Extended Abstract Track

Hamiltonian Matching for Symplectic Neural Integrators

Editors: List of editors' names

Abstract

Hamilton's equations of motion form a fundamental framework in various branches of physics, including astronomy, quantum mechanics, particle physics, and climate science. Classical numerical solvers are typically employed to compute the time evolution of these systems. However, when the system spans multiple spatial and temporal scales numerical errors can accumulate, leading to reduced accuracy. To address the challenges of evolving such systems over long timescales, we propose SympFlow, a novel neural network-based symplectic integrator, which is the composition of a sequence of exact flow maps of parametrised time-dependent Hamiltonian functions. This architecture allows for a backward error analysis: we can identify an underlying Hamiltonian function of the architecture and use it to define a *Hamiltonian matching* objective function, which we use for training. In numerical experiments, we show that SympFlow exhibits promising results, with qualitative energy conservation behaviour similar to that of time-stepping symplectic integrators.

Keywords: Hamiltonian systems, backward error analysis, physics-informed machine learning, scientific machine learning.

1. Introduction

In this work, we will study the use of neural networks to integrate Hamiltonian systems, which were first defined in the context of classical mechanics and which have since found many applications in physics (Arnold, 1978). More specifically, when we speak of Hamiltonian systems, we are considering ordinary differential equations (ODEs) of the following form for a state variable $x \in \mathbf{R}^{2d}$ and *Hamiltonian* function $H : \mathbf{R}^{2d} \to \mathbf{R}$:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \mathbf{J}\nabla H(x), \quad \text{with} \quad \mathbf{J} = \begin{pmatrix} 0 & \mathrm{id}_d \\ -\mathrm{id}_d & 0 \end{pmatrix}. \tag{1}$$

Typically, the state variable is partitioned into a position $q \in \mathbf{R}^d$ and momentum $p \in \mathbf{R}^d$. Under standard (non-restrictive) assumptions on H (Arnold, 1991), the corresponding initial value problem has a unique solution for any initial condition and initial time, which can be used to define the corresponding *flow map* $\phi_H : \mathbf{R} \times \mathbf{R}^{2d} \to \mathbf{R}^{2d}$ by

$$\frac{\mathrm{d}}{\mathrm{d}t}\phi_{H,t}(x) = \mathbf{J}\nabla H(\phi_{H,t}(x)) \quad \text{and} \quad \phi_{H,0}(x) = x. \tag{2}$$

Since the exact flow map in eq. (2) is generally not accessible, it is necessary to make a "satisfactory" approximation to such an exact flow map. Depending on the context, this can have different meanings, but for Hamiltonian systems, there are important structural properties that can provide guidance: the flow map is *symplectic*, meaning that the Jacobian matrix $D\phi_{H,t}(x)$ satisfies the identity $[D\phi_{H,t}(x)]^\top \mathbf{J}[D\phi_{H,t}(x)] = \mathbf{J}$, and the Hamiltonian H is conserved, i.e. $H(\phi_{H,t}(x)) = H(x)$. Symplecticity also implies that volumes in phase space are preserved. When numerically approximating the flow map, it is desirable to take

these structural properties into account, and doing so has led to the celebrated field of *geometric numerical integration* (Hairer et al., 2006).

The neural network architecture that we propose takes inspiration from the methods of geometric numerical integration to define a time-dependent symplectic neural network architecture, SympFlow, that can be used to approximate Hamiltonian flow maps. There are similarities between SympFlow and previous works on symplectic neural networks (Jin et al., 2020; Burby et al., 2021), but SympFlow differs from these approaches by incorporating a time dependence which allows an underlying Hamiltonian to be identified.

In this work, we study SympFlow from the perspective of physics-informed machine learning (Karniadakis et al., 2021): we design an unsupervised learning approach for approximating Hamiltonian flow maps. It is in principle also possible to go in the other direction, fitting the approximate flow map to trajectories, and extracting the underlying Hamiltonian to discover a physical model, a task which has previously been studied in the scientific machine learning literature (Bertalan et al., 2019; Greydanus et al., 2019).

2. Methodology

2.1. The SympFlow architecture

Fundamentally, SympFlow is defined by iterated composition of exact flow maps of time-dependent Hamiltonians, each of which depends either on position or momentum, but not both. Given an arbitrary C^2 function $V_\mathrm{p} : \mathbf{R} \times \mathbf{R}^d \to \mathbf{R}$, we can consider the map

$$\phi_{\mathrm{p},t}((q,p)) = \begin{pmatrix} q \\ p - (\nabla_q V_\mathrm{p}(t,q) - \nabla_q V_\mathrm{p}(0,q)) \end{pmatrix}, \tag{3}$$

which is the flow map (starting from time 0) corresponding to the Hamiltonian $H_{\mathrm{p},t}((q,p)) = \dot{V}_\mathrm{p}(t,q)$, where $\dot{V}_\mathrm{p}$ stands for $\frac{\mathrm{d}}{\mathrm{d}t}V_\mathrm{p}$ and the subscript p indicates that the Hamiltonian depends on position, but not momentum. Similarly, for a C^2 function $V_\mathrm{m} : \mathbf{R} \times \mathbf{R}^d \to \mathbf{R}$, we can consider the map

$$\phi_{\mathrm{m},t}((q,p)) = \begin{pmatrix} q + (\nabla_p V_\mathrm{m}(t,p) - \nabla_p V_\mathrm{m}(0,p)) \\ p \end{pmatrix}, \tag{4}$$

which is the flow map (starting from time 0) corresponding to the Hamiltonian $H_{\mathrm{m},t}((q,p)) = \dot{V}_\mathrm{m}(t,p)$. As above, the subscript m indicates that the Hamiltonian depends on momentum but not position. Although the Hamiltonians above take a very particular form, they naturally arise when applying splitting integration methods to separable Hamiltonians. By parametrising V_p and V_m as multi-layer perceptrons (MLPs), say, and composing such steps, we get a time-dependent symplectic map, the parameters of which can be optimised to fit data or, more generally, minimise an objective function.

2.2. The Hamiltonian of the SympFlow architecture

As we will see in Proposition 1, we can find a time-dependent Hamiltonian function corresponding to the SympFlow architecture. This allows us to essentially perform a *backward error analysis*: while SympFlow does not generally solve the true ODE under consideration, it solves a time-dependent Hamiltonian ODE, the Hamiltonian of which we can give an expression for. We can apply the following result for this purpose:

Proposition 1 (Proposition 1.4.D from Polterovich (2001)) *Let $H^1, H^2 : \mathbf{R}\times\mathbf{R}^{2d} \to \mathbf{R}$ be continuously differentiable functions, and $\phi_{H_t^1,t}, \phi_{H_t^2,t} : \mathbf{R}^{2d} \to \mathbf{R}^{2d}$ the exact flows (starting from time 0) of the Hamiltonian systems they define. Then, the map $\psi_t = \phi_{H_t^2,t} \circ \phi_{H_t^1,t} : \mathbf{R}^{2d} \to \mathbf{R}^{2d}$ is the exact flow of the time-dependent Hamiltonian system defined by the Hamiltonian function*

$$H_t^3(x) = H_t^2(x) + H_t^1\left(\phi_{H_t^2,t}^{-1}(x)\right).$$

As a result, given an overall SympFlow of the following form (with $\phi_{\mathrm{m},t}^i$ taking the form of eq. (4) for some V_m^i and $\phi_{\mathrm{p},t}^i$ taking the form of eq. (3) for some V_p^i)

$$\bar{\psi}_t = \phi_{\mathrm{m},t}^L \circ \phi_{\mathrm{p},t}^L \circ \ldots \circ \phi_{\mathrm{m},t}^1 \circ \phi_{\mathrm{p},t}^1, \tag{5}$$

we can associate the SympFlow with a time-dependent Hamiltonian. To denote this Hamiltonian, we introduce the operator $\mathcal{H}$ sending a SympFlow into one of its generating Hamiltonian functions (all of which differ just by a constant), so one has $\mathcal{H}(\bar{\psi}) : \mathbf{R} \times \mathbf{R}^{2d} \to \mathbf{R}$.

To assemble such a function, we can group the pairs of alternated momentum and position flows, finding the Hamiltonian associated with $\phi_{\mathrm{m},t}^i \circ \phi_{\mathrm{p},t}^i$, which is $H_t^i((q,p)) = \dot{V}_\mathrm{m}^i(t,p) + \dot{V}_\mathrm{p}^i(t, q - (\nabla_p V_\mathrm{m}^i(t,p) - \nabla_p V_\mathrm{m}^i(0,p)))$. The Hamiltonian $\mathcal{H}(\bar{\psi})$ can then be expressed iteratively, aggregating from last layer to first as

$$H_t^{L:i}(x) = H_t^{L:(i+1)}(x) + H_t^i\left(\phi_{H_t^{L:(i+1)},t}^{-1}(x)\right), \; i = 1, \ldots, L-1,$$

where $H_t^{L:L} = H_t^L$, and

$$\phi_{H_t^{L:i},t}^{-1} = \left(\phi_{H_t^{L:(i+1)},t} \circ \phi_{H_t^i,t}\right)^{-1} = \phi_{H_t^i,t}^{-1} \circ \phi_{H_t^{L:(i+1)},t}^{-1}.$$

The Hamiltonian of the network $\mathcal{H}(\bar{\psi})$ then corresponds to $H_t^{L:1}$.

2.3. Training SympFlow: Flow learning with Hamiltonian matching

The natural task to use the SympFlow architecture for is to approximate the flow map of a Hamiltonian system as in eq. (1). We will assume the existence of a compact subset $\Omega \subset \mathbf{R}^{2d}$ which is forward invariant, meaning that $\phi_{H,t}(\Omega) \subseteq \Omega$ for every $t \geq 0$.

To train SympFlow, we consider a loss function composed of two terms. The first is the usual physics-informed (PI) loss function (Raissi et al., 2019), based on the residual of eq. (2),

$$\mathcal{L}_1(\bar{\psi}) = \frac{1}{N}\sum_{i=1}^N \left\| \frac{\mathrm{d}}{\mathrm{d}t}\bar{\psi}_{t_i}(x_0^i) - \mathbf{J}\nabla H(\bar{\psi}_{t_i}(x_0^i)) \right\|_2^2,$$

where $x_0^i \in \Omega$ and $t_i \in [0, \Delta t]$ for every $i = 1, \ldots, N$. Unlike classical numerical integrators, Δt can be chosen to be large. Using the analysis in the previous section, it is also possible to extract the underlying Hamiltonian of a given instance of SympFlow. This gives rise to a natural training objective, which we call the *Hamiltonian matching loss* and which constitutes the second term in our loss, defined as

$$\mathcal{L}_2(\bar{\psi}) = \frac{1}{M}\sum_{i=1}^M \left(\mathcal{H}(\bar{\psi})(t_i, x^i) - H(x^i)\right)^2,$$

where $x^i \in \Omega$ and $t_i \in [0, \Delta t]$ for every $i = 1, \ldots, N$. The loss function we optimise over the space of SympFlows, is then $\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2$, and we use PyTorch (Paszke et al., 2019) to get its gradient for use in the Adam optimiser (Kingma and Ba, 2017).

Given a trained SympFlow network, representing a function $\bar{\psi} : [0, \Delta t] \times \mathbf{R}^{2d} \to \mathbf{R}^{2d}$, we can apply it to longer time horizons as follows: extend $\bar{\psi}$ to $[0, +\infty)$ via $\psi : [0, +\infty) \times \mathbf{R}^{2d} \to \mathbf{R}^{2d}$ defined as

$$\psi_t(x_0) := \bar{\psi}_{t - \Delta t \lfloor t/\Delta t \rfloor} \circ \left(\bar{\psi}_{\Delta t}\right)^{\lfloor t/\Delta t \rfloor} (x_0), \tag{6}$$

which can be considered an approximation of $\phi_{H,t}(x_0)$ for every $t \geq 0$.

3. Experiments

We consider two Hamiltonian systems: the harmonic oscillator, with $H(q, p) = (q^2 + p^2)/2$, and the Hénon-Heiles system, with $H(q_1, q_2, p_1, p_2) = (p_1^2 + p_2^2)/2 + (q_1^2 + q_2^2)/2 + q_1^2 q_2 - q_2^3/3$. The Hénon-Heiles system is notable for exhibiting chaos. We train SympFlow and a comparable MLP (using only physics-informed loss), on an interval with $\Delta t = 1$. We also compare to an adaptive integrator, ODE45, as implemented in SciPy (with the default tolerances) (Virtanen et al., 2020). More results are shown in appendix A, but note in particular the good long-time energy behaviour of SympFlow, compared to the other methods:

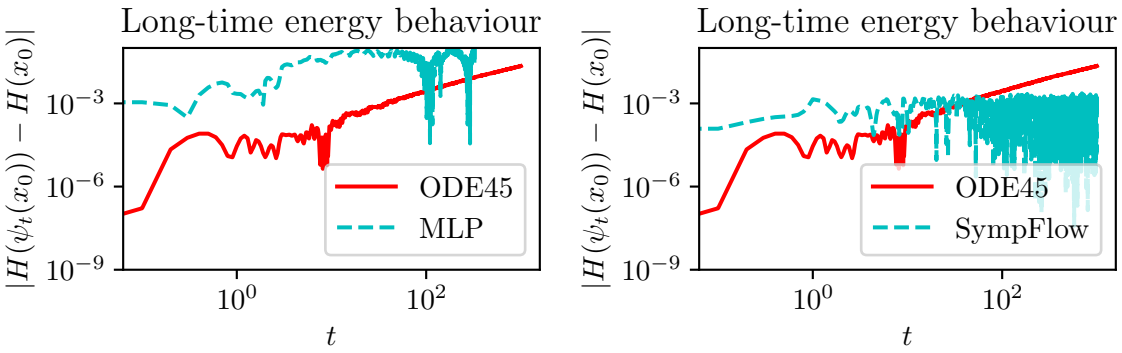

Figure 1: A comparison of the long-time energy conservation of SympFlow and an unconstrained neural flow map approximator on the Hénon-Heiles system.

4. Conclusions and discussion

We have given a demonstration of the SympFlow architecture and its application to integrating Hamiltonian systems. As shown in our experiments in section 3, SympFlow exhibits good long-time conservation of energy, as is common for classical time-stepping symplectic numerical integrators as well. There is a large body of theoretical research on such properties for time-stepping integrators, which may be leveraged in future research to theoretically support the behaviour that we have observed for SympFlow. The setting of Hamiltonian systems is not as restrictive as it may appear at a first glance: we have only considered ODEs here, but extensions are possible to PDEs (Bridges and Reich, 2006), and even to non-conservative systems Galley (2013); Galley et al. (2014); Tsang et al. (2015). Studying such extensions in more detail is a promising avenue for future research.

References

V. I. Arnold. *Mathematical Methods of Classical Mechanics*, volume 60 of *Graduate Texts in Mathematics*. Springer New York, New York, NY, 1978. ISBN 978-1-4757-1695-5 978-1-4757-1693-1. doi: 10.1007/978-1-4757-1693-1.

V. I. Arnold. *Ordinary Differential Equations*. MIT Press, Cambridge, Mass., 8. print edition, 1991. ISBN 978-0-262-51018-9 978-0-262-01037-5.

Tom Bertalan, Felix Dietrich, Igor Mezić, and Ioannis G. Kevrekidis. On learning Hamiltonian systems from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29 (12):121107, December 2019. ISSN 1054-1500, 1089-7682. doi: 10.1063/1.5128231.

Thomas J Bridges and Sebastian Reich. Numerical methods for Hamiltonian PDEs. *Journal of Physics A: Mathematical and General*, 39(19):5287–5320, May 2006. ISSN 0305-4470, 1361-6447. doi: 10.1088/0305-4470/39/19/S02.

J. W. Burby, Q. Tang, and R. Maulik. Fast neural Poincaré maps for toroidal magnetic fields. *Plasma Physics and Controlled Fusion*, 63(2):024001, 2021. ISSN 0741-3335, 1361-6587. doi: 10.1088/1361-6587/abcbaa.

Chad R. Galley. Classical mechanics of nonconservative systems. *Physical Review Letters*, 110(17), 2013. doi: 10.1103/physrevlett.110.174301. URL https://doi.org/10.1103%2Fphysrevlett.110.174301.

Chad R. Galley, David Tsang, and Leo C. Stein. The principle of stationary nonconservative action for classical mechanics and field theories, 2014. URL https://arxiv.org/abs/1412.3082.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/26cd8ecadce0d4efd6cc8a8725cbd1f8-Paper.pdf.

Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Number 31 in Springer Series in Computational Mathematics. Springer, Berlin ; New York, 2nd ed edition, 2006. ISBN 978-3-540-30663-4.

Pengzhan Jin, Zhen Zhang, Aiqing Zhu, Yifa Tang, and George Em Karniadakis. SympNets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. *Neural Networks*, 132:166–179, 2020. ISSN 0893-6080. doi: 10.1016/j.neunet.2020.08.017.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, June 2021. ISSN 2522-5820. doi: 10.1038/s42254-021-00314-5.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037, 2019.

Leonid Polterovich. *The Geometry of the Group of Symplectic Diffeomorphisms*. Lectures in Mathematics ETH Zürich. Springer Basel AG, Basel, 2001. ISBN 978-3-7643-6432-8.

M Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.045.

David Tsang, Chad R. Galley, Leo C. Stein, and Alec Turner. "Slimplectic" Integrators: Variational Integrators for General Nonconservative Systems. *The Astrophysical Journal*, 809(1):L9, 2015. ISSN 2041-8213. doi: 10.1088/2041-8205/809/1/l9.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Appendix A. Additional experimental results

A.1. Harmonic oscillator

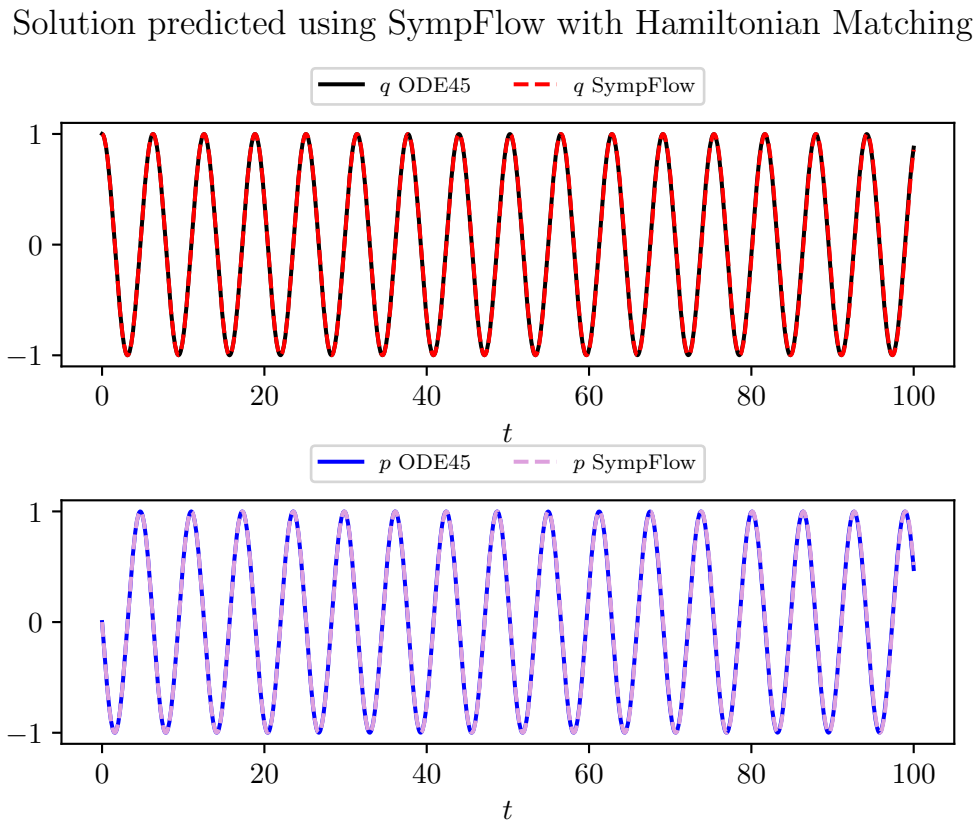

Figure 2: A trajectory of the harmonic oscillator predicted by SympFlow and ODE45.

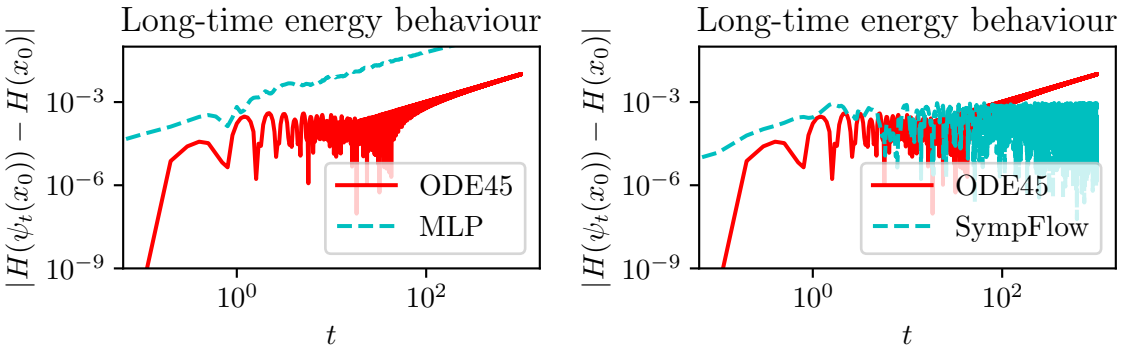

Figure 3: A comparison of the long-time energy conservation of SympFlow and an unconstrained neural flow map approximator on the harmonic oscillator.

A.2. Hénon-Heiles system

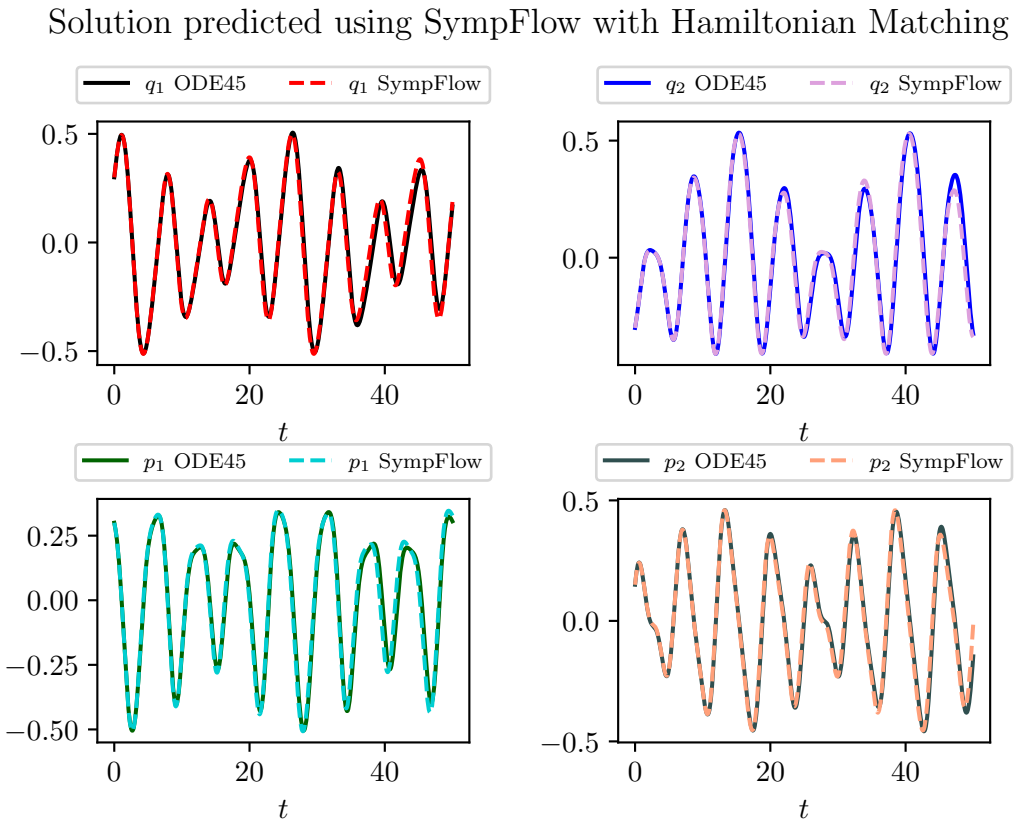

Figure 4: A trajectory of the Hénon-Heiles system predicted by SympFlow and ODE45.