Bayesian Identification of Hamiltonian Dynamics from Symplectic Data

Nicholas Galioto and Alex A. Gorodetsky

***Abstract*— We propose a Bayesian probabilistic formulation for system identification of Hamiltonian systems. This approach uses an approximate marginal Markov Chain Monte Carlo algorithm to directly discover a system Hamiltonian from data. Our approach improves upon existing methods in two ways: first we encode the fact that the data generating process is symplectic directly into our learning objective, and second we utilize a learning objective that simultaneously accounts for unknown parameters, model form, and measurement noise. This objective is the log marginal posterior of a probabilistic model that embeds a symplectic and reversible integrator within an uncertain dynamical system. We demonstrate that the resulting learning problem yields dynamical systems that have improved accuracy and reduced predictive uncertainty compared to existing state-of-the-art approaches. Simulation results are shown on the Hénon-Heiles Hamiltonian system.**

I. Introduction

We consider system identification where we seek to learn a system when the data generating process is known to follow certain conservation laws. Within the context of this problem we seek to address two primary concerns: (1) how do we endow the learning process with knowledge that the underlying system conserves energy, and (2) how can we enable the learning process to be robust with respect to measurement, model, and parameter uncertainty. Specifically, we focus on how to set up the optimization objective, or in our case the probabilistic model, for the process that solves the above two problems.

A wide variety of recent work has focused on the same problem where a physics-informed identification of dynamical systems is sought by prioritizing adherence to physical laws and/or phenomena such as stability [1], [2], [3], [4], conservation of energy [5], [6], [7], the principle of least action [8], [9], [10], [11], and symmetries in the dynamics [12], [10], [11]. These methods all follow a common approach in setting up their learning problem.

Specifically, they set up a least squares objective function that learns a propagator by placing a penalty on the gradient of the Hamiltonian. For example in [6], the authors learn a Hamiltonian system (denoted a Hamiltonian neural network) by minimizing the squared difference between the derivative of the parameterized Hamiltonian and the numerical derivatives of the states

$$J(\theta) = \sum_{i=1}^{N} \left\| \frac{q_i - q_{i-1}}{\Delta t} - \frac{\partial \mathcal{H}_\theta}{\partial p} \right\|^2 + \left\| \frac{p_i - p_{i-1}}{\Delta t} + \frac{\partial \mathcal{H}_\theta}{\partial q} \right\|^2, \tag{1}$$

N. Galioto and A.A. Gorodetsky are with the Department of Aerospace Engineering, University of Michigan, Ann Arbor. {ngalioto,goroda}@umich.edu.

where q is the position, p the momentum, $\mathcal{H}_\theta$ the Hamiltonian dependent on parameters θ, and N the number of measurements. The authors used finite difference approximations indicated above to approximate true time derivatives. With this approximation, the objective function is equivalent to the sum of the squared difference between the measurements and the integrated values of the parameterized gradients using a forward Euler integration scheme.

Other works have sought to improve this objective function by using a more accurate integration scheme. In [13], the authors use 4th-order Runge-Kutta integration, but this introduces additional cost and can still stray from the manifold over a long enough time. In [14], the authors use the symplectic leapfrog integrator and show improved performance over a forward Euler integrator; however, this comparison does not distinguish between gains due to symplecticness and gains due to the second-order accuracy of leapfrog. Furthermore, neither of these approaches optimally manage the uncertainty in the learning problem that enters through the selection of model structure, the noise on the data, and the uncertainty on the inferred parameters. As a result, these approaches require thousands of data points to train. In this paper, we extend a previously developed method [15] derived from the first principles of probability to optimally manage uncertainty to the physics-informed objective 1. We show that while this method exhibits strong performance with very little data, using a symplectic integrator can improve its performance further over using a non-symplectic integrator of the same order of integration accuracy. Thus the contributions of this work are the following

1) tailor the method of [15] to this physics-informed context by embedding a symplectic integrator within the objective
2) provide a direct comparison to an objective with a non-symplectic integrator of equivalent order accuracy.

The rest of the paper is structured as follows: Section II gives some preliminary background on Hamiltonian mechanics and symplectic integrators and probabilistically formulates the system identification problem. Section III outlines our proposed algorithm for learning Hamiltonian systems and includes a discussion on why it is more effective than existing approaches. Section IV compares the proposed algorithm to one that is not equipped with a symplectic integrator on the Hénon-Heiles system. Lastly, Section V summarizes the findings and key points from this paper.

II. Problem definition

In this section, we review relevant mechanics and our probabilistic modeling framework.

A. Hamiltonian mechanics and symplectic integration

Here we briefly review Hamiltonian systems and their integration.

1) Hamilton's equations: For a mechanical system, the Hamiltonian is defined as the total energy in the system and is a function of the generalized position q, generalized momentum p, and possibly time t. Because we are interested here in conservative systems, the Hamiltonian will be independent of time and take the form

$$\mathcal{H}(q,p) = \frac{1}{2}p^T M^{-1}(q)p + U(q,p), \tag{2}$$

where M is the inertia matrix and U the potential energy. Time derivatives of the position and momentum can be derived from the Hamiltonian according to Hamilton's equations

$$\dot{q} = \frac{\partial \mathcal{H}}{\partial p} \qquad \dot{p} = -\frac{\partial \mathcal{H}}{\partial q}. \tag{3}$$

Hamiltonian systems possess the following properties: conservation, reversibility, and symplecticness [16].

2) Model structure and symplectic integrators: We propose to learn discrete-time propagators and to ensure that they conserve the aforementioned physical properties. As the name suggests, reversible symplectic integrators preserve reversibility and symplectiness and will serve as one basis of our learning approach. A multitude of symplectic integration schemes exist, and the selection of one generally depends on the structure of the model to be integrated. We focus on a basic model structure that can be integrated with the particularly simple leapfrog integration method, which enables us to focus on the contributions of our proposed learning approach. However, as long as a compatible symplectic integration scheme exists for one's model structure, restrictions that we place on our model can be ammended with similar results.

The model structure we use here assumes an autonomous system, a constant and known inertia matrix, and a potential energy function independent of the momentum. Without loss of generality, we set the inertia matrix to be the identity. These assumptions admit Hamiltonians of the form

$$\mathcal{H}(q,p) = \frac{1}{2}p^T p + U(q). \tag{4}$$

Hamiltonians of this form are separable and therefore compatible with leapfrog integration. The leapfrog method is traditionally defined in half-step intervals for the momentum, but here we provide the equations at integer steps, which is more useful to our setting:

$$q^{k+1} = q^k + \Delta t p^k - \frac{\Delta t^2}{2}\left.\frac{\partial U}{\partial q}\right|_{q^k}, \tag{5}$$

$$p^{k+1} = p^k - \frac{\Delta t}{2}\left(\left.\frac{\partial U}{\partial q}\right|_{q^k} + \left.\frac{\partial U}{\partial q}\right|_{q^{k+1}}\right). \tag{6}$$

These update equations form a symplectic integrator that is also reversible [17]. As a result, they ensure that an approximate (there is an error due to discretization) Hamiltonian is conserved along the the trajectory of the dynamics.

B. Probabilistic learning of dynamical systems

In this section, we describe a hidden Markov model on which we will base our learning approach. Here we follow the notation of our recently introduced analysis [15].

1) Probabilistic model: Let $\mathbb{R}$ denote the set of reals and $\mathbb{Z}_+$ denote the set of positive integers. Let $d_x \in \mathbb{Z}_+$ denote the size of a state space, $d_y \in \mathbb{Z}_+$ denote the size of an observation space, and $k \in \mathbb{Z}_+$ denote a time index corresponding to a time $t_k \geq 0$. Sequential time indices will typically occur with a constant interval Δt so that $t_k = t_{k-1} + \Delta t$. A discrete-time dynamical system evolves hidden states $X_k \in \mathbb{R}^{d_x}$ at these time intervals. The hidden states are observed through a noisy and potentially indirect measurement operator providing us data $y_k \in \mathbb{R}^{d_y}$. These data are viewed as realizations of a state-dependent stochastic process Y_k.

All unknown parameters are denoted by $\theta \in \mathbb{R}^{d_\theta}$ for $d_\theta \in \mathbb{Z}_+$. These define a search space over models. These parameters are partitioned $\theta = (\theta_\Psi, \theta_h, \theta_\Sigma, \theta_\Gamma)$ across several uncertain sources including dynamics model parameters θ_Ψ, observation model parameters θ_h, process noise parameters θ_Σ, and observation noise parameters θ_Γ. Together these states, observations, and parameters are related through the following discrete-time stochastic process [18]

$$\begin{aligned} X_k &= \Psi(X_{k-1}, \theta_\Psi) + \xi_k; \quad & \xi_k &\sim \mathcal{N}(0, \Sigma(\theta_\Sigma)) \\ Y_k &= h(X_k, \theta_h) + \eta_k; & \eta_k &\sim \mathcal{N}(0, \Gamma(\theta_\Gamma)), \end{aligned} \tag{7}$$

for $k = 1, \ldots, n$ where $\mathcal{N}$ denotes a normal distribution, $\Psi : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_x}$ is the dynamics operator, ξ_k is the process noise with uncertain covariance $\Sigma(\theta_\Sigma)$, $h : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_y}$ is the observation/measurement operator, Y_k is the predictive stochastic process for the observable, and η_k is the observation noise with uncertain covariance $\Gamma(\theta_\Gamma)$. Although we have added noise to the state, the underlying system that we are trying to learn is still deterministic. This noise represents the error introduced from numerical integration and from using an incorrect/imperfect model.

In the context of this paper, we would like to use the theory of Hamiltonian systems and symplectic integrators described previously in subsection II-A to inform the design of the propagator Ψ. The propagator design that we have selected is described in detail in Section III.

2) System identification goals: Our system identification goal is to compute the marginal posterior $\pi(\theta \mid \mathcal{Y}_n)$, which is given by Bayes rule as

$$\pi(\theta \mid \mathcal{Y}_n) = \pi(\theta)\frac{\mathcal{L}(\theta; \mathcal{Y}_n)}{\pi(\mathcal{Y}_n)}, \qquad \text{where } \mathcal{Y}_n = (y_1, \ldots, y_n), \tag{8}$$

where $\pi(\theta)$ is the prior, and the marginal likelihood is a function of the unknown parameter

$$\mathcal{L}(\theta; \mathcal{Y}_n) \equiv \pi(\mathcal{Y}_n \mid \theta). \tag{9}$$

The challenge is that only the joint likelihood $\mathcal{L}(\theta; \mathcal{X}_n, \mathcal{Y}_n) \equiv \pi(\mathcal{X}_n, \mathcal{Y}_n \mid \theta)$ is available for us to compute where $\mathcal{X}_n = (x_0, \ldots, x_n)$.

To obtain the marginal likelihood, we must compute the integral $\mathcal{L}(\theta;\mathcal{Y}_n) = \int \mathcal{L}(\theta;\mathcal{X}_n,\mathcal{Y}_n)d\mathcal{X}_n$ which is in general intractable. However, by approximating certain distributions as Gaussian, this integral can be approximated with the recursive algorithm given in Section II-B.3.

In the context of optimization, a point estimate would be found by optimizing the log posterior to obtain a maximum *a-posterior* (MAP) estimate. However, in the Bayesian approach we sample from the posterior to account for the full uncertainty. Unlike the least squares objectives described in the introduction, this posterior accounts for parameter, model, and measurement uncertainty.

To make future predictions, in this work, we select the MAP point of the parameter posterior $\pi(\theta|\mathcal{Y}_n)$ and compute $X_k = \Psi^k(X_0,\theta_\Psi)$, where Ψ^k refers to k applications of the dynamics. Alternatively, one may also use a point selected from the predictive posterior obtained by weighting the dynamics with respect to the dynamics model parameters $\pi(X_k \mid \mathcal{Y}_n) = \int \pi(X_k \mid \theta)\pi(\theta \mid \mathcal{Y}_n)d\theta$.

3) Computational algorithm: The Bayesian inference problem (8) is solved using an approximate marginal delayed-rejection Markov Chain Monte Carlo method called UKF-MCMC [19]. This method involves using an unscented Kalman filter (UKF) to compute an approximation to the marginal likelihood. This marginal likelihood is represented recursively and given by the Th. 12.1 [18].

Theorem 1 (Marginal likelihood (Th. 12.1 [18])): Let $\mathcal{Y}_k \equiv \{y_i; i \leq k\}$ denote the set of all observations up to time k. Let the initial condition be uncertain with distribution $\pi(X_0 \mid \theta)$. Then the marginal likelihood (9) is defined recursively in three stages: prediction

$$\pi(X_{k+1} \mid \theta, \mathcal{Y}_k) = \sqrt{2\pi}^{-d_x}|\Sigma(\theta_\Sigma)|^{-\frac{1}{2}} \times \int \exp\left(-\frac{1}{2}\|X_{k+1} - \Psi(X_k,\theta_\Psi)\|^2_{\Sigma(\theta_\Sigma)}\right)\pi(X_k \mid \theta, \mathcal{Y}_k)dX_k \tag{10}$$

update,

$$\pi\left(X_{k+1} \mid \theta, \mathcal{Y}_{k+1}\right) = \pi(X_{k+1} \mid \theta, \mathcal{Y}_k)\times \frac{\exp\left(-\frac{1}{2}\|Y_{k+1} - h(X_{k+1},\theta_h)\|^2_{\Gamma(\theta_\Gamma)}\right)}{\sqrt{2\pi}^{d_y}|\Gamma(\theta_\Gamma)|^{\frac{1}{2}}\pi(Y_{k+1} \mid \theta, \mathcal{Y}_k)} \tag{11}$$

and marginalization,

$$\mathcal{L}_{k+1}(\theta \mid \mathcal{Y}_{k+1}) = \sqrt{2\pi}^{-d_y}|\Gamma(\theta_\Gamma)|^{-\frac{1}{2}} \times \int \pi(X_{k+1} \mid \theta, \mathcal{Y}_k)\exp\left(-\frac{1}{2}\|Y_{k+1} - h(X_{k+1},\theta_h)\|^2_{\Gamma(\theta_\Gamma)}\right)dX_{k+1} \tag{12}$$

for $k = 1, 2, \ldots$.

The UKF-MCMC algorithm evaluates these prediction, update, and marginalization steps by running a UKF every time the marginal likelihood needs to be evaluated within the Metropolis-Hastings step of the the MCMC procedure. In [15] we showed that the asymptotic cost of each log posterior evaluation scales cubically in the number of states and observations, and linearly in the number of data points.

III. Probabilistic learning of Hamiltonian systems

In this section, we describe an algorithm that adapts the probabilistic approach of Section II-B to the context of Hamilton's equations. As previously mentioned, the motivation is to encode additional structure into the learning problem, when it is available, to both reduce data requirements and enforce physical constraints on the predictions.

Our goal is to design a propagator $\Psi(\theta_\Psi)$ used in the objective that conserves certain phenomena in our problem. We achieve this goal primarily through two methods: (1) parameterizing the Hamiltonian rather than the ODEs so that the form of all models in our search space conserves the Hamiltonian properties, and (2) embedding the propagator with a leapfrog integrator so as not to violate these properties during the numerical integration necessary for training.

A. Parameterizing the Hamiltonian

As we previously mentioned, we restrict ourselves in this work to considering Hamiltonians of the form $\mathcal{H}(q,p,\theta_\Psi) = \frac{1}{2}p^Tp + U(q,\theta_\Psi)$, and we assume the generalized coordinate frame is known. Therefore, the only parameters we must learn are those that parameterize the potential.

Many potentials can typically be well approximated by polynomial approximations, see Section IV. However, neural networks or other nonlinear approximation approaches can also be applied. Our goal is not to advocate for a specific type of parameterization, but rather to show how any approach can be embedded in our proposed approach. Here we parameterize the potential as $U(q,\theta_\Psi) = f_\ell(q;\theta_\Psi)$, where f is a polynomial up to order ℓ — terms whose exponent add to greater than ℓ are not included — and θ_Ψ are their coefficients. In this paper, we will not conisder polynomial orders beyond three and therefore, we will use a monomial basis. For higher dimensional problems, orthogonal polynomials can be used to improve conditioning [20].

The Hamiltonian now becomes

$$\mathcal{H}(q,p,\theta_\Psi) = \frac{1}{2}p^Tp + f_\ell(q;\theta_\Psi). \tag{13}$$

Differentiating the potential energy to obtain the negative generalized forces we obtain $\frac{\partial U(q,\theta_\Psi)}{\partial q_i} = \frac{\partial f_\ell(q;\theta_\Psi)}{\partial q_i}$.

B. Embedded learning within a symplectic propagator

In this section, we demonstrate how the uncertain potential enters into the probabilistic learning framework of Section II-B. The main idea is to endow Ψ with a structure that both embeds the uncertain Hamiltonian and reflects the conservative nature of the underlying system. Here we do this by combining the symplectic integrator equations (5) and (6) with the Hamiltonian parameterization (13) to obtain

$$\Psi(q_k,p_k;\theta_\Psi) = \begin{bmatrix} q_k + \Delta t p_k - \frac{\Delta t^2}{2}\left.\frac{\partial U(q,p,\theta_\Psi)}{\partial q}\right|_{q_k} \\ p_k - \frac{\Delta t}{2}\left(\left.\frac{\partial U(q,p,\theta_\Psi)}{\partial q}\right|_{q_k} + \left.\frac{\partial U(q,p,\theta_\Psi)}{\partial q}\right|_{q_{k+1}}\right)\end{bmatrix}, \tag{14}$$

where each time that Ψ needs to be evaluated, the derivative of $U(q,p,\theta_\Psi)$ must be evaluated at both the current and

the updated coordinate. Equation (14) essentially embeds the learning problem within a structure that implicitly enforces that the learned model will be conservative, symplectic, and reversible — thus consistent with the true process.

C. Discussion and analysis

In this section, we comment and analyze the effect of the structure of Equation (14) on the observed dynamics. First and foremost, this propagator is structurally designed so that it is symplectic and reversible. Specifically, the learning is embedded within a leapfrog integrator. This directly contrasts with similar approaches [21], [6], [7] that only focus on learning a Hamiltonian but do not actually account for data generated by a conservative process. In our setting the leapfrog integrator is symplectic and reversible, and therefore the propagator is also. This proves the following main result.

Theorem 2 (Symplectic and reversible): Consider the propagator Ψ parameterized according to Equation (14). Then the mapping $x_{k+1} = (q_{k+1}, p_{k+1}) = \Psi^k(q_0, p_0; \theta_\Psi) = \Psi(x_0; \theta_\Psi)$ is symplectic and reversible.

The ramifications of this result are that the objective function can more accurately assess the performance of the model because the propagator enforces phenomena that are also enforced in nature.

Next, it is interesting to consider how the chosen propagator affects the marginal likelihod calculation. Primarily, it enters through the prediction step Equation (10), where the probability of a future state X_{k+1} is the probability of its deviation from $\Psi(X_k, \theta_\Psi)$ weighted by the previous probability $\pi(X_k \mid \theta, \mathcal{Y}_k)$. The deviations from $\Psi(X_k, \theta_\Psi)$ are weighted by the process covariance $\Sigma(\theta_\Sigma)$. This process covariance represents model error; in our setting this model error is a combination of the integration error of the leapfrog integrator and the error of the potential function U.

In another black-box approach where some other integrator was chosen [15], there would be an additional source of error whereby the propagator would not respect the solution manifold. This source of error would grow with time, and therefore yield significant deviations causing an overestimation of the process noise covariance. In our approach, this source of deviation does not exist, and therefore we are able to learn a much smaller process noise when the model form $U(q, \theta)$ is appropriate. We show this feature in Section IV.

IV. Numerical examples

In this section, we provide numerical results for the canonical Hénon-Heiles Hamiltonian system to show the value of embedding physical knowledge about the data into the learning process. We will compare the approach in the previous section to a 'physics-ignorant' approach where the propagator is equipped with a non-symplectic integrator that does not conserve the Hamiltonian. This approach is akin to the derivative reconstruction approaches used by, for example, Hamiltonian neural networks [6]. We modify this physics-ignorant approach to use a classical second-order Runge-Kutta integrator [22], [23] to match our second-order accurate leapfrog. We will sometimes refer to these approaches as the symplectic and non-symplectic approaches. We also point out that one can learn a Hamiltonian with the physics-ignorant approach and then predict with it using a symplectic integrator. Indeed this is the approach that seems to be espoused by a majority of the literature mentioned in the introduction. Here we will show that this approach leads to poorer reconstruction of the Hamiltonian because it neglects the symplectic process actually generating the data.

For each of the numerical examples, we use an approximate marginal MCMC method with the delayed rejection adaptive Metropolis (DRAM) [24] algorithm with a $\gamma = 0.01$ covariance scaling factor for the second-tier proposal. We place Laplace priors on the dynamical parameters to encourage sparsity [25] and half-normal priors on the covariance parameters as recommended in [26]. Sampling begins at the MAP estimate found using MATLAB's `fmincon` function. The initial proposal covariance is set as the inverse Hessian of the negative log posterior evaluated at the MAP point, and MATLAB's `fminunc` function is used to calculate this covariance. To ensure proper convergence, each chain is run for 200,000 iterations, and the first 100,000 samples are discarded as a conservative burn-in period.

For generating the "true" data in each simulation, we use a leapfrog integrator with conservatively small timestep $\Delta t = 10^{-3}$ to minimize integration error. For learning, we use a large timestep of $\Delta t = 0.25$ to reduce computation time and demonstrate the robustness of our approach. In the results that follow, we will refer to the timestep $\Delta t = 0.25$ used for learning as the 'learning timestep,' and the timestep $\Delta t = 0.01$ used for prediction after learning as the 'fine timestep.'[1]

The Hénon and Heiles [27] potential is given as

$$U(q_1, q_2) = \frac{1}{2}q_1^2 + \frac{1}{2}q_2^2 + q_1^2 q_2 - \frac{1}{3}q_2^3. \tag{15}$$

After differentiation we obtain the momentum derivatives

$$\dot{p}_1 = -q_1 - 2q_1 q_2, \qquad \dot{p}_2 = -q_2 - q_1^2 + q_2^2. \tag{16}$$

For the initial condition, we select the point $(q_1, q_2, p_1, p_2)(0) = \left(\frac{1}{5}, \frac{-1}{4}, 0.3, -\frac{-1}{4}\right)$.

During learning, we parameterize the unknown potential with polynomials up to total order three such that Hamilton's equations for the momenta in the learning system are

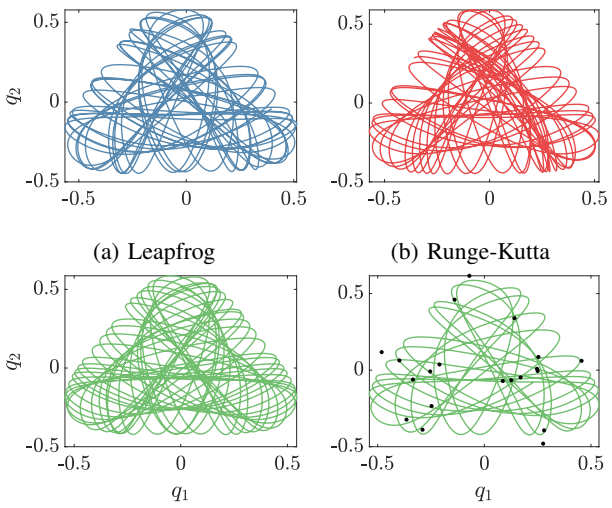
$$\dot{p}_1 = \theta_1 + 2\theta_3 q_1 + \theta_4 q_2 + 3\theta_6 q_1^2 + 2\theta_7 q_1 q_2 + \theta_8 q_2^2 \tag{17}$$

$$\dot{p}_2 = \theta_2 + \theta_4 q_1 + 2\theta_5 q_2 + \theta_7 q_1^2 + 2\theta_8 q_1 q_2 + 3\theta_9 q_2^2. \tag{18}$$

Note that these equations share parameters because they are derived from a single parameterized potential function. We also parameterize the process and measurement noise as $\Sigma(\theta) = \theta_{10} I_4$ and $\Gamma(\theta) = \theta_{11} I_4$. Finally, data are collected on the full state $n = 20$ times at regular intervals over 100 seconds with noise standard deviation of $\sigma = 0.05$.

The MAP points from each posterior are used to simulate the system for 200 seconds using the leapfrog integrator with the fine timestep, and the resulting phase plots are shown alongside the truth and the data points in Figure 1.

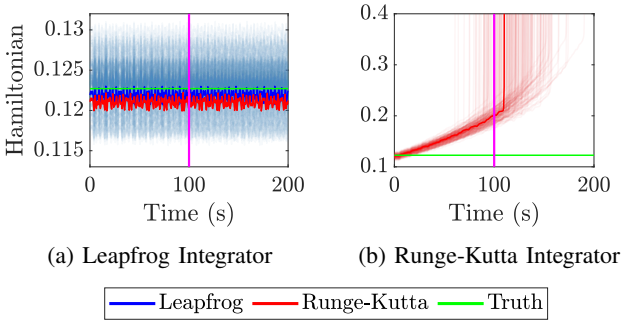
[1] After learning a Hamiltonian, any relevant numerical scheme can be used for prediction.



(a) Leapfrog (b) Runge-Kutta

(c) True (d) Position Data

Fig. 1: Phase diagrams of the Hénon-Heiles system. The top row shows the models learned by the approaches equipped with the leapfrog and Runge-Kutta integrators. All trajectories were integrated with the leapfrog method with the fine timestep for 200s, except for the bottom right, which was only integrated for the 100s period of data collection.



(a) Leapfrog Integrator (b) Runge-Kutta Integrator

Fig. 2: The learned Hénon-Heiles Hamiltonians. The lighter blue and red lines are samples from the posteriors from the symplectic and non-symplectic learning approaches. The dark lines represent the MAP of the respective parameter posterior, and the subcaptions denote the integrator used to integrate each solution post-learning, e.g. the two dark red lines are the same model integrated with different methods. The vertical magenta line indicates the end of the data collection period.

We see that both approaches learn that the system evolves on a manifold because the parameterization discussed in III restricts the model search space to Hamiltonian systems, each of which possesses a corresponding manifold.

From visual inspection, it first appears that the manifolds learned by both the symplectic and non-symplectic approaches are virtually identical. Figure 2a, however, reveals that the system learned by the non-symplectic approach exists on a lower energy manifold than that learned by the symplectic approach. The true Hamiltonian is 0.1227, and the Hamiltonian learned by the symplectic approach is 0.1219 and by the non-symplectic approach is 0.1211.

The explanation for this discrepancy is shown in Figure 2b, where we see that the Hamiltonian of the system grows very quickly when the Runge-Kutta integrator with the learning timestep is used. To compensate for this growth, the non-symplectic approach learns a lower energy system such that in the time it takes to integrate from one data point to the next, the energy added by the integrator has now pushed the Hamiltonian up to its true value. Overall, the Hamiltonian learned by the symplectic approach is closer to the truth than that learned by the non-symplectic approach.

In addition to accuracy, the second main benefit to using a symplectic integrator is a reduction of uncertainty in one's estimate. Figure 3 shows the estimates of the trajectory of the system's first state and their corresponding posteriors. For the sake of direct comparison, each trajectory is integrated using the leapfrog method with the fine timestep. In this figure, the region between the 2.5% and 97.5% quantiles of the posterior is shaded and 100 individual samples have been overlaid. The posterior from the non-symplectic approach grows much more quickly than the posterior from the symplectic approach as the system evolves, reflecting greater uncertainty in the estimate of the non-symplectic approach.

To understand this discrepancy in the variance of the estimates, we turn to the shape of the log posterior. The inability of the Runge-Kutta integrator to preserve the Hamiltonian of the system introduces error into the system learning process not present in the process using the leapfrog integrator. This additional error increases the process noise of the system, shown in Figure 4, and, as discussed in [15], this order of magnitude difference in the process noise can introduce significant bias. Indeed, we see in Figure 3 that the MAP estimate from the symplectic approach aligns closely with the truth for about 150 seconds, while the non-symplectic approach can only last about 20 seconds.

V. Conclusion

We have developed a learning approach for system identification that (1) enforces natural phenomena in the data generation process and (2) is robust to uncertainties in the models, parameters, and measurements. This approach extended the work in [15] by explicitly embedding the dynamical model within a symplectic integrator. We then showed that when learning Hamiltonian systems, this modification of the algorithm can yield two main benefits: greater accuracy and greater certainty. The increase in accuracy is a direct result of the conservation of the Hamiltonian achieved by the symplectic integrators. We also showed that using a symplectic integrator reduced the variance in the posterior predictive estimates of system behavior.

Future work will seek to utilize more stable and accurate numerical integrators as well as develop an expansion to general variational integrators on manifolds. We envision that these methods can be widely applicable to system identification problems where the underlying systems are mechnical in nature. For instance, identifying the kinematics and dynamics of robotic systems or the orbits of satellites.

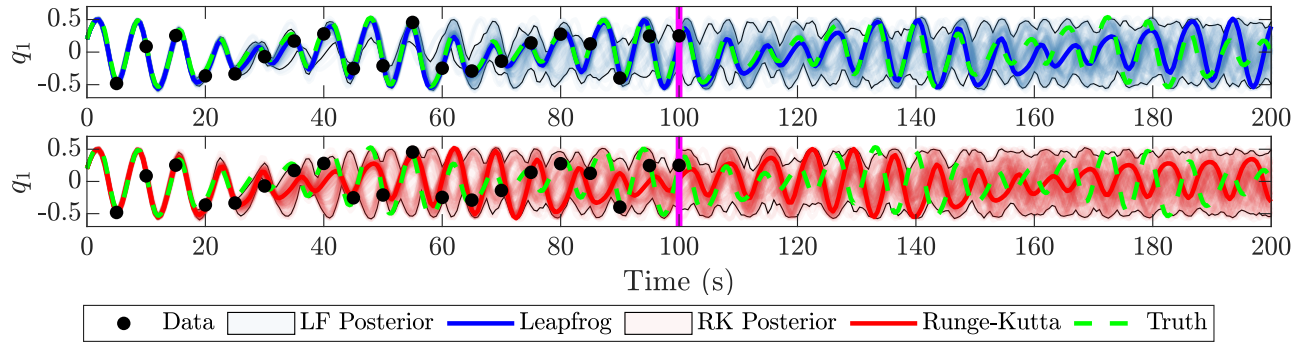
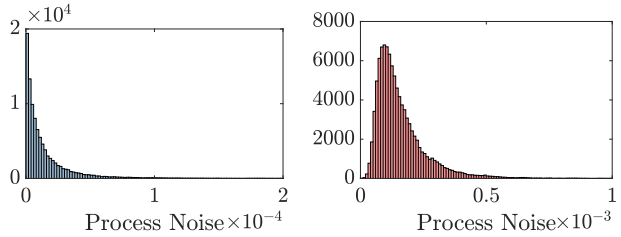


Fig. 3: Hénon-Heiles reconstructed trajectories using the MAP point and samples from the posterior. The top row shows the results from the symplectic approach and the bottom row shows results from the non-symplectic approach, each integrated with leapfrog. The vertical magenta line indicates the end of the data collection period.



(a) Leapfrog (b) Runge-Kutta

Fig. 4: Marginal posteriors of the process noise. Due to the symplectic nature of the leapfrog integrator, the process noise is reduced by an order of magnitude.

VI. Acknowledgments

This research was primarily supported by the DARPA Physics of AI Program under the grant "Physics Inspired Learning and Learning the Order and Structure of Physics."

References

[1] S. M. Khansari-Zadeh and A. Billard, "Learning control lyapunov function to ensure stability of dynamical system-based robot reaching motions," *Robotics and Autonomous Systems*, vol. 62, no. 6, pp. 752–765, 2014.

[2] ——, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[3] K. Neumann, A. Lemme, and J. J. Steil, "Neural learning of stable dynamical systems based on data-driven lyapunov candidates," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1216–1222.

[4] C. Blocher, M. Saveriano, and D. Lee, "Learning stable dynamical systems using contraction theory," in *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*. IEEE, 2017, pp. 124–129.

[5] S. Chmiela, A. Tkatchenko, H. E. Sauceda, I. Poltavsky, K. T. Schütt, and K.-R. Müller, "Machine learning of accurate energy-conserving molecular force fields," *Science advances*, vol. 3, no. 5, p. e1603015, 2017.

[6] S. Greydanus, M. Dzamba, and J. Yosinski, "Hamiltonian neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 353–15 363.

[7] T. Bertalan, F. Dietrich, I. Mezić, and I. G. Kevrekidis, "On learning hamiltonian systems from data," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 29, no. 12, p. 121107, 2019.

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

[9] D. J. Hills, A. M. Grütter, and J. J. Hudson, "An algorithm for discovering lagrangians automatically from data," *PeerJ Computer Science*, vol. 1, p. e31, 2015.

[10] M. Ahmadi, U. Topcu, and C. Rowley, "Control-oriented learning of lagrangian and hamiltonian systems," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 520–525.

[11] P. Vernaza, D. D. Lee, and S.-J. Yi, "Learning and planning high-dimensional physical trajectories via structured lagrangians," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 846–852.

[12] L. A. Aguirre, R. A. Lopes, G. F. Amaral, and C. Letellier, "Constraining the topology of neural networks to ensure dynamics with symmetry properties," *Physical Review E*, vol. 69, no. 2, p. 026701, 2004.

[13] Y. D. Zhong, B. Dey, and A. Chakraborty, "Symplectic ode-net: Learning hamiltonian dynamics with control," *arXiv preprint arXiv:1909.12077*, 2019.

[14] P. Toth, D. J. Rezende, A. Jaegle, S. Racanière, A. Botev, and I. Higgins, "Hamiltonian generative networks," *arXiv preprint arXiv:1909.13789*, 2019.

[15] N. Galioto and A. A. Gorodetsky, "Bayesian system id: optimal management of parameter, model, and measurement uncertainty," *Nonlinear Dynamics*, Sep 2020. [Online]. Available: https://doi.org/10.1007/s11071-020-05925-8

[16] V. I. Arnol'd, *Mathematical methods of classical mechanics*. Springer Science & Business Media, 2013, vol. 60.

[17] R. W. Hockney and J. W. Eastwood, *Computer simulation using particles*. crc Press, 1988.

[18] S. Särkkä, *Bayesian Filtering and Smoothing*, ser. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.

[19] M. Khalil, A. Sarkar, S. Adhikari, and D. Poirel, "The estimation of time-invariant parameters of noisy nonlinear oscillatory systems," *Journal of Sound and Vibration*, vol. 344, pp. 81 – 100, 2015.

[20] R. Askey and J. A. Wilson, *Some basic hypergeometric orthogonal polynomials that generalize Jacobi polynomials*. American Mathematical Soc., 1985, vol. 319.

[21] K. Wu, T. Qin, and D. Xiu, "Structure-preserving method for reconstructing unknown hamiltonian systems from trajectory data," *arXiv preprint arXiv:1905.10396*, 2019.

[22] C. Runge, "Über die numerische auflösung von differentialgleichungen," *Mathematische Annalen*, vol. 46, no. 2, pp. 167–178, 1895.

[23] W. Kutta, "Beitrag zur naherungsweisen integration totaler differentialgleichungen," *Z. Math. Phys.*, vol. 46, pp. 435–453, 1901.

[24] H. Haario, M. Laine, A. Mira, and E. Saksman, "Dram: efficient adaptive mcmc," *Statistics and Computing*, vol. 16, pp. 339–354, 12 2006.

[25] P. M. Williams, "Bayesian regularization and pruning using a laplace prior," *Neural Computation*, vol. 7, no. 1, pp. 117–143, 1995.

[26] A. Gelman, "Prior distributions for variance parameters in hierarchical models," *Bayesian Analysis*, vol. 1, 09 2006.

[27] M. Hénon and C. Heiles, "The applicability of the third integral of motion: some numerical experiments," *The Astronomical Journal*, vol. 69, p. 73, 1964.