Latent Space Energy-based Neural ODEs

Anonymous authors
Paper under double-blind review

Abstract

This paper introduces novel deep dynamical models designed to represent continuous-time sequences. Our approach employs a neural emission model to generate each data point in the time series through a non-linear transformation of a latent state vector. The evolution of these latent states is implicitly defined by a neural ordinary differential equation (ODE), with the initial state drawn from an informative prior distribution parameterized by an Energy-based model (EBM). This framework is extended to disentangle dynamic states from underlying static factors of variation, represented as time-invariant variables in the latent space. We train the model using maximum likelihood estimation with Markov chain Monte Carlo (MCMC) in an end-to-end manner. Experimental results on oscillating systems, videos and real-world state sequences (MuJoCo) demonstrate that our model with the learnable energy-based prior outperforms existing counterparts, and can generalize to new dynamic parameterization, enabling long-horizon predictions.

1 Introduction

Top-down dynamic generators, such as Pathak et al. (2017); Tulyakov et al. (2018); Xie et al. (2019; 2020), are the predominant models for representing and generating high-dimensional, regularly-sampled, discrete-time sequences, such as text and video. However, they are less suited for irregularly-sampled, continuous-time sequences, commonly found in medical datasets Goldberger et al. (2000), physical science Karniadakis et al. (2021). Neural ordinary differential equations (ODEs) (Chen et al., 2018) have emerged as a great approach for the continuous representation of time sequences, demonstrating impressive results in synthesizing new continuous-time sequences. Neural ODEs model hidden state dynamics using an ODE defined by a neural network, which takes the current state and time-step as input and outputs the derivative. Given an initial state, a neural ODE defines a continuous-time trajectory of hidden states through numerical ODE solver. As Chen et al. (2018) has shown, the solver's gradient with respect to the neural network parameters can be efficiently computed, enabling neural ODEs to serve as building blocks in advanced deep learning frameworks.

Neural ODEs have been adapted for continuous-time sequences through approaches like ODE-RNNs and Latent ODEs (Rubanova et al., 2019). ODE-RNNs generalize RNNs with continuous-time hidden dynamics, while Latent ODEs (Chen et al., 2018; Rubanova et al., 2019) extend latent variable sequential models to continuous dynamics. They assume the initial state follows an isotropic Gaussian distribution as the prior and recruit an additional inference network for variational inference (Kingma & Welling, 2014) during training. However, Latent ODEs face some limitations. First, the design of the inference network is not trivial since it has to capture the continue-time hidden dynamics effectively. Second, the Latent ODEs optimize the evidence lower bound as in variational auto-encoder (VAE) (Kingma & Welling, 2014), which introduces a non-zero gap, known as the Kullback–Leibler (KL) divergence, between the true posterior and the approximated Gaussian posterior. This gap can be large without careful design of the inference network and may limit the model's ability to capture complex data distributions accurately. Third, the simple Gaussian prior employed in Latent ODEs may insufficiently capture the complexity of initial latent state spaces, thus hindering the effective learning of latent dynamics.

To address the aforementioned issues, we propose ODE-LEBM, a novel latent space neural-ODE based dynamic generative model for continuous-time sequences. Our model integrates three key components: a neural ODE for flexible modeling of continuous latent dynamics, a top-down neural network as emission model mapping from latent to data space, and an energy-based model (EBM) prior (Pang et al., 2020) for the initial ODE state, enhancing expressiveness.

We employ an empirical Bayes approach, simultaneously training all components from observed sequences using maximum likelihood estimation (MLE) with Markov chain Monte Carlo (MCMC)-based inference. This eliminates the need for an additional inference network. The training process involves MCMC sampling of latent initial states from both EBM prior and posterior distributions. We update the EBM priors based on the statistical difference between these samples, while the ODE-based dynamic generator (including latent dynamics and emission model) is updated using posterior samples and observed data. This method provides a principled and statistically rigorous approach to training the ODE-LEBM model, addressing the limitations of previous neural-ODE based generative models for continuous-time sequential data.

Our experiments demonstrate that the proposed ODE-LEBM, coupled with an MCMC-based learning algorithm, effectively eliminates the need for complex inference network design. The model showcases robust capabilities in interpolation and extrapolation beyond the training distribution, while discovering meaningful global representations and adapting to real-world applications. Furthermore, the learned expressive EBM priors facilitate the disentanglement of latent dynamics from trajectory-specific semantics, thereby enhancing model interpretability.

Our work makes the following contributions:

1. We introduce the latent space energy-based neural ODE (ODE-LEBM), a novel ODE-based dynamic generative model with an energy-based prior, designed for continuous-time sequence generation.
2. We train the ODE-LEBM model MLE combined with MCMC-based inference. This training method is principled, statistically rigorous and eliminates the need for inference networks.
3. We present two variants of ODE-LEBM that can discover static and dynamic latent variables in complex systems, improving the model's generalizability and interpretability.
4. We conduct extensive experiments to assess the effectiveness and performance of the proposed ODE-LEBM model and the associated learning algorithms. These experiments evaluate the model's ability to generate realistic continuous-time sequences, learn disentangled and interpretable representations, and outperform existing approaches.

2 Related Work

Neural ODEs Neural ODEs Chen et al. (2018) aim to model continuous-time sequential data by using neural networks to parameterize the continuous dynamics of latent states. Rubanova et al. (2019) and Brouwer et al. (2019) propose continuous-time RNNs by incorporating neural ODE dynamics. Rubanova et al. (2019) and Yildiz et al. (2019) extend the state transition of dynamic latent variable models to continuous-time dynamics specified by neural ODEs and train these models within the VAE framework (Kingma & Welling, 2014). Our work also falls under the category of ODE-based latent variable models. However, unlike Chen et al. (2018); Rubanova et al. (2019); Yildiz et al. (2019), our model learns an informative prior for the neural-ODE-based dynamic generator instead of using an uninformative Gaussian prior. Additionally, we train our model via MCMC inference, eliminating the need for an additional assisting network for variational inference.

Latent space energy-based model As an intriguing branch of deep energy-based models (Xie et al., 2016; Nijkamp et al., 2019), Latent space EBMs (LEBMs) (Pang et al., 2020) have demonstrated that an EBM can serve as an informative prior model in the latent space of a top-down generator, and can be jointly trained with the generator using maximum likelihood. LEBMs have been successfully applied to a variety of tasks, including text generation (Pang & Wu, 2021; Yu et al., 2022), image generation (Zhu et al., 2023; Cui et al., 2023), trajectory generation (Pang et al., 2021), salient object detection (Zhang et al., 2021; 2023), unsupervised foreground extraction (Yu et al., 2021), and molecule design (Kong et al., 2023; 2024a) and offline reinforcement learning (Kong et al., 2024b). From a modeling perspective, Zhu et al. (2023) integrate a LEBM and a neural radiance field (NeRF) (Mildenhall et al., 2022), while Zhang et al. (2021) combine the LEBM with a vision transformer. Cui et al. (2023) incorporate the LEBM into a hierarchical generator, and Yu et al. (2023) pair the LEBM with a diffusion model (Ho et al., 2020). Our work leverages an LEBM alongside an ODE-based generator for modeling continuous-time sequences.

3 Latent Space Energy-based Neural ODE

Here, we present our method. Let $\mathbf{x} = (x_{t_i}, i = 1, ..., T)$ denote a sequence of data points, where $(t_i, i = 1, .., T)$ are their timestamps and T is the sequence length. The time t_i is assumed to be known for the observation x_{t_i}.

We present a generative model, ODE-LEBM, to represent continuous-time sequences. The model consists of (1) a latent energy-based prior model for the initial state of the latent trajectory, (2) a neural ODE to describe the dynamics of the latent trajectory, and (3) an emission model that produces the time series from the latent trajectory. To be specific, we have

$$z_{t_0} \sim p_\alpha(z_{t_0}), \tag{1}$$

$$z_{t_i} = z_{t_0} + \int_{t_0}^{t_i} f_\gamma(z(t), t)\, \mathrm{d}t, \tag{2}$$

$$x_{t_i} \sim p_\beta(x_{t_i}|z_{t_i}). \tag{3}$$

The energy-based prior model for the initial state in Eq. (1) can be defined as the exponential tilting of a Gaussian distribution:

$$p_\alpha(z_{t_0}) = \frac{1}{Z} \exp\left(f_\alpha(z_{t_0})\right) p_0(z_{t_0}), \tag{4}$$

where f_α is a multilayer perceptron (MLP) with parameters α and $p_0 \sim \mathcal{N}(0, \sigma^2 I)$ is a known Gaussian distribution. $Z = \int \exp(f_\alpha(z_{t_0})) p_0(z_{t_0}) \mathrm{d}z_{t_0}$ is the intractable normalizing constant. The emission model in Eq. (3) is represented by $p_\beta(x|z_{t_i}) = \mathcal{N}(x; f_\beta(z_{t_i}), \sigma_\epsilon^2 I)$, where f_β is a neural network with parameters β. In Eq. (2), f_γ is a neural network parameterized by γ, which specifies a neural ODE Chen et al. (2018) given by:

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f_\gamma(z(t), t). \tag{5}$$

Given an initial state $z(t_0) = z_{t_0}$, the trajectory of the latent states $z(t)$ is implicitly defined by the ODE, and can be evaluated at any desired time using a numerical ODE solver:

$$z_{t_1}, z_{t_2}, \ldots, z_{t_T} = \mathrm{ODESolve}(f_\gamma, z_{t_0}, t_0, t_1, \ldots, t_T). \tag{6}$$

Following (Chen et al., 2018), we can treat the ODE solver as a differentiable building block for constructing a more complex neural network system.

For notation simplicity, let $\theta = (\alpha, \beta, \gamma)$ be the parameters of the whole model, and $\phi = (\beta, \gamma)$ be those in the ODE-based generation model. Although each x_{t_i} is generated by its latent state z_{t_i}, all $\{z_{t_i}, i = 1, ..., T\}$ are generated by the initial state z_{t_0}. We can actually write a summarized form for our model

$$\mathbf{x} = F_\phi(z_{t_0}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2 I), \quad z_{t_0} \sim p_\alpha(z_{t_0}), \tag{7}$$

where F_ϕ composes f_γ (i.e., ODESolve) and f_β.

The marginal distribution of our ODE-LEBM is defined as

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, z_{t_0}) \mathrm{d}z_{t_0} = \int p_\phi(\mathbf{x}|z_{t_0}) p_\alpha(z_{t_0}) \mathrm{d}z_{t_0} = \int \prod_{i=1}^{T} p_\phi(x_{t_i}|z_{t_0}) p_\alpha(z_{t_0}) \mathrm{d}z_{t_0}. \tag{8}$$

Suppose we observe a training set of continuous-time sequences $\{\mathbf{x}_m, m = 1, ..., M\}$. The model can be learned by maximizing the likelihood of the training set. The gradient of the log-likelihood is

$$\nabla_\theta \log p_\theta(\mathbf{x}) = \mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}[\nabla_\theta \log p_\theta(\mathbf{x}, z_{t_0})] = \mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}[\nabla_\alpha \log p_\alpha(z_{t_0}) + \nabla_\phi \log p_\phi(\mathbf{x}|z_{t_0})], \tag{9}$$

which involves the computation of

$$\mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}[\nabla_\alpha \log p_\alpha(z_{t_0})] = \mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}[\nabla_\alpha f_\alpha(z_{t_0})] - \mathbb{E}_{p_\alpha(z_{t_0})}[\nabla_\alpha f_\alpha(z_{t_0})], \tag{10}$$

and

$$\mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}[\nabla_\phi \log p_\phi(\mathbf{x}|z_{t_0})] = \mathbb{E}_{p_\theta(z_{t_0}|\mathbf{x})}\left[\frac{1}{\sigma_\epsilon^2}(\mathbf{x} - F_\phi(z_{t_0}))\nabla_\phi F_\phi(z)\right]. \tag{11}$$

Estimating the expectations in Eq. (10) and Eq. (11) relies on Markov chain Monte Carlo sampling to evaluate the prior distribution $p_\alpha(z_{t_0})$ in Eq. (4) and the posterior distribution $p_\theta(z_{t_0}|\mathbf{x})$.

We could use Langevin dynamics (Neal et al., 2011) to sample from a target distribution of the form $\pi(z) \propto \exp(-U(z))$ by performing a Γ-step iterative procedure with a step size δ, starting with an initial value $z^{(0)}$ sampled from a standard normal distribution,

$$z^{(\tau)} = z^{(\tau-1)} - \delta\nabla_z U(z) + \sqrt{2\delta}e^{(\tau)}, z^{(0)} \sim \mathcal{N}(0, I), e^{(\tau)} \sim \mathcal{N}(0, I), \quad \text{for} \quad \tau = 1, ..., \Gamma. \tag{12}$$

For the prior distribution $p_\alpha(z)$, the gradient of the potential function is given by $\nabla_z \log p_\alpha(z) = -\nabla_z f_\alpha(z) + z/\sigma^2$. For the posterior distribution $p_\alpha(z|\mathbf{x}) \propto p(\mathbf{x}, z) = p_\alpha(z)p_\phi(\mathbf{x}|z)$, the gradient of the potential function is given by $\nabla_z \log p_\theta(z|\mathbf{x}) = \nabla_z \left[||\mathbf{x} - F_\phi(z)||^2/2\sigma_\epsilon^2 - f_\alpha(z) + ||z||^2/2\sigma^2\right]$.

Algorithm 1 provides a complete description of the training algorithm for our ODE-LEBM.

Algorithm 1 Learning algorithm for our ODE-LEBM

Input: (1) Training sequences $\{\mathbf{x}_m\}_{m=1}^M$; (2) Number of learning iterations J; (3) Numbers of Langevin steps for prior and posterior $\{K^-, K^+\}$; (4) Langevin step sizes for prior and posterior $\{\delta^-, \delta^+\}$; (5) Learning rates for energy-based prior and ODE-based generator $\{\xi_\alpha, \xi_\phi\}$; (6) batch size n'.
Output: Parameters $\phi = (\beta, \gamma)$ for the neural-ODE-based generator and α for the energy-based prior model

1: Initialize α, β, and γ.
2: **for** $j \leftarrow 1$ to J **do**
3: Sample observed sequences $\{\mathbf{x}_m\}_{m=1}^{n'}$
4: For each $\mathbf{x}_m$, sample the prior $z_i^- \sim p_{\alpha_j}(z)$ using Γ^- Langevin steps in Eq. (12) with a step size δ^-.
5: For each $\mathbf{x}_m$, sample the posterior $z_m^+ \sim p_{\theta_j}(z|\mathbf{x}_m)$ using Γ^+ Langevin steps in Eq. (12) with a step size δ^+.
6: Update energy-based prior using Adam with the gradient $\nabla\alpha$ computed in Eq. (10) and a learning rate ξ_α.
7: Update the ODE-based dynamic generator using Adam with the gradient $\nabla\phi$ computed in Eq. (11) and a learning rate ξ_ϕ.
8: **end for**

4 Experiments

This section investigates the performance and adaptability of our proposed ODE-LEBM in various scenarios. By incorporating an EBM prior with an explicit probability density, ODE-LEBM provides a more expressive generative model compared to latent ODE. In the experiments, we evaluate the model's ability to capture complex dynamics from irregularly sampled sequences, discover different time-invariant variables, detect out-of-distribution samples, and demonstrate its applicability in real-world scenarios.

Baseline methods All our baseline methods use neural ODEs as the dynamical model. Their primary difference is in how they encode the latent space variables. Latent ODE Chen et al. (2018) encodes the initial latent space using an RNN. To address the issue of irregular sampling, Rubanova et al. (2019) proposes the ODE-RNN as an encoder. Additionally, to identify time-invariant variables, Modulated Neural ODEs (MoNODEs) Auzina et al. (2024) employs an additional inference network for encoding.

Implementation details The network architectures and training details are shown in the Appendix A.

4.1 Overview

Our empirical study adopts the convention from Neural ODEs and test our model in the following scenarios.

Irregularly-sampled time series We generate a synthetic dataset of 1000 one-dimensional trajectories, with each containing 100 irregularly sampled time points within the interval $[0, 5]$. The trajectories are generated using a sinusoidal function with a fixed amplitude of 1, frequency randomly sampled from $[0.5, 1]$, and starting points sampled from $\mathcal{N}(\mu = 1, \sigma = 0.1)$. The dataset is split into 80% for training and 20% for testing. The data generation setting follows Rubanova et al. (2019).

Bouncing balls with friction We use a modified version of the bouncing ball video sequences, a benchmark commonly employed in temporal generative modeling (Gan et al., 2015; Yildiz et al., 2019; Sutskever et al., 2008). The dataset is generated using the original implementation by Sutskever et al. (2008), with an additional friction term sampled from $U[0, 0.1]$ added to each trajectory. The dataset consists of 1000 training sequences of length 20, and 100 validation and 100 test trajectories with lengths of 20 and 40, respectively. This setup allows for a comprehensive evaluation of the model's ability to learn and predict the dynamics of the bouncing ball under varying friction. The data generation setting follows the implementation by Auzina et al. (2024).

Rotating MNIST The data is generated following the implementation by Casale et al. (2018); Auzina et al. (2024), with the total number of rotation angles set to $T = 16$. We include all ten digits from MNIST dataset, with the initial rotation angle sampled from all possible angles $\theta^n \sim \{0°, 24°, \ldots, 312°, 336°\}$. The training data consists of $N = 1000$ trajectories, each with a length of $T = 16$, corresponding to one cycle of rotation. The validation and test data consist of $N_{\text{val}} = N_{\text{test}} = 100$ trajectories, each with a sequence length of $T_{\text{val}} = T_{\text{test}} = 45$.

MuJoCo physics simulation MuJoCo physics simulation (Todorov et al., 2012), widely used for training reinforcement learning models, is employed in our experiments with three physical environments: *Hopper*, *Swimmer*, and *HalfCheetah* using the DeepMind control suite (Tunyasuvunakool et al., 2020). We sampled 10,000 trajectories for each environment with 100 timesteps each, with initial states randomly selected. The data was split into training and testing sets at an 80/20 ratio.

4.2 Irregularly-Sampled Time Series

First, we aim to verify that our proposed ODE-LEBM, along with its associated posterior inference and learning algorithm, performs effectively. Among various choices in the sinusoidal function dataset, those with irregular timesteps pose the greatest challenge. Rubanova et al. (2019) claims that RNN-based inference networks struggle to model time series with non-uniform intervals, proposing ODE-RNNs as an alternative to handle arbitrary time gaps between observations. Instead, we propose using an MCMC-based posterior inference method to avoid the intricate design of inference networks for learning latent ODEs. By incorporating the EBM prior, we are interested in the model's capability to handle irregularly sampled time series.

In training, we randomly subsample a small percentage of time points to simulate sparse observations. For evaluation, we measure the mean squared error (MSE) on the full time series, testing the learned model's interpolation capability. Experiments in Table 1 show the mean squared error for models trained on different percentages of observed points ranging from 0.1 to 0.5. The visualizations are shown in Figure 1. Our ODE-LEBM outperforms Latent ODE with both the RNN encoder and ODE-RNN variant, as well as NCDSSM (Ansari et al., 2023), which is a state space model designed for irregular sampling, validating that ODE-LEBM with MCMC-based posterior inference achieves better performance while eliminating the complexity of designing an inference network. Similarly to Latent ODEs, ODE-LEBMs reconstruct the time series pretty well even with sparse observations and improve performance with increasing observations.

4.3 Disentangle Trajectory-specific Latent Variables

Discovering variables that steer the evolution of underlying dynamics in complex dynamical systems can improve generalization to new dynamics and enhance interoperability. Latent variable modeling facilitates the learning of underlying dynamics, particularly when observations and dynamics reside in different spaces, such as in a video recording of a moving car on the road.

Table 1: Test Mean Squared Error (MSE) on the irregularly-sampled sinusoidal dataset.

Observed ratio	0.1	0.2	0.3	0.5
GRU-D	2.3562	0.0599	0.0483	0.0411
RNN-VAE	0.4200	0.4189	0.4180	0.4160
NCDSSM	0.1710	0.1105	0.1072	0.1029
Latent ODE (RNN enc)	0.1524	0.0374	0.0311	0.0327
Latent ODE (ODE-RNN enc)	0.0713	0.0311	0.0303	0.0246
ODE-LEBM	**0.0700**	**0.0304**	**0.0266**	**0.0233**

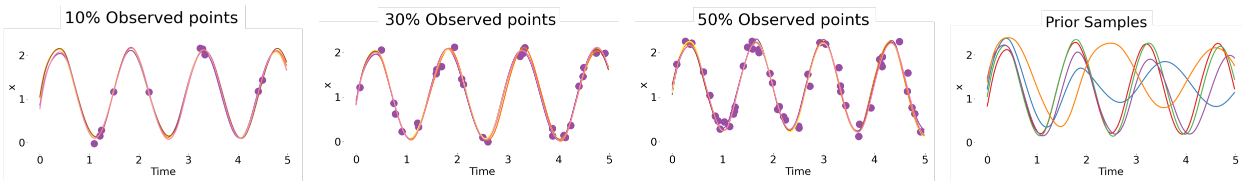


Figure 1: Interpolation results on irregularly-sampled time series. The latent initial state is sampled based on partial observations, $z_{t_0} \sim p_\theta(z_{t_0}|\mathbf{x})$, and then used to predict the entire sequence. For comparison, we also show the results of directly sampling from the learned latent EBM prior, $z_{t_0} \sim p_\alpha(z_{t_0})$ in the last column.

The top-down latent variable model (i.e., our ODE-LEBM) can be viewed as a specifically designed model for trajectory-specific variable discovery. The EM-type MLE learning algorithm updates the model parameters using the gradient $\mathbb{E}_{p_\theta(z|\mathbf{x})}[\nabla_\theta \log p_\theta(\mathbf{x}, z)]$ in Eq. (9). The learning process consists of two steps: (1) Infer the latent variable $z \sim p_\theta(z|\mathbf{x})$ from each trajectory using MCMC. The Langevin sampling of the posterior distribution can be viewed as a gradient-based learning of the trajectory-specific parameters, i.e., z. (2) Update the model parameters using $\nabla_\theta \log p_\theta(\mathbf{x}, z)$, where all trajectories share the same model parameters θ.

To further enhance model generalization, we disentangle the latent variable into dynamic latent variables z_d, which play a role in ODE evolution, and static variables z_s, which capture environment statistics unrelated to dynamics. For simplicity, we concatenate these variables with the initial latent state z_{t_0} and let them share the joint energy-based prior distribution. Unlike MoNODEs (Auzina et al., 2024), which infer deterministic dynamic and static variables using the average of the observation embeddings, our approach infers both latent variables based on their posterior distribution using Langevin dynamics.

4.3.1 Trajectory-specific static variables

The first variant of our proposed ODE-LEBM includes a time-invariant static variable z_s to capture variations not directly related to dynamics. z_s plays a role in the emission model, where $x_{t_i} \sim p_\beta(x_{t_i}|z_{t_i}, z_s)$. We assume $z \sim p_\alpha(z)$, where $z = [z_{t_0}, z_s]$, and $[\cdot]$ denotes concatenation. The model, learned by MLE using Algorithm 1, is summarized as follows:

$$[z_{t_0}, z_s] \sim p_\alpha([z_{t_0}, z_s]), \tag{13}$$

$$z_{t_i} = z_{t_0} + \int_{t_0}^{t_i} f_\gamma(z(t), t)\mathrm{d}t, \tag{14}$$

$$x_{t_i} \sim p_\beta(x_{t_i}|z_{t_i}, z_s). \tag{15}$$

We investigate our model's capability on the rotating-MNIST dataset in scenarios where sequence dynamics remain the same but observations differ due to varying static variables in the environment. We explore the following aspects: (1) Interpolation and extrapolation with the learned model; (2) Interpretability of both z_{t_0} and z_s; (3) Learned energy functions for out-of-distribution (OOD) detection.

Interpolation and Extrapolation To ensure a fair comparison with the baseline model (Auzina et al., 2024), we set the length of the sequences to 15 during training. For evaluation, we measure the MSE on test sequences with lengths of 15 and 45 to demonstrate our model's ability in both interpolation and extrapolation tasks. The interpolation performance is assessed on sequences of the same length as the training data, while the extrapolation performance is evaluated on three times longer sequences. Figure 2 visualizes the model predictions for sequences with 45 time steps, showcasing the model's ability to generate accurate predictions beyond the training sequence length. The experimental results, presented in Table 2, indicate that our model outperforms MoNODEs, which encode static variables with an additional convolutional neural network, in both interpolation and extrapolation tasks. Additionally, our model outperforms recent approaches like LatentApproxSDE (Solin et al., 2021) and Neural ODE Processor (Norcliffe et al., 2021). These findings demonstrate the effectiveness of the learned energy functions in the latent space, enabling our model to capture the underlying dynamics and generalize well to longer sequences.

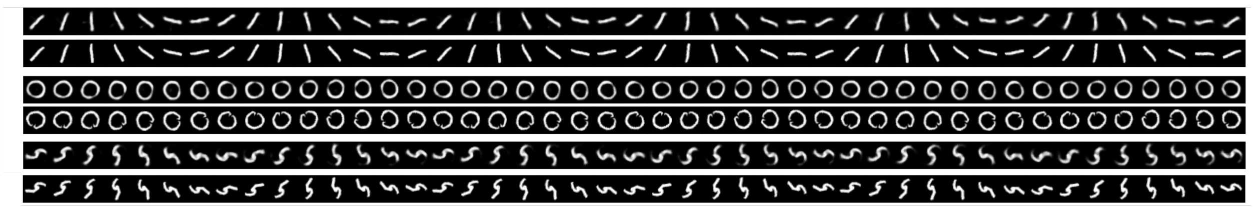


Figure 2: Interpolation and extrapolation results on the rotating-MNIST test set. The first 15 steps represent interpolation, while the last 30 steps represent extrapolation. The first row shows the model predictions for each digit, and the second row presents the ground-truth observations.

Table 2: Mean Squared Error (MSE) on Rotating-MNIST.

	T=15	T=45	NMI
Latent ODE	0.015(0.001)	0.039(0.001)	-
LatentApproxSDE	0.086(0.003)	0.099(0.003)	-
Neural ODE Processor	0.032(0.004)	0.035(0.002)	-
Modulated ODE	0.027(0.002)	0.030(0.001)	0.131
ODE-LEBM	**0.020**(0.000)	**0.024**(0.001)	**0.576**

Interpretability of initial states z_{t_0} and static variables z_s We analyze the interpretability of the learned latent variables z_{t_0} and z_s by sampling a random digit and generating 16 sequences of length 16 with initial angles set to $0°, 24°, \ldots, 312°, 336°$. After inferring the latent variables using posterior sampling (i.e., $[z_{t_0}, z_s] \propto p([z_{t_0}, z_s]|\mathbf{x})$), we project z_s and z_{t_0} into 2D using principal component analysis (PCA) to discover the rotating behavior. We study two types of sequences with the same observations but different time-step labels. We reset the time steps for the first type from 0 to 15, while for the second type, we keep the original time steps. In the first case (Figures 5a and 5b), z_{t_0} starts from the same initial position, and z_s captures the rotational behavior of each sequence. By resetting the starting time step, the ODEs focus on the underlying dynamics, which are the same since they originate from the same digit, and the rotational behavior emerges in z_s. In contrast, when not resetting the starting time step (Figures 5c and 5d), the ODEs infer the same starting point at time step 0, causing z_{t_0} to capture the rotational behavior, while z_s is reduced to the same cluster. This projection demonstrates that the static variables encode the initial rotations when the time steps are not reset. Furthermore, projecting z_{t_i} into 2D reveals closely aligned trajectories of the 16 sequences, indicating that our model learns consistent dynamics within the dataset, while observation differences are attributed to static variables.

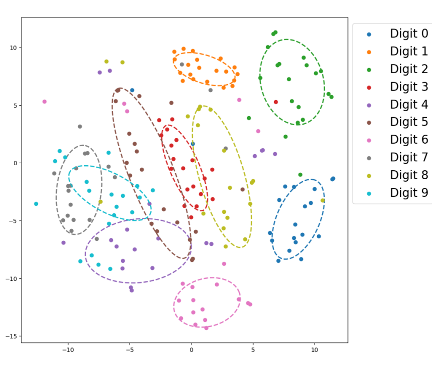
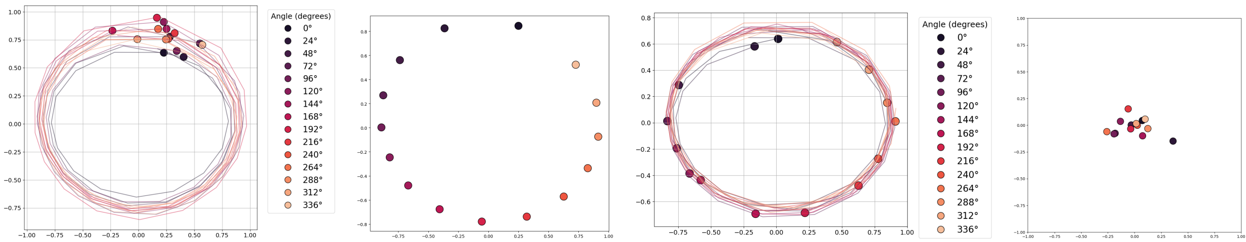


Figure 3: t-SNE plot of static variable z_s with randomly sampled sequences.

To investigate whether z_s can capture global information, such as the label of the digits, we randomly sample 20 sequences for each digit, with the initial rotation angle randomly set. We project the z_s of each sequence into 2D using t-SNE (Van der Maaten & Hinton, 2008), as shown in Figure 3. The projection reveals sequences corresponding to the same digit cluster together, indicating that z_s successfully encodes the digit label as global information.



(a) PCA plot of z_{t_0}. (b) PCA plot of z_s. (c) PCA plot of z_{t_0}. (d) PCA plot of z_s.

Figure 5: PCA embeddings of both z_{t_0} and z_s for the Rotating MNIST dataset, as inferred by posterior sampling. We generate 16 trajectories from a single digit, incrementing the initial angle of each trajectory by 24°, starting from 0° until 360°. In (a) and (c), circles denote the start of the trajectory (the initial angle), and lines represent the ODE trajectory. The color gradient corresponds to the initial angle of the trajectory in the observation space. (a) and (b) are from the same sequences with time reset, while (c) and (d) are from the same sequences without time reset.

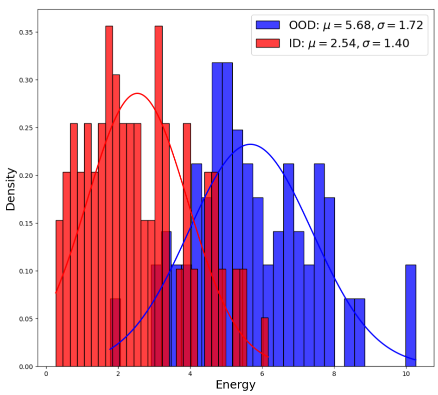


Figure 4: Out-of-distribution detection by energy function.

Furthermore, we compute the Normalized Mutual Information (NMI) of the clustering results to assess the quality of the learned representations quantitatively. MoNODE achieves an NMI of 0.131, while our method obtains a significantly higher NMI of 0.576, demonstrating that our approach produces more interpretable and informative latent variables regarding digit labels.

Out-of-distribution detection Energy-based model has been widely used to detect the out-of-distribution samples by treating the energy function as a generative classifier Elflein et al. (2021). To verify this ability, we generate OOD samples by randomly shuffling the sequence order of real samples. We then visualize the energy distribution of both real and OOD samples in Figure 4. The visualization shows that OOD samples predominantly fall into the high-energy region, while real samples are primarily located in the low-energy region.

4.3.2 Trajectory-specific dynamic variables

We investigate whether latent variables that steer the latent dynamics evolution can emerge with the simplest inductive bias, representing time-invariant hyperparameters in ODEs. We use a dataset of bouncing balls with friction, where each trajectory is sampled from a sequence with different friction coefficients drawn from $U[0, 0.1]$. During training, we observe videos of bouncing balls with length $T = 20$. At test time, we generate sequences with $T = 20$ for interpolation and $T = 40$ for extrapolation. During training and testing, only the first 10 timesteps are used to infer z_{t_0} and z_d.

The model, learned by MLE as in Algorithm 1, can be summarized as follows:

$$[z_{t_0}, z_d] \sim p_\alpha([z_{t_0}, z_d]), \tag{16}$$

$$z_{t_i} = z_{t_0} + \int_{t_0}^{t_i} f_\gamma(z(t), z_d, t)\mathrm{d}t, \tag{17}$$

$$x_{t_i} \sim p_\beta(x_{t_i}|z_{t_i}). \tag{18}$$

Table 3 shows the MSE on the test set, demonstrating that our model with dynamic latent variables outperforms other ODE-based counterparts. This highlights the effectiveness of incorporating trajectory-specific dynamic variables to make accurate predictions.

Table 3: Mean Squared Error (MSE) on bouncing balls with friction (BB).

	T=20	T=40	R^2
Latent ODE	0.0178(0.001)	0.0199(0.001)	-0.29
Modulated ODE	0.0110(0.000)	0.0164(0.001)	0.58
ODE-LEBM	**0.0099**(0.001)	**0.0159**(0.001)	**0.62**

4.4 Real-world application: MuJoCo physics simulation

We demonstrate that our ODE-LEBM can be adapted to complex systems like MuJoCo simulation Todorov et al. (2012). We experiment with three physical environments: *Hopper*, *Swimmer*, and *Cheetah*. To evaluate our method's ability to interpolate and extrapolate in these complex state sequences, we use a subsequence of length 50 during training. At test time, the model observes the first 50 time steps, infers z_{t_0}, and predicts the entire 100 time steps. The first 50 steps resemble the interpolation setting, while the last 50 steps require extrapolation beyond the training distribution. Figure 6 visualizes the prediction results compared to the ground truth. Table 4 shows that ODE-LEBM significantly outperforms latent ODE, demonstrating the capability of our proposed model in real-world applications.

Table 4: Mean Squared Error (MSE) on MuJoCo physics simulation.

	Hopper		Swimmer		HalfCheetah	
	T=50	T=100	T=50	T=100	T=50	T=100
Latent ODE	0.0097	0.0145	5.8e-05	6.3e-05	0.0112	0.0087
ODE-LEBM	**0.0076**	**0.0138**	**3.0e-06**	**4.9e-06**	**0.0081**	**0.0079**

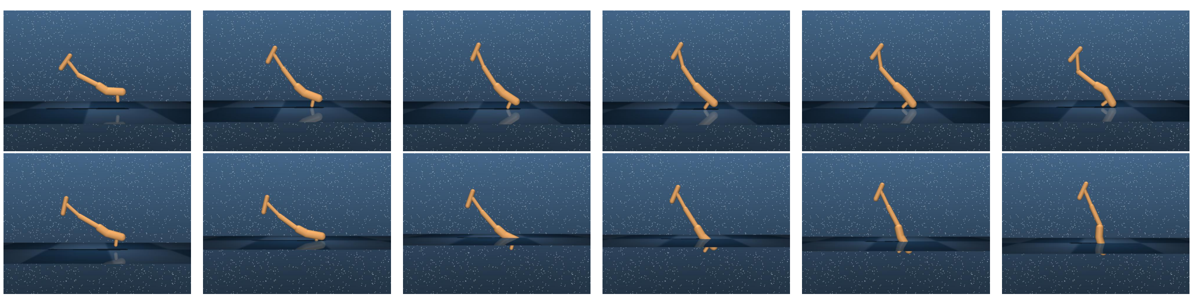


Figure 6: Visualization of prediction results on the MuJoCo *Hopper* environment with time interval 20 steps. The upper row shows the ground truth, while the lower row presents the model predictions. The first 50 time steps (first three columns) represent interpolation, where the model is conditioned on observed data, and the remaining 50 time steps (last three columns) correspond to extrapolation.

4.5 Ablation Studies

Number of Langevin sampling steps during test We conduct the experiment on the rotating MNIST, starting from 20 steps, which are set for training. We show that increasing the number of Langevin steps for the posterior during the test can further improve the performance as shown in Table 5.

The necessity of Latent EBM prior and MCMC-based inference To demonstrate the significance of the EBM prior model and MCMC-based inference in posterior inference, we conducted research using the Spiral2D experiment from Chen et al. (2018). We generate 1000 samples each for training and validation, and 200 samples for testing. The sequence length for training and validation is 100, while the test sequence length is 500. We train four models: Latent ODE, MCMC-based posterior inference without an EBM prior, Latent ODE with an EBM prior, and MCMC-based posterior inference with an EBM prior. The test MSE

Table 5: Ablation study on the number of steps in Langevin sampling.

Steps	20	50	100	1000
T=15	0.020	0.014	0.012	0.011
T=45	0.024	0.022	0.020	0.019

losses for these models are shown in Table 6. The substantial improvement achieved by using the EBM prior underscores its importance.

Table 6: The necessity of Latent EBM prior and MCMC-based inference.

MCMC-based inference	EBM prior	Test MSE
✗	✗	22.76
✗	✓	17.92
✓	✗	18.33
✓	✓	1.36

5 Conclusion

This paper presents a novel time series model, which is a latent space energy-based neural ODE (ODE-LEBMs) designed for continuous-time sequences. Our model specifies the dynamics of latent states using neural ordinary differential equations (Neural ODEs), represents the initial state space through a latent energy-based prior model, and maps latent states to data space using a top-down neural network. Through extensive experiments, we demonstrate that our model can effectively interpolate and extrapolate beyond the training distribution, unveil meaningful representations, and adapt to various scenarios, including learning from irregularly sampled sequences and identifying different time-invariant variables. Compared to its counterpart model, Latent ODE, our approach is more statistically rigours, accurate, and design friendly.

References

Abdul Fatir Ansari, Alvin Heng, Andre Lim, and Harold Soh. Neural continuous-discrete state space models for irregularly-sampled time series. In *International Conference on Machine Learning*, pp. 926–951. PMLR, 2023.

Ilze Amanda Auzina, Çağatay Yıldız, Sara Magliacane, Matthias Bethge, and Efstratios Gavves. Modulated neural odes. *Advances in Neural Information Processing Systems*, 36, 2024.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7377–7388, 2019.

Francesco Paolo Casale, Adrian Dalca, Luca Saglietti, Jennifer Listgarten, and Nicolo Fusi. Gaussian process prior variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 6572–6583, 2018.

Jiali Cui, Ying Nian Wu, and Tian Han. Learning hierarchical features with joint latent space energy-based prior. In *International Conference on Computer Vision (ICCV)*, pp. 2218–2227, 2023.

Gianfranco Doretto, Alessandro Chiuso, Ying Nian Wu, and Stefano Soatto. Dynamic textures. *Int. J. Comput. Vis.*, 51(2):91–109, 2003.

Sven Elflein, Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. On out-of-distribution detection with energy-based models. *arXiv preprint arXiv:2107.08785*, 2021.

Zhe Gan, Chunyuan Li, Ricardo Henao, David E Carlson, and Lawrence Carin. Deep temporal sigmoid belief networks for sequence modeling. *Advances in Neural Information Processing Systems*, 28, 2015.

A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations, (ICLR)*, 2014.

Deqian Kong, Bo Pang, Tian Han, and Ying Nian Wu. Molecule design by latent space energy-based modeling and gradual distribution shifting. In *Uncertainty in Artificial Intelligence (UAI)*, volume 216, pp. 1109–1120, 2023.

Deqian Kong, Yuhao Huang, Jianwen Xie, Edouardo Honig, Ming Xu, Shuanghong Xue, Pei Lin, Sanping Zhou, Sheng Zhong, Nanning Zheng, and Ying Nian Wu. Molecule design by latent prompt transformer. *Advances in Neural Information Processing Systems*, 2024a.

Deqian Kong, Dehong Xu, Minglu Zhao, Bo Pang, Jianwen Xie, Andrew Lizarraga, Yuhao Huang, Sirui Xie, and Ying Nian Wu. Latent plan transformer for trajectory abstraction: Planning as latent space inference. *Advances in Neural Information Processing Systems*, 2024b.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, 65(1):99–106, 2022.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent non-persistent short-run MCMC toward energy-based model. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5233–5243, 2019.

Alexander Norcliffe, Cristian Bodnar, Ben Day, Jacob Moss, and Pietro Liò. Neural {ode} processes. In *International Conference on Learning Representations*, 2021.

Bo Pang and Ying Nian Wu. Latent space energy-based model of symbol-vector coupling for text generation and classification. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 8359–8370, 2021.

Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Bo Pang, Tianyang Zhao, Xu Xie, and Ying Nian Wu. Trajectory prediction with latent belief energy-based model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11814–11824, 2021.

Jaideep Pathak, Zhixin Lu, Brian R Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12), 2017.

Yulia Rubanova, Tian Qi Chen, and David Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5321–5331, 2019.

Arno Solin, Ella Tamir, and Prakhar Verma. Scalable inference in sdes by direct matching of the fokker–planck–kolmogorov equation. *Advances in Neural Information Processing Systems*, 34:417–429, 2021.

Ilya Sutskever, Geoffrey E Hinton, and Graham W Taylor. The recurrent temporal restricted boltzmann machine. *Advances in neural information processing systems*, 21, 2008.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1526–1535, 2018.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dmcontrol: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Jianwen Xie, Yang Lu, Song-Chun Zhu, and Yingnian Wu. A theory of generative convnet. In *International Conference on Machine Learning (ICML)*, pp. 2635–2644, 2016.

Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Learning dynamic generator model by alternating back-propagation through time. In *The Thirty-Third AAAI Conference on Artificial Intelligence, (AAAI)*, pp. 5498–5507. AAAI Press, 2019.

Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Motion-based generator model: Unsupervised disentanglement of appearance, trackable and intrackable motions in dynamic patterns. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI))*, pp. 12442–12451. AAAI Press, 2020.

Çagatay Yildiz, Markus Heinonen, and Harri Lähdesmäki. ODE2VAE: deep generative second order odes with bayesian neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 13412–13421, 2019.

Peiyu Yu, Sirui Xie, Xiaojian Ma, Yixin Zhu, Ying Nian Wu, and Song-Chun Zhu. Unsupervised foreground extraction via deep region competition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 14264–14279, 2021.

Peiyu Yu, Sirui Xie, Xiaojian Ma, Baoxiong Jia, Bo Pang, Ruiqi Gao, Yixin Zhu, Song-Chun Zhu, and Ying Nian Wu. Latent diffusion energy-based model for interpretable text modeling. In *International Conference on Machine Learning (ICML)*, 2022.

Peiyu Yu, Yaxuan Zhu, Sirui Xie, Xiaojian Ma, Ruiqi Gao, Song-Chun Zhu, and Ying Nian Wu. Learning energy-based prior model with diffusion-amortized MCMC. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Jing Zhang, Jianwen Xie, Nick Barnes, and Ping Li. Learning generative vision transformer with energy-based latent space for saliency prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 15448–15463, 2021.

Jing Zhang, Jianwen Xie, Nick Barnes, and Ping Li. An energy-based prior for generative saliency. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(11):13100–13116, 2023.

Yaxuan Zhu, Jianwen Xie, and Ping Li. Likelihood-based generative radiance field with latent space energy-based model for 3d-aware disentangled image representation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 206, pp. 4164–4180. PMLR, 2023.

A Training details

For the emission model and neural ODE model, the network architecture designs are the same as the baseline model in Auzina et al. (2024); Rubanova et al. (2019). For EBM prior, all experiments use 3 layers MLP with activation GELU. The number of steps in Langevin sampling is 20 for training. We list the remaining training details in Table 7.

Table 7: The training details of prior model

Experiment name	hidden dim	step size	learning rate
Irregular sampling	25	0.01	1e-4
Bouncing balls	20	0.5	1e-5
Rotating MNIST	20	0.5	1e-5
Mojuco	25	0.001	1e-5

Computing resources for training All experiments can be done within an 11GB GPU like GTX 1080Ti. However, to speed up the training process, we use a single V100 for training.

B Limitations

This paper proposes using posterior inference to learn ODE-LEBM. Posterior sampling using Langevin dynamics as a form of test-time computation provides a flexible sampling process at the cost of multi-step model forward and backward computation. In the future, we shall study the potential behavior of sampling using Langevin dynamics as a stochastic differential equation or persistent Markov Chain to speed up the sampling process. In the future, we shall leverage the expressive ODE-LEBM to tackle more challenging large-scale real-world applications.

C Additional Related Work

Dynamical system. The dynamic texture model (Doretto et al., 2003) employs linear transition and emission models. In contrast, non-linear dynamic generator model has been used to approximate chaotic systems (Pathak et al., 2017), where innovation vectors are given as inputs, and the model is deterministic. The dynamic generator (Tulyakov et al., 2018; Xie et al., 2019; 2020) uses independent innovation vectors at each discrete time step, making the model stochastic. Our model does not incorporate randomness in the dynamical system and is focused on generating continuous-time sequences by leveraging neural ODEs (Chen et al., 2018).

D More visualization

In this section, we will provide more visualization results.

D.1 Bouncing balls with friction

We present the reconstruction results over 20 timesteps in Figure 7, demonstrating that our model can almost reconstruct the movement of the ball. In contrast, according to Auzina et al. (2024), the latent ODE is unable to achieve this, and MoNODE's reconstruction is also inadequate.

D.2 Hopper

We present one more pair of Hopper environment results in Figure 8.

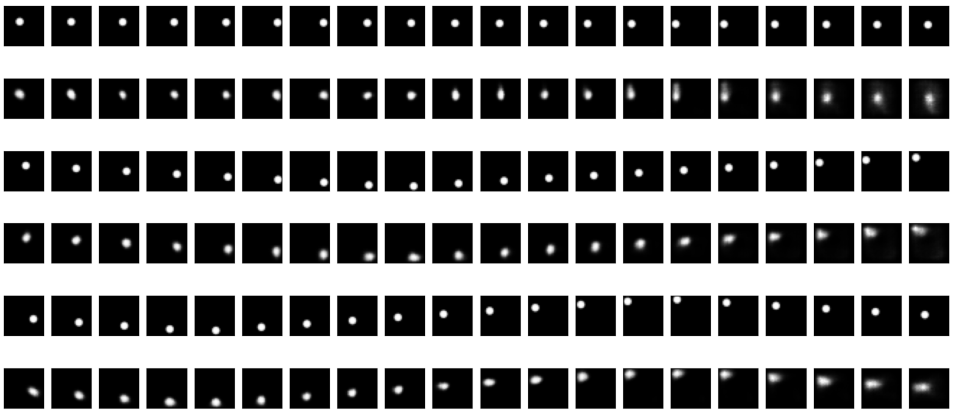


Figure 7: Visualization of prediction results on the Bouncing balls with friction. The first row is the ground truth and the second row is the prediction.

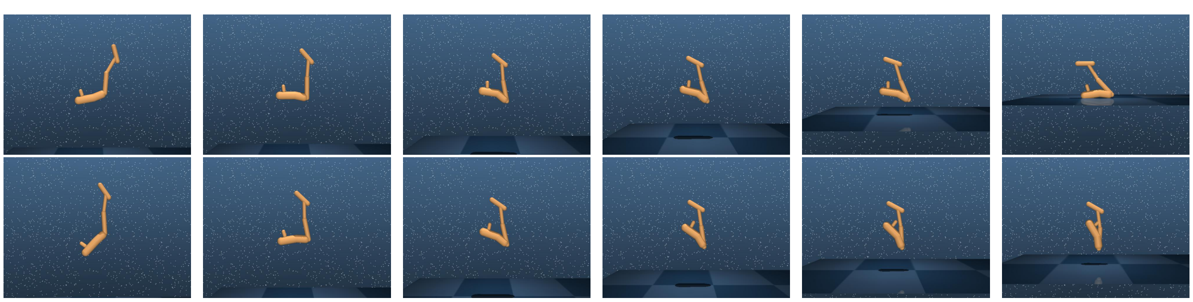


Figure 8: Visualization of prediction results on the MuJoCo Hopper environment. The first row is the ground truth and the second row is the prediction.