases} \dot{l} = v \ \text{ and } \ \dot{v} = 11 - 10\,l, & l \leqslant 1 \\ \dot{l} = v \ \text{ and } \ \dot{v} = -1, & l > 1 \end{cases}. \tag{8}$$

The generated positions and velocities are concatenated $[l, v]$ and used as observations. Instead of generating only a few samples in Hybrid-SINDy (Mangan et al., 2019) (3 for training and 5 for validation), we scale up the datasets and sample 240 initial conditions from the ranges $(0.5, 3)$ and $(-1, 1)$ for positions a and velocities b, respectively. Among them, 200 samples are for training, 20 for validation, and 20 for testing. The system is simulated to generate 150 time steps for each time series, with sampling intervals of $\triangle_\tau = 0.033$. We add Gaussian noise with mean zero and standard derivation 10^{-6} to generated samples. By default, we use the first 100 time steps as context and predict the following next 50 time steps one by one based on the ground truth of the previous time step, i.e. one-step prediction. By default, the order of polynomial functions is set as 2, and the maximal number of possible modes is 3.

C.1.2. Susceptible, Infected and Recovered (SIR) Disease Dataset

The SIR disease model in the epidemiological community has been widely studied in the literature (Toda, 2020; McMahon et al., 2020). The model can be defined as

$$\begin{aligned}\dot{S} &= vN - \frac{\beta_t}{N} IS - dS,\\ \dot{I} &= \frac{\beta_t}{N} IS - (\gamma + d)I,\\ \dot{R} &= \gamma I - dR,\end{aligned}$$

where the rate of transmission β_t is time-varying, which takes two discrete values according to whether the school is in session or not

$$\beta_t = \begin{cases} \hat{\beta} \cdot (1 + b), & t \in \text{school in session},\\ \hat{\beta}/(1 + b), & t \in \text{school out of session}.\end{cases}$$

Following Hybrid-SINDy (Mangan et al., 2019), for dataset generation, the rates that define at which students enter and leave the population are set as $v = 1/365$ and $d = 1/365$. The total population of students is set as $N = 1000$. The recovery rate is set as $\gamma = 1/5$ assuming 5 days is the average infectious period. The base transmission rate is set as $\hat{\beta} = 9.336$ and $b = 0.8$ tunes the transmission rate change. Following Hybrid-SINDy (Mangan et al., 2019), the concatenation $[S, I]$ of S and I are used as observations. Thus the target closed-form ordinary differential equations are

$$\begin{cases} \dot{S} = 2.74 - 0.0168\ IS - 0.0027\ S \ \text{ and } \ \dot{I} = 0.0168\ IS - 0.20\ I, & t \in \text{school in session}\\ \dot{S} = 2.74 - 0.0052\ IS - 0.0027\ S \ \text{ and } \ \dot{I} = 0.0052\ IS - 0.20\ I, & t \in \text{school out of session}.\end{cases} \tag{9}$$

In a school year, the in-class periods are 35-155 and 225-365 days. The break periods are 0-35 and 155-225 days. Instead of creating only one time series for training and one for validation in Hybrid-SINDy, we scale up the datasets and sample 240 initial conditions for S_0, I_0, and R_0. For instance, in each sample, we first sample a R_0 from the range $(900, 980)$, and then sample a I_0 from the range $(0, 1000 - R_0)$, and then calculate S_0 by $S_0 = 1000 - R_0 - I_0$. We simulate each time series for 2 years with a daily interval, thus producing 730 time steps for each time series. We add a random perturbation to the start of each session by changing the states of S, I and R by either -2, -1, 0, 1, or 2, independently. By default, we use the first 600 time steps as context and predict the next 130 time steps one by one based on the ground truth of the previous time step, i.e. one-step prediction. By default, the order of polynomial functions is set as 2, and the maximal number of possible modes is 3 for our methods.

C.1.3. Non-hybrid Physical Systems

Following Course & Nair (2023), non-hybrid physical systems include the Coupled linear, Cubic oscillator, Lorenz' 63, Hopf bifurcation, Seklov glycolysis, and Duffing oscillator. Equations of a Damped linear oscillator are defined as $\dot{x} = -0.1x + 2y$ and $\dot{y} = -2x - 0.1y$. A Damped cubic oscillator is $\dot{x} = -0.1x^3 + 2y^3$ and $\dot{y} = -2x^3 - 0.1y^3$. A coupled linear system is $\ddot{x} = -6x + 2y$ and $\ddot{y} = 2x - 6y$. A Duffing oscillator is $\dot{x} = y$ and $\dot{y} = -x^3 + x - 0.35y$. A Selkov glycolysis is $\dot{x} = -x + 0.08y + x^2y$ and $\dot{y} = 0.6 - 0.08y - x^2y$. A Lorenz'63 system is $\dot{x} = 10y - 10x$, $\dot{y} = 28x - xz - y$, and $\dot{z} = xy - 2.67z$. A Hopf bifurcation is $\dot{x} = 0.5x + y - x^3 - xy^2$ and $\dot{y} = -x + 0.5y - x^2y - y^3$. We refer readers to see the details in (Course & Nair, 2023). By default, the order of polynomial functions of the Coupled linear, Cubic oscillator, Lorenz' 63, Hopf bifurcation, Seklov glycolysis, and Duffing oscillator are 2, 3, 2, 3, 3, and 3, respectively for our methods.

C.1.4. ODE-driven Particle Dataset

Following GRASS (Liu et al., 2023), Ordinary Differential Equations are introduced as motion modes to generate trajectories of particles, i.e. Lotka-Volterra, Spiral, and Bouncing Ball

$$\begin{aligned}&\mathrm{Lotka-Volterra}\colon \dot{x} = x - xy;\ \dot{y} = -y + xy,\\&\mathrm{Spiral}\colon \dot{x} = -0.1x^3 + 2y^3;\ \dot{y} = -2x^3 - 0.1y^3,\\&\mathrm{Bouncing\ Ball+}: \dot{x} = 0;\ \dot{y} = 2,\\&\mathrm{Bouncing\ Ball-}: \dot{x} = 0;\ \dot{y} = -2\end{aligned}\tag{10}$$

Balls are introduced on a squared 2d canvas of size $64 * 64$ which are with radius r and whose locations are randomly initialized. Trajectories of balls are generated by numerical values of different equations over time which are mapped to the canvas field. To simulate mode-switching behaviors, the driven ODE modes of two objects are switched when they collide in the canvas. Different from GRASS (Liu et al., 2023), one mode $\mathrm{Bouncing\ Ball}$ is regarded as two modes $\mathrm{Bouncing\ Ball+}$ and $\mathrm{Bouncing\ Ball-}$ in this work as they have different explicit equations for equation discovery. In summary, 4,928 samples are for training, 191 samples for validation, and 204 samples for testing. Each trajectory has 150 time steps with 10 frames per second. By default, the order of polynomial functions is set as 3, and the maximal number of possible modes is 5 for our methods.

C.1.5. Salsa-dancing Dataset

Following GRASS (Liu et al., 2023), four modes are annotated and used in the Salsa-dancing dataset, i.e. “moving forward”, “moving backward”, “clockwise turning”, and “counter-clockwise turning”. In summary, 1,321 samples are for training and 156 samples are for testing. Each sample has 100 time steps, among which 80 for context and the remaining 20 for prediction with 5 frames per second. The coordinates of the skeletal joints of dancers in 3D space are as observations. In practice, for all methods, we utilize two representative joints, i.e. right hip and left hip. By default, the order of polynomial functions is set as 3, and the maximal number of possible modes is 5 for our methods.

C.2. More Implementation Details

For each dataset, we set different numbers of modes K and orders of polynomial functions D for our model. By default, $K = 3$ and $D = 2$ for the Mass-spring Hopper dataset. $K = 3$ and $D = 2$ for the SIR dataset. D of the Coupled linear, Cubic oscillator, Lorenz’ 63, Hopf bifurcation, Seklov glycolysis, and Duffing oscillator are 2, 3, 2, 3, 3, and 3, respectively. $K = 5$ and $D = 3$ for the ODE-driven particle dataset. $K = 5$ and $D = 3$ for the Salsa-dancing dataset.

C.3. Statistics of Experiments

C.3.1. Mass-spring Hopper

Experiments with statistics on the Mass-spring Hopper dataset are reported in Tables 13 and 14, which are extended versions of Tables 1 and 2 in the main paper.

Table 13. Segmentation results with statistics on Mass-spring Hopper dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.426	0.383	0.705	0.691
AMORE (ours)	**0.928±0.011**	**0.967±0.013**	**0.991±0.005**	**0.993±0.007**

Table 14. Forecasting results with statistics on Mass-spring Hopper dataset.

Method	NMAE ↓	NRMSE ↓
LLMTime	0.113±0.032 / 0.305±0.036	0.417±0.051 / 0.454±0.072
SVI	0.068±0.016 / 0.075±0.011	0.148±0.023 / 0.262±0.030
AMORE (ours)	**0.008±0.003 / 0.039±0.008**	**0.026±0.005 / 0.059±0.006**

C.3.2. SIR DISEASE

Experiments with statistics on the SIR disease dataset are reported in Tables 15 and 16, which are extended versions of Tables 3 and 4 in the main paper.

Table 15. Segmentation results with statistics on the SIR disease dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.296	0.283	0.538	0.519
AMORE (ours)	**0.475±0.027**	**0.483±0.032**	**0.731±0.054**	**0.735±0.051**

Table 16. Forecasting results of Susceptible/Infected with statistics on the SIR disease dataset.

Method	NMAE ↓	NRMSE ↓
LLMTime	0.352±0.073 / 0.396±0.091	0.481±0.084 / 0.523±0.096
SVI	0.257±0.031 / 0.273±0.054	0.355±0.050 / 0.401±0.078
AMORE (ours)	**0.088±0.012 / 0.113±0.018**	**0.142±0.029 / 0.181±0.035**

C.4. Additional Ablation Studies

C.4.1. SAMPLING INTERVALS ANALYSIS

In our experiments, we followed the experimental setup of Hybrid-SINDy on the sampling intervals of the Mass-spring Hopper dataset and the SIR disease dataset. That means we use their standard sampling intervals Δ_t, e.g. $\Delta_t = 0.033$ on the Mass-spring Hopper dataset. In Table 17, we report the segmentation comparison results when Δ_t increases. We double the previous Δ_t each time and thus get $\{0.033, 0.066, 0.132, 0.264\}$. We can see that when $\Delta_t \geqslant 0.132$, the segmentation performance of Hybrid-SINDy decreases considerably due to the temporal pattern disruption, while our model has a smaller decrease in performance. When Δ_t increases (e.g. $\Delta_t \geqslant 0.132$), the discretization obviously disrupts the original temporal patterns of time series. Thus, after learning on the discretization, the model shows significantly decreased performance on *labels that are annotated based on the original temporal patterns*.

Table 17. Analyses of Δ_t on segmentation results of the Mass-spring Hopper dataset.

Sampling interval Δ_t	Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
0.033	Hybrid-SINDy	0.426	0.383	0.705	0.691
0.033	AMORE (ours)	**0.928±0.011**	**0.967±0.013**	**0.991±0.005**	**0.993±0.007**
0.066	Hybrid-SINDy	0.422	0.385	0.701	0.697
0.066	AMORE (ours)	**0.925±0.017**	**0.973±0.014**	**0.986±0.007**	**0.982±0.010**
0.132	Hybrid-SINDy	0.235	0.201	0.447	0.413
0.132	AMORE (ours)	**0.458±0.021**	**0.369±0.016**	**0.627±0.013**	**0.644±0.017**
0.264	Hybrid-SINDy	0.226	0.183	0.382	0.376
0.264	AMORE (ours)	**0.417±0.015**	**0.335±0.008**	**0.574±0.020**	**0.580±0.012**

C.4.2. NUMBER OF TRAINING SAMPLES ANALYSIS

To answer the question: “Given the significantly smaller datasets used by Hybrid-SINDy, can the proposed method maintain this level of performance difference?”, we rerun experiments on the Mass-spring Hopper dataset by varying the number of samples in the training set from 3 (the same as Hybrid-SINDy) to 20 and 200. The comparison results are summarized in Tables 18, 19, and 20. In the few-shot setting with a very low number of samples, e.g. 3 samples, Hybrid-SINDy outperforms AMORE. This is expected and a common limitation of deep learning methods, which usually require larger numbers of samples for training. On the other hand, when given more samples, e.g. larger than 20, AMORE outperforms Hybrid-SINDy consistently.

Table 18. Analyses of numbers of training samples on segmentation results of the Mass-spring Hopper dataset.

Number of training samples	Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
3	Hybrid-SINDy	**0.425**	**0.377**	**0.693**	**0.684**
3	AMORE (ours)	0.238±0.052	0.217±0.065	0.474±0.134	0.429±0.110
20	Hybrid-SINDy	0.422	0.383	0.698	0.693
20	AMORE (ours)	**0.774±0.037**	**0.762±0.025**	**0.846±0.094**	**0.853±0.071**
200	Hybrid-SINDy	0.426	0.383	0.705	0.691
200	AMORE (ours)	**0.928±0.011**	**0.967±0.013**	**0.991±0.005**	**0.993±0.007**

Table 19. Analyses of numbers of training samples on forecasting results of Location/Velocity on the Mass-spring Hopper dataset.

Number of training samples	Method	NMAE ↓	NRMSE ↓
3	LLMTime	0.113±0.032 / 0.305±0.036	0.417±0.051 / 0.454±0.072
3	SVI	0.173±0.039 / 0.341±0.053	0.450±0.081 / 0.481±0.094
3	AMORE (ours)	**0.091±0.018 / 0.160±0.026**	**0.315±0.049 / 0.348±0.042**
20	LLMTime	0.113±0.032 / 0.305±0.036	0.417±0.051 / 0.454±0.072
20	SVI	0.094±0.020 / 0.147±0.024	0.302±0.038 / 0.381±0.044
20	AMORE (ours)	**0.036±0.012 / 0.057±0.018**	**0.106±0.025 / 0.129±0.031**
200	LLMTime	0.113±0.032 / 0.305±0.036	0.417±0.051 / 0.454±0.072
200	SVI	0.068±0.016 / 0.075±0.011	0.148±0.023 / 0.262±0.030
200	AMORE (ours)	**0.008±0.003 / 0.039±0.008**	**0.026±0.005 / 0.059±0.006**

Table 20. Analyses of numbers of training samples on reconstruction errors (RER) of discovered equations on the Mass-spring Hopper dataset. Numbers are of e^{-3}.

Number of training samples	Method	RER (e^{-3}) ↓
3	Hybrid-SINDy	**8.3**
3	AMORE (ours)	17.2 ± 2.4
20	Hybrid-SINDy	8.2
20	AMORE (ours)	**5.1±0.6**
200	Hybrid-SINDy	7.5
200	AMORE (ours)	**2.4±0.3**

C.4.3. Count Variables Analysis

The count variables are introduced by REDSDS (Ansari et al., 2021) to learn the duration distributions of each mode from the data and to avoid frequent mode-switching behaviors. We show below some ablations studies on count variables in the Mass-spring Hopper system, where the flying mode usually takes more than twice as many time steps as the compression mode. To quantitatively compare the discovered equations, we first report the equation reconstruction error (RER) for Hybrid-SINDy, AMORE, and AMORE w/o count variable, which are respectively $7.5e^{-3}$, $2.4e^{-4}$, and $2.8e^{-4}$. We can see that with count variables, AMORE has a lower equation reconstruction error than its counterpart without count variables. In Tables 21 and 22, we can see that count variables help AMORE learn fewer false-positive mode-switching behaviors, benefitting segmentation and forecasting.

Table 21. Analyse of count variables on segmentation results of the Mass-spring Hopper dataset.

Method	NMI ↑	ARI ↑	Accuracy ↑	F_1 ↑
Hybrid-SINDy	0.426	0.383	0.705	0.691
AMORE (ours)	**0.928±0.011**	**0.967±0.013**	**0.991±0.005**	**0.993±0.007**
AMORE w/o count (ours)	0.903±0.017	0.929±0.019	0.970±0.012	0.975±0.013

Table 22. Analyse of count variables on forecasting results of Location/Velocity on the Mass-spring Hopper dataset.

Method	NMAE ↓	NRMSE ↓
LLMTime	0.113±0.032 / 0.305±0.036	0.417±0.051 / 0.454±0.072
SVI	0.068±0.016 / 0.075±0.011	0.148±0.023 / 0.262±0.030
AMORE (ours)	**0.008±0.003 / 0.039±0.008**	**0.026±0.005 / 0.059±0.006**
AMORE w/o count (ours)	0.014±0.004 / 0.046±0.007	0.052±0.011 / 0.068±0.014

C.4.4. Polynomial orders and mode numbers analysis

To qualitatively show the discovered equations when the order of candidates and the number of modes are increased, we increase the order of candidates from 2 to 5, i.e. $D = 5$, and the number of modes from 3 to 5, i.e. $K = 5$ on the Mass-spring Hopper dataset. The discovered equations are summarized in Table 23. We can see that our model can categorize exactly 2 modes, i.e. the same as the ground truth, no matter how many potential modes are introduced. Besides, the discovered equations of the 2 modes are regularized by sparsity promotion and do not involve irrelevant function terms thanks to the sparsity regularization when increasing the order of polynomial basis functions.

Table 23. Analyse of equation discovery of AMORE when increasing the number of modes K and the order of candidate basis functions D on the Mass-spring Hopper dataset.

Settings	Discovered Equations	Ground-truth Equations
$K = 3$ and $D = 2$	$\dot{l} = v$ and $\dot{v} = 11.03 - 10.08l$; $\dot{l} = v$ and $\dot{v} = -1$	$\dot{l} = v$ and $\dot{v} = 11 - 10l$; $\dot{l} = v$ and $\dot{v} = -1$
$K = 5$ and $D = 5$	$\dot{l} = v$ and $\dot{v} = 10.95 - 10.06l$; $\dot{l} = v$ and $\dot{v} = -1$	$\dot{l} = v$ and $\dot{v} = 11 - 10l$; $\dot{l} = v$ and $\dot{v} = -1$