NONSEPARABLE SYMPLECTIC NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting the behaviors of Hamiltonian systems has been drawing increasing attention in scientific machine learning. However, the vast majority of the literature was focused on predicting separable Hamiltonian systems with their kinematic and potential energy terms being explicitly decoupled, while building data-driven paradigms to predict nonseparable Hamiltonian systems that are ubiquitous in fluid dynamics and quantum mechanics were rarely explored. The main computational challenge lies in the effective embedding of symplectic priors to describe the inherently coupled evolution of position and momentum, which typically exhibits intricate dynamics with many degrees of freedom. To solve the problem, we propose a novel neural network architecture, Nonseparable Symplectic Neural Networks (NSSNNs), to uncover and embed the symplectic structure of a nonseparable Hamiltonian system from limited observation data. The enabling mechanics of our approach is an augmented symplectic time integrator to decouple the position and momentum energy terms and facilitate their evolution. We demonstrated the efficacy and versatility of our method by predicting a wide range of Hamiltonian systems, both separable and nonseparable, including vortical flow and quantum system. We showed the unique computational merits of our approach to yield long-term, accurate, and robust predictions for large-scale Hamiltonian systems by rigorously enforcing symplectomorphism.

1 INTRODUCTION

A Hamiltonian dynamic system refers to a category of physical systems exhibiting certain forms of energy conservation during its temporal evolution. A typical example is a pendulum whose total energy (referred as the system's Hamiltonian) is conserved as a temporally invariant sum of its kinematic energy and potential energy. Mathematically, such energy conservation indicates a specific geometric structure underpinning its time integration, named as a *symplectic* structure, which further spawns a wide range of numerical time integrators to model Hamiltonian systems. These symplectic time integrators have proven their effectiveness on simulating a variety of energy-conserving dynamics, when Hamiltonian expressions are known as a prior. Examples encompass applications in plasma physics (Morrison, 2005), electromagnetics (Li et al., 2019), fluid mechanics (Salmon, 1988), and celestial mechanics (Saari & Xia, 1996), to name a few.

On another front, the emergence of the various machine learning paradigms with their particular focus on uncovering the hidden invariant quantities and their evolutionary structures (e.g. DiPietro et al., 2020; Jin et al., 2020) enable a faithful prediction of Hamiltonian dynamics without knowing its analytical energy expression beforehand. The key mechanics underpinning these learning models lies in a proper embedding of the strong mathematical inductive priors to ensure Hamiltonian conservation in a neural network data flow. Typically, such priors are realized in a variational way or a structured way. For example, in Greydanus et al. (2019), the Hamiltonian conservation is encoded in the loss function. This category of methods (see also Chen et al., 2020) does not assume any combinatorial pattern of the energy term and therefore relies on the inherent expressiveness of neural networks to distill the Hamiltonian structure from abundant training data. Another category of Hamiltonian networks, which we refer to as structured approaches, implements the conservation law indirectly by embedding a symplectic time integrator into the design of the network architecture (e.g., see Tong et al., 2020).

One of the main limitations of the current structured methods lies in the *separable* assumption of the Hamiltonian expression. Namely, the system's Hamiltonian has to be written as the sum of two

separable energy terms, i.e., $H(p, q) = V(p) + T(q)$, with V and T solely depending on p and q in separate. Examples of separable Hamiltonian systems include the pendulum (Jin et al., 2020), the Lotka–Volterra (Zhu et al., 2016), the Kepler (Antohe & Gladwell, 2004), and the Hénon–Heiles systems (Zotos, 2015). However, beyond this scope exist various nonseparable systems whose Hamiltonian has no explicit expression to decouple the position and momentum energies. Examples include incompressible flows (Suzuki et al., 2007), quantum systems (Bonnabel et al., 2009), rigid body dynamics (Chadaj et al., 2017), charged particle dynamics (Zhang et al., 2016), and nonlinear Schrödinger equation (Brugnano et al., 2018). This nonseperability typically causes chaos and instability, which further complicates the systems' dynamics. Further, these nonseparable systems exhibit plateau of degrees of freedom, demonstrating complexities that are orders-of-magnitude higher than separable systems (whose degree of Freedoms are typically below 10). Such chaotic and large-scale nature jointly adds shear difficulties for a conventional machine learning model to deliver faithful predictions. In particular, it is noteworthy that variational models are not well suited to tackling such challenges due to their inherent weaknesses in uncovering mechanics of large-scale systems featured by complicated dynamics, especially when the training data is partial and sparse.

In this paper, we propose an effective machine learning paradigm to predict nonseparable Hamiltonian systems. We build a novel neural network architecture, named as Nonseparable Symplectic Neural Networks (NSSNNs), to enable accurate and robust predictions of long-term Hamiltonian dynamics based on short-term observation data. Our proposed method belongs to the category of structured network architectures: it intrinsically embeds the symplectomorphisms into the network design to strictly preserve the symplectic evolution and further conserves the unknown, nonseparable Hamiltonian energy. The enabling techniques we adopted in our learning framework consist in an augmented symplectic time integrator to *asymptotically* "decouple" the position and momentum quantities that were nonseparable in their original form. To realize this, we enhance the dimensionality of the phase space by introducing a set of auxiliary variables that constitute an array of augmented Lagrangian multipliers to enhance the Hamiltonian expression. Our network design is motivated by ideas originated from physics (Tao, 2016) and optimization (Boyd et al., 2004). The combination of these mathematical observation and numerical paradigms results in a novel neural network architecture that can drastically enhance both the *scale* and *scope* of the state-of-the-art Hamiltonian predictions.

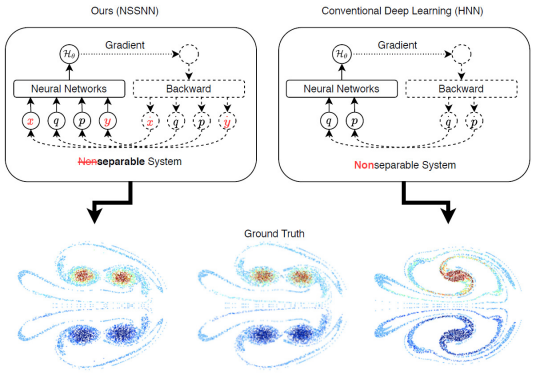

Figure 1: Comparison between NSSNN and HNN regarding the network design and prediction results of a vortex flow example.

.

We show a motivational example in Figure 1 by comparing our approach with a traditional HNN method (see Greydanus et al., 2019) regarding their structural designs and predicting abilities. We refer the readers to Section 4.3 for a detailed discussion. As shown in Figure 1, the vortices evolved using NSSNN are separated nicely as the ground truth, while the vortices merge together using HNN due to the failure of conserving the symplectic structure of a nonseparable system. The conservative capability of NSSNN springs from our novel design of the auxiliary variables (red x and y) which converts the original nonseparable system into a higher dimensional quasi-separable system where we can adopt a symplectic integrator.

2 Related Works

Data-driven physical prediction. Data-driven approaches have been widely applied in physical systems including fluid mechanics (Brunton et al., 2020), wave physics (Hughes et al., 2019), quantum physics (Sellier et al., 2019), thermodynamics (Hernandez et al., 2020), and material science (Teicherta et al., 2019). Among these different physical systems, data-driven fluid receives increasing attention. We refer the readers to Brunton et al. (2020) for a thorough survey of the fundamental

machine learning methodologies as well as their uses for understanding, modeling, optimizing, and controlling fluid flows in experiments and simulations based on training data. One of the motivations of our work is to design a versatile learning approach that can predict complex fluid motions. On another front, many pieces of research focus on incorporating physical priors into the learning framework, e.g., by enforcing incompressibility (Mohan et al., 2020; Baiges et al., 2020), the Galilean invariance (Ling et al., 2016), quasistatic equilibrium (Geng et al., 2020), and the invariant quantities in Lagrangian systems (Cranmer et al., 2020; Lutter et al., 2019) and Hamiltonian systems (Hernandez et al., 2020; Greydanus et al., 2019; Jin et al., 2020; Zhong et al., 2020). Here, inspired by the idea of embedding physics priors into neural networks, we aim to accelerate the learning process and improve the accuracy of our model.

Neural networks for Hamiltonian systems. Greydanus et al. (2019) introduced Hamiltonian neural networks (HNNs) to conserve the Hamiltonian energy of the system by reformulating the loss function. Based on HNNs, Chen et al. (2020) developed symplectic recurrent neural networks (SRNN), which is a recurrent HNN that relies on a symplectic integrator. Toth et al. (2020) developed the Hamiltonian Generative Network (HGN), learning Hamiltonian dynamics from high-dimensional observations. Moreover, Zhong et al. (2020) introduced Symplectic ODE-Net (SymODEN), which adds an external control term to the standard Hamiltonian dynamics. Unlike these works, our model 1) embeds a symmetric structure intrinsically into the neural networks, instead of manipulating the loss function, 2) implement symplectic integrator within a neural ODE-net (Chen et al., 2018) architecture, which enables approximation of the continuous time evolution of dynamical systems. Our work shares some similarities with (Jin et al., 2020)'s Symplectic networks (SympNets). Moreover, an intrinsic way to encode the symplectic structure is introduced by Tong et al. (2020). However, our work focuses on nonseperable Hamiltonian systems, which is too complicated to be fully addressed by any of the previous methods.

3 Framework

3.1 Augmented Hamiltonian equation

We start by considering a Hamiltonian system with N pairs of canonical coordinates (i.e. N generalized positions and N generalized momentum). The time evolution of canonical coordinates is governed by the symplectic gradient of the Hamiltonian (Hand & Finch, 2008). Specifically, the time evolution of the system is governed by Hamilton's equations as

$$\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}t} = \frac{\partial \mathcal{H}}{\partial \boldsymbol{p}}, \quad \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\frac{\partial \mathcal{H}}{\partial \boldsymbol{q}}, \tag{1}$$

with the initial condition $(\boldsymbol{q}, \boldsymbol{p})|_{t=t_0} = (\boldsymbol{q}_0, \boldsymbol{p}_0)$. In a general setting, $\boldsymbol{q} = (q_1, q_2, \cdots, q_N)$ represents the positions and $\boldsymbol{p} = (p_1, p_2, ...p_N)$ denotes their momentum. Function $\mathcal{H} = \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})$ is the Hamiltonian, which corresponds to the total energy of the system. An important feature of Hamilton's equations is its symplectomorphism (see Appendix A for a detailed overview).

The symplectic structure underpinning our proposed network architecture draws inspirations from the original research of (Tao, 2016) in computational physics. In Tao (2016), a generic, high-order, explicit and symplectic time integrator was proposed to solve (1) of an arbitrary separable and nonseparable Hamiltonian $\mathcal{H}$. This is implemented by considering an augmented Hamiltonian

$$\overline{\mathcal{H}}(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{x}, \boldsymbol{y}) := \mathcal{H}_A + \mathcal{H}_B + \omega \mathcal{H}_C, \tag{2}$$

with

$$\mathcal{H}_A = \mathcal{H}(\boldsymbol{q}, \boldsymbol{y}), \;\; \mathcal{H}_B = \mathcal{H}(\boldsymbol{x}, \boldsymbol{p}), \;\; \mathcal{H}_C = \frac{1}{2}\left(\|\boldsymbol{q} - \boldsymbol{x}\|_2^2 + \|\boldsymbol{p} - \boldsymbol{y}\|_2^2\right), \tag{3}$$

in an extended phase space with symplectic 2 form $\mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} + \mathrm{d}\boldsymbol{x} \wedge \mathrm{d}\boldsymbol{y}$, where ω is a constant that controls the binding of the original system and the artificial restraint.

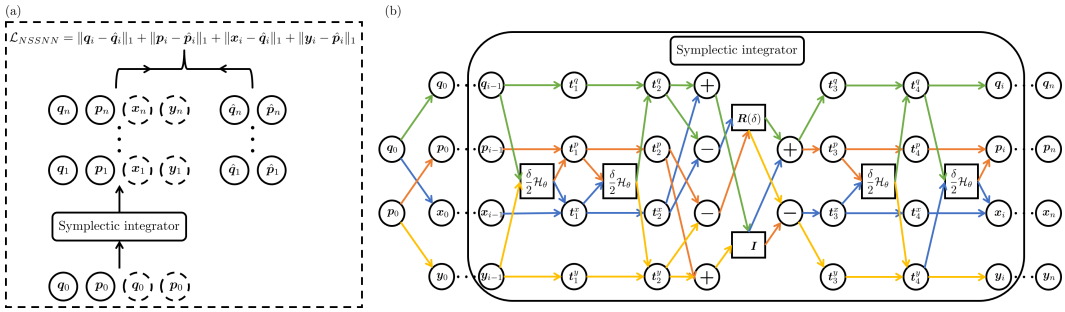

Figure 2: (a) The forward pass of an NSSNN is composed of a forward pass through a differentiable symplectic integrator as well as a backpropagation step through the model. (b) The schematic diagram of NSSNN.

Notice that the Hamilton's equations for $\overline{\mathcal{H}}$

$$\begin{cases} \dfrac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}t} = \dfrac{\partial \overline{\mathcal{H}}}{\partial \boldsymbol{p}} = \dfrac{\partial \mathcal{H}(\boldsymbol{x},\boldsymbol{p})}{\partial \boldsymbol{p}} + \omega(\boldsymbol{p}-\boldsymbol{y}), \\ \dfrac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\dfrac{\partial \overline{\mathcal{H}}}{\partial \boldsymbol{q}} = -\dfrac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{y})}{\partial \boldsymbol{q}} - \omega(\boldsymbol{q}-\boldsymbol{x}), \\ \dfrac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \dfrac{\partial \overline{\mathcal{H}}}{\partial \boldsymbol{y}} = \dfrac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{y})}{\partial \boldsymbol{y}} - \omega(\boldsymbol{p}-\boldsymbol{y}), \\ \dfrac{\mathrm{d}\boldsymbol{y}}{\mathrm{d}t} = -\dfrac{\partial \overline{\mathcal{H}}}{\partial \boldsymbol{x}} = -\dfrac{\partial \mathcal{H}(\boldsymbol{x},\boldsymbol{p})}{\partial \boldsymbol{x}} + \omega(\boldsymbol{q}-\boldsymbol{x}), \end{cases} \tag{4}$$

with the initial condition $(\boldsymbol{q},\boldsymbol{p},\boldsymbol{x},\boldsymbol{y})|_{t=t_0} = (\boldsymbol{q}_0,\boldsymbol{p}_0,\boldsymbol{q}_0,\boldsymbol{p}_0)$ have the same exact solution as (1) in the sense that $(\boldsymbol{q},\boldsymbol{p},\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{q},\boldsymbol{p},\boldsymbol{q},\boldsymbol{p})$. Hence, we can get the solution of (1) by solving (4). Furthermore, it is possible to construct high-order symplectic integrators for $\overline{\mathcal{H}}$ in (4) with explicit updates. Our model aims to learn the dynamical evolution of $(\boldsymbol{q},\boldsymbol{p})$ in (1) by embedding (4) into the framework of ODE-net. We remark that the coefficient ω in Hamiltonian (2) should be larger than a certain threshold for stability. Thus, we choose $\omega = 2000$ in this paper (see Appendix B). The coefficient ω acts as a regularizer, which stabilizes the numerical results.

3.2 Nonseperable Hamiltonian Neural Network

We learn the nonseparable Hamiltonian dynamics (1) by constructing an augmented system (4), from which we can obtain the energy function $\mathcal{H}(\boldsymbol{q},\boldsymbol{p})$ by training the neural network $\mathcal{H}_\theta(\boldsymbol{q},\boldsymbol{p})$ with parameter $\boldsymbol{\theta}$ and calculate the gradient $\nabla\mathcal{H}_\theta(\boldsymbol{q},\boldsymbol{p})$ by taking the in-graph gradient.

Figure 2 shows the architecture of NSSNN. For the constructed network $\mathcal{H}_\theta(\boldsymbol{q},\boldsymbol{p})$, we integrate (4) by using the second-order symplectic integrator (Tao, 2016). Specifically, we will have an input layer $(\boldsymbol{q},\boldsymbol{p},\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{q}_0,\boldsymbol{p}_0,\boldsymbol{q}_0,\boldsymbol{p}_0)$ at $t = t_0$ and an output layer $(\boldsymbol{q},\boldsymbol{p},\boldsymbol{x},\boldsymbol{y}) = (\boldsymbol{q}_n,\boldsymbol{p}_n,\boldsymbol{x}_n,\boldsymbol{y}_n)$ at $t = t_0 + n\mathrm{d}t$. The recursive relations of $(\boldsymbol{q}_i,\boldsymbol{p}_i,\boldsymbol{x}_i,\boldsymbol{y}_i), i = 1,2,\cdots,n$, can be expressed by the algorithm 1. The input functions in algorithm 1 are

$$\begin{aligned} \boldsymbol{\phi}_1^\delta : \begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}\\ \boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} &= \begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}-\delta[\partial\mathcal{H}_\theta(\boldsymbol{q},\boldsymbol{y})/\partial\boldsymbol{q}]\\ \boldsymbol{x}+\delta[\partial\mathcal{H}_\theta(\boldsymbol{q},\boldsymbol{y})/\partial\boldsymbol{p}]\\ \boldsymbol{y}\end{bmatrix}, \quad \boldsymbol{\phi}_2^\delta : \begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}\\ \boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} = \begin{bmatrix}\boldsymbol{q}+\delta[\partial\mathcal{H}_\theta(\boldsymbol{x},\boldsymbol{p})/\partial\boldsymbol{p}]\\ \boldsymbol{p}\\ \boldsymbol{x}\\ \boldsymbol{y}-\delta[\partial\mathcal{H}_\theta(\boldsymbol{x},\boldsymbol{p})/\partial\boldsymbol{q}]\end{bmatrix}, \\ \boldsymbol{\phi}_3^\delta : \begin{bmatrix}\boldsymbol{q}\\ \boldsymbol{p}\\ \boldsymbol{x}\\ \boldsymbol{y}\end{bmatrix} &= \frac{1}{2}\begin{bmatrix}\begin{pmatrix}\boldsymbol{q}+\boldsymbol{x}\\ \boldsymbol{p}+\boldsymbol{y}\end{pmatrix} + \boldsymbol{R}(\delta)\begin{pmatrix}\boldsymbol{q}-\boldsymbol{x}\\ \boldsymbol{p}-\boldsymbol{y}\end{pmatrix}\\ \begin{pmatrix}\boldsymbol{q}+\boldsymbol{x}\\ \boldsymbol{p}+\boldsymbol{y}\end{pmatrix} - \boldsymbol{R}(\delta)\begin{pmatrix}\boldsymbol{q}-\boldsymbol{x}\\ \boldsymbol{p}-\boldsymbol{y}\end{pmatrix}\end{bmatrix}, \end{aligned} \tag{5}$$

with

$$\boldsymbol{R}(\delta) := \begin{bmatrix}\cos(2\omega\delta)\boldsymbol{I} & \sin(2\omega\delta)\boldsymbol{I}\\ -\sin(2\omega\delta)\boldsymbol{I} & \cos(2\omega\delta)\boldsymbol{I}\end{bmatrix}, \quad \text{where } \boldsymbol{I} \text{ is a identity matrix.} \tag{6}$$

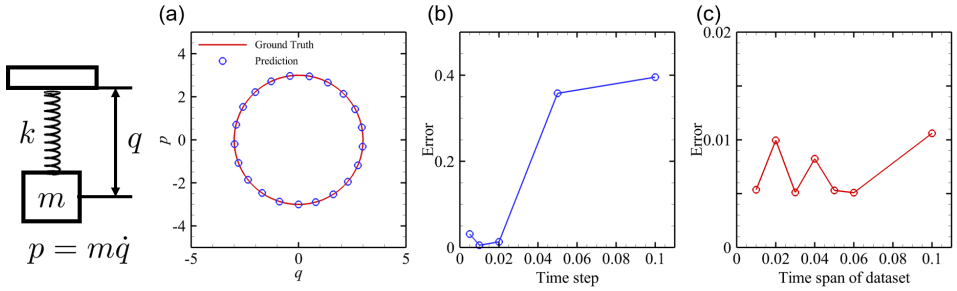

Figure 3: Ablation test of a simple ideal spring system. (a) A trace state with $(q, p) = (0, -3)$ in the phase space, (b) validation error with different integral time step, (c) validation error with different datasets time span.

Relationship (5) is obtained by replacing $\partial\mathcal{H}/\partial\boldsymbol{p}$ and $\partial\mathcal{H}/\partial\boldsymbol{q}$ in the symplectic integrator with deliberately designed neural networks $\partial\mathcal{H}_\theta/\partial\boldsymbol{p}$ and $\partial\mathcal{H}_\theta/\partial\boldsymbol{q}$, respectively. Figure 2 plots a schematic diagram of NSSNN which is described by algorithm 1. The input of symplectic neural networks is $(\boldsymbol{q}_0, \boldsymbol{p}_0)$, and the output is $(\boldsymbol{q}_n, \boldsymbol{p}_n, \boldsymbol{x}_n, \boldsymbol{y}_n)$. NSSNN consists of n iterations of second-order symplectic integrator. The input of the integrator is $(\boldsymbol{q}_{i-1}, \boldsymbol{p}_{i-1}, \boldsymbol{x}_{i-1}, \boldsymbol{y}_{i-1})$, and the output is $(\boldsymbol{q}_i, \boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{y}_i)$. We remark that $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$ are just auxiliary variables, which are theoretically equal to $\boldsymbol{q}$ and $\boldsymbol{p}$. Therefore, we can use the data set of $(\boldsymbol{q}, \boldsymbol{p})$ to construct the data set containing variables $(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{x}, \boldsymbol{y})$. By constructing the network $\mathcal{H}_\theta$, we show that theorem A.1 in Appendix A holds, so the networks $\boldsymbol{\phi}_1^\delta$, $\boldsymbol{\phi}_2^\delta$, and $\boldsymbol{\phi}_3^\delta$ in (5) preserve the symplectic structure of the system. Suppose that Φ_1 and Φ_2 are two symplectomorphisms. Then, it is easy to show that their composite map $\Phi_2 \circ \Phi_1$ is also symplectomorphism due to the chain rule. Thus, the symplectomorphism of algorithm 1 can be guaranteed by the theorems A.1.

Algorithm 1 Integrate (4) by using the second-order symplectic integrator

Input: $\boldsymbol{q}_0, \boldsymbol{p}_0, t_0, t, \mathrm{d}t$; $\boldsymbol{\phi}_1^\delta$, $\boldsymbol{\phi}_2^\delta$, and $\boldsymbol{\phi}_3^\delta$ in (5);
Output: $(\hat{q}, \hat{p}, \hat{x}, \hat{y}) = (\boldsymbol{q}_n, \boldsymbol{p}_n, \boldsymbol{x}_n, \boldsymbol{y}_n)$

$(\boldsymbol{q}_0, \boldsymbol{p}_0, \boldsymbol{x}_0, \boldsymbol{y}_0) = (\boldsymbol{q}_0, \boldsymbol{p}_0, \boldsymbol{q}_0, \boldsymbol{p}_0)$
$n = \mathrm{floor}[(t - t_0)/\mathrm{d}t]$
for $i = 1 \to n$ **do**
  $(\boldsymbol{q}_i, \boldsymbol{p}_i, \boldsymbol{x}_i, \boldsymbol{y}_i) = \boldsymbol{\phi}_1^{\mathrm{d}t/2} \circ \boldsymbol{\phi}_2^{\mathrm{d}t/2} \circ \boldsymbol{\phi}_3^{\mathrm{d}t} \circ \boldsymbol{\phi}_2^{\mathrm{d}t/2} \circ \boldsymbol{\phi}_1^{\mathrm{d}t/2} \circ (\boldsymbol{q}_{i-1}, \boldsymbol{p}_{i-1}, \boldsymbol{x}_{i-1}, \boldsymbol{y}_{i-1})$;
end

3.2.1 Data Training

We use 6 linear layers with hidden size 64 to model $\mathcal{H}_\theta$, all of which are followed by a Sigmoid activation function except the last one. The derivatives $\partial\mathcal{H}_\theta/\partial\boldsymbol{p}, \partial\mathcal{H}_\theta/\partial\boldsymbol{q}, \partial\mathcal{H}_\theta/\partial\boldsymbol{x}, \partial\mathcal{H}_\theta\boldsymbol{\partial y}$ are all obtained by automatic differentiation in Pytorch (Paszke et al., 2019). The weights of the linear layers are initialized by Xavier initializaiton (Glorot & Bengio, 2010). The loss function is:

$$\mathcal{L}_{NSSNN} = \sum_{j=1}^{N_s} \|q^{(j)} - \hat{q}^{(j)}\|_1 + \|p^{(j)} - \hat{p}^{(j)}\|_1 + \|\hat{x}^{(j)} - \hat{q}^{(j)}\|_1 + \|\hat{y}^{(j)} - \hat{p}^{(j)}\|_1 \tag{7}$$

where N_s is the number of training samples; $\hat{p}, \hat{q}, \hat{x}, \hat{y}$ is the output of Algorithm 1. We use the Adam optimizer (Kingma & Ba, 2015) with learning rate 0.05. The learning rate is multiplied by 0.8 for every 10 epoches. We use a batch of 512 examples.

4 Experiments

4.1 Ablation Test

We embed the Sympletic integrator to the network training, in which the integral time step is a vital parameter, taking ideal spring system as an example in figure 3, we compare the validation error generated by various integral time step $\mathrm{d}t$ based on fixed dataset time span $T_{train} = 0.1$. Generally, as shown in figure 3 (b), the smaller the time step is, the less the validation error. And we set the time step as $\mathrm{d}t = 0.01$ in our training. In addition, as shown in 3 (c), the algorithm used is not sensitive to

Table 1: Comparison between NeuralODE, HNN, NSSNN, TaylorNet, and SSINN. ✓ represents the method preserves such property.

Methods	NSSNN	HNN	NeuralODE	TaylorNet	SSINN
Solve nonseparable systems	✓	✓	✓		
Solve separable systems	✓	✓	✓	✓	✓
Preserve symplectic structure	✓	Partially		✓	Partially
Utilize continuous dynamics	✓		✓	✓	✓
No need for derivatives in the dataset	✓		✓	✓	✓
Strong stability	✓				
Long-term predictability	✓	Partially		✓	✓
Extend to N-body system	✓	✓		✓	

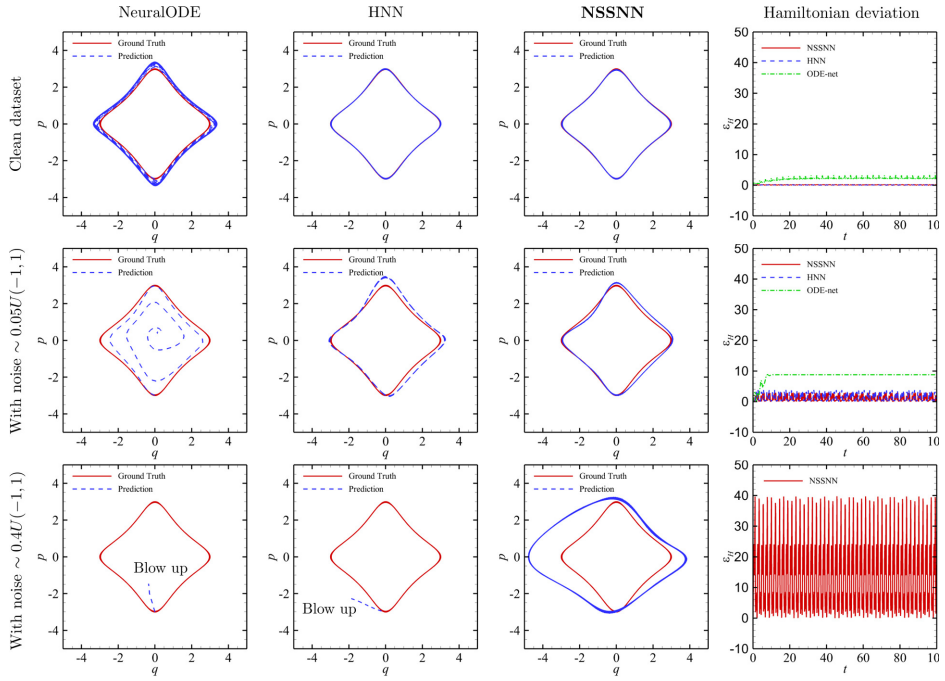

Figure 4: Comparison of prediction results of (q, p) for the problem $\mathcal{H} = 0.5(q^2 + 1)(p^2 + 1)$ from $t = 0$ to $t = 100$. The network is trained by the clean dataset (first row), the dataset with noise $\sim 0.05U(-1, 1)$ (second row), and the dataset with noise $\sim 0.4U(-1, 1)$ (third row). The first three columns are three different methods NeuralODE, HNN, and NSSNN, respectively. The fourth column is the Hamiltonian deviation $\epsilon_H = \|\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{truth}} - \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{predict}}\|_2 \,/\, \|\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{truth}}\|_2$.

the time span of the datasets, which contributes robust for wider datasets. We set the dataset time span as $T_{train} = 0.01$ in our training.

4.2 Comparison with other methods

We consider the pendulum, the Lotka–Volterra, the Spring, the Hénon–Heiles, the Tao's example (Tao, 2016), the Fourier form of nonlinear Schrödinger and the vortex particle systems in our implementation. The Hamiltonian energies of these systems (except vortex particle system) are summarized as follows:

Pendulum system: $\mathcal{H}(q, p) = 3(1 - \cos(q)) + p^2$. Lotka–Volterra system: $\mathcal{H}(q, p) = p - e^p + 2q - e^q$. Spring system: $\mathcal{H}(q, p) = q^2 + p^2$. Hénon–Heiles system: $\mathcal{H}(q_1, q_2, p_1, p_2) = (p_1^2 + p_2^2)/2 + (q_1^2 + q_2^2) + (q_1^2 q_2 - q_2^3/3)/2$. Tao's example (Tao, 2016): $\mathcal{H}(q, p) = (q^2 + 1)(p^2 + 1)/2$. Fourier form of nonlinear Schrödinger equation: $\mathcal{H}(q_1, q_2, p_1, p_2) = \left[(q_1^2 + p_1^2)^2 + (q_2^2 + p_2^2)^2\right]/4 - (q_1^2 q_2^2 + p_1^2 p_2^2 - q_1^2 p_2^2 - p_1^2 q_2^2 + 4 q_1 q_2 p_1 p_2)$.

Table 2: Comparison of Prediciton error and Hamiltonian deviation between NeuralODE, HNN and NSSNN

	Prediciton error			Hamiltonian deviation		
Problems	NeuralODE	HNN	NSSNN	NeuralODE	HNN	NSSNN
Pendulum	3.4×10^{-2}	3.1×10^{-2}	$\mathbf{2.6 \times 10^{-2}}$	1.5×10^{-2}	1.3×10^{-2}	$\mathbf{7.4 \times 10^{-3}}$
Lotka-Volterra	2.2×10^{-2}	3.9×10^{-2}	$\mathbf{2.7 \times 10^{-2}}$	7.4×10^{-3}	8.8×10^{-3}	$\mathbf{7.5 \times 10^{-3}}$
Spring	2.1×10^{-2}	2.1×10^{-2}	$\mathbf{1.6 \times 10^{-2}}$	9.3×10^{-3}	6.7×10^{-3}	$\mathbf{6.7 \times 10^{-3}}$
Hénon–Heiles	1.0×10^{-1}	9.4×10^{-2}	$\mathbf{8.4 \times 10^{-2}}$	3.7×10^{-2}	4.0×10^{-2}	$\mathbf{3.5 \times 10^{-2}}$
Tao's example	3.7×10^{-2}	2.6×10^{-2}	$\mathbf{2.2 \times 10^{-2}}$	1.4×10^{-2}	1.1×10^{-2}	$\mathbf{8.2 \times 10^{-3}}$
Schrödinger	8.7×10^{-2}	5.9×10^{-2}	$\mathbf{5.7 \times 10^{-2}}$	3.8×10^{-2}	2.3×10^{-2}	$\mathbf{2.0 \times 10^{-2}}$
Vortex (2 particles)	2.1×10^{-2}	7.7×10^{-3}	$\mathbf{3.4 \times 10^{-3}}$	1.5×10^{-2}	2.8×10^{-3}	$\mathbf{2.1 \times 10^{-3}}$
Vortex (4 particles)	3.4×10^{-2}	9.4×10^{-3}	$\mathbf{6.9 \times 10^{-3}}$	8.2×10^{-2}	1.4×10^{-2}	$\mathbf{3.4 \times 10^{-3}}$

We compare our method with other state-of-the-art methods, such as NeuralODE, HNN, TaylorNet, and SSIN. There are several features that distinguishes our method with others, shown in Table 1. First, our method is designed to solve both nonseparable and separable systems. While HNN and NeuralODE can be used on nonseparable systmes, they exhibit significant disadvantages. NSSNN preserves symplectic structure intrincally, while HNN enforces conservative features of the system by reformulating the loss function and NeuralODE does not preserve symplectic structure. Under the framework of NeuralODE, NSSNN utilizes continuously-defined dynamics in the construction of the neural network, which gives the capability to learn the continuous time evolution of dynamical systems. HNN requires the temporal derivatives of the momentum and the position of the system in the training dataset to calculate loss. These data are often not obtainable from real systems. NSSNN and NeuralODE need only the momentum and the position of the system. With the augmented Lagrangian multiplier, a regularizer of the integrator, HSSNN exhibits much stronger stability than the other two methods. From the results that we show in figure 4, it is clear that NSSNN can make long-term prediction accurately, which the other two methods fail to do. Additionally, TaylorNet and SSINN target towards seperable Hamiltonian systems. Although they preserve symplectic structure in some ways and utilize continuous dynamics, they are unable to be applied to nonseperable systems Therefore, we did not compare them in the experimental results.

In Figure 4, we show comparison of prediction results of (q, p) for the problem $\mathcal{H} = (q^2+1)(p^2+1)/2$ from $t = 0$ to $t = 100$. The network is trained by the clean dataset, the dataset with noise $\sim 0.05U(-1, 1)$, and the dataset with noise $\sim 0.4U(-1, 1)$. The first three columns are three different methods NeuralODE, HNN, and NSSNN, respectively. To quantititively compare the different between the three methods, we compare the Hamiltonian deviation, defined as $_H = \|\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{truth}} - \mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{predict}}\|_2 / \|\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})_{\text{truth}}\|_2$. We can observe that NeuralODE and HNN blow up quickly as the noise increases, while the results using NSSNN remain close to the ground truth. In Table 2, we compare the prediction error and the Hamiltonian deviation for the pendulum problem, the Lotka-Volterra problem, the Spring system, the Hènon–Heiles syetem, Tao's example, the Schrödinger equation, and the vortex system with 2 particles and 4 particles. The prediction error is defined as $\epsilon_p = \|\boldsymbol{q}_{\text{truth}} - \boldsymbol{q}_{\text{predict}}\|_1 + \|\boldsymbol{p}_{\text{truth}} - \boldsymbol{p}_{\text{predict}}\|_1$. It is clearly from the table that NSSNN either outperforms or has similar performance as NeuralODE and HNN.

4.3 Modeling vortex dynamics of multi-particle system

For two dimensional vortex particle system, the dynamical equations of particles positions $(x_j, y_j),\ j = 1, 2, \cdots, N_v$ with particle strengths $\Gamma_j,\ j = 1, 2, \cdots, N_v$ can be written in the generalized Hamiltonian form as

$$\begin{cases} \Gamma_j \dfrac{\mathrm{d}x_j}{\mathrm{d}t} = -\dfrac{\partial \mathcal{H}_v}{\partial y_j}, \\ \Gamma_j \dfrac{\mathrm{d}y_j}{\mathrm{d}t} = \dfrac{\partial \mathcal{H}_v}{\partial x_j}, \end{cases} \tag{8}$$

with

$$H_p = \frac{1}{4\pi} \sum_{j,k} \Gamma_j \Gamma_k \log(|X_j - X_k|). \tag{9}$$

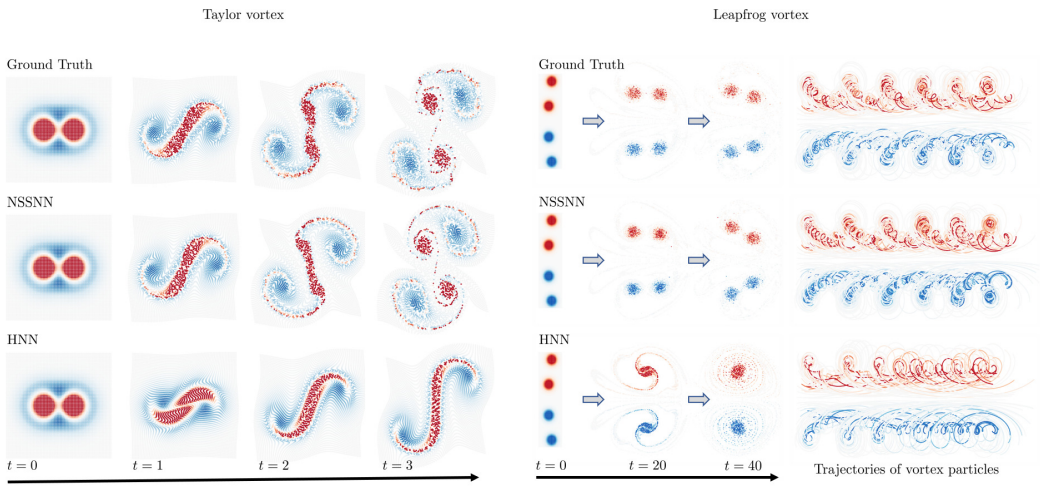

Figure 5: Taylor and Leapfrog vortex. We generate results of Taylor vortex and Leapfrop vortex using NSSNN and HNN, and compare them with the ground truth. 6000 vortex elements are used with corresponding initial vorticity conditions of Taylor vortex and Leapfrop vortex. In both cases, the vortices evolved using NSSNN are seperated nicely as the ground truth shows, while the vortices merge together using HNN.

.

We can still adopt the method mentioned in last section to learn the Hamiltonian when there are less particles. However, consider a system with $N \gg 2$ particles. The cost to collect training data from all N particles may be high, and the training process may be time inefficient. Thus, instead of collecting information from all N particles to train our model, we only use data collected from two bodies as training data to make prediction of the dynamics of N particles.

The setup of multi-particle problem is similar to the previous problems. The training time span is $T_{train} = 0.01$ while the prediction period can be up to $T_{predict} = 40$. We use 2048 clean data samples to train our model. The training process takes about 100 epochs for the loss to converge. In figure 5, we use our trained model to predict the dynamics of a 6000-particle systems, including Taylor and Leapfrog vortices. We generate results of Taylor vortex and Leapfrop vortex using NSSNN and HNN, and compare them with the ground truth. Vortex elements are used with corresponding initial vorticity conditions of Taylor vortex and Leapfrop vortex (see Qu et al., 2019). The difficulty of the numerical modeling of these two systems lies in the separation of different dynamical vortices instead of having them merging into a bigger structure. In both cases, the vortices evolved using NSSNN are seperated nicely as the ground truth shows, while the vortices merge together using HNN.

5 Conclusions

We incorporate a classic ideal that maps a nonseparable system to a higher dimensional space making it quasi-separable into nonseparable Hamiltonian system to construct sympletic networks. With the intrinsic symplectic structure, NSSNN possesses many benefits compared with other methods. In particular, NSSNN is the first method that can learn the vortex dynamical system, and accurately predict the evolution of complex vortex structures, such as Taylor and Leapfrog vortices. NSSNN, based on the first principle learning complex systems, has potential applications in fields of physics, astronomy, and weather forecast, etc. We will further explore the possibilities of neural networks with inherent structure preserving ability in fields like 3D vortex dynamics and quantum turbulence. We will also work on general applications of auxiliary variables in the field of machine learning.

REFERENCES

V. Antohe and I. Gladwell. Performance of variable step size methods for solving model separable hamiltonian systems. *Math. COMPUT. MODEL.*, 40(11-12):1245–1262, 2004.

J. Baiges, R. Codina, I. Castañar, and E. Castillo. A finite element reduced-order model based on adaptive mesh refinement and artificial neural networks. *International Journal for Numerical Methods in Engineering*, 121(4):588–601, 2020.

S. Bonnabel, M. Mirrahimi, and P. Rouchon. Observer-based hamiltonian identification for quantum systems. *Automatica*, 45(5):1144–1155, 2009.

S. Boyd, S. P Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

L. Brugnano, C. Zhang, and D. Li. A class of energy-conserving hamiltonian boundary value methods for nonlinear schrödinger equation with wave operator. *Communications in Nonlinear Science and Numerical Simulation*, 60:33–49, 2018.

S. L. Brunton, B. R. Noack, and P. Koumoutsakos. Machine Learning for Fluid Mechanics. *Annu. Rev. Fluid Mech.*, 52:477–508, 2020.

K. Chadaj, P. Malczyk, and J. Frączek. A parallel hamiltonian formulation for forward dynamics of closed-loop multibody systems. *Multibody Syst. Dyn.*, 39(1-2):51–77, 2017.

R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *Conference on Neural Information Processing Systems*, pp. 6571–6583, 2018.

Z. Chen, J. Zhang, M. Arjovsky, and L. Bottou. Symplectic recurrent neural networks. In *International Conference on Learning Representations*, 2020.

M. Cranmer, S. Greydanus, S. Hoyer, P. Battaglia, D. Spergel, and S. Ho. Lagrangian neural networks. *arXiv:2003.04630*, 2020.

D. M. DiPietro, S. Xiong, and B. Zhu. Sparse symplectically integrated neural networks. *arXiv:2006.12972*, 2020.

Z. Geng, D. Johnson, and R. Fedkiw. Coercing machine learning to output physically accurate results. *J. Comput. Phys.*, 406:109099, 2020.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

S. Greydanus, M. Dzamba, and J. Yosinski. Hamiltonian neural networks. In *Conference on Neural Information Processing Systems*, pp. 15379–15389, 2019.

L. N. Hand and J. D. Finch. *Analytical Mechanics*. Cambridge University Press, 2008.

Q. Hernandez, A. Badias, D. Gonzalez, F. Chinesta, and E. Cueto. Structure-preserving neural networks. *arXiv:2004.04653*, 2020.

T. W. Hughes, I. A. D. Williamson, M. Minkov, and S. Fan. Wave physics as an analog recurrent neural network. *Sci. Adv.*, 5:6946, 2019.

P. Jin, A. Zhu, G. E. Karniadakis, and Y. Tang. Symplectic networks: intrinsic structure-preserving networks for identifying Hamiltonian systems. *arXiv:2001.03750*, 2020.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

A. N. Kolmogorov. On the Conservation of Conditionally Periodic Motions under Small Perturbation of the Hamiltonian. *Dokl. akad. nauk SSSR*, 98:527–530, 1954.

Y. Li, Y. He, J. Niesen Y. Sun, H. Qin, and J. Liu. Solving the vlasov–maxwell equations using hamiltonian splitting. *J. Comput. Phys.*, 396:381–399, 2019.

J. Ling, R. Jones, and J. Templeton. Machine learning strategies for systems with invariance properties. *J. Comput. Phys.*, 318:22–35, 2016.

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

A. T. Mohan, N. Lubbers, D. Livescu, and M. Chertkov. Embedding hard physical constraints in convolutional neural networks for 3d turbulence. In *International Conference on Learning Representations*, 2020.

P. J. Morrison. Hamiltonian and action principle formulations of plasma physics. *Phys. Plasmas*, 12: 058102, 2005.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pp. 8026–8037, 2019.

Z. Qu, X. Zhang, M. Gao, C. Jiang, and B. Chen. Efficient and conservative fluids using bidirectional mapping. *ACM Trans. Graph.*, 38:1–12, 2019.

D. G. Saari and Z. Xia. *Hamiltonian Dynamics and Celestial Mechanics*. American Mathematical Society, 1 edition, 1996.

R. Salmon. Hamiltonian fluid mechanics. *Ann. Rev. Fluid Mech.*, 20:225–256, 1988.

J. M. Sellier, G. M. Caron, and J. Leygonie. Signed particles and neural networks, towards efficient simulations of quantum systems. *J. Comput. Phys.*, 387:154–162, 2019.

Y. Suzuki, S. Koshizuka, and Y. Oka. Hamiltonian moving-particle semi-implicit (hmps) method for incompressible fluid flows. *Computer Methods in Applied Mechanics and Engineering*, 196 (29-30):2876–2894, 2007.

M. Tao. Explicit symplectic approximation of nonseparable Hamiltonians: algorithm and long time performance. *Phys. Rev. E*, 94:043303, 2016.

G. H. Teicherta, A. R. Natarajanc, A. Van der Venc, and K. Garikipati. Machine learning materials physics: Integrable deep neural networks enable scale bridging by learning free energy functions. *Comput. Methods Appl. Mech. Engrg.*, 353:201–216, 2019.

Y. Tong, S. Xiong, X. He, G. Pan, and Bo Zhu. Symplectic neural networks in taylor series form for hamiltonian systems. *arXiv:2005.04986*, 2020.

P. Toth, D. J. Rezende, A. Jaegle, S. Racaniére, A. Botev, and I. Higgins. Hamiltonian generative networks. In *International Conference on Learning Representations*, 2020.

R. Zhang, H. Qin, Y. Tang, J. Liu, Y. He, and J. Xiao. Explicit symplectic algorithms based on generating functions for charged particle dynamics. *Physical Review E*, 94(1):013205, 2016.

Y. D. Zhong, B. Dey, and A. Chakraborty. Symplectic ode-net: learning hamiltonian dynamics with control. In *International Conference on Learning Representations*, 2020.

B. Zhu, R. Zhang, Y. Tang, X. Tu, and Y. Zhao. Splitting k-symplectic methods for non-canonical separable hamiltonian problems. *J. Comput. Phys.*, 322:387–399, 2016.

E. E Zotos. Classifying orbits in the classical hénon–heiles hamiltonian system. *Nonlinear Dynamics*, 79(3):1665–1677, 2015.

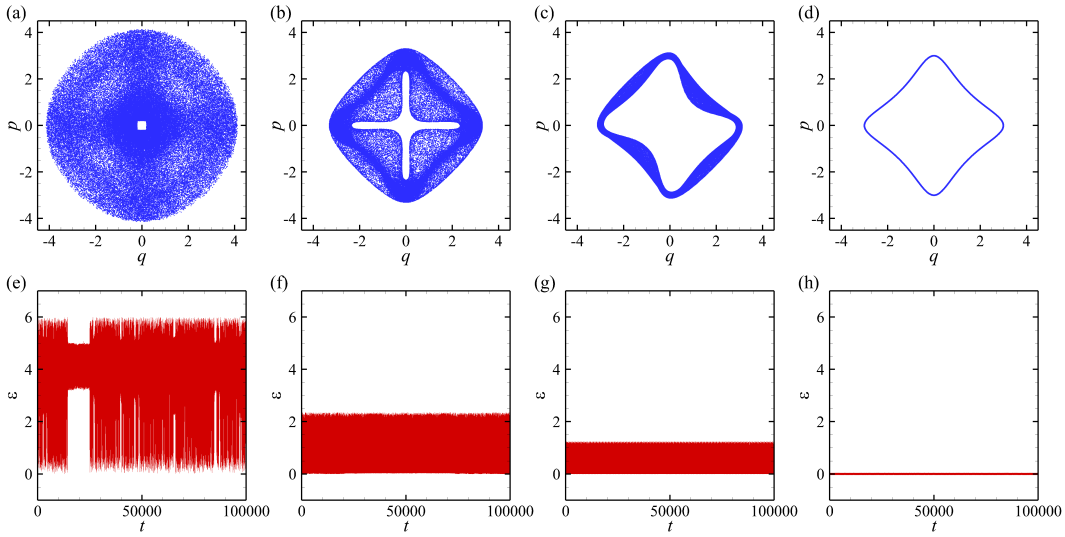

Figure 6: Comparison of different ω (from left to right columns) with (a) and (e) $\omega = 0$, (b) and (f) $\omega = 0.8$, (c) and (g) $\omega = 0.9$, (d) and (h) $\omega = 10$ in the symplectic integrator for nonseparable Hamiltonian $H = (q^2 + 1)(p^2 + 1)/2$. The upper row: projection of a trace $[q(t), p(t), x(t), y(t)]$ with $[q(0), p(0), x(0), y(0)] = (-3, 0, -3, 0)$ onto the $q - p$ plane. The bottom row: deviation $\epsilon = \|(q, p) - (x, y)\|_2$ between (q, p) and (x, y).

.

A SYMPLECTOMORPHISMS

One of the important features of the time evolution of Hamilton's equations is that it is a symplectomorphism, representing a transformation of phase space that is volume-preserving. In the setting of canonical coordinates, symplectomorphism means the transformation of the phase flow of a Hamiltonian system conserves the symplectic two-form

$$\mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} \equiv \sum_{j=1}^{N} \left(\mathrm{d}q_j \wedge \mathrm{d}p_j\right), \tag{10}$$

where $\wedge$ denotes the wedge product of two differential forms. As proved below, our constructed network structure intrinsically preserves Hamiltonian structure.

Theorem A.1. *For a given δ, the mapping $\boldsymbol{\phi}_1^\delta$, $\boldsymbol{\phi}_2^\delta$, and $\boldsymbol{\phi}_3^\delta$ in (5) are symplectomorphisms.*

Proof. Let

$$(\boldsymbol{t}_j^q, \boldsymbol{t}_j^p, \boldsymbol{t}_j^x, \boldsymbol{t}_j^y) = \boldsymbol{\phi}_j^\delta(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{x}, \boldsymbol{y}), \quad j = 1, 2, 3. \tag{11}$$

From the first equation of (5), we have

$$\begin{aligned}
&\mathrm{d}\boldsymbol{t}_1^q \wedge \mathrm{d}\boldsymbol{t}_1^p + \mathrm{d}\boldsymbol{t}_1^x \wedge \mathrm{d}\boldsymbol{t}_1^y \\
=&\mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\left[\boldsymbol{p} - \delta \frac{\partial \mathcal{H}_\theta(\boldsymbol{q}, \boldsymbol{y})}{\partial \boldsymbol{q}}\right] + \mathrm{d}\left[\boldsymbol{x} + \delta \frac{\partial \mathcal{H}_\theta(\boldsymbol{q}, \boldsymbol{y})}{\partial \boldsymbol{p}}\right] \wedge \mathrm{d}\boldsymbol{y} \\
=&\mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} + \mathrm{d}\boldsymbol{x} \wedge \mathrm{d}\boldsymbol{y} + \delta \left[\frac{\partial \mathcal{H}_\theta(\boldsymbol{q}, \boldsymbol{y})}{\partial \boldsymbol{q} \partial \boldsymbol{y}} - \frac{\partial \mathcal{H}_\theta(\boldsymbol{q}, \boldsymbol{y})}{\partial \boldsymbol{y} \partial \boldsymbol{q}}\right] \mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{y} \\
=&\mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} + \mathrm{d}\boldsymbol{x} \wedge \mathrm{d}\boldsymbol{y}.
\end{aligned} \tag{12}$$

Similarly, we can prove that $\mathrm{d}\boldsymbol{t}_2^q \wedge \mathrm{d}\boldsymbol{t}_2^p + \mathrm{d}\boldsymbol{t}_2^x \wedge \mathrm{d}\boldsymbol{t}_2^y = \mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} + \mathrm{d}\boldsymbol{x} \wedge \mathrm{d}\boldsymbol{y}$. In addition, from the third equation of (5), we can directly deduce that $\mathrm{d}\boldsymbol{t}_3^q \wedge \mathrm{d}\boldsymbol{t}_3^p + \mathrm{d}\boldsymbol{t}_3^x \wedge \mathrm{d}\boldsymbol{t}_3^y = \mathrm{d}\boldsymbol{q} \wedge \mathrm{d}\boldsymbol{p} + \mathrm{d}\boldsymbol{x} \wedge \mathrm{d}\boldsymbol{y}$. □

Suppose that Φ_1 and Φ_2 are two symplectomorphisms. Then, it is easy to show that their composite map $\Phi_2 \circ \Phi_1$ is also symplectomorphism due to the chain rule. Thus, the symplectomorphism of algorithm 1 can be guaranteed by the theorem A.1.

B DETERMINING COEFFICIENT ω

To further elucidation, the Hamiltonian $\mathcal{H}_A + \mathcal{H}_B$ without the binding, i.e., $\overline{\mathcal{H}}$ with $\omega = 0$, in extended phase space $(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{x}, \boldsymbol{y})$ may not be integrable, even if $\mathcal{H}(\boldsymbol{q}, \boldsymbol{p})$ is integrable in the original phase space $(\boldsymbol{q}, \boldsymbol{p})$. However, $\mathcal{H}_C$ is integrable. Thus, as ω increases, a larger proportion in the phase space for $\overline{\mathcal{H}}$ corresponds to regular behaviors (Kolmogorov, 1954). For $H(q, p) = (q^2 + 1)(p^2 + 1)/2$, shown in Fig. 6, we compare the trajectories starting from $[q(0), p(0), x(0), y(0)] = (-3, 0, -3, 0)$ calculated by the symplectic integrator (Tao, 2016) with different ω, where the calculation accuracy is second order accuracy and the time interval is 0.001. As Figs. 6(a), (b), (c), and (d) shown, the chaotic region in phase space is significantly decreasing until forming a stable limit cycle. We define $\epsilon = \|(q, p) - (x, y)\|_2$ as the calculation error of this system, shown in Figs. (e), (f), (g), and (h) that the error is decreasing with ω increasing, which fits the quantitative results of phase trajectory well.