Neural Canonical Transformation with Symplectic Flows

Shuo-Hui Li,1,2 Chen-Xiao Dong,1,2 Linfeng Zhang,3,* and Lei Wang1,4,5,†

1Institute of Physics, Chinese Academy of Sciences, Beijing 100190, China
2University of Chinese Academy of Sciences, Beijing 100049, China
3Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ 08544, USA
5CAS Center for Excellence in Topological Quantum Computation, University of Chinese Academy of Sciences, Beijing 100190, China
5Songshan Lake Materials Laboratory, Dongguan, Guangdong 523808, China

Canonical transformation plays a fundamental role in simplifying and solving classical Hamiltonian systems. We construct flexible and powerful canonical transformations as generative models using symplectic neural networks. The model transforms physical variables towards a latent representation with an independent harmonic oscillator Hamiltonian. Correspondingly, the phase space density of the physical system flows towards a factorized Gaussian distribution in the latent space. Since the canonical transformation preserves the Hamiltonian evolution, the model captures nonlinear collective modes in the learned latent representation. We present an efficient implementation of symplectic neural coordinate transformations and two ways to train the model. The variational free energy calculation is based on the analytical form of physical Hamiltonian. While the phase space density estimation only requires samples in the coordinate space for separable Hamiltonians. We demonstrate appealing features of neural canonical transformation using toy problems including two-dimensional ring potential and harmonic chain. Finally, we apply the approach to real-world problems such as identifying slow collective modes in alanine dipeptide and conceptual compression of the MNIST dataset.

I. INTRODUCTION

The inherent symplectic structure of classical Hamiltonian mechanics has profound theoretical and practical implications [1]. For example, the symplectic symmetry underlies the Liouville’s theorem [1], which states that the phase space density is incompressible under the Hamiltonian evolution. Both the Hamiltonian evolution and canonical transformations are symplectic flows in the phase space. Respecting the intrinsic symplectic symmetry of Hamiltonian systems are crucial for investigations such as celestial mechanics and molecular dynamics which requires stable and energy-conserving numerical integration schemes [2].

Molecular dynamics (MD) simulation investigates the dynamical and statistical properties of matter by integrating the equations of motion of a large number of atoms. MD is a vital tool for understanding complex physical, chemical, and biological phenomena, as well as for practical applications in material discovery and drug design. Modern MD simulation generates huge datasets, which encapsulate the full microscopic details of the molecular system [3]. However, this also poses challenges to the development of data analysis tools. In particular, one typically interests in the emerging slow modes, which are often related to the collective property of the system. Moreover, identifying such degrees of freedom is also crucial for enhanced sampling of molecular conformations. See Refs. [4, 5] for recent reviews.

Machine learning techniques provide promising solutions to these problems in MD. For example, the time-lagged independent component analysis (t-ICA) [6–8] separates a linear mixture of independent time-series signals. The approach shows a close connection to the dynamic mode decomposition scheme developed in the fluid mechanics’ community [9, 10]. More recently, several deep learning approaches have been proposed to identify nonlinear coordinate transformation of dynamical systems [11–15]. On the other hand, the slow feature analysis [16] and its generalizations [17] are used for identifying slow features from time-series data in the machine learning community.

In this paper, we develop a different approach by exploiting the symplectic symmetry of Hamiltonian dynamics and its inherent connection to the flow-based generative models [18–22]. We design neural networks to implement a symplectic transformation of the canonical variables. The resulting neural canonical transformations can simplify complex Hamiltonian dynamics of the physical variables towards independent collective motions in the latent phase space. Correspondingly, the symplectic neural network transforms the physical phase space densities towards an independent prior distribution while preserving the canonical relation of the dynamical variables. Conversely, the neural network can also directly generate complex phase space distribution as a flow-based generative model. We present learning algorithms and discuss possible applications of the neural canonical transformation, including the identification of slow collective variables and conceptual compression of the realistic dataset. We stress that most techniques targeting at extracting dynamical information require time-series data. However, in the present approach, the dynamical information is imposed on the structure of the neural canonical transformation, so that the training scheme does not necessarily follow a specific time step, and the data sample may come from other types of sampling methods, such as Monte Carlo, biased dynamics, etc.

There have been related research works exploiting the symplectic property in machine learning tasks. Ref. [23] solves Hamiltonian equations using neural networks. Refs. [24, 25] learns a Hamiltonian dynamics from data using neural net-
works. Both studies found that exploiting the symplectic structure in the learning helps in boosting the performance. Our work finds the closest connections to Ref. [26] which investigate classical integrable systems using constructed symplectic neural networks. Our paper targets at more general settings and trains the neural canonical transformation as a flow-based generative model with either variational or data-driven approaches.

The organization of the paper is as follows. In section II, we review canonical transformation for classical Hamiltonian dynamics and statistical physics. We then reveal its connection to the flow-based generative models. In section III, we present the symplectic neural network architecture and training approaches. In section IV we apply the neural canonical transformation first to toy problems and then to more sophisticated problems with realistic data. Finally, we discuss possible prospects of neural canonical transformation in Sec. V.

II. THEORETICAL BACKGROUND

We review the canonical transformations with an emphasize on its probabilistic aspect and intimate connection to the flow-based generative models.

A. Hamiltonian dynamics and Canonical Transformation

Consider a Hamiltonian system whose momenta and coordinates are collected into a row vector with $2n$ elements $x \equiv (p, q)$. The Hamiltonian equation can be concisely written as $\dot{x} = \nabla_x H(x)J$, where $\dot{x}$ denotes time derivative of the canonical variables. $H(x)$ is the Hamiltonian function and $J = \begin{pmatrix} 0 & -I \\ I & 0 \end{pmatrix}$ is a $2n \times 2n$ symplectic metric matrix.

The canonical transformation is a bijective mapping from the original variables to a new set of canonical variables, $\mathcal{T} : x \mapsto z = (P, Q)$, whose Jacobian matrix $M_J = \nabla_z z_i$ satisfies the symplectic condition

$$MJM^T = J. \quad (1)$$

We denote $x$ as the physical and $z$ as the latent phase space variables, respectively. The Hamiltonian equation is invariant under the canonical transformation, i.e. one has $\dot{z} = \nabla_z K(z)J$, where $K(z) = H \circ \mathcal{T}^{-1}(z)$ is the Hamiltonian corresponding to the latent phase space. The symplectic condition Eq. (1) implies that the all possible canonical transformations for this system form a group. Therefore, one can form a new canonical transformation by composing canonical transformations, and the inverse of a canonical transformation is also a canonical transformation.

The canonical transformation establishes a bijective mapping between the Hamiltonian trajectories in the latent space and the physical space, i.e., one has $\mathcal{T} (x + \int \dot{x} \, dt) = z + \int \dot{z} \, dt$ for an arbitrary evolution time. Therefore, although the equations of motion may appear different in the physical and the latent spaces, they describe the same Hamiltonian evolution as illustrated in Fig. 1.

Based on the ergodicity hypothesis and the Liouville’s theorem, the phase space points along the evolution trajectory, namely momenta and coordinates, can be considered as samples in the phase space. One can access the ensemble average of physical properties at equilibrium by simply computing the time average along the trajectory. In this paper, we focus on the canonical ensemble with a fixed temperature. The phase space density reads $\pi(x) = e^{-\beta H(x)}/Z$, and $Z = \int dxe^{-\beta H(x)}$ is the partition function, $\beta = k_B T$ is the inverse temperature.

Considering the ensemble distribution in the phase space highlights the probabilistic interpretation of the canonical transformations. Arriving at a simplified Hamiltonian function $K(z)$ also implies reaching a simplified phase density in the latent space. Interpreting the canonical transformation as a symplectic change-of-variables allows further investigations using the flow-based generative models.

B. Flow-based generative models

Generative modeling aims at modeling the joint distribution of complex high dimensional data [27]. A generative model can capture the key variations of the dataset and draw samples efficiently from the learned probability distribution. A large class of generative models achieves these goals by learning a latent representation that encodes collective patterns of the data. For example, the generative adversarial networks (GAN) transform latent variables which follow a simple prior distribution to the target distribution [28]; the variational autoencoders (VAE) model the conditional probabilities between the target distribution and the latent distribution using an encoder and a decoder network [29].

On the other hand, the flow-based generative models perform an invertible change of variables between the latent $z$ and the physical $x$ space. Thus, with a simple prior distribution for the latent variable, e.g. the Gaussian distribution $\mathcal{N}(z; 0, \Sigma)$ with zero mean and covariance $\Sigma$, the flow model has a tractable likelihood for any data $p(x) = \mathcal{N}(z; 0, \Sigma)\det(\nabla_x z)$. There have been large flavors of flow-based generative mod-
els [18–22] which achieve nice performance in machine learning applications. One of the key problems for the flow models is to design flexible bijective neural networks whose Jacobian determinants are efficiently computable. In this regard, one can view the canonical transformation as a generative model. The corresponding flow in the phase space is volume-preserving since the Jacobian determinant is always unity according to the symplectic condition Eq. (1). On the other hand, the flow-based generative models with the symplectic network also serve as flexible and adaptive canonical transformations.

Compared to GAN and VAEs, the flow models have appealing features such as tractable likelihood for any given data \( x \) and exact reversibility between \( z \) from \( x \). These features are particularly attractive for quantitative and principled scientific applications such as learning renormalization group flow [30], holographic mapping [31], Monte Carlo sampling [32–34] and molecular simulations [35]. Interestingly, in the continuous-time limit, the flow model exhibits intriguing connections to a variety of topics including dynamical systems, ordinary differentiation equations, optimal transport theory and fluid dynamics [36–38]. See Ref. [39] for a recent survey of flow-based generative models.

### III. CANONICAL TRANSFORMATION USING FLOW-BASED MODELS

It is usually difficult to devise useful canonical transformations for generic Hamiltonians since it typically involves solving a large set of coupled nonlinear equations. However, building on the connections of canonical transformation and flow-based generative models [18–22], we can construct a family of flexible canonical transformations with symplectic neural networks and learn them with optimization.

The training can follow either the variational approach or the data-driven approach. As a result, the neural canonical transformation helps simplify the dynamics and identify nonlinear slow collective variables of complex Hamiltonians.

#### A. Model Architectures

As a flow-based generative model, the neural canonical transformation consists of a prior distribution and a symplectic network. Although in the most general setting, the canonical transformation can even mix the momenta and coordinates, we restrict ourself to canonical coordinate transformations for balanced flexibility and interpretability. We list several other possible implementations of the neural canonical transformations in Appendix A. We note that one can compose symplectic neural networks to form more richer canonical transformations.

#### 1. Neural point transformations

The point transformation performs a nonlinear transformation to the coordinates, and a linear transformation to the momenta accordingly [40]

\[
Q = \mathcal{F}(q),
\]

\[
P = p(\nabla_q \mathcal{F})^{-1} = p\nabla_q q.
\]

The overall canonical transformation \( \mathcal{T} : x = (p, q) \mapsto z = (P, Q) \) satisfies the symplectic condition Eq. (1). In the second equality of Eq. (3) we have exploited the reversibility of the coordinate transformation. Interestingly, the momentum transformation has the form of a vector-Jacobian product, which is commonly implemented for the reverse mode automatic differentiation [41]. This is intuitively understandable since the momentum is covariant under the coordinate transformation.

We implement the coordinate transformation \( \mathcal{T} : q \mapsto Q \) in Eq. (2) using a bijective neural network, e.g., the real NVP model [20]. Since the coordinate transformation Eq. (2) is independent of momenta, we can use this sub-network alone as collective coordinates after training. We leverage the automatic differentiation mechanism for the momenta transformation \( P = \nabla_Q (p \cdot \mathcal{F}^{-1}(Q)) \). In practice, we run the coordinate transformation forwardly and then reversely for \( q = \mathcal{F}^{-1}(Q) \). Then, we compute its inner product with the initial momenta \( p \cdot q \). Finally, we compute its derivative with respect to \( Q \) to obtain the result of Eq. (3).

#### 2. Latent space Hamiltonian and prior distribution

We assume the latent space Hamiltonian has the form of an independent harmonic oscillator

\[
K(z) = \sum_{k=1}^{n} \frac{p_k^2}{2} + \omega_k^2 q_k^2,
\]

where \( \omega_k \) are learnable frequencies for each pair of conjugated canonical variables. Without loss of generality, we set the inverse temperature in the latent space to be one. Therefore, the prior distribution in the latent space is an independent Gaussian \( \mathcal{N}(z; 0, \Sigma) = \frac{1}{(2\pi)^{n/2}} \exp(-K(z)) \), where \( \Sigma = \text{diag}(1, \ldots, 1, \omega_1^{-2}, \ldots, \omega_n^{-2}) \) is a diagonal covariance matrix.

#### B. Training Approaches

The principle for training is to match the phase space density of the generative model \( \rho \) to the target density \( \pi \). However, depending on specific applications one may either have direct access to the Hamiltonian function or the samples from the target distribution. Thus, there are two training schemes of the neural canonical transformation based on variational calculation and the data-driven approach respectively.
\section{Variational approach}

We can learn the canonical transformation based on the analytical expression of the physical Hamiltonian. For this purpose, we minimize the variational free energy

$$\mathcal{L} = \int dx \rho(x) \left[ \ln \rho(x) + \beta H(x) \right].$$

(5)

This objective function is upper bounded by the free energy since $\mathcal{L} + \ln Z = \mathcal{KL} (\rho || \pi) \geq 0$, where the Kullback-Leibler (KL) divergence is a nonnegative measure of the dissimilarity between the model and the target distributions. The equality is reached only when two distributions match ether other. The objective function of this form was recently employed in probability density distillation of generative models [42].

To minimize Eq. (5) we employ the reparametrization trick [29], where we first draw samples from the prior distribution $z \sim \mathcal{N}(z; \mathbf{0}, \Sigma)$. then pass them through the symplectic transformation, shown in Fig. 1(a). Since the symplectic transformation preserves the probability density, the model density reads $\rho(x = T^{-1}(z)) = \mathcal{N}(z; \mathbf{0}, \Sigma)$. The reparametrization trick provides unbiased and low variance gradient estimator for training the learnable parameters.

\section{Maximum likelihood estimation}

Alternatively, one can also learn the canonical transformation in a purely data-driven approach. Assuming one already has access to independent and identically distributed samples from the target distribution $\pi(x)$, one can learn the neural canonical transformation with the maximum likelihood estimation on the data. This amounts to density estimation in the phase space with a flow-based probabilistic generative model. The goal is to minimize the negative log-likelihood (NLL) on the dataset $\mathcal{D} = \{x\}$

$$\text{NLL} = -E_{x \sim \mathcal{D}} \left[ \ln \rho(x) \right],$$

(6)

which amounts to minimize the observed phase space density and the model density $\mathcal{KL}(\pi || \rho)$ based on empirical observations. To train the network we run the transformation from the physical to latent space as shown in Fig. 1(b) and compute the model density $\rho(x) = \mathcal{N}(x; T(x); \mathbf{0}, \Sigma)$.

The density estimation Eq. (6) requires the phase space data, which involves both the coordinates and the momenta information. However, in practical MD simulations, one typically only records the trajectory data in the coordinate space. Fortunately, the momenta and coordinates distribution are factorized for a separable Hamiltonians. In particular, the momenta follow an independent Gaussian distribution whose variances are given by the atom masses and temperature. Therefore, one can exploit this fact and extend the training dataset by sampling momenta directly from a Gaussian distribution.

\section{Applications}

Neural canonical transformation learns a latent representation with simplified dynamics. In principle, the learned representation is useful for the prediction, estimation, and control of the original dynamical system. We list a few concrete applications below.

\subsection{Thermodynamics and excitation spectra}

Since the training approach of Sec. III B 1 satisfies the variational principle, the loss function Eq. (6) provides an upper bound to the physical free energy of the system. Besides, one can also estimate entropy and free energy differences of the Hamiltonian with different parameters. Similar variational free energy calculation of statistical physics problem using deep generative models has been carried out recently in Refs. [30, 35, 38, 43]. The present approach differs in the sense that it works in the phase space which involves both momenta and coordinates.

Moreover, since the neural canonical transformation preserves the dynamics of the system, the learned frequencies in the latent space Eq. 4 reflect the intrinsic time scale of the target problem. In this way, the present approach captures coherent excitation of the system in the latent space harmonic motion. One can also estimate the spectral density based on the learned frequencies.

\subsection{Identifying collective variables from slow modes}

Since the neural canonical transformation automatically separates collective modes with different frequencies in the latent space, one identifies slow collective motion of the original physical system by selecting the latent variables with small frequencies.

The neural canonical transformation differs fundamentally with these general time-series analysis approaches [6–10] which do not exploit the domain-specific symplectic symmetry of the dynamical system. Another fundamental difference is that the canonical transformation is done for the phase space which contains both momenta and coordinates, rather than for the time sequence of the coordinates. Since the explicit time information was never used in the present approach, there is no need to choose the time lag hyperparameter as in the tICA and related approaches. Lastly, variational training of the transformation also allows one to identify the canonical transformation directly from the microscopic Hamiltonian even without the time-series data.

We note that in practice, exact dynamical information is usually lost when one cares only about the structural, or static, information of a system. Sophisticated thermosetting and enhanced sampling techniques are used to accelerate the sampling, but, meanwhile, the dynamics are destroyed. In this case, MD plays the role of a sampler, rather than a real-time simulator, yet dynamical information can be extracted from our neural canonical transformation approach.
IV. EXAMPLES

We demonstrate the application of the neural canonical transformation with concrete examples. We start from simple toy problems and then move on to more challenging realistic problems. In all examples, the trainable parts of the network are the coordinate transformation $\mathcal{F}$ [Eq. (2)] and the latent space frequencies [Eq. (4)]. The code implementation is publicly available at [44].

A. Ringworld

First, we consider a two-dimensional toy problem with the Hamiltonian $H = \frac{1}{2} (p_1^2 + p_2^2) + \left( \sqrt{q_1^2 + q_2^2} - 2 \right)^2 / 0.32$ [32]. Canonical distribution of this Hamiltonian resides in four-dimensional phase space. The canonical ensemble samples from $\pi(x) = e^{-H(x)} / Z$ are confined in a manifold embedded in the phase space due to the potential term. In the Euclidean space, the coordinates are correlated.

Taking the Hamiltonian and train it with the variational approach, we obtain a neural point transformation from the original variables to a new set of canonical variables. Figure 2 shows the samples projected to the plane of latent coordinates $Q_k$ and the polar coordinate variables $\varphi = \arctan(q_2 / q_1), r = \sqrt{q_1^2 + q_2^2}$. One observes a significant correlation between the slowest variable $Q_1$ and the polar angle $\varphi$. While the other transformed coordinate $Q_2$ shows a strong correlation with the radius variable $r$.

This example demonstrates that, as a bottom line, the neural point transformation can automatically identify nonlinear transformation (such as polar coordinates) of the original coordinates. In the learned representation, the dynamics of each degree of freedom becomes independent.

B. Harmonic chain

Next, we consider a harmonic chain with the Hamiltonian $H = \frac{1}{2} \sum_{n=1}^{N} \left[ p_n^2 + (q_n - q_{n+1})^2 \right]$. We set $q_0 = q_{N+1} = 0$ to fix both ends of the chain. The system can be readily diagonalized by finding the normal modes representation $H = \frac{1}{2} \sum_{k=1}^{N} \left[ Q_k^2 + \omega_k^2 Q_k^2 \right]$, where $\omega_k = 2 \sin \left( \frac{\pi}{2N+2} \right)$ is the normal modes frequency and $Q_k = \sqrt{\frac{2}{\pi N}} \sum_{i=1}^{N} q_i \sin \left( \frac{ik\pi}{N+1} \right)$ is the normal coordinate.

We train a neural point transformation with the variational loss Eq. (5) at the inverse temperature $\beta = 1$. Figure 3(a) shows learned frequencies in the latent space harmonic Hamiltonian Eq. (4) together with the analytical dispersion. The agreement is particularly good for the low frequencies which are populated at the given temperature. Moreover, we pick the two slowest coordinates $Q_k$ and compute their Jacobians with respect to the physical variables $q_i$ as shown in Fig. 3(b). The comparison shows that the neural canonical transformation nicely identifies slow collective modes of the system bases on its Hamiltonian. On the other hand, the model is also able to learn these slow modes from data. In this simple case, modes with small frequency correspond to latent variables with large covariance, which could also be captured by the principal component analysis (PCA) [45].

Having demonstrated that the neural point transformation reproduces conventional normal modes analysis and PCA for the harmonic chain, we move on to show the major strength of the present approach in extracting nonlinear collective modes.

C. Alanine Dipeptide

Proteins show rich dynamics with multiple emergent time scales. As one of protein’s building block, the alanine dipeptide is a standard benchmark problem. Despite being small, it shows nontrivial dynamics. The backbone of alanine dippe-
We train a neural canonical transformation to identify the slow modes of the alanine dipeptide molecule based on raw MD simulation data. For the training, we use the MD trajectory [12, 46] released at [47]. The dataset consists of 250 ns of Euclidean space trajectory of the 10 heavy atoms in the alanine dipeptide at 300K with an integration step of 2 fs. Since the density estimation Eq. (6) requires the phase space data, we add to coordinates dataset the momenta sampled from the Gaussian distributions. The variances of the momenta Gaussian distribution are determined by the atom masses and temperature. Note that for the phase space density estimation we randomly shuffle the trajectory data, thus we do not use any of the time information in the training. We use the three MD independent trajectories at [47] for the training, validation, and testing, respectively. Each of them contains 250,000 snapshots. We use the Adam optimizer [48] with a minibatch size 200 and an initial learning rate $10^{-3}$ for training. We reduce the learning rate by a factor of 10 if there is no improvement of the loss function on the validation set for 10 training steps.

Figure 4(a) shows the learned frequencies, which spans a wide range and suggests the emergence of slow collective modes in the system. The frequency of the slowest modes is smaller than the fastest mode by more than three orders of magnitude. Assuming the fastest mode is of the order of the integration time step, the time scale of the slowest modes correspond to more than a few thousands of femtoseconds. Moreover, there is a notable gap in the frequencies, which suggests a separation of the fast and slow modes in the system.

We identify the latent coordinates with the smallest frequencies as the slowest nonlinear collective variables. To further reveal the physical meaning of these learned collective variables, we estimate the mutual information [49] between them and the two torsion angles $\Phi$ and $\Psi$ in Fig. 4(b). One sees that the first two latent coordinates show a significant correlation with $\Psi$ and $\Phi$, respectively. The mutual information between the other latent coordinates and the torsion angles decrease rapidly. Therefore, we conclude that the symplectic network has successfully identified the two slowest collective modes which capture the low energy physics. It is remarkable this is done without having any access to the time information in the MD trajectory. This success highlights the usefulness of imposing the symplectic symmetry in the transformation for identifying fast and slow modes. We note that with the current training objectives, the two slowest modes identified by the network will not exactly reproduce the torsion angles since their marginal distributions are not Gaussians. However, for a practical purpose, identifying the low dimensional manifold spanned by nonlinear slow modes provide an equally good description of the collective behavior of the system.

Thanks to the reversibility of the network, one can directly generate molecular configurations by sampling or interpolating or the collective coordinates. Figure 5 shows the molecule configurations of the heavy atoms as we tune the two slowest variables in the range $Q_k \in [-3/\omega_k, 3/\omega_k]$. One sees that the generated molecule configuration changes smoothly as the latent coordinate changes. Thus, by directly manipulating the latent variables one can directly control the collective degrees of freedom of the molecule. Note that a crucial difference of this approach compared to generating molecule configuration with VAE or GANs is that one has access to the likelihood of each configuration thanks to the tractability of the flow model. Having an exact likelihood allows unbiased sampling of the configuration space with rejection sampling [50, 51] or latent space Monte Carlo updates [30].

D. MNIST handwritten digits

Finally, we apply the neural canonical transformation to machine learning problems. We consider the MNIST handwritten digits dataset, which contains 50000 grayscale images of the size $28 \times 28$. These images are divided into ten-digit classes. Treating the pixel values as coordinate variables, we can view the digits classes as stable conformations of a physical system [52]. Similar to the dipeptide studied in the Sec. IV C, one conjectures that the transition between conformations are slow, while the variations within the digits classes are the fast degrees of freedom. For training, we augment the dataset with momenta and perform density estimation in the phase space.

Figure 6(a) shows the dispersion of the MNIST dataset, which clearly contains a small portion of slow frequencies over all the variables. To show that these slow modes contain the relevant information of the digits classes, we pass only these slow modes to a multilayer perceptron classifier and perform supervised training. The classifier contains a single hidden of neurons with rectified linear units. The trained symplectic neural network has its parameter fixed and works as a feature extractor. By varying the number of kept slow
modes from 5 up to 35 out of the total 784 dimensions, one sees that the classification accuracy on the test dataset quickly increases a plateau around 97% as shown in Fig. 6(b).

Reaching high classification accuracy with only a few of the slow collective variables motivates us further to perform the conceptual compression experiment to the MNIST dataset [20, 53]. Conceptual compression is a lossy compression scheme which aims at capturing the global information of the input data. The standard application makes use of the VAEs or the hierarchical network structure. However, we perform conceptual compression based on the learned frequencies since the symplectic network naturally separates fast and slow degrees of freedom of the dataset.

The top of Figure 7 shows the setup for conceptual compression. First, we use the learned nonlinear coordinate transformation Eq. (2) to map the data to the latent space. Then, we only pass a few slow modes to a decoder network. The decoder restores the image from the latent space by running the inverse transformation as the of the encoder network. To make up the missing information, we simply sample the high-frequency modes from the prior distribution and feed them into the decoder. The bottom of Figure 7 shows results of the conceptual compression, that from left to right we keep 5, 10, 15, 20, 25, 30, 35 of the slowest collective variables. The conceptual compression experiments show that the symplectic transformation captures the global information in the slow modes in the latent space since one can restore the image with only a small number of the slowest variables.

Figure 5. Latent space interpolation in the plane of the two slowest collective coordinates generates various molecule conformations.
n slowest collective variables. The remaining fast modes are thrown away and resampled from the prior distribution. The conceptual compression of the MNIST images. Images on the left-most column are from the test dataset. The remaining columns from left to right are reconstructions by keeping $n_{\text{slow}} = 5, 10, 15, 20, 25, 30, 35$ slowest collective variables.

V. DISCUSSIONS

Neural canonical transformation extends conventional theoretical tools by training symplectic neural networks as flow-based generative models. Symplectic flow models can extract dynamical information from statistical correlations in the phase space. These flexible and adaptive canonical transformations provide a systematic way to simplify Hamiltonian dynamics and extract nonlinear slow modes of dynamical systems.

Besides data analysis, the neural canonical transformation may open the door to address the sampling problem in MD simulations. As a bottom line, one is ready to employ the existing enhance sampling approaches [54–56] with the learned slow modes as the collective variables. Since these collective variables are differentiable and exhibit slow dynamics, it fulfills the typical requirement of collective variables. Moreover, one can even directly sample feasible molecule configurations by exploiting the fact that the learned canonical transformation is also a flow-based generative model. These samples would approximate the target distribution well if the generative model is trained well. Furthermore, one can correct the sampling bias by using the generative model as proposals in the Markov chain Monte Carlo [50, 51]. Lastly, one can perform Monte Carlo sampling in the latent space and then map the latent vectors to the physical samples [30, 35]. These latent space Monte Carlo updates extend the enhanced sampling approach based on variable transformations [57] to an adaptive setting.

It is also instructive to compare the present approach to probabilistic generative models [18–22]. Conventional generative models only concern about the statistical properties of data in the configuration space. While neural canonical transformations deal with phase space density. Therefore, they allow access to dynamical information by exploiting the symplectic structure in the transformation. Since the phase space density is factorized to momenta and coordinates parts for separable Hamiltonians, the KL divergence can be written as a summation of KL divergences for the marginal distributions. The KL divergence in the phase space is lower bounded by the KL divergence of the marginal distributions for the coordinates [58]. In this sense, one can view the KL divergence for the momenta distribution as a form of regularization for the neural canonical transformations. In addition, composing momenta together with the coordinates data in the density estimation can be viewed as a form of data augmentation.

A central problem with the latent variable generative model is that after learning one typically can not judge which latent variable plays a major role in the data generation. There have been various attempts to design hierarchical generative models [20, 59–62] to capture global information of data with a few variables. The neural canonical transformation differs by selecting the collective variables according to the learned frequencies. Therefore, one can assign a dynamical interpretation to the statistically learned latent representation. On the other hand, currently, we use a general real NVP model to perform the transformation of the coordinates. Additional symmetry constraints like the invariance under translation, rotation, and identical particle permutation, might be important for some future applications. In this case, we may devise the transformation by leveraging a symmetry-preserving energy model and using it to drive a gradient flow [38, 63].

We remark that reaching a perfect harmonic Hamiltonian Eq. (4) in the latent space is generally not possible because it implies the original system is fully integrable. In fact, a perfectly trained neural canonical transformation would reveal the invariant torus of integrable systems as shown in Ref. [26].
More generally, the Kolmogorov-Arnold-Moser theory \[64\] shows that the phase space trajectory of nearly-integrable systems would only be deformed from quasiperiodic motions. Thus, we expect the neural canonical transformation would work well for systems with coherent collective motion. Along this line, the present approach may also be useful to study synchronization phenomena \[65\], where a collective motion emerges out of complex dynamical systems. Finally, extending the present work to more general time-dependent canonical transformations may give even more powerful tool to study many-particle dynamical systems.

VI. ACKNOWLEDGMENT

We thank Austen Lamacraft, Masatoshi Imada, Yuan Wan, Pan Zhang, Yi-Zhuang You, Zi Cai, Lei-Han Tang, Yueshui Zhang, Dian Wu, Yantao Wu and Yixiao Chen for discussions. The work is supported by the Ministry of Science and Technology of China under the Grant No. 2016YFA0300603, the National Natural Science Foundation of China under Grant No. 11774398, and the Strategic Priority Research Program of Chinese Academy of Sciences Grant No. XDB28000000.

[1] V. I. Arnold, *Mathematical Methods of Classical Mechanics* (Springer, 1989).
[2] K. Feng and M. Qin, *Symplectic Geometric Algorithms for Hamiltonian Systems* (Springer, 2011).
[3] D. E. Shaw, P. Maragakis, K. Lindorff-Larsen, S. Piana, R. O. Dror, M. P. Eastwood, J. A. Bank, J. M. Jumper, J. K. Salmon, Y. Shan, and W. Wriggers, *Science* **330**, 341 (2010).
[4] F. Noé, (2018), arXiv:1812.07669.
[5] Y. Wang, J. M. L. Ribeiro, and P. Tiwary, (2019), arXiv:1909.11748.
[6] L. Molgedey and H. G. Schuster, *Phys. Rev. Lett.* **72**, 3634 (1994).
[7] G. Pérez-Hernández, F. Paul, T. Giorgino, G. De Fabritiis, and F. Noé, *J. Chem. Phys.* **139** (2013).
[8] C. R. Schwantes and V. S. Pande, *J. Chem. Theory Comput.* **9**, 2000 (2013).
[9] P. J. Schmid, *J. Fluid Mech.* **656**, 5 (2010).
[10] S. Klus, F. Nüske, P. Koltaï, H. Wu, I. Kevekridis, C. Schütte, and F. Noé, *J. Nonlinear Sci.* **28**, 985 (2018).
[11] A. Mardt, L. Pasquali, H. Wu, and F. Noé, *Nat. Commun.* **9**, 5 (2018), arXiv:1710.06012.
[12] C. Wehmeyer and F. Noé, *J. Chem. Physics* **148**, 241703 (2018).
[13] C. X. Hernández, H. K. Waymant-Steele, M. M. Sultan, B. E. Husic, and V. S. Pande, *Phys. Rev. E* **97**, 062412 (2018).
[14] M. M. Sultan and V. S. Pande, (2018), arXiv:1802.10510.
[15] B. Lusch, J. N. Kutz, and S. L. Brunton, *Nat. Commun.* **9** (2018), 10.1038/s41467-018-07210-0.
[16] L. Wiskott and T. J. Sejnowski, *Neural Comput.* **14**, 715 (2002).
[17] D. Pfau, S. Petersen, A. Agarwal, D. G. T. Barrett, and K. L. Stachenfeld, (2018), arXiv:1806.02215.
[18] D. J. Rezende and S. Mohamed, (2015), arXiv:1505.05770.
[19] L. Dinh, D. Krueger, and Y. Bengio, (2014), arXiv:1410.8516.
[20] L. Dinh, J. Sohl-Dickstein, and S. Bengio, (2016), arXiv:1605.08803.
[21] D. P. Kingma and P. Dhariwal, (2018), arXiv:1807.03039.
[22] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, (2018), arXiv:1810.01367.
[23] M. Mattheakis, P. Protopenas, D. Sondak, M. Di Giovanni, and E. Kaxiras, arXiv (2019), arXiv:1904.08991v1.
[24] S. Greydanus, M. Dzamba, and J. Yosinski, (2019), arXiv:1906.01563.
[25] Y. D. Zhong, B. Dey, and A. Chakraborty, (2019), arXiv:1909.12077.
[26] R. Bondezan and A. Lamacraft, (2019), arXiv:1906.04645.
[27] I. Goodfellow, (2016), arXiv:1701.00160.
[28] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, (2014), arXiv:1406.2661.
[29] D. P. Kingma and M. Welling, (2013), arXiv:1312.6114.
[30] S.-H. Li and L. Wang, *Phys. Rev. Lett.* **121**, 260601 (2018), arXiv:1802.02840.
[31] H.-Y. Hu, S.-H. Li, L. Wang, and Y.-Z. You, (2019), arXiv:1903.00804.
[32] J. Song, S. Zhao, and S. Ermon, (2017), arXiv:1706.07561.
[33] D. Levy, M. D. Hoffman, and J. Sohl-Dickstein, (2017), arXiv:1711.09268.
[34] M. S. Albergo, G. Kanwar, and P. E. Shanahan, *Phys. Rev. D* **100**, 34515 (2019), arXiv:1904.12072.
[35] F. Noé, S. Olsson, J. Köhler, and H. Wu, *Science* **365** (2019), 10.1126/science.aaw1147, arXiv:1812.01729.
[36] W. E, *Commun. Math. Stat.* **5**, 1 (2017).
[37] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, (2018), arXiv:1806.07366.
[38] L. Zhang, W. E, and L. Wang, (2018), arXiv:1809.10188.
[39] I. Kobyzev, S. Prince, and M. A. Brubaker, (2019), arXiv:1908.09257.
[40] G. Sussman and J. Wisdom, *Structure and interpretation of classical mechanics*, 2nd ed. (The MIT Press, 2014).
[41] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, *J. Mach. Learn. Res.* **18**, 1 (2015), arXiv:1502.05767.
[42] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Vukucuglu, G. van den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis, (2017), arXiv:1711.10433.
[43] D. Wu, L. Wang, and P. Zhang, *Phys. Rev. Lett.* **122**, 080602 (2018), arXiv:1809.10606.
[44] See https://github.com/til012589/neuralCT for code implementation in PyTorch.
[45] K. Pearson, *Philosophical Magazine* **2**, 559 (1901).
[46] F. Nüske, H. Wu, J. H. Prinz, C. Wehmeyer, C. Clementi, and F. Noé, *J. Chem. Phys.* **146**, 094104 (2017).
[47] https://markovmodel.github.io/mdshare/ALA2/#alanine-dipeptide.
[48] D. Kingma and J. Ba, in *Proceedings of the International Conference on Learning Representations (ICLR)* (2015).
[49] A. Kraskov, H. Stögbauer, and P. Grassberger, *Phys. Rev. E* **69**, 066138 (2004).
[50] L. Huang and L. Wang, Phys. Rev. B 95, 035105 (2017), arXiv:1610.02746.
[51] J. Liu, Y. Qi, Z. Y. Meng, and L. Fu, Phys. Rev. B 95, 041101(R) (2017), arXiv:1610.03137.
[52] See Ref. [38] for the preprocessing steps to map MNIST dataset to continuous variables.
[53] K. Gregor, F. Besse, D. J. Rezende, I. Danihelka, and D. Wierstra, (2016), arXiv:1604.08772.
[54] A. Laio and M. Parrinello, PNAS 99, 12562 (2002).
[55] O. Valsson and M. Parrinello, Phys. Rev. Lett. 113, 090601 (2014).
[56] L. Zhang, H. Wang, and W. E, The Journal of Chemical Physics 148, 124113 (2018), https://doi.org/10.1063/1.5019675.
[57] Z. Zhu, M. E. Tuckerman, S. O. Samuelson, and G. J. Martyna, Phys. Rev. Lett. 88, 100201 (2002).
[58] T. Salimans, D. P. Kingma, and M. Welling, (2014), arXiv:1410.6460.
[59] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, (2016), arXiv:1606.03657.
[60] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, in International Conference on Learning Representations, Vol. 3 (2017).
[61] T. Karras, S. Laine, and T. Aila, (2018), arXiv:1812.04948.
[62] H. P. Das, P. Abbeel, and C. J. Spanos, (2019), arXiv:1908.01686.
[63] L. Zhang, J. Han, H. Wang, W. A. Saidi, R. Car, and W. E, in Advances of the Neural Information Processing Systems (NIPS) (2018).
[64] H. S. Dumas, The KAM Story: A Friendly Introduction to the Content, History, and Significance of Classical Kolmogorov-Arnold-Moser Theory (World Scientific Publishing Company, 2014).
[65] J. A. Acebrón, L. L. Bonilla, C. J. Pérez Vicente, F. Ritort, and R. Spigler, Rev. Mod. Phys. 77, 137 (2005).
[66] B. Hall, Lie Groups, Lie Algebras and Representations, 2nd ed. (Springer, 2015).
[67] M. Lezcano-Casado, (2019), arXiv:1909.09501.
Appendix A: Symplectic Flows

We list several other forms of neural symplectic transformation and discuss their relation to known constructions in the literature.

1. Linear symplectic transformation

The simplest canonical transformation is a linear transformation to the input variables. We parameterize the linear symplectic transformation using the exponential map of its Lie algebra \([66]\),

\[
z = x e^Y \quad \text{with} \quad Y = \begin{pmatrix} A & B \\ C & -A^T \end{pmatrix},
\]

where \(B, C\) are real symmetric matrices and \(A\) is an arbitrary real matrix. One can implement Eq. (A1) via efficient vector-matrix exponential multiplication. Since the symplectic group is connected \([66]\), the exponential map covers all linear symplectic transformations. Moreover, one can obtain the reverse of the transformation by acting \(e^{-Y}\) instead. Accurate and efficient differentiation through the matrix exponential is discussed in Ref. \([67]\).

In the special case of \(B = C = 0\) and \(A\) is a skew symmetric matrix, i.e. \(A = -A^T\), the linear symplectic transformation reduces to the orthogonal transformation of both momenta and coordinates, which corresponds to the normal mode transformation.

2. Continuous symplectic flow

In general, one can parameterize the canonical transformation using a scalar generating function \(G(\lambda)\). Integrating the ordinary differential equation (ODE)

\[
\dot{\lambda} = \nabla_{\lambda} G(\lambda) J
\]

from time 0 to \(\tau\), one can transform the original variables from \(\lambda(t = 0) = x\) to \(\lambda(t = \tau) = z\). As a consequence, the Hamiltonian evolution is a special form of symplectic flow in the phase space with the generating function being the Hamiltonian \([1]\).

Equation (A2) corresponds to the infinitesimal canonical transformation \([1]\), which covers a broad family of symplectic transformations discussed so far. For example, if the generating function is a linear function of the momenta, we will arrive at the neural point transformation Eqs. (2, 3) introduced in the main texts. While if the generating function is a quadratic function of \(\lambda\) we obtain the linear symplectic transformation Eq. (A1).

The continuous symplectic flow falls into the framework of Monge-Ampère flow in the optimal transport theory \([38]\), where the transportation is induced by a gradient flow under a scalar potential function. The symplectic structure in Eq. (A2) simplifies the computation due to the volume-preserving property. In practice, the continuous transformation Eq. (A2) can be implemented via the neural ODE \([37]\). Since the symplectic symmetry is crucial for the canonical transformation, it is crucial to employ symplectic integrators \([2]\) in the NeuralODE implementation. In particular, if one employs a symplectic leap-frog discretization of the Eq. (A2) for a separable generating function, one will arrive at the transformations discussed in \([26, 33]\).