Learning Permutation-invariant Macroscopic Dynamics

Zhichao Han [1] **Mengyi Chen** [2] **Qianxiao Li** [2 1]

[1] *Institute for Functional Intelligent Materials, National University of Singapore* [2] *Department of Mathematics, National University of Singapore*. Correspondence to: Qianxiao Li qianxiao@nus.edu.sg.

1. Introduction

Accurately modeling the macroscopic dynamics of high-dimensional microscopic systems is of broad interest across the sciences [1]. Examples include modeling stretching dynamics of polymers [2], discovering reduced kinetic models for molecular conformational dynamics [3, 4], and learning macroscopic internal variables for history-dependent heterogeneous materials [5].

Most existing approaches learn closure variables, which are combined with given macroscopic variables for dynamics modeling, using an autoencoder trained to minimize the point-wise reconstruction loss. These methods commonly assume that the microscopic degrees of freedom admit a fixed ordering to be represented as vectors or tensors (Fig. 1(a)). However, many physical systems have no intrinsic ordering, such as a set of interacting particles. The desired closure variable must be *permutation-invariant* with respect to the input (Fig. 1(b)). Applying existing methods, which require a canonical ordering on the input, can be prohibitively difficult in practice.

This gap arises because the autoencoders parameterized by MLP [2, 6] or CNN [7] are not symmetric to permutation. One may consider adopting models for sets (e.g., DeepSet [8] or Set Transformer [9]) as the encoder to produce a permutation-invariant latent variable. Yet, a fundamental difficulty remains on the decoding side: without a specified ordering, there is no natural mechanism to reconstruct the microstate with the index-wise loss. For instance, consider a microstate $\{x^i\}_{i=1}^N$ with $N = 3$ particles. If a decoder is trained with a point-wise MSE loss to output an ordered list $(\hat{x}^1, \dots, \hat{x}^N)$, the loss will penalize $(\hat{x}^1, \hat{x}^2, \hat{x}^3)$ and $(\hat{x}^3, \hat{x}^2, \hat{x}^1)$ differently, even though they correspond to the same physical configuration. Crucially, the number of such physically equivalent permutations scales as $N!$. Therefore, training often relies on explicit matching [10, 11] or permutation-invariant losses [12, 13]; both add substantial computational cost [14] and may introduce optimization instability [15, 16].

To overcome this difficulty, we propose an alternative reconstruction objective for closure modeling. Unlike reconstructing microstates directly, we reconstruct their distribution. This distributional reconstruction eliminates the need to align individual points or impose the ordering, thereby enabling learning of permutation-invariant latent variables.

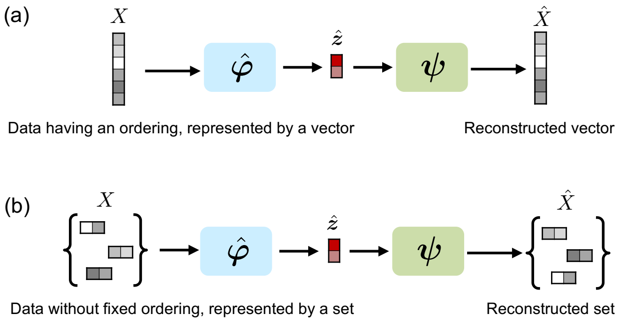

Fig. 1: Existing and desired autoencoder for closure modeling. (a) Standard autoencoder to learn the latent $\hat{z}$. Microstate X is represented by a vector based on the ordering. The encoder $\hat{\varphi}$ and decoder ψ are typically implemented as MLPs. (b) The desired autoencoder to learn $\hat{z}$. The model should produce a permutation-invariant latent variable $\hat{z}$ for inputs that lack the canonical ordering.

2. Method

2.1 Problem Statement

Let $X_t = \{x_t^1, x_t^2, \dots, x_t^n\} \in \mathcal{X}$ be the state of n *unordered* particles at time t, where $x_t^i \in \mathbb{R}^d$ is the state of particle with index i. A deterministic function $\bar{\varphi}$ is given beforehand to extract the macroscopic quantities of interest from the microscopic state, $\bar{\varphi} : \mathcal{X} \to \bar{\mathcal{Z}}$ where $\bar{\mathcal{Z}}$ is a space containing low-dimensional vectors. We restrict our discussion to intensive macroscopic observables such as the average energy of the interacting particle system, so that their magnitude does not scale with the system size n. The goal is to learn a permutation-invariant macroscopic dynamical model that can be applied to microscopic systems with varying numbers of particles.

To this end, an autoencoder is used to learn a low-dimensional representation that serves as the closure variable. The encoder, $\hat{\varphi}$, maps the microstate $X \in \mathcal{X}$ to a low-dimensional latent vector $\hat{z} = \hat{\varphi}(X)$. The latent $\hat{z}$ should be invariant to the ordering of $X = \{x^1, x^2, \dots, x^n\}$, i.e.,

$$\hat{\varphi}(\sigma X) = \hat{\varphi}(X), \quad \forall \sigma \in S_n, \tag{1}$$

where S_n denotes the symmetric group on n elements and $\sigma X := \{x^{\sigma(1)}, x^{\sigma(2)}, \dots, x^{\sigma(n)}\}$ is the permuted particle set. In parallel, we extract the macroscopic quantity of interest $\bar{z}_t = \bar{\varphi}(X_t)$, and form the closed macroscopic state $z_t = [\bar{z}_t, \hat{z}_t]$.

2.2 Closure Modeling

Our model to learn closure variables is illustrated by Fig. 2. First, the encoder $\hat{\varphi}$ learns permutation-invariant representation $\hat{z}$ (Eq. 1). We

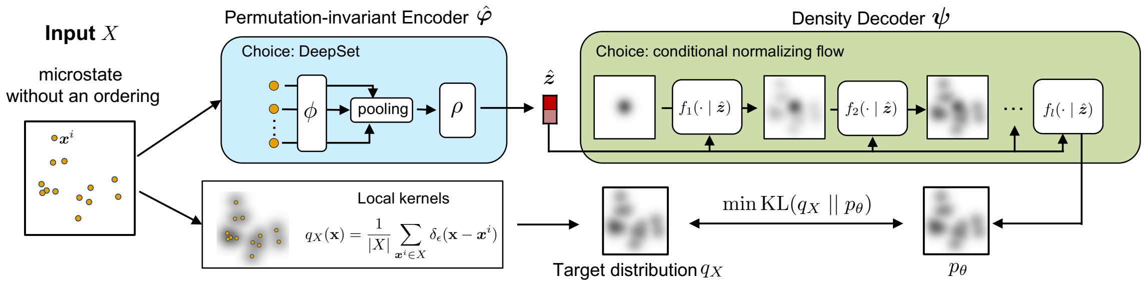

Fig. 2: Overview of our distribution-aware autoencoder for closure modeling. The encoder $\hat{\varphi}$ maps unordered microstate X to a permutation-invariant latent variable $\hat{z}$. The microstate X induces a target distribution q_X. Conditioned on $\hat{z}$, the decoder ψ generates a density $p_\theta(\mathbf{x}|\hat{z})$ to approximate the target density q_X. In general, $\hat{\varphi}$ is a permutation-invariant set function and ψ is a conditional density function. We instantiate $\hat{\varphi}$ with DeepSet and ψ with the conditional normalizing flow.

choose deepset [8] as the encoder, but it can be any permutation-invariant function. Then, for each $X \in \mathcal{X}$, we induce a continuous density $q_X(\mathbf{x})$ over the input space $\mathcal{X}$ that corresponds to a mass distribution centered at the observation points, i.e.,

$$q_X(\mathbf{x}) = \frac{1}{|X|} \sum_{x^i \in X} \delta_\epsilon(\mathbf{x} - x^i), \tag{2}$$

where $\delta_\epsilon(\cdot)$ is an isotropic Gaussian kernel with variance ϵ^2. The $q_X(\mathbf{x})$ serves as the reconstruction target. We construct the decoder ψ to generate a conditional density $p(\mathbf{x}|\hat{z})$ via $p_\theta(\mathbf{x}|\hat{z}) = \psi(\mathbf{x}; \hat{\varphi}(X))$, where θ denotes all learnable parameters in $\hat{\varphi}$ and ψ. We implement $p(\mathbf{x}|\hat{z})$ with a conditional normalizing flow, though any conditional density model can be used [17, 18]. We use the KL divergence to measure the discrepancy between $q_X(\mathbf{x})$ and $p_\theta(\mathbf{x}|\hat{z})$ [19]:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_X \left[\mathrm{KL}(q_X(\mathbf{x}) \,\|\, p_\theta(\mathbf{x}|\hat{z})) \right] . \tag{3}$$

The KL divergence can be estimated with Monte Carlo samples drawn from $q_X(\cdot)$. The sampling is easy since $q_X(\cdot)$ is a mixture of Gaussians.

2.3 Macroscopic Dynamics Modeling

We model the dynamics of the augmented macroscopic state $z_t = [\bar{z}_t, \hat{z}_t]$ by

$$dz_t = g(z_t)\, dt + \Sigma(z_t)\, dW_t, \tag{4}$$

where $g(z_t)$ is the drift term, $\Sigma(z_t)$ is the diffusion term, and W_t denotes standard Brownian motion. In the deterministic case, we set $\Sigma(z) \equiv 0$. In practice, we first train the $\hat{\varphi}$ and ψ by minimizing Eq. 3. Then, applying $\hat{\varphi}$ to get $\hat{z}$ and combining with $\bar{z}$, we train g and Σ by maximizing the likelihood of one-step transitions implied by an Euler–Maruyama discretization. Learning the dynamics of z naturally yields the dynamics of $\bar{z}$, which is our quantity of interest.

3. Experiment

We study energy evolution in particle systems as a test case to validate the effectiveness of our method. Specifically, we adopt the 2D interacting particle system from [20], where particle positions form the microstate X and $\bar{\varphi}$ computes the *normalized pairwise interaction energy*, which is the total pairwise interaction energy divided by the number of particle pairs. Fig. 3(a) visualizes an exemplary trajectory. After training, we select three trajectories in the test set, provide only their initial macroscopic state z_0, and recursively predict the future states. The prediction closely matches the ground truth throughout the rollout, as shown in Fig. 3(b). Furthermore, we take one test trajectory, apply three random permutations of the particle ordering, and compare the macroscopic prediction from our model and the baseline that uses MLP autoencoder. As shown in Fig. 3(c), our method is exactly permutation-invariant: the three prediction curves overlap, appearing as a single curve. Details about the experiment setup and the baseline are provided in Appendix A and B.

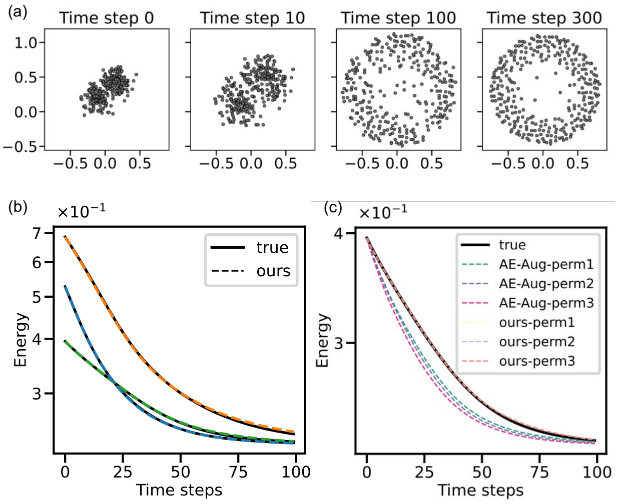

Fig. 3: Experiment on the interacting particle system. (a) Microstate snapshots examples. (b) $\bar{z}$ predicted by our model versus the ground truth. (c) Permutation-invariant verification.

4. Conclusion

In this work, we develop an autoencoder framework that learns permutation-invariant latent representations from microstates without the ordering, which can serve as the closure variables for learning macroscopic dynamics. We expect to apply the method to solve more real-world problems.

Acknowledgments

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG3-RP-2022-028).

References

[1] Dror Givon, Raz Kupferman, and Andrew Stuart. Extracting macroscopic dynamics: model problems and algorithms. *Nonlinearity*, 17(6):R55, 2004.

[2] Xiaoli Chen, Beatrice W. Soh, Zi-En Ooi, Eleonore Vissol-Gaudin, and Haijun Yu. Constructing custom thermodynamics using deep learning. *Nature Computational Science*, 4(1):66–85, 2023.

[3] Christoph Wehmeyer and Frank Noé. Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics. *The Journal of chemical physics*, 148(24), 2018.

[4] Andreas Mardt, Luca Pasquali, Hao Wu, and Frank Noé. Vampnets for deep learning of molecular kinetics. *Nature communications*, 9(1):5, 2018.

[5] Burigede Liu, Eric Ocegueda, Margaret Trautner, Andrew M Stuart, and Kaushik Bhattacharya. Learning macroscopic internal variables and history dependence from microscopic models. *Journal of the Mechanics and Physics of Solids*, 178:105329, 2023.

[6] Mengyi Chen and Qianxiao Li. Learning macroscopic dynamics from partial microscopic observations. *Advances in Neural Information Processing Systems*, 37:48996–49021, 2024.

[7] Kookjin Lee and Kevin T Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020.

[8] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

[9] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pages 3744–3753. PMLR, 2019.

[10] Yan Zhang, Jonathon Hare, and Adam Prugel-Bennett. Deep set prediction networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[11] Hamid Rezatofighi, Tianyu Zhu, Roman Kaskman, Farbod T Motlagh, Javen Qinfeng Shi, Anton Milan, Daniel Cremers, Laura Leal-Taixé, and Ian Reid. Learn to predict sets using feed-forward neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9011–9025, 2021.

[12] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018.

[13] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 206–215. IEEE, 2018.

[14] Trung Nguyen, Quang-Hieu Pham, Tam Le, Tung Pham, Nhat Ho, and Binh-Son Hua. Point-set distances for learning representations of 3d point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10478–10487, 2021.

[15] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13619–13627, 2022.

[16] Sasan Sharifipour, Constantino Álvarez Casado, Mohammad Sabokrou, and Miguel Bordallo López. Apml: Adaptive probabilistic matching loss for robust 3d point cloud reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[17] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[18] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[19] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[20] Theodore Kolokolnikov, Hui Sun, David Uminsky, and Andrea L. Bertozzi. Stability of ring patterns arising from two-dimensional particle interactions. *Phys. Rev. E*, 84:015203, Jul 2011.

[21] Ryan L. Murphy, Balasubramaniam Srinivasan, Vinayak Rao, and Bruno Ribeiro. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *International Conference on Learning Representations*, 2019.

Appendix A. Experiment Setup and Data Generation

We adopt the 2D interacting particle system from [20], where each particle interacts with all others through a pairwise step-force law $F(r) = \tanh(a(1-r)) + b$. We set $a = 4$ and $b = 0.1$, and simulate trajectories using a forward Euler integrator with time step $\Delta t = 0.002$. The initial particle positions are sampled from a two-component Gaussian mixture model, with component means and isotropic covariances randomly drawn per trajectory. The total pairwise energy of this particle system is computed as $E_{\text{total}} = \sum_{i<j} \left[\frac{1}{a} \log\cosh\left(a(1-r_{ij})\right) + b(1-r_{ij}) \right]$, where r_{ij} is the distance between particle i and particle j. We divide this total energy by $n \times (n-1)/2$ to get the normalized pairwise interaction energy. We generate 10k simulated trajectories and split them into training/validation sets with an 8/2 split. Each trajectory contains 300 particles recorded at 301 time steps. Since this macroscopic evolution is deterministic, we model z_t using an ODE (i.e., setting $\Sigma(z) = 0$ in Eq. 4). For testing, we generate 1k trajectories with the same number of particles as training, and sample initial positions also from a two-component Gaussian mixture model with independently resampled component means and isotropic covariances.

Appendix B. Baseline Setup

The baseline used in Sec. 3 that uses MLPs as encoder and decoder is specified as follows. To apply the MLP-style autoencoder to the microstates without the ordering, we adopt the common strategy of sampling random input permutations during training [21]. This can be interpreted as averaging over permutations, making the latent representation permutation-invariant in expectation. Combining this permutation-augmentation strategy with a standard MLP autoencoder trained using point-wise MSE reconstruction loss gives our baseline, denoted as *AE-Aug*.