





**Physics-enhanced neural networks learn order and chaos**Anshul Choudhary <sup>1</sup>, John F. Lindner <sup>1,2,\*</sup>, Elliott G. Holliday,<sup>1</sup> Scott T. Miller <sup>1</sup>, Sudeshna Sinha,<sup>1,3</sup>  
and William L. Ditto <sup>1</sup><sup>1</sup>*Nonlinear Artificial Intelligence Laboratory, Physics Department, North Carolina State University, Raleigh, North Carolina 27607, USA*<sup>2</sup>*Physics Department, The College of Wooster, Wooster, Ohio 44691, USA*<sup>3</sup>*Indian Institute of Science Education and Research Mohali, Knowledge City, SAS Nagar, Sector 81, Manauli PO 140 306, Punjab, India*

(Received 26 November 2019; revised manuscript received 22 May 2020; accepted 24 May 2020; published 18 June 2020)

Artificial neural networks are universal function approximators. They can forecast dynamics, but they may need impractically many neurons to do so, especially if the dynamics is chaotic. We use neural networks that incorporate Hamiltonian dynamics to efficiently learn phase space orbits even as nonlinear systems transition from order to chaos. We demonstrate Hamiltonian neural networks on a widely used dynamics benchmark, the Hénon-Heiles potential, and on nonperturbative dynamical billiards. We introspect to elucidate the Hamiltonian neural network forecasting.

DOI: [10.1103/PhysRevE.101.062207](https://doi.org/10.1103/PhysRevE.101.062207)**I. INTRODUCTION**

Newton wrote, “My brain never hurt more than in my studies of the moon (and Earth and Sun)” [1]. Unsurprising sentiment, as the seemingly simple three-body problem is intrinsically intractable and practically unpredictable. Nonetheless, Hamilton remarkably reimagined Newton’s laws as an incompressible energy-conserving flow in phase space [2], highlighting the difference between integrable and nonintegrable systems [3], and heralding the discovery of deterministic chaos [4,5].

Today, artificial neural networks are popular tools in industry and academia [6], especially for classification and regression, and are beginning to elucidate nonlinear dynamics [7] and fundamental physics [8–10]. Recent neural networks outperform traditional techniques in symbolic integration [11] and numerical integration [12] and outperform humans in strategy games such as chess and Go [13]. But neural networks have a blind spot; they do not understand that “Clouds are not spheres, mountains are not cones, coastlines are not circles, . . .” [14]. They are unaware of the chaos and strange attractors of nonlinear dynamics, where exponentially separating trajectories bounded by finite energy repeatedly stretch and fold into complicated self-similar fractals. Their attempts to learn and predict nonlinear dynamics can be frustrated by ordered and chaotic orbits coexisting at the same energy for different initial positions and momenta.

Recent research [15–19] features artificial neural networks that incorporate Hamiltonian structure to learn fundamental

dynamical systems. But from stormy weather to swirling galaxies, natural dynamics is far richer and more challenging. In this article, we exploit the Hamiltonian structure of conservative systems to provide neural networks with the physics intelligence needed to learn the mix of order and chaos that often characterizes natural phenomena. After reviewing Hamiltonian chaos and neural networks, we apply Hamiltonian neural networks to the Hénon-Heiles potential, a numerical and dynamical benchmark, which models both stellar [20,21] and molecular [22–24] dynamics. Even as these systems transition from order to chaos, Hamiltonian neural networks correctly learn their dynamics, overcoming deep learning’s chaos blindness. If chaos is a nonlinear “super power,” enabling deterministic dynamics to be practically unpredictable, then the Hamiltonian is a neural network “secret sauce,” a special ingredient that enables learning and forecasting order and chaos.

**II. HAMILTONIAN CHAOS**

The Hamiltonian formalism describes phenomena from astronomical scales to quantum scales; even dissipative systems involving friction or viscosity are microscopically Hamiltonian. It reveals underlying structures in position-momentum phase space and reflects essential symmetries in physical systems. Its elegance stems from its geometric structure, where positions  $q$  and conjugate momenta  $p$  form a set of  $2N$  canonical coordinates describing a physical system with  $N$  degrees of freedom. A *single* Hamiltonian function  $\mathcal{H}$  uniquely generates the time evolution of the system via the  $2N$  coupled differential equations

$$\{\dot{q}, \dot{p}\} = \{dq/dt, dp/dt\} = \{+\partial\mathcal{H}/\partial p, -\partial\mathcal{H}/\partial q\}, \quad (1)$$

where the overdots are Newton’s notation for time derivatives.

This classical formalism exhibits two contrasting dynamics: simple integrable motion suggests a “clockwork universe,” while complicated nonintegrable motion suggests

\*Corresponding author: [jlindner@wooster.edu](mailto:jlindner@wooster.edu)

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International](https://creativecommons.org/licenses/by/4.0/) license. Further distribution of this work must maintain attribution to the author(s) and the published article’s title, journal citation, and DOI.

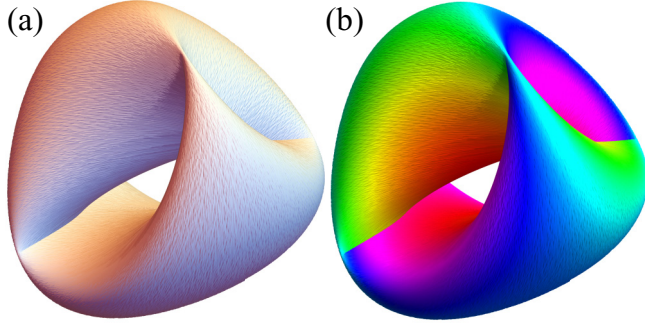


FIG. 1. Hamiltonian flow. (a) Hénon-Heiles orbit wrapped many times around the hypertorus appears to intersect at the creases in this 3D projection. (b) Different colors indicating the fourth dimension show that the apparent intersections are actually separated in 4D phase space.

a chaotic one. Additional conserved quantities constrain integrable orbits to smooth  $N$ -dimensional “kamtori” [25] in  $2N$ -dimensional phase space, as in Fig. 1, but increasing nonlinearity can “fractalize” adjacent kamtori into infinitely intersecting “cantori” [26], allowing nonintegrable orbits to wander over the entire phase space, extremely sensitive to initial conditions, and constrained only by energy.

The Hénon-Heiles potential [20], which models phenomena ranging from the orbits of stars to the vibrations of molecules, provides a famous example of such an order-to-chaos transition. In a four-dimensional phase space  $\{q, p\} = \{q_x, q_y, p_x, p_y\}$ , its nondimensionalized Hamiltonian

$$\mathcal{H} = (p_x^2 + p_y^2)/2 + (q_x^2 + q_y^2)/2 + (q_x^2 q_y - q_y^3/3) \quad (2)$$

is the sum of the kinetic and potential energies, including quadratic harmonic terms perturbed by cubic nonlinearities that convert a circularly symmetric potential into a triangularly symmetric potential. Bounded motion is possible in a triangular region of the  $\{x, y\}$  plane for energies  $0 < E < 1/6$ . As orbital energy increases, circular symmetry degenerates to triangular symmetry, integrable motion degrades to nonintegrable motion, kamtori become cantori, and ordered islands give way to a chaotic sea.

### III. NEURAL NETWORKS

While traditional analyses focus on forecasting orbits or understanding fractal structure, understanding the entire landscape of dynamical order and chaos requires new tools. Artificial neural networks are today widely used and studied partly because they can approximate any continuous function [27,28]. Recent efforts to apply them to chaotic dynamics involve the *recurrent* neural networks of *reservoir computing* [29–31]. We instead exploit the dominant *feed-forward* neural networks of *deep learning* [6].

Inspired by natural neural networks, the activity  $a_\ell = \sigma[W_\ell a_{\ell-1} + b_\ell]$  of each layer of a conventional feed-forward neural network is the nonlinear step or ramp of the linearly transformed activities of the previous layer, where  $\sigma$  is a vectorized nonlinear function that mimics the on-off activity of a natural neuron,  $a_\ell$  are activation vectors, and  $W_\ell$  and  $b_\ell$  are adjustable weight matrices and bias vectors that mimic the

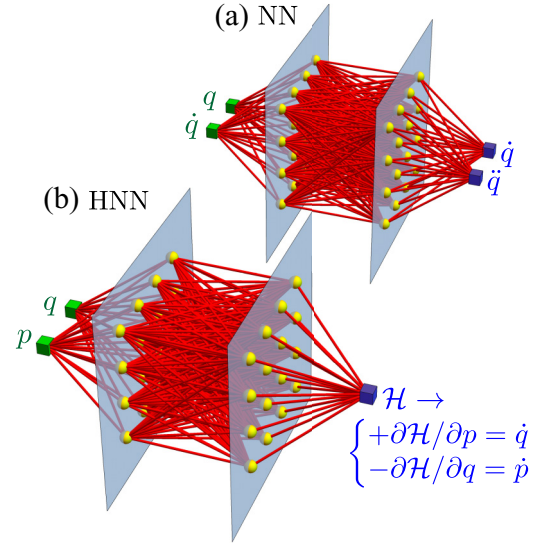


FIG. 2. Neural network schematics. Weights (red lines) and biases (yellow spheres) connect inputs (green cubes) through neuron hidden layers (gray planes) to outputs (blue cubes). (a) NN has  $2N$  inputs and  $2N$  outputs. (b) HNN has  $2N$  inputs and 1 output but internalizes the output’s gradient in its weights and biases.

dendrite and axon connectivity of natural neurons. Concatenating multiple layers eliminates the hidden neuron activities, so the output  $y = f_P[x]$  is a parametrized nonlinear function of just the input  $x$  and the weights and biases  $P = \{W_\ell, b_\ell\}$ . A training session inputs multiple  $x$  and adjusts the weights and biases to minimize the difference or “loss”  $\mathcal{L} = (y_t - y)^2$  between the target  $y_t$  and the output  $y$  so the neural network learns the correspondence.

Recently, neural networks have been proposed [15–19] that not only learn dynamics but also capture invariants and symmetries, including Hamiltonian phase space structure. In particular, Greydanus *et al.* [15] introduced Hamiltonian neural networks, and Toth *et al.* [16] added automatic learning of the canonical coordinates, among other refinements.

The Fig. 2 conventional neural network NN intakes positions and velocities  $\{q, \dot{q}\}$  and outputs approximations to

TABLE I. Python neural network parameters. We explored all values but generated our final results with the **bold** ones.

Description	Values
Number of layers	2, 4, 6, 8
Neurons per layer	100, <b>200</b> , 400
Optimizer	<b>Adam</b> , SGD
Training loss threshold	<b><math>10^{-4}</math></b> , $10^{-5}$
Batch size	128, <b>256</b> , 512
Epochs	10, <b>50</b> , 100
Activations	<b>Tanh</b> , ReLU
Orbit time	1000, <b>5000</b>
Energy samples	<b>20</b>
Orbits per energy	<b>20</b> , 50
Integration scheme	RK4, <b>RK45</b> , Symplectic
Data sampling rate	0.01, <b>0.1</b> , 1, 10

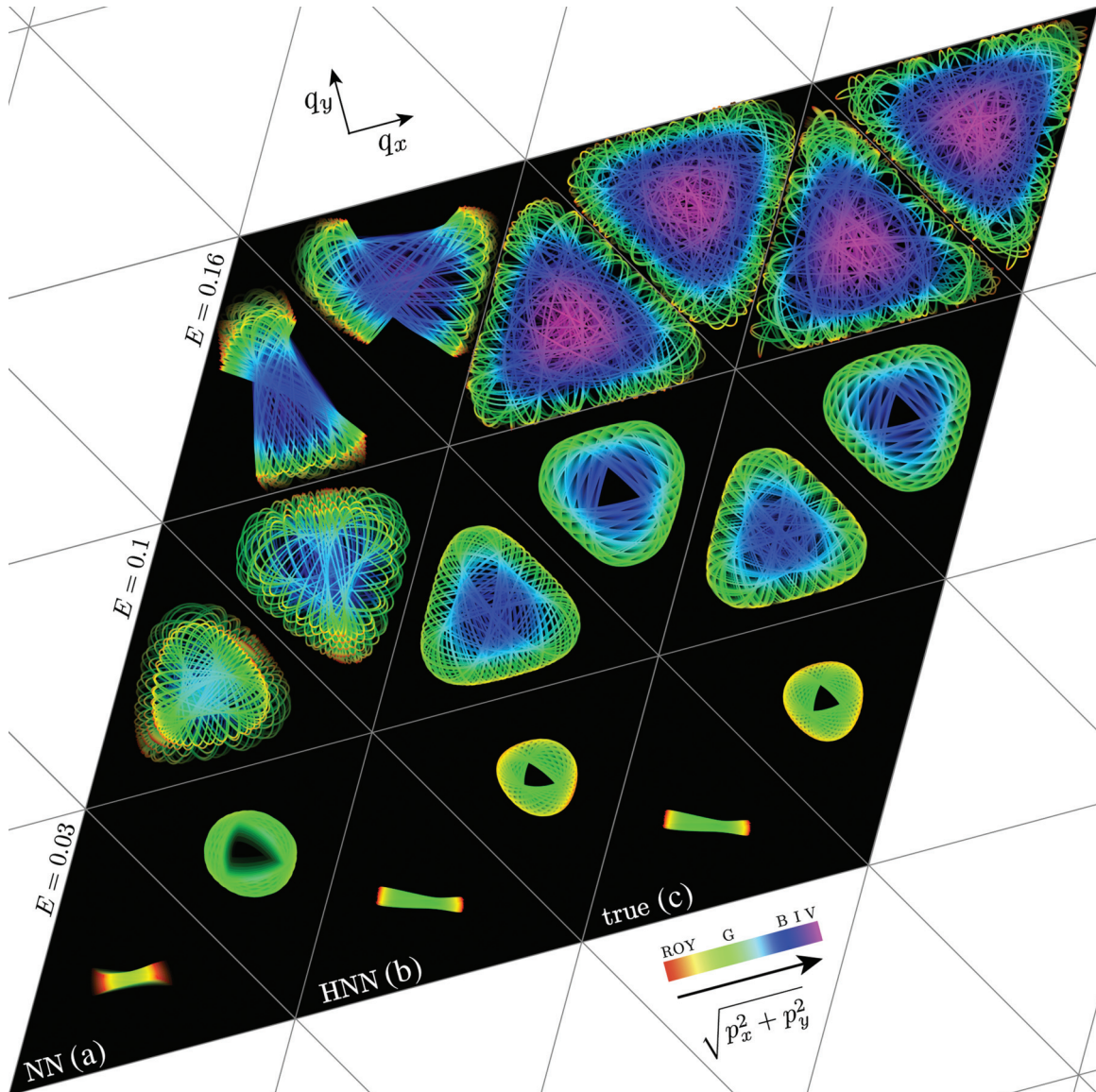


FIG. 3. Hénon-Heiles flows. Two sample Hénon-Heiles flows  $\{q_x, q_y\}$  for different initial conditions forecast by conventional neural network (a), Hamiltonian neural network (b), and Hénon-Heiles differential equations (c), for small, medium, and large bounded energies  $0 < E < 1/6$ . Hues code momentum magnitudes, from red to violet; orbit tangents code momentum directions. Orbits fade into the past. HNN's physics-informed forecasting is especially better than NN's at high energies.

their time derivatives  $\{\dot{q}, \ddot{q}\}$ , adjusting its weights and biases to minimize the loss

$$\mathcal{L}_{\text{NN}} = (\dot{q}_t - \dot{q})^2 + (\ddot{q}_t - \ddot{q})^2 \quad (3)$$

until it learns the correct mapping. In contrast, the Fig. 2 Hamiltonian neural network (HNN) intakes position and momenta  $\{q, p\}$ , outputs the scalar function  $\mathcal{H}$ , takes its gradient to find its position and momentum rates of change, and minimizes the loss

$$\mathcal{L}_{\text{HNN}} = (\dot{q}_t - \partial\mathcal{H}/\partial p)^2 + (\dot{p}_t + \partial\mathcal{H}/\partial q)^2, \quad (4)$$

which enforces Hamilton's equations of motion. For a given time step  $dt$ , each trained network can extrapolate a given initial condition with an Euler update  $\{q, p\} \leftarrow \{q, p\} + \{\dot{q}, \dot{p}\}dt$  or some better integration scheme [32].

Loosely, NN learns the orbits, while HNN learns the Hamiltonian. Geometrically, NN learns the generalized velocities, the *dual* mappings  $\{q, \dot{q}\} \rightarrow \dot{q}$  and  $\{q, \dot{q}\} \rightarrow \ddot{q}$ , while HNN learns the Hamiltonian generator function, the *single* mapping  $\{q, p\} \rightarrow \mathcal{H}$ , whose (symplectic) gradient gives the generalized velocities  $\{\dot{q}, \dot{p}\}$ . With the same resources, HNN outperforms NN, and the advantage grows as the phase space dimension increases, where  $q$  and  $p$  are multicomponent vectors.

#### IV. HÉNON-HEILES EXAMPLE

We “stress test” our neural networks on the Hénon-Heiles system, as its mixed phase space of order and chaos is an especially challenging dynamical scenario to identify and decipher. For selected bounded energies, and for the same

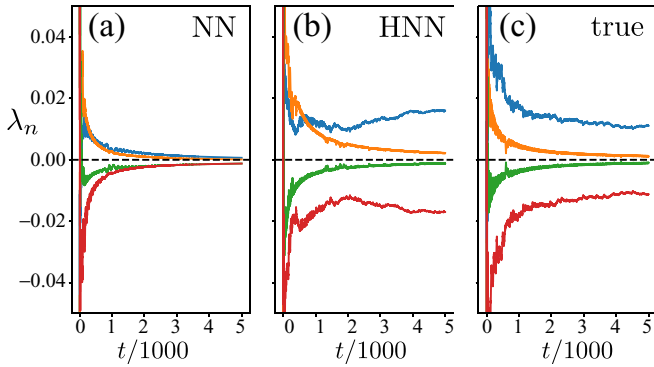


FIG. 4. Lyapunov spectrum. Integrating the variational equations estimates the Hénon-Heiles Lyapunov exponents  $\lambda_n$  after time  $t$  for NN (a), HNN (b), and true (c) for an example initial condition. HNN better approaches the expected Hamiltonian spectrum  $\{-\lambda, 0, 0, +\lambda\}$ .

learning parameters detailed in Table I, we train NN and HNN on multiple Hénon-Heiles trajectories starting in the triangular basin that enables bounded motion. We use the neural networks to forecast new trajectories and then compare them to the “true” trajectories obtained by numerically integrating Hamilton’s Eq. (1). Figure 3 shows these results. HNN captures the nature of the global phase space structures well and effectively distinguishes qualitatively different dynamical regimes. NN forecasts are especially poor at high energies.

To quantify the ability of NN and HNN to paint a full portrait of the global, mixed phase space dynamics, we use their knowledge of the system to estimate the Hénon-Heiles Lyapunov spectrum [33], which characterizes the separation rate of infinitesimally close trajectories, one exponent for each dimension. Since perturbations along the flow do not cause divergence away from it, at least one exponent will be zero. For a Hamiltonian system, the exponents must exist in diverging-converging pairs to conserve phase space volume. Hence we expect a spectrum like  $\{-\lambda, 0, 0, +\lambda\}$ , as in Fig. 4, with the maximum exponent increasing at large energies like  $\lambda \propto E^{3.5}$  [34], as in Fig. 5. HNN satisfies both these expectations, which are nontrivial consistency checks that it

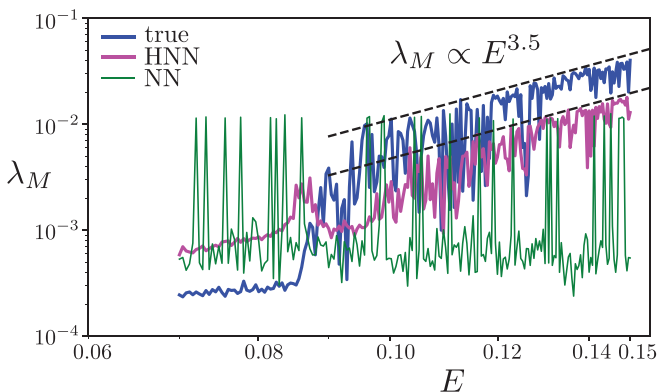


FIG. 5. Lyapunov scaling. Maximum Lyapunov exponent  $\lambda_M$  versus energy  $E$  for NN, HNN, and true. HNN reproduces the correct energy exponent, while NN shows no trend at all.

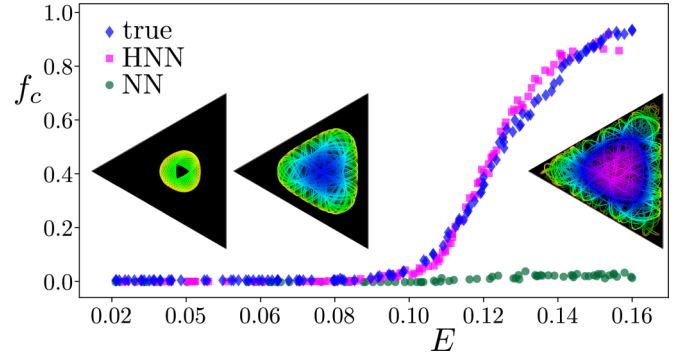


FIG. 6. Order to chaos. Fraction of chaotic orbits  $f_c$  for random energies  $E$ , as inferred by the smaller alignment index  $\alpha$ , for conventional neural network (green circles), Hamiltonian neural network (magenta squares), and Hénon-Heiles differential equations (blue diamonds). NN is especially poor for chaotic high-energy orbits. Insets are sample orbits.

has authentically learned a conservative flow similar to the Hénon-Heiles flow. NN satisfies neither check.

Using NN and HNN we also compute the smaller alignment index  $\alpha$ , a metric of chaos that allows us to quickly find the fraction of orbits that are chaotic at any energy [35]. We compute  $\alpha$  for a specific orbit by following the time evolution of two different normalized deviation vectors along the orbit and computing the minimum of the norms of their difference and sum. Via extensive testing, an orbit is chaotic if  $\alpha < 10^{-8}$ , indicating that its deviation vectors have been aligned or antialigned by a large positive Lyapunov exponent. Figure 6 shows the fraction of chaotic trajectories for each energy, including a distinct transition between islands of order at low energy and a sea of chaos at high energy. The chaos fractions computed with HNN forecasts are good at all energies, but those computed by NN forecasts are poor at high energies.

Finally, to understand what NN and HNN have learned when they forecast orbits, we use an *autoencoder*—a neural network with a sparse “bottleneck” layer—to examine their hidden neurons. The autoencoder’s mean-square-error loss function forces the input to match the output, so its weights and biases adjust to create a compressed, low-dimensional representation of the neural networks’ activity, a process called introspection [36]. For HNN, the loss function  $\mathcal{L}_b$  drops precipitously for  $N_b = 4$  (or more) bottleneck neurons, which appear to encode a combination of the four phase space coordinates, thereby capturing the dimensionality of the system, as in Fig. 7. NN shows no similar drop, and the uncertainty in its loss function is orders of magnitude larger than HNN’s.

## V. BILLIARDS EXAMPLE

Billiards model a wide range of real-world systems, spanning lasers [37], optical fibers, ray optics [38], acoustic and microwave cavities [38–40], quantum dots [41], and nanodevices [42]. Billiards also elucidates the subtleties of quantum-classical correspondence and the challenging notion of quantum chaos [43–46].

Dynamicists typically idealize billiard tables with hard boundaries and discontinuous potentials. With similar

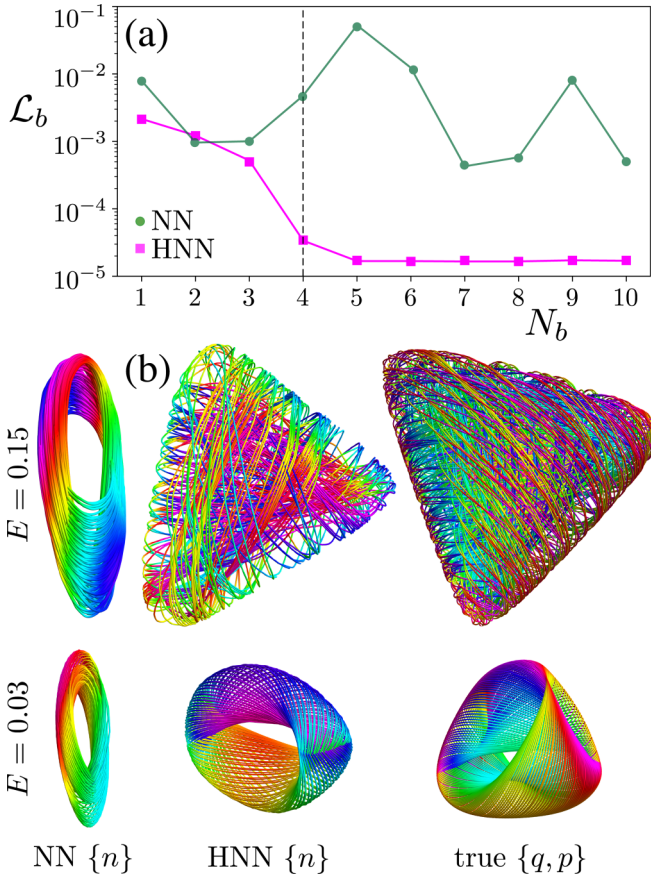


FIG. 7. Introspection. (a) When an autoencoder compresses HNN forecasting, its loss function  $\mathcal{L}_b$  (magenta squares) drops precipitously as its bottleneck layer increases past  $N_b = 4$  neurons, the dimensionality of the Hénon-Heiles system; when an autoencoder compresses NN forecasting, its loss function (green circles) wiggles irregularly, oblivious to this transition. (b) HNN’s compressed representation  $\{n_1, n_2, n_3, n_4\}$  resembles the low- or high-energy orbit  $\{q_x, q_y, p_x, p_y\}$  it is forecasting, where color indicates the fourth dimension. NN hardly notices the differences between the low- or high-energy orbits.

phenomenology, we model billiard tables with soft boundaries and continuous potentials, so the billiard balls’ momenta change rapidly but continuously at each bounce. Our Hamiltonian

$$\mathcal{H} = (p_x^2 + p_y^2)/2 + \mathcal{V} \tag{5}$$

with potential energy

$$\mathcal{V}[q_x, q_y] = + \frac{1}{1 + e^{(r_o - \sqrt{q_x^2 + q_y^2})/\delta r}} - \frac{1}{1 + e^{(r_i - \sqrt{(q_x - \delta q_x)^2 + q_y^2})/\delta r}}, \tag{6}$$

where  $r_o$  is the radius of the outer circle,  $r_i$  is the radius of the inner circle,  $\delta q_x$  is the shift of the inner circle, and  $\delta r$  is the softness of the walls, as in Fig. 8.

Our tables are bounded by two circles, and the dynamics exhibits fascinating transitions as the circles resize or shift. Ordered and chaotic trajectories coexist for different initial

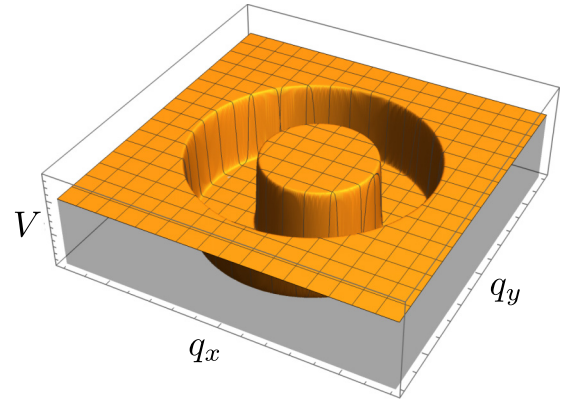


FIG. 8. Billiards potential energy. Potential energy versus position for circular billiards.

conditions at the same energy. Figure 9 shows typical trajectories, where grays code potential and colors code time.

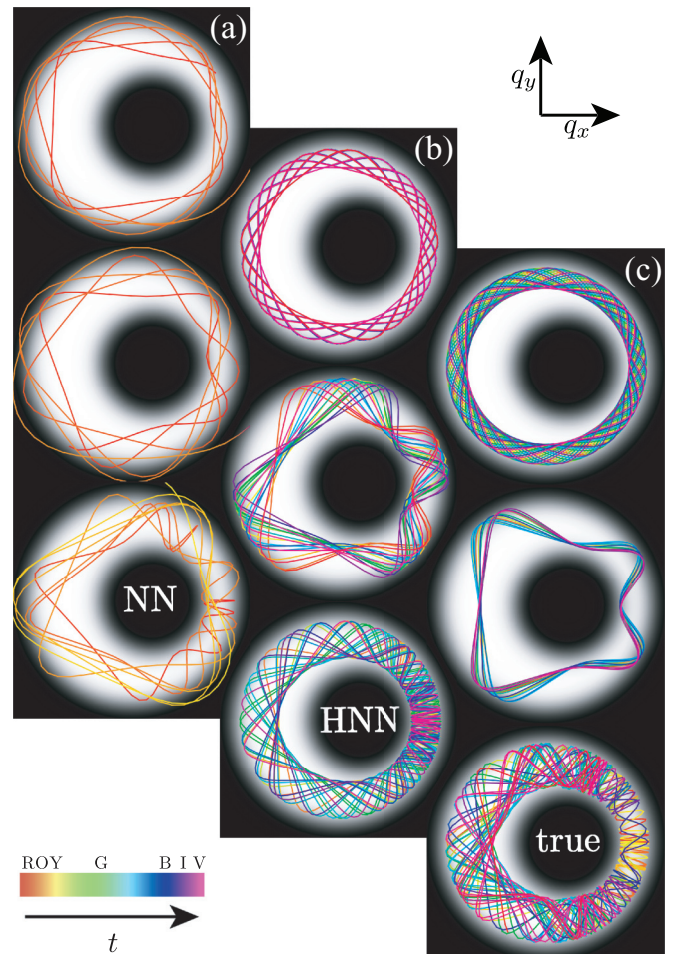


FIG. 9. Billiard flows. Three sample billiard flows  $\{q_x, q_y\}$  for different initial conditions, regular and bouncing from the outer circle only, regular and bouncing from both circles, and chaotic, forecast by conventional neural network (a), Hamiltonian neural network (b), and billiard differential equations (c). Hues code time, from red to violet. HNN physics bias significantly improves its dynamics forecasting over NN, which typically diverges after short times (and low-frequency colors).

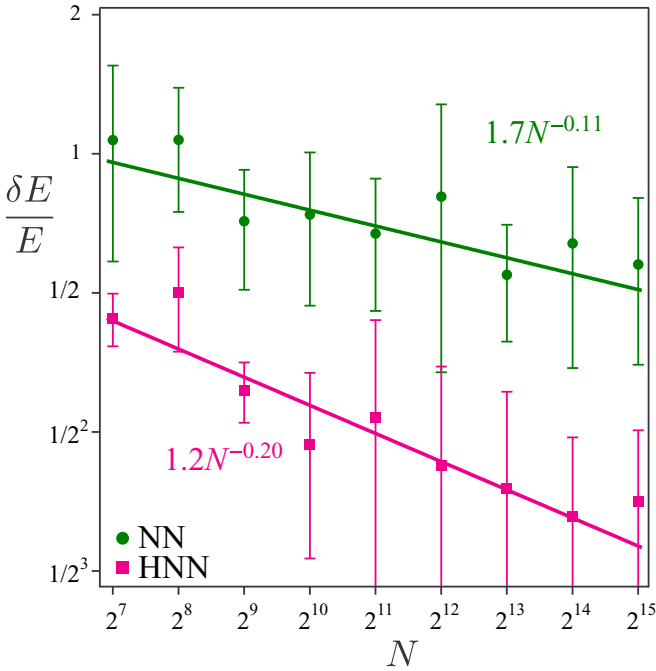


FIG. 10. Double pendulum. Relative energy error  $\delta E/E$  for forecasted orbits versus number of training pairs  $N$  for a double pendulum averaged over six different initial distributions of weights and biases for NN and HNN. Power-law fits guide the eye, and HNN’s advantage grows with training pairs.

After identical training, HNN easily outperforms NN, which often forecasts energy-violating escapes at early times (low-frequency colors).

**VI. OTHER EXAMPLES**

We have demonstrated the efficient forecasting of Hamiltonian neural networks in diverse systems, both perturbative, like Hénon-Heiles with  $\mathcal{H} = \mathcal{H}_0 + \epsilon \mathcal{H}_1$ , and nonperturbative, like dynamical billiards. Other successful implementations include higher-dimensional Hénon-Heiles, and simple harmonic oscillators and pendulums with noisy training data.

We have successfully used Hamiltonian neural networks to learn the dynamics of librating (back-and-forth) and rotating (end-over-end) single, double, and triple pendulums. The angles of rotating pendulums diverge, which makes them difficult for neural networks to learn. Consequently, we compactify the pendulum phase space by wrapping it into a cylinder. We find that both NN and HNN can learn the

pendulums, and that the learning improves with the number of training pairs, but HNN is always significantly better, as in Fig. 10 for the double pendulum.

To handle nonconservative systems, we are currently investigating Lagrangian and Newtonian neural network variations of HNN. The essential idea behind building such networks is manifold learning [47], but instead of learning a Hamiltonian as in HNN, we now learn a Lagrangian, from which we obtain the equations of motion. For instance, we are currently studying a Lagrangian neural network for a damped pendulum whose Hamiltonian conserves phase space volume but not energy.

We are also investigating potential applications of physics-informed neural networks like HNN to flights of autonomous drones through turbulent air characterized by chaotic changes in velocity and pressure. Such drones will use neural networks to navigate and maneuver, and physics-informed networks might efficiently learn and forecast their movements, including landings when the interaction between the rotors and the ground induce turbulence.

**VII. CONCLUSION**

Neural networks that respect Hamiltonian time-translational symmetry can learn order and chaos, including mixed phase space flows, as quantified by metrics like Lyapunov spectra and smaller alignment indices. Incorporating other symmetries [19] in deep learning may produce comparable qualitative performance improvements. While we have focused on feed-forward neural networks, nonlinear recurrent neural networks (RNNs) can exhibit coexisting chaotic and ordered orbits [48], and an algorithm for training an RNN without a Hamiltonian loss function may be possible.

Time series can be analyzed using multiple techniques, including genetic algorithms [49] or neural networks [9] to find algebraic or differential equations of motion or simple compressed representations [8]. But Newton and Poincaré had equations hundreds of years ago and still just glimpsed their complexity without fully understanding it.

Conventional neural networks extrapolating time series do not conserve energy, and their orbits can drift off the energy surface, jump into the sea of chaos from islands of order, or fly out to infinity. By incorporating energy-conserving and volume-preserving flows arising from an underlying Hamiltonian function—*without invoking any details of its form*—Hamiltonian neural networks can recognize the presence of

**ALGORITHM 1.** Generating orbit from learned model.

```

1: procedure EVOLVE(state, model)
2:    $q_1^0, q_2^0, p_1^0, p_2^0 \leftarrow state$ 
3:   while  $t \leq T$  do
4:      $flow \leftarrow \hat{q}_1, \hat{q}_2, \hat{p}_1, \hat{p}_2 \leftarrow \mathbf{MUPDATE}(state, model)$ 
5:      $q_1^{t+1}, q_2^{t+1}, p_1^{t+1}, p_2^{t+1} \leftarrow \mathbf{RK45}(flow, state)$ 
6:      $state \leftarrow q_1^{t+1}, q_2^{t+1}, p_1^{t+1}, p_2^{t+1}$ 
7:   end while
8:   return  $q_1, q_2, p_1, p_2$ 
9: end procedure

```

▷ Evolve from start with NN or HNN.  
 ▷ Integration Loop  
   ▷ Model update  
 ▷ Integrate one step  
 ▷ Return the full orbit

**ALGORITHM 2.** Model update subroutine.

---

```

1: procedure MUPDATE(state, model)                                ▷ advances the state using model
2:   if model == NN then
3:     flow ←  $\dot{q}_1, \dot{q}_2, \dot{p}_1, \dot{p}_2$  ← NN.Forward(state)          ▷ Forward pass
4:   else
5:      $\mathcal{H}$  ← HNN.Forward(state)                                ▷ Forward pass
6:     flow ←  $\dot{q}_1, \dot{q}_2, \dot{p}_1, \dot{p}_2$  ← backprop( $\mathcal{H}$ , state)        ▷ Gradients using back-propagation
7:   end if
8:   return flow                                                ▷ Return the vector field
9: end procedure

```

---

order and chaos as well as challenging regimes where both these very distinct dynamics coexist.

**ACKNOWLEDGMENTS**

This research was supported by ONR Grant No. N00014-16-1-3066 and a gift from United Therapeutics. J.F.L. thanks The College of Wooster for making possible his sabbatical at NCSU. S.S. acknowledges support from the J.C. Bose Fellowship (Grant No. SB/S2/JCB-013/2015).

**APPENDIX: NEURAL NETWORK IMPLEMENTATIONS**

We implement our baseline neural network NN and Hamiltonian neural network HNN in the Python programming language using the PyTorch open source machine learning library. Table I summarizes the parameters used to train our neural networks. On a 12-core desktop CPU, our model takes approximately 3 h to reach the desired training accuracy without any GPU acceleration. Our code is available upon request.

Much recent research on deep learning implements artificial neural networks in the Python ecosystem using automatic differentiation. (Unlike either numerical or symbolic differentiation, given a sequence of elementary arithmetic operations on elementary functions, automatic differentiation applies the calculus chain rule to decompose a compound derivative into many elementary derivatives, which are evaluated and combined numerically.) As an independent check of the HNN efficacy, we also implement NN and HNN in *Mathematica* using symbolic differentiation. We symbolically derive (long) formulas for the output, loss function, and gradients, which we then compile to fast machine code before training, validating, and testing the neural networks. The entire implementation is self-contained in a single *Mathematica* notebook. As in our Python implementation, HNN significantly outperforms NN.

Algorithms 1 and 2 outline how trajectories are generated using learned neural nets. **NN.Forward** and **HNN.Forward** are simply the conventional feed-forward pass for NN and HNN, respectively.

- 
- [1] V. G. Szebehely and H. Mark, *Adventures in Celestial Mechanics*, 2nd ed. (Wiley, New York, 1998).
  - [2] W. R. Hamilton, *Phys. Trans. Roy. Soc.* **124**, 247 (1834).
  - [3] H. Poincaré, *Il Nuovo Cimento* **10**, 128 (1899).
  - [4] E. N. Lorenz, *J. Atmos. Sci.* **20**, 130 (1963).
  - [5] J. Gleick, *Chaos: Making a New Science* (Penguin Books, New York, 1987).
  - [6] S. O. Haykin, *Neural Networks and Learning Machines*, 3rd ed. (Pearson, London, 2008).
  - [7] B. Lusch, J. N. Kutz, and S. L. Brunton, *Nat. Commun.* **9**, 4950 (2018).
  - [8] R. Iten, T. Metger, H. Wilming, L. del Rio, and R. Renner, *Phys. Rev. Lett.* **124**, 010508 (2020).
  - [9] S.-M. Udrescu and M. Tegmark, *Sci. Adv.* **6**, eaay2631 (2020).
  - [10] T. Wu and M. Tegmark, *Phys. Rev. E* **100**, 033311 (2019).
  - [11] G. Lample and F. Charton, [arXiv:1912.01412](https://arxiv.org/abs/1912.01412).
  - [12] P. G. Breen, C. N. Foley, T. Boekholt, and S. P. Zwart, *Mont. Not. R. Astron. Soc.* **494**, 2465 (2020).
  - [13] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, *Science* **362**, 1140 (2018).
  - [14] B. B. Mandelbrot, *The Fractal Geometry of Nature* (Freeman, San Francisco, CA, 1982).
  - [15] S. Greynanus, M. Dzamba, and J. Yosinski, [arXiv:1906.01563](https://arxiv.org/abs/1906.01563).
  - [16] P. Toth, D. J. Rezende, A. Jaegle, S. Racanière, A. Botev, and I. Higgins, [arXiv:1909.13789](https://arxiv.org/abs/1909.13789).
  - [17] M. Mattheakis, P. Protopapas, D. Sondak, M. Di Giovanni, and E. Kaxiras, [arXiv:1904.08991](https://arxiv.org/abs/1904.08991).
  - [18] T. Bertalan, F. Dietrich, I. Mezić, and I. G. Kevrekidis, *Chaos* **29**, 121107 (2019).
  - [19] R. Bondesan and A. Lamacraft, [arXiv:1906.04645](https://arxiv.org/abs/1906.04645).
  - [20] M. Hénon and C. Heiles, *Astron. J.* **69**, 73 (1964).
  - [21] J. Binney and S. Tremaine, *Galactic Dynamics*, Vol. 20 (Princeton University Press, Princeton, NJ, 2011).
  - [22] B. A. Waite and W. H. Miller, *J. Chem. Phys.* **74**, 3910 (1981).
  - [23] M. D. Feit and J. A. Fleck, *J. Chem. Phys.* **80**, 2578 (1984).
  - [24] O. Vendrell and H.-D. Meyer, *J. Chem. Phys.* **134**, 044135 (2011).
  - [25] J. Moser, *Stable and Random Motions in Dynamical Systems: With Special Emphasis on Celestial Mechanics*, Annals of Mathematics Studies (Princeton University Press, Princeton, NJ, 1973).
  - [26] R. Mackay, J. Meiss, and I. Percival, *Phys. D (Amsterdam, Neth.)* **13**, 55 (1984).
  - [27] G. Cybenko, *Math. Control Signals Syst.* **2**, 303 (1989).
  - [28] K. Hornik, *Neural Networks* **4**, 251 (1991).
  - [29] H. Jaeger and H. Haas, *Science* **304**, 78 (2004).
  - [30] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, *Phys. Rev. Lett.* **120**, 024102 (2018).
  - [31] T. L. Carroll and L. M. Pecora, *Chaos* **29**, 083130 (2019).

- [32] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing* (Cambridge University Press, New York, 2007).
- [33] S. H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*, 2nd ed. (Westview Press, Boulder, CO, 2015).
- [34] A. V. Melnikov and I. I. Shevchenko, in *Order and Chaos in Stellar and Planetary Systems* (Astronomical Society of the Pacific, San Francisco, CA, 2004), Vol. 316, p. 34.
- [35] C. Skokos, *J. Phys. A: Math. Gen.* **34**, 10029 (2001).
- [36] C. Wang, H. Zhai, and Y.-Z. You, *Sci. Bull.* **64**, 1228 (2019).
- [37] A. Kamor, F. Mauger, C. Chandre, and T. Uzer, *Phys. Rev. E* **85**, 016204 (2012).
- [38] S. Koyanagi, T. Nakano, and T. Kawabe, *J. Acoust. Soc. Am.* **124**, 719 (2008).
- [39] J. U. Nöckel and A. D. Stone, *Nature (London)* **385**, 45 (1997).
- [40] H.-J. Stöckmann and J. Stein, *Phys. Rev. Lett.* **64**, 2215 (1990).
- [41] L. A. Ponomarenko, F. Schedin, M. I. Katsnelson, R. Yang, E. W. Hill, K. S. Novoselov, and A. K. Geim, *Science* **320**, 356 (2008).
- [42] S. Chen, Z. Han, M. M. Elahi, K. M. Habib, L. Wang, B. Wen, Y. Gao, T. Taniguchi, K. Watanabe, J. Hone *et al.*, *Science* **353**, 1522 (2016).
- [43] M. V. Berry, *Proc. R. Soc. London, Ser. A* **400**, 229 (1985).
- [44] M. C. Gutzwiller, *J. Math. Phys.* **12**, 343 (1971).
- [45] D. Biswas and S. Sinha, *Phys. Rev. Lett.* **71**, 3790 (1993).
- [46] D. Biswas and S. Sinha, *Phys. Rev. Lett.* **70**, 916 (1993).
- [47] J. B. Tenenbaum, V. De Silva, and J. C. Langford, *Science* **290**, 2319 (2000).
- [48] R. Laje and D. V. Buonomano, *Nat. Neurosci.* **16**, 925 (2013).
- [49] M. Schmidt and H. Lipson, *Science* **324**, 81 (2009).