Sigma Journal of Engineering and Natural Sciences
Web page info: https://sigma.yildiz.edu.tr
DOI: 10.14744/sigma.2026.2021

Research Article

Solving Hamiltonian systems based on a data-driven deep learning algorithm

Tayfun ÜNAL[1,2,*], Ayten İrem IŞIK[2], Ünver ÇİFTÇİ[2]

[1]*Department of Information Technologies, Kırklareli University, Kırklareli, 39100, Türkiye*
[2]*Department of Mathematics, Tekirdağ Namık Kemal University, Tekirdağ, 59030, Türkiye*

ARTICLE INFO

Article history
Received: 12 June 2024
Revised: 18 September 2024
Accepted: 02 February 2025

Keywords:
Hamiltonian Systems, Data-driven, Deep Learning, Energy Conserving

ABSTRACT

Hamiltonian systems possess important properties, such as the preservation of the symplectic structure and the conservation of energy. Traditional numerical iteration methods generally violate these properties. This paper proposes a data-driven deep learning algorithm to solve separable Hamiltonian systems. It is computationally efficient and scalable. The proposed algorithm tries to hold the balance between symplectic and energy-conserving. Additionally, it can work efficiently with few data points, and the obtained solution is continuous. The algorithm consists of three main steps: (i) obtaining data points from a solver, (ii) data augmentation, and (iii) ensuring energy conservation. The algorithm was evaluated on the simple harmonic oscillator, nonlinear oscillator, and Lotka-Volterra systems. The proposed algorithm has demonstrated successful performance across all these systems despite the few data. In conclusion, it has been experimentally demonstrated that the proposed method is effective even with few data points. Furthermore, the proposed algorithm can effectively work regardless of the solver chosen in all examples.

Cite this article as: Ünal T, Işık Aİ, Çiftçi Ü. Solving Hamiltonian systems based on a data-driven deep learning algorithm. Sigma J Eng Nat Sci 2026;44(2):994–1003.

INTRODUCTION

Solving differential equations with deep learning algorithms dates back approximately three decades [1, 2]. To date, two main approaches have been widely used in research for this purpose. One approach involves solving differential equations using equation-driven deep learning algorithms [1, 3, 4]. Another one is using data-driven deep learning algorithms [5, 6]. Recently, the utilization of both equation-driven and data-driven deep learning algorithms has gained popularity in the numerical study of dynamical systems [7]. One of the most significant motivations for this approach is that it can incorporate the laws of physics [8], as the time-dependent dynamics of a system are governed by physical laws such as energy and momentum. An example of this is Hamiltonian systems, which are dynamical systems that conserve energy [9]. To solve Hamiltonian systems, a significant amount of research has been conducted utilizing both approaches. However, they have employed automatic differentiation, which is more computationally expensive,

***Corresponding author.**
*E-mail address: tayfununal39@gmail.com

This paper was recommended for publication in revised form by Editor-in-Chief Ahmet Selim Dalkilic

or utilized complex deep learning models. In this context, this paper proposes a data-driven deep learning algorithm to solve separable Hamiltonian systems without automatic differentiation and complex deep learning architectures.

In recent years, an equation-driven approach for Hamiltonian systems has been introduced by Mattheakis et al. [4]. They proposed a Hamiltonian neural network architecture designed to approximate Hamiltonian trajectories. Motivated by [4], this paper utilizes their model but in a different manner. The first difference is that a data-driven model is employed here. The second point is that they employed automatic differentiation to obtain the derivatives of the model with respect to the time variable. However, the derivatives of the neural network were not computed here, as the data was obtained from an ordinary differential equation solver.

For Hamiltonian systems, there has been increased research that employs the data-driven approach. In particular, Greydanus et al. [10] presented Hamiltonian Neural Networks (HNN). They designed a neural network that represents the Hamiltonian function. Automatic differentiation was used in the training of HNN. The notable aspect of the study, and another difference from this paper, is the learning of the Hamiltonian function using pixel data. Subsequently, HNNs were developed to solve chaotic systems by Han et al. [11] and named adaptable HNN. Meanwhile, HNNs were utilized to learn the trajectories of Hamiltonian systems in [12]. Furthermore, there are various works concerning the representation of the Hamiltonian function using neural networks [13–15]. These works employed automatic differentiation, which, as previously mentioned, is computationally expensive. Moreover, they utilized advanced architectures, such as autoencoders [13] and symmetry-preserving extensions of HNNs [14], among others.

Physics-inspired neural networks have been introduced for solving differential equations [8, 16–22], where they incorporate the laws of physics. Karniadakis et al. [8] proposed the physics-informed neural network (PINN). They constructed a loss function to satisfy differential equations and added a new term to the loss function to ensure the conservation of physical laws. The derivatives of the neural network were obtained using automatic differentiation once again. Recent advancements in PINNs encompass improvements in training algorithms, architectural designs, and computational efficiency. A neural network-based architecture called SciNet was presented by Iten et al. [16] with the goal of simulating human physical thinking in order to extract basic physical principles from empirical data without the need for prior knowledge. SciNet demonstrates how these neural networks can independently discover physical notions. Yu et al. [17] developed Gradient Enhanced Physics Knowledge Neural Networks (gPINNs) to improve training performance and accuracy, using residual gradients of PDEs. To further improve performance, they also combined gPINNs with a residual-based adaptive refinement method. First-Order PINNs (FO-PINNs) were developed by Glasdstone et al. [18]. It reformulates higher-order PDEs into a series of first-order equations. Thus, FO-PINNs reduce training time per iteration by avoiding the computation of higher-order derivatives using automatic differentiation. Moreover, it enhanced accuracy compared to standard PINNs. Wang et al. [19] developed the NAS-PINN, which employs Neural Architecture Search (NAS) to automatically identify optimal neural network architectures for solving PDEs. Using PINNs, Norambuena et al. [20] suggested a computational approach to optimal quantum control problems. They show that PINNs are more flexible and energy efficient than traditional control methods when it comes to solving state-to-state transmission in open quantum systems. For conventional numerical integrators in Hamiltonian systems, maintaining phase space structures and energy conservation over extended time scales is a major difficulty. In this regard, SympFlow, a neural network-based symplectic integrator that incorporates time-dependent Hamiltonian functions, was developed by Canizares et al. [21]. By leveraging physics-informed machine learning techniques, SympFlow effectively addresses these challenges. Another work, called the physics-enhanced neural network, was presented by Choudhary et al. [22]. They applied HNN to chaotic systems.

Moreover, there is a research perspective focused on defining Hamiltonian systems using neural networks that preserve the symplectic structure [23–25]. Tong et al. [23] presented a fourth-order integrator using a neural network architecture called TaylorNet. Additionally, symplectic networks (SympNets) were introduced by [24]. Hamiltonian systems are learned using SympNet in the paper. Subsequently, SympNets were improved by [25], and the architecture was designed to locally preserve the symplectic structure.

Another aspect of the study involves learning dynamical systems using Poisson neural networks [26], as well as employing deep learning to represent the Koopman operator [27] and using thermodynamics-aware time integrators for learning dynamical systems [28].

In this paper, a data-driven deep learning algorithm is proposed to solve separable Hamiltonian systems. The algorithm is applied to the simple harmonic oscillator, nonlinear oscillator, and Lotka-Volterra as examples. The results were then evaluated and compared with the scipy.integrate.odeint method in Python. The algorithm outperforms its counterparts, even with few data. The requirements for the proposed algorithm are a separable Hamiltonian system and a traditional solver, such as Euler, symplectic Euler, Verlet, or Runge-Kutta. Numerical symplectic integrators cannot conserve energy. Conversely, numerical methods that conserve energy cannot preserve the symplectic structure [29]. Thus, the algorithm is designed to ensure conservation in both cases. To achieve this, a symplectic integrator, such as symplectic Euler or Verlet, is employed to obtain data. The contributions of this work are outlined as follows:

- The proposed algorithm solves Hamiltonian equations with few data.
- It provides a continuous solution and scales to high dimensions.
- It balances the preservation of symplectic structure and energy conservation.

The structure of this paper is organized as follows: In Section 2, Hamiltonian systems will be introduced. Subsequently, the algorithm will be described in Section 3. Section 4 includes the examples. In Section 5, the results will be presented. Finally, the results are evaluated, and the discussion will focus on future work.

Hamiltonian Systems

A Hamiltonian system is a dynamical system characterised by a function $\mathcal{H}$: $\mathbb{R}^{2d} \to \mathbb{R}$ that satisfies the following equations:

$$\frac{d\boldsymbol{q}}{dt} = \frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{p}}, \quad \frac{d\boldsymbol{p}}{dt} = -\frac{\partial \mathcal{H}(\boldsymbol{q},\boldsymbol{p})}{\partial \boldsymbol{q}}, \tag{1}$$

where $q = (q_1, q_2, \ldots, q_d) \in \mathbb{R}^d$ and $p = (p_1, p_2, \ldots, p_d) \in \mathbb{R}^d$ are usually called as position and momentum vectors in the phase space, respectively. Additionally, t represents the time variable here.

In this paper, separable Hamiltonian systems of the form

$$\mathcal{H}(q,p) \quad = \mathcal{T}(p) + \mathcal{V}(q) \tag{2}$$

are considered, where $\mathcal{T}(p)$ represents the kinetic energy function and $\mathcal{V}(q)$ denotes the potential energy function. One of the most significant properties of Hamiltonian systems is the conservation of energy. This property arises from the fact that the derivative of $\mathcal{H}$ with respect to t is zero, as demonstrated by Equation .

$$\frac{\partial \mathcal{H}}{\partial t} = \frac{\partial \mathcal{H}}{\partial q}\frac{dq}{dt} + \frac{\partial \mathcal{H}}{\partial p}\frac{dp}{dt} = \frac{\partial \mathcal{H}}{\partial q}\left(\frac{\partial \mathcal{H}}{\partial p}\right) + \frac{\partial \mathcal{H}}{\partial p}\left(-\frac{\partial \mathcal{H}}{\partial q}\right) = 0 \tag{3}$$

Therefore, it can be concluded that $\mathcal{H}$ remains constant for all t. From this perspective, the deep learning algorithm proposed in this study ensures that the solution of the system adheres to the constraint of energy conservation.

MATERIALS AND METHODS

The proposed method consists of the following three steps:

1. Solve the system with few data points t values (denoted by t_s) within the specified time interval using a symplectic solver, such as symplectic Euler or Verlet.
2. Develop a deep learning model and train it on the data points obtained from the solver, utilizing the Mean Squared Error (MSE) loss function. After the training phase, augment the dataset by generating additional t points (denoted by t_m) using the trained model. Specifically, the model predicts the values of q and p corresponding to the additional t_m data points within the specified time interval.
3. Finally, the augmented dataset is trained using an energy-conserving loss function for the same deep learning model. Upon completion of this training, the model is capable of providing a continuous approximate solution for the specified time interval t.

Figure 1 provides an overview of the solution steps.

In this study, both the symplectic Euler and Verlet schemes are employed as solvers for all examples. For the Hamiltonian system, the symplectic Euler and Verlet schemes solve system using the algorithms provided in equations and , respectively:

$$\begin{aligned} q_{i+1} &= q_i + h\nabla_p \mathcal{H}(q_{i+1}, p_i) \\ p_{i+1} &= p_i - h\nabla_q \mathcal{H}(q_{i+1}, p_i) \end{aligned} \tag{4}$$

$$\begin{aligned} p_{i+1/2} &= p_i - \tfrac{h}{2}\nabla_q \mathcal{H}(q_i, p_{i+1/2}), \\ q_{i+1} &= q_i + \tfrac{h}{2}(\nabla_p \mathcal{H}(q_i, p_{i+1/2}) + \nabla_p \mathcal{H}(q_{i+1}, p_{i+1/2})), \\ p_{i+1} &= p_{i+1/2} - \tfrac{h}{2}\nabla_q \mathcal{H}(q_{i+1}, p_{i+1/2}). \end{aligned} \tag{5}$$

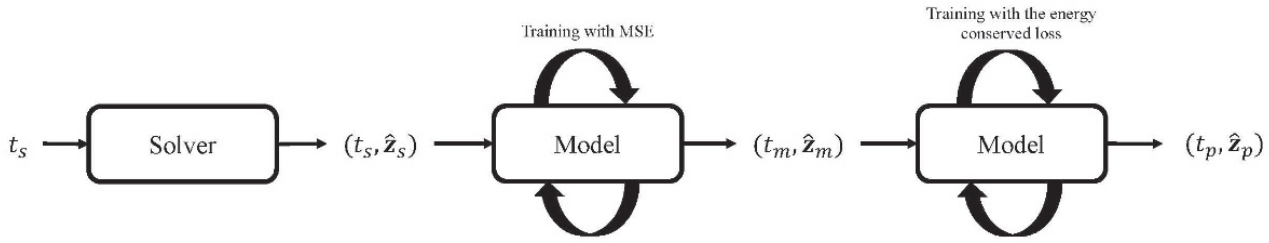

Figure 1. The proposed method consists of three steps. In the first step, the system is solved with a solver for a small number of points, t_s. In the second step, training is performed using the data points obtained, with a model that employs an MSE loss function, and additional data points are predicted for t_m using the model. In the final step, training is conducted with a model that employs an energy-conserving loss function, and predictions are made for t_p. As a result, the final model gains the capability to make predictions for any t.

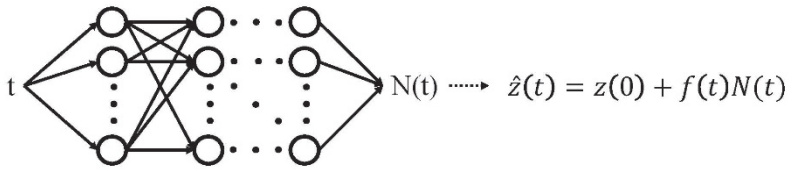

Figure 2. In both models, the structure is identical to that of a fully connected deep learning model, except for the final layer. In each model, the final layer is designed as $\hat{z}(t) = z(0) + f(t)N(t)$, where $N(t)$ represents the output of the last layer of the fully connected model, ensuring the initial conditions of the system are satisfied. The input is the time t, and the output consists of $\hat{z}(t) = (q, p)$ values.

In these equations (4) and (5), h represents the step size and q_i, p_i are the solutions at the i-th time point. Moreover, it is explicit that $\nabla_q \mathcal{H}(q, p) = \nabla_q \mathcal{V}(q)$ and $\nabla_p \mathcal{H}(q, p) = \nabla_p \mathcal{T}(p)$ for seperable Hamiltonian systems.

As mentioned above, given an initial point defined as $z_0 = z(t_0 = 0) = (q_0, p_0)$ and a time interval t, the system described in Equation is initially solved using either the symplectic Euler or Verlet scheme. As a result, a dataset is obtained that comprises $\{(t_0, \hat{z}_0), (t_1, \hat{z}_1), \ldots, (t_s, \hat{z}_s)\}$. Secondly, this dataset is used in the deep learning algorithm that has the architecture in Figure 1.

In this model, the architecture is similar to multilayer perceptron (MLP) except the last layer (6) where the output is:

$$\hat{\boldsymbol{z}}(t) = \boldsymbol{z}(0) + f(t)\boldsymbol{N}(t). \tag{6}$$

Here, $N(t)$ represents the output of model before the last layer. The last layer have been just designed for satisfying the given initial point [4]. Accordingly, f is defined as:

$$f(t) = 1 - e^{-t^2} \tag{7}$$

Equation so that $\hat{z}(0) = z(0)$ when $t = 0$. After training the model with the MSE loss function defined in Equation, predictions of $\hat{z}$ can be made for various t points. The model has just fitted to the dataset. In contrast to the Hamiltonian, the model has been employed to approximate a different trajectory so far. However, the model provides flexibility for augmenting data points. For $s < m$, m sample is taken by uniform distribution from given interval t. Then for each t_i, $i = 1, 2, \ldots, m$, model makes a prediction. Thus, an augmented dataset is obtained, denoted as $\{(t_1, \hat{z}_1), (t_2, \hat{z}_2), \ldots, (t_m, \hat{z}_m)\}$, which will be utilized by the model in the second training phase with the energy-conserved loss function defined in Equation. In the final step, the energy-conserved loss function was designed to enhance the model's performance.

As a result, this method guarantees excellent performance even when using sparse data points obtained from a symplectic solver. The proposed method outperforms competing strategies by avoiding neural network derivative calculations. It reduces computational load and increases numerical stability. The approach is also scalable. By simply changing the parameters, it can be applied to a wide variety of complex physical problems.

MSE Loss Functions

In the first training, the widely recognized loss function known as the Mean Squared Error (MSE) was utilized, defined as follows:

$$\mathcal{L}_{MSE}(\boldsymbol{z}, \hat{\boldsymbol{z}}(t)) = \frac{1}{k}\sum_{i=1}^{k}(\boldsymbol{z}_i - \hat{\boldsymbol{z}}(t_i))^2. \tag{8}$$

The significance of this training lies in its capacity to learn from the dataset, thereby facilitating the augmentation of data points after the training process.

Energy-Conserving Loss Function

The second loss function, referred to as the energy-conserving loss function, is defined as follows:

$$\begin{aligned}\mathcal{L}_{Energy}(\boldsymbol{z}, \hat{\boldsymbol{z}}(t)) = \gamma \frac{1}{k}\sum_{i=1}^{k}(\boldsymbol{z}_i - \hat{\boldsymbol{z}}(t_i))^2 \\ + (1-\gamma)\frac{1}{m}\sum_{i=1}^{m}(\mathcal{H}(\boldsymbol{z}_0) \\ - \mathcal{H}(\hat{\boldsymbol{z}}(t_m)))^2\end{aligned} \tag{9}$$

This loss function consists of two components: the first part ensures that the model preserves the basic structure of the data, while the second part ensures the conservation of the Hamiltonian. Here, γ is a crucial hyperparameter because it controls the balance between fitting the data and ensuring the conservation of the Hamiltonian. Its value directly affects the degree to which the Hamiltonian is preserved throughout the optimization process. This implies that, if γ is chosen sufficiently small, the model places more emphasis on energy conservation, thus prioritizing the conservation of the Hamiltonian over the precise fitting of the data.

EXAMPLES

Deep learning algorithms are highly effective tools for learning invariant patterns, capturing complex patterns and relationships within a dataset. Even when trained on relatively small datasets, they have the ability to effectively generalize by identifying underlying patterns and extending beyond the given data. In this study, the proposed model is applied to Hamiltonian dynamical systems, especially the

simple harmonic oscillator, the nonlinear oscillator, and the Lotka-Volterra system. The performance of the proposed model is compared to that of the 'scipy.integrate.odeint' solver, which employed 1500 time points to solve the non-linear oscillator and Lotka-Volterra systems. However, for the simple harmonic oscillator, the analytical solution is utilized owing to its accessibility. The hyperparameters were selected based on insights gained from experimental observations. The common parameters and hyperparameters of the neural network are presented in Table 1.

The sine function is utilized as the activation function due to the enhancement of the model's predictive capability by trigonometric functions [4]. The experiments were conducted on virtual machines provided by Google Colab, equipped with 2 vCPUs and 13GB of RAM. Additionally, the Python programming language and the TensorFlow library were used in all examples[1].

Table 1. The common parameters and hyper-parameters

	Values		Values
Input dim	1	Activation function	sinus
Output dim	2	Augmented t (tm)	10000
Hidden layer	20	Learning rate	0.0001
Neurons	64	γ (Gamma)	0.0001
Epoch	5000		

Simple Harmonic Oscillator

The kinetic and potential energies are defined in Equation , respectively:

$$\mathcal{T}(p) = \frac{p^2}{2m}, \quad \mathcal{V}(q) = \frac{kq^2}{2}. \tag{10}$$

For simplicity, let $k = m = 1$; then the Hamiltonian is given by:

$$\mathcal{H}(q,p) = \frac{q^2}{2} + \frac{p^2}{2}. \tag{11}$$

The initial point, solution interval, sample size, and batch size for the simple harmonic oscillator are provided in Table 2.

Table 2. The parameters for simple harmonic oscillator

	Symplectic-Euler and Verlet
Initial point	(0, 1)
Interval of t	[0, 2π]
Samples of t	5
Batch Size	1000

Nonlinear Oscillator

The kinetic and potential energies are defined, respectively, as follows:

$$\mathcal{T}(p) = \frac{p^2}{2}, \quad \mathcal{V}(q) = \frac{q^2}{2} + \frac{q^4}{4}. \tag{12}$$

Thus, the Hamiltonian takes the form:

$$\mathcal{H}(q,p) = \frac{p^2}{2} + \frac{q^2}{2} + \frac{q^4}{4}. \tag{13}$$

For the nonlinear oscillator, the initial point, solution interval, sample size, and batch size are provided in Table 3.

Table 3. The parameters for nonlinear oscillator

	Symplectic-Euler and Verlet
Initial point	(1.3, 1)
Interval of t	[0, 4π]
Samples of t	25
Batch size	1000

Lotka-Volterra

The Hamiltonian of the Lotka-Volterra dynamical system is defined as:

$$\mathcal{H}(x,y) = \delta x - \gamma ln(x) + \beta y - \alpha ln(y), \tag{14}$$

where x and y do not represent canonical coordinates. Assuming $q = ln(y)$, $p = ln(x)$, and setting $\alpha = 2/3$, $\beta = 4/3$, $\gamma = \delta = 1$, the Hamiltonian takes the form:

$$\mathcal{H}(q,p) = e^p - p + \frac{4}{3}e^q - \frac{2}{3}q, \tag{15}$$

where

$$\begin{aligned} \mathcal{T}(p) &= \delta e^p - \gamma p = e^p - p, \\ \mathcal{V}(q) &= \beta e^q - \alpha q = \frac{4}{3}e^q - \frac{2}{3}q. \end{aligned} \tag{16}$$

For the Lotka-Volterra problem, the initial point, solution interval, sample size, and batch sizes are specified in Table 4.

Table 4. The parameters for Lotka-Volterra

	Symplectic-Euler and Verlet
Initial Point	(-0.105, -0.105)
Interval of t	[0, 50]
Samples of t	45
Batch Size	5000

1 Project page is available at https://tayfununal.github.io/Article-2/

RESULTS AND DISCUSSION

Initially, the symplectic Euler method is employed as the solver in the proposed model. Only 5 data points are utilized for the harmonic oscillator, 25 data points for the nonlinear oscillator, and 45 data points for the Lotka-Volterra system during the training phase. Subsequent to the training, the results are presented in Figures Figure 3, Figure 4, and Figure 5. Despite training the model with a minimal amount of data, high performance is achieved across all examples. In the cases of the simple harmonic oscillator, nonlinear oscillator, and Lotka-Volterra system, the model demonstrated significantly better energy conservation compared to the symplectic-Euler method and provided a closer approximation to the ground truth.

MSE values for energy conservation were computed by comparing both the symplectic-Euler method and the proposed model to the ground truth in Table 5. This

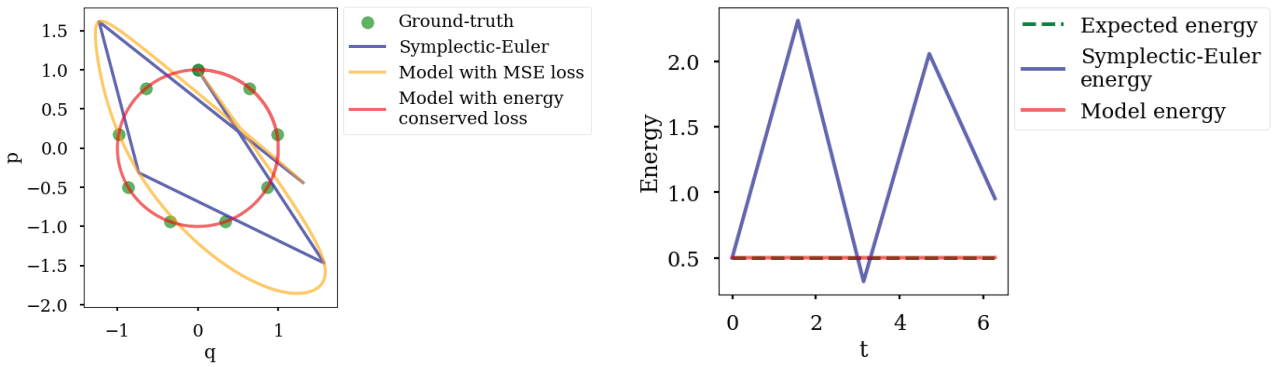

Figure 3. Phase diagram and energy graph of the harmonic oscillator using the symplectic-Euler solver.

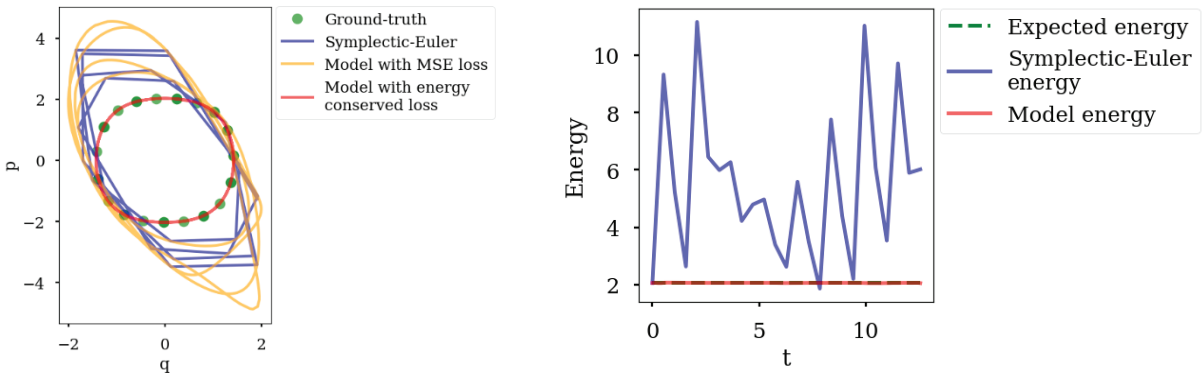

Figure 4. Phase diagram and energy graph of the nonlinear oscillator using the symplectic-Euler solver.

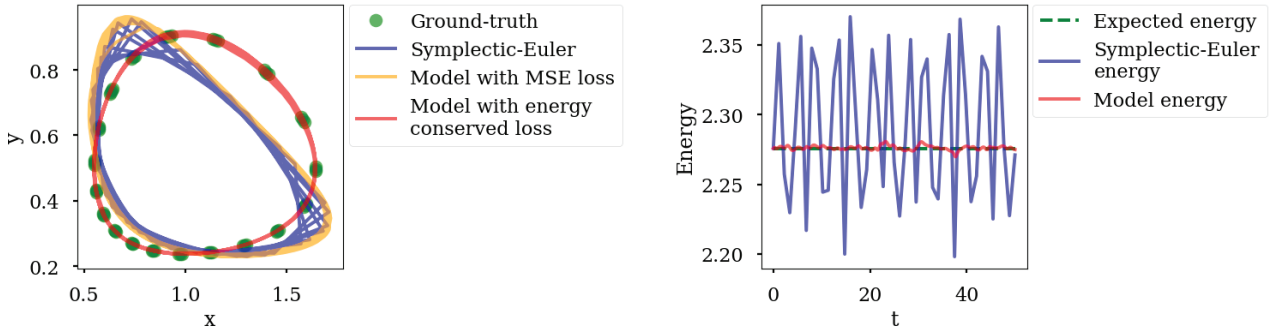

Figure 5. Phase diagram and energy graph of the Lotka-Volterra system using the symplectic-Euler solver.

Table 5. MSE values for both symplectic-Euler and the proposed model according to ground-truth

	MSE values	
Examples	**Symplectic -Euler**	**The proposed model**
Simple harmonic oscillator	1.1874350	0.0000010
Non-linear oscillator	18.471113	0.0000940
Lotka-Volterra	0.0029499	0.0000028

comparison highlights the effectiveness of the proposed model in maintaining energy conservation more accurately than the symplectic-Euler method.

The second solver employed in this model is the Verlet method. As previously mentioned, the examples utilize 5, 25, and 45 data points, respectively. The results are presented in Figures Figure 6, Figure 7, and Figure 8.

MSE values for energy conservation were computed by comparing both the Verlet method and the proposed model to the ground truth in Table 6. This comparison highlights

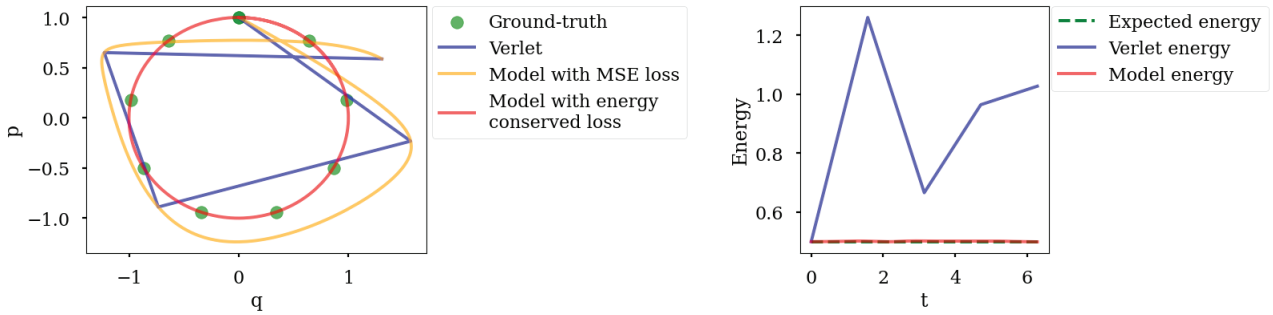

Figure 6. Phase diagram and energy graph of the harmonic oscillator using the Verlet method as the solver.

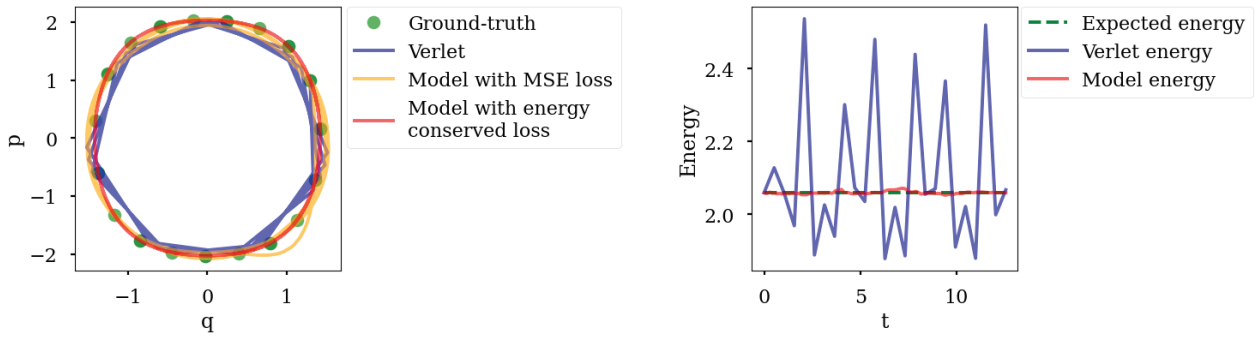

Figure 7. Phase diagram and energy graph of the nonlinear oscillator using the Verlet method as the solver.

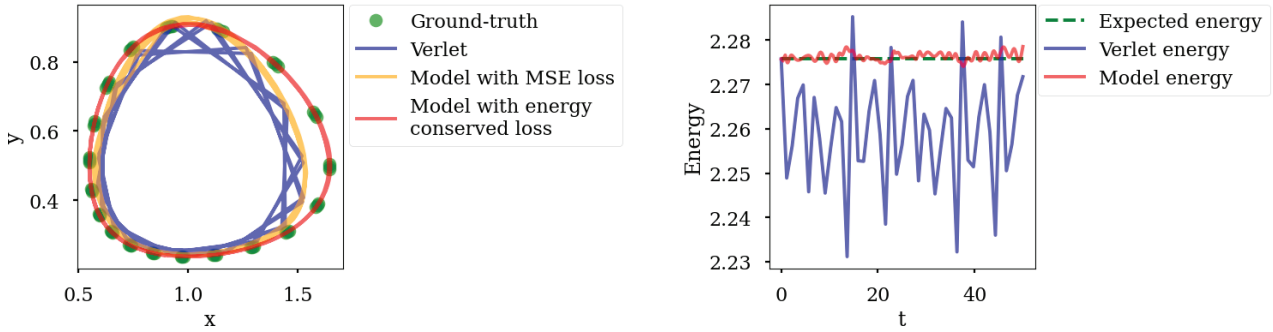

Figure 8. Phase diagram and energy graph of the Lotka-Volterra system using the Verlet method as the solver.

Table 6. MSE values for both Verlet and the proposed model according to ground-truth

	MSE values	
Examples	**Verlet**	**The proposed model**
Simple harmonic oscillator	0.220253	0.0000030
Non-linear oscillator	0.044066	0.0000250
Lotka-Volterra	0.000431	0.0000003

the effectiveness of the proposed model in maintaining energy conservation more accurately than the Verlet method.

Lastly, a comparison between the results obtained using the symplectic-Euler method and the Verlet method is presented for all examples, as illustrated in Figures Figure 9, Figure 10, and Figure 11. Although the Verlet method conserves the symplectic structure more effectively than the symplectic-Euler method, the results demonstrate that the model performs exceptionally well with both solvers.

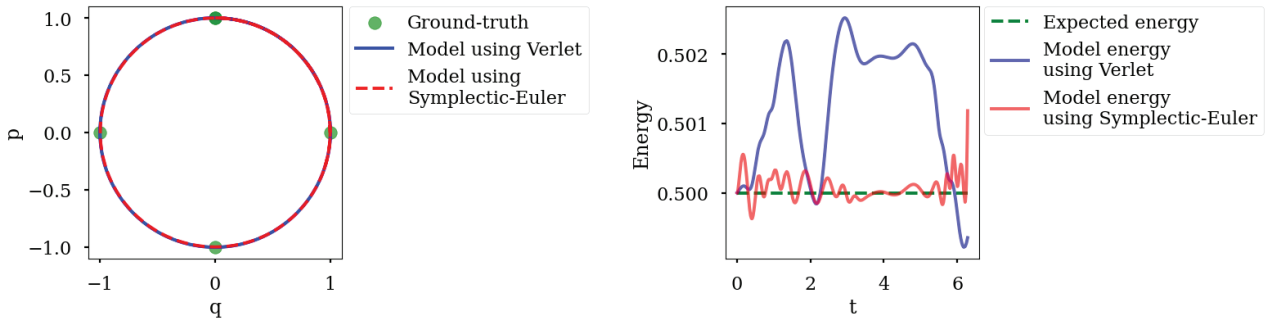

Figure 9. Symplectic-Euler versus Verlet methods in the phase diagram and energy graph of the harmonic oscillator.

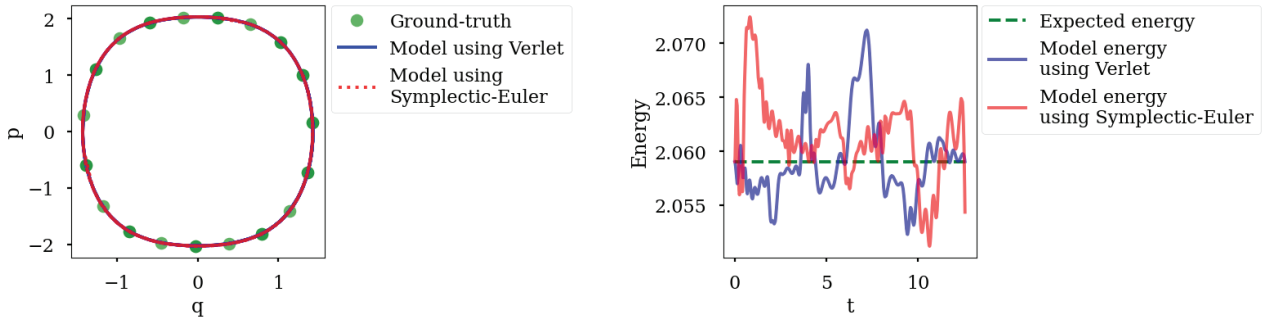

Figure 10. Symplectic-Euler versus Verlet methods in the phase diagram and energy graph of the nonlinear oscillator.

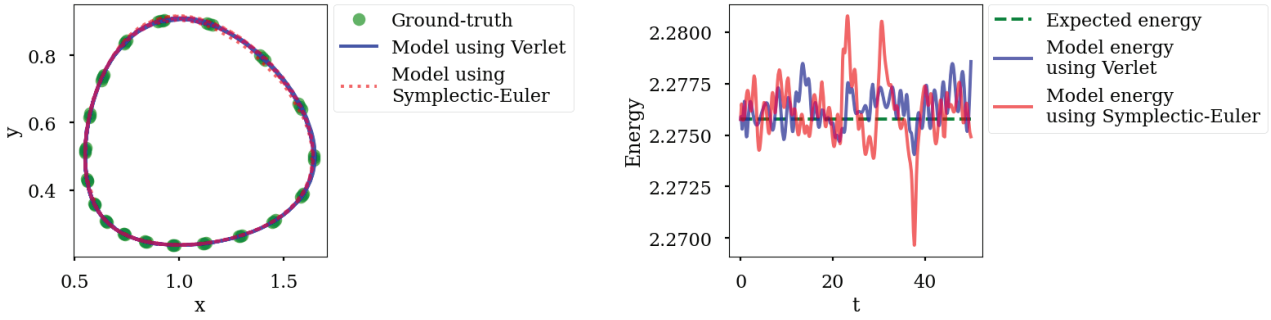

Figure 11. Symplectic-Euler versus Verlet methods in the phase diagram and energy graph of the Lotka-Volterra system.

CONCLUSION

A data-driven deep learning algorithm for solving separable Hamiltonian systems has been presented. With the proposed algorithm, Hamiltonian systems have been solved numerically. Since Hamiltonian systems are widely used in physics and engineering, the proposed algorithm will be beneficial in these fields. For instance, many models in mechanics can be expressed in Hamiltonian form. Therefore, the suggested algorithm can be applied in the study and control of mechanical systems. The proposed algorithm offers advantages over existing schemes by providing a continuous solution that balances the preservation of symplectic structure and energy conservation even with few data. There are no limitations on the choice of an ordinary differential equation solver in the initial step. In this study, Symplectic-Euler and Verlet methods were employed; however, alternative solvers such as Euler, Runge-Kutta, or neural ordinary differential equation solvers may also be utilized. Experimental results demonstrate that the algorithm performs effectively, even when utilizing few data points. An important distinction of the proposed method compared to its counterparts is that it does not utilize the differential of the neural network. This approach renders the method easy to implement, allowing it to be applied similarly to ordinary regression problems, which is advantageous in terms of numerical stability. In future work, the research will be extended to address partial differential equations utilizing the proposed method with certain modifications. Furthermore, other deep learning models, such as autoencoders and flow-based models, will be employed to augment the data.

REFERENCES

[1] Lagaris IE, Likas A, Fotiadis DI. Artificial neural netwoxrks for solving ordinary and partial differential equations. IEEE Trans Neural Netw 1998;9:987-1000. [CrossRef]

[2] Lee H, Kang IS. Neural algorithm for solving differential equations. J Comput Phys 1990;91:110-131. [CrossRef]

[3] Chen RTQ, Rubanova Y, Bettencourt J, Duvenaud D. Neural ordinary differential equations. Adv Neural Inf Process Syst 2018;31.

[4] Mattheakis M, Sondak D, Dogra AS, Protopapas P. Hamiltonian neural networks for solving equations of motion. Phys Rev E 2022;105:65305. [CrossRef]

[5] Yildiz S, Goyal P, Bendokat T, Benner P. Data-driven identification of quadratic symplectic representations of nonlinear Hamiltonian systems. arXiv [Preprint] 2023:arXiv:2308.01084. [CrossRef]

[6] Chen S, Kevrekidis PG, Zhang H, Zhu W. Data-driven discovery of conservation laws from trajectories via neural deflation. arXiv [Preprint] 2024:arXiv:2410.05445. [CrossRef]

[7] Aydın M, Yakut O, Alli H. Implementation of sliding mode control with artificial neural network to the rotary inverted pendulum system. Sigma J Eng Nat Sci 2013;5:1-7.

[8] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2018;378:686-707. [CrossRef]

[9] Khoo Z, Wu D, Low JSC, Bressan S. Separable Hamiltonian neural networks. Phys Rev E 2024;110:44205. [CrossRef]

[10] Greydanus S, Dzamba M, Yosinski J. Hamiltonian neural networks. arXiv [Preprint] 2019:arXiv:1906.01563.

[11] Han CD, Glaz B, Haile M, Lai YC. Adaptable Hamiltonian neural networks. Phys Rev Res 2021;3:23156. [CrossRef]

[12] Haitsiukevich K, Ilin A. Learning trajectories of Hamiltonian systems with neural networks. arXiv [Preprint] 2022:arXiv:2204.05077. [CrossRef]

[13] Bertalan T, Dietrich F, Mezić I, Kevredikis IG. On learning Hamiltonian systems from data. Chaos 2019;29:121107. [CrossRef]

[14] Dierkes E, Flaßkamp K. Learning Hamiltonian systems considering system symmetries in neural networks. IFAC-PapersOnLine 2021;54:210-216. [CrossRef]

[15] Celledoni E, Leone A, Murari D, Owren B. Learning Hamiltonians of constrained mechanical systems. J Comput Appl Math 2023;417:114608. [CrossRef]

[16] Iten R, Metger T, Wilming H, Rio LD, Renner R. Discovering physical concepts with neural networks. Phys Rev Lett 2020;124:10508. [CrossRef]

[17] Yu J, Lu L, Meng X, Karniadakis GE. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. Comput Methods Appl Mech Eng 2022;393:114823. [CrossRef]

[18] Gladstone RJ, Nabian MA, Meidani H. FO-PINNs: A first-order formulation for physics informed neural networks. arXiv [Preprint] 2022:arXiv:2210.14320.

[19] Wang Y, Zhong L. NAS-PINN: Neural architecture search-guided physics-informed neural network for solving PDEs. J Comput Phys 2024;496:112603. [CrossRef]

[20] Norambuena A, Mattheakis M, González FJ, Coto R. Physics-informed neural networks for quantum control. Phys Rev Lett 2024;132:10801. [CrossRef]

[21] Canizares P, Murari D, Schönlieb CB, Sherry F, Shumaylov Z. Hamiltonian matching for symplectic neural integrators. arXiv [Preprint] 2024:arXiv:2410.18262.

[22] Choudhary A, Lindner JF, Holliday EG, Miller ST, Sinha S, Ditto WL. Physics-enhanced neural networks learn order and chaos. Phys Rev E 2020;101:62207. [CrossRef]

[23] Tong Y, Xiong S, He X, Pan G, Zhu B. Symplectic neural networks in Taylor series form for Hamiltonian systems. J Comput Phys 2021;437:110325. [CrossRef]

[24] Jin P, Zhang Z, Zhu A, Tang Y, Karniadakis GE. SympNets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. Neural Netw 2020;132:166-179. [CrossRef]

[25] Bajars J. Locally-symplectic neural networks for learning volume-preserving dynamics. J Comput Phys 2023;476:111911. [CrossRef]

[26] Jin P, Zhang Z, Kevrekidis IG, Karniadakis GE. Learning Poisson systems and trajectories of autonomous systems via Poisson neural networks. IEEE Trans Neural Netw Learn Syst 2022;34:8271-8283. [CrossRef]

[27] Lusch B, Kutz NJ, Brunton SL. Deep learning for universal linear embeddings of nonlinear dynamics. Nat Commun 2018;9:4950. [CrossRef]

[28] Hernandez Q, Badías A, González D, Chinesta F, Cueto E. Deep learning of thermodynamics-aware reduced-order models from data. Comput Methods Appl Mech Eng 2021;379:113763. [CrossRef]

[29] Zhong G, Marsden JE. Lie-Poisson Hamilton-Jacobi theory and Lie-Poisson integrators. Phys Lett A 1988;133:134-139. [CrossRef]