Spring 6-1-2022

Symplectically Integrated Symbolic Regression of Hamiltonian Dynamical Systems

Daniel DiPietro
Dartmouth College, Daniel.M.DiPietro.22@Dartmouth.edu

Symplectically Integrated Symbolic Regression of Hamiltonian Dynamical Systems

Daniel M. DiPietro
daniel.m.dipietro.22@dartmouth.edu

Undergraduate Thesis in Computer Science, Dartmouth College

Advised By: Professor Bo Zhu

May 16th, 2022

Abstract

Here we present Symplectically Integrated Symbolic Regression (SISR), a novel technique for learning physical governing equations from data. SISR employs a deep symbolic regression approach, using a multi-layer LSTM-RNN with mutation to probabilistically sample Hamiltonian symbolic expressions. Using symplectic neural networks, we develop a model-agnostic approach for extracting meaningful physical priors from the data that can be imposed on-the-fly into the RNN output, limiting its search space. Hamiltonians generated by the RNN are optimized and assessed using a fourth-order symplectic integration scheme; prediction performance is used to train the LSTM-RNN to generate increasingly better functions via a risk-seeking policy gradients approach. Employing these techniques, we extract correct governing equations from oscillator, pendulum, two-body, and three-body gravitational systems with noisy and extremely small datasets.

1. Introduction

Newton, Kepler, and many other physicists are remembered for uncovering mathematical governing equations of complex physical dynamical systems. Ultimately, they were performing exceedingly difficult free-form regression tasks, finding relationships in data sourced from real-world physical observations.

Modern machine learning techniques have demonstrated a broad and impressive ability to extract meaningful relationships from data (Radford et al., 2019; Wang et al., 2017; Krizhevsky et al., 2012). Researchers are increasingly interested in applying such techniques to scientific discovery tasks, particularly in the context of physical dynamical systems and their governing equations (Long et al., 2018; 2019; Qin et al., 2019; Brunton et al., 2020; Raissi & Karniadakis, 2018; Brunton et al., 2016; Tong et al., 2021; Liu & Tegmark, 2021). When compared to traditional machine learning tasks, governing equation discovery offers additional changes in that observation datasets tend to be relatively small and are inevitably tainted with noise or measurement error. Constructing approaches that can learn from a small number of observations while being resistant to noise is a core goal of using machine learning for governing equation discovery. Much of this work has focused on specific formalisms of governing equations, such as Hamiltonians, which provide a consistent approach for many physical dynamical systems and the easy incorporation of priors to shrink the solution space (Greydanus et al., 2019; DiPietro et al., 2020; Chen et al., 2019).

In the past literature, there have been two lines of approaches to learn and predict the time evolution of a physical dynamical system. The first approach emphasizes prediction, relying upon black-box machine learning techniques that can time-evolve the dynamical system with very high accuracy but offer no insight into its underlying governing equations (Greydanus et al., 2019; Chen et al., 2019; Raissi et al., 2020; Breen et al., 2020; Cranmer et al., 2020a). The second approach emphasizes interpretability, prioritizing the discovery of true, concise governing equations for the system (DiPietro et al., 2020; Brunton et al., 2016; Cranmer et al., 2020b; Schmidt & Lipson, 2009; Udrescu & Tegmark; Liu & Tegmark, 2021). Interpretable approaches have yielded noteworthy successes, but they often require large amounts of data, fail to learn systems with many degrees of freedom, or require the user to specify function spaces that the governing equation must be contained in. Both black-box and interpretable methods commonly incorporate physics priors, such as symplectic integration schemes, modified loss functions, structure-enforcing formalisms, and constraining equations derived from known physical laws.

In this paper, we present a novel interpretable method: Symplectically Integrated Symbolic Regression (SISR). SISR adopts the Hamiltonian formalism, seeking to uncover governing Hamiltonians of physical dynamical systems solely from position-momenta time series data. First, SISR employs a model-agnostic technique using black-box symplectic networks to extract common Hamiltonian priors, such as coupling, from data. Next, SISR uses the deep symbolic regression paradigm, employing an autoregressive LSTM-RNN that probabilistically generates sequences of function operators, variables, and placeholder constants. These sequences correspond to pre-order traversals of free-form symbolic expressions: in this case, Hamiltonians. The RNN is specially constructed so that only separable Hamiltonian functions are generated. Additionally, the Hamiltonian priors extracted in the initial step are imposed into RNN architecture so that all generated Hamiltonians obey them, drastically reducing the search space. Once generated, these Hamiltonian sequences are translated at run-time into trainable expressions that support auto-differentiation. Each expression has its constants optimized via back-propagation through a symplectic integrator; its prediction performance (reward) is recorded during this process. Then, using a risk-seeking policy gradients approach, the RNN is trained to generate increasingly better symbolic expressions.

Using these techniques, SISR manages to extract correct governing Hamiltonians from four noteworthy dynamical systems, including harmonic oscillator, pendulum, two-body gravitational, and three-body gravitational. All experiments are performed on small training datasets (none them exceed 200 points, as compared to 1000+ in other state-of-the-art approaches), with and without Gaussian noise. To our knowledge, SISR is the first free-form, interpretable machine learning approach that can discover governing Hamiltonians of systems with many degrees of freedom using small-scale, noisy data samples.

In summary, we propose SISR, a novel technique for extracting governing Hamiltonians from observation data, which makes the following key architectural contributions:

1. Employs a model-agnostic approach using symplectic neural networks to extract common Hamiltonian priors from data.

2. Uses a specially constructed LSTM-RNN with mutation to generate separable Hamiltonians that obey the extracted Hamiltonian priors.

3. Imposes a further symplectic prior on the RNN-generated expressions by training their constants via backpropagation through a symplectic integrator.

2. Related Work

Symbolic Regression: Traditional and Deep Often, regression techniques parameterize underlying functions using specific functional forms, such as lines or polynomials. Symbolic regression, on the other hand, performs its regression search over the space of all mathematical functions that can be constructed from some given set of operators: the form is essentially arbitrary. Traditionally, symbolic regression has relied upon genetic programming techniques (O'Neill, 2009; Lu et al., 2016; Cranmer et al., 2020b). Recently, researchers have begun incorporating modern deep learning approaches. Sahoo et al. (2018) parameterize spaces of symbolic functions using neural networks with specially constructed activation functions. More recently, Petersen et al. (2019) introduce the deep symbolic regression model, which uses an autoregressive RNN to generate symbolic expressions and then trains it using a novel policy-gradients approach. Mundhenk et al. (2021) augment the deep symbolic regression approach, using an RNN-guided symbolic expression search to construct populations for use with genetic programming. Note that most of these techniques are for the general symbolic regression task, lacking any priors unique to physical dynamical systems.

Computational Techniques for Governing Equation Discovery There has been considerable research done on machine learning models that can extract mathematically concise governing functions from physical systems. Brunton et al. (2016) propose SINDy, which operates within a pre-specified function space (i.e. polynomial, trigonometric, etc.) and performs a L1-regularized sparse regression, learning which terms to keep. DiPietro et al. (2020) introduce SSINNs, which use a similar approach to SINDy but frame the problem using the Hamiltonian formalism and perform the sparse optimization by back-propagating through a symplectic integrator, which incorporates a symplectic bias into the symbolic expressions. These sparse approaches are limited in that the governing function must exist in the pre-specified function space in order to be found. Cranmer et al. (2020b) use graph neural networks, representing particles as nodes and forces between pairs of particles as edges. After training these graph models, they extract symbolic expressions by using Eureqa to fit the edge, node, and global models of their graph networks. However, their experiments involve a relatively large volume of clean data. Schmidt & Lipson (2009) employ a genetic, free-form symbolic regression approach that focuses upon conserved quantities such as Lagrangians and Hamiltonians; they succeed in learning governing equations for systems such as the double pendulum. Finally, there are numerous other approaches for equation discovery that follow different paradigms altogether (Long et al., 2018; 2019; Raissi & Karniadakis, 2018; Saemundsson et al., 2020; Sun et al., 2019).

Symplectic Machine Learning Models In recent years, many works have incorporated symplectomorphism as a physics prior; as a result, they tend to use the Hamiltonian formalism as well. Greydanus et al. (2019) propose Hamiltonian Neural Networks, which parameterize Hamiltonians using black-box deep neural networks. They optimize the gradients of these networks via back-propagation through a leapfrog numerical integrator, which gives rise to conserved quantities in the model. Chen et al. (2019) employ a similar approach but opt for a symplectic integrator and recurrent neural network; they achieve noteworthy prediction results but, due to their choice of integrator, can only learn separable Hamiltonians. Xiong et al. (2020) provide a novel architecture that incorporates symplectic priors but is not limited to separable Hamiltonians; they achieve noteworthy results on both separable and non-separable systems, including chaotic vortical flows. DiPietro et al. (2020), mentioned above, use a symplectic prior to augment an interpretable sparse regression approach. Symplectic priors are used by a variety of other works as well (Tong et al., 2021; Zhu et al., 2020; Mattheakis et al., 2019; Jin et al., 2020).

3. Background

SISR learns governing equations in the Hamiltonian formalism and employs a model-agnostic approach to extract Hamiltonian coupling priors. Here we offer mathematical background on Hamiltonian mechanics and coupling.

3.1. Hamiltonian Mechanics

Hamiltonian mechanics is a reformulation of classical mechanics introduced in 1833 by William Rowland Hamilton. The Hamiltonian formalism provides a consistent approach for mathematically describing a variety of physical dynamical systems; it is heavily used in quantum mechanics and statistical mechanics, often for systems with many degrees of freedom (Morin, 2008; Reichl, 2016; Taylor, 2005; Sakurai & Napolitano, 2010; Hand & Finch, 1998).

In Hamiltonian mechanics, dynamical systems are governed via their Hamiltonian $\mathcal{H}(\mathbf{p}, \mathbf{q})$, a function of the momenta and position vectors ($\mathbf{q}$ and $\mathbf{p}$ respectively) of each object in the system. The value of the Hamiltonian generally measures the total energy of the system; for systems where energy is conserved, $\mathcal{H}$ is time-invariant. Further, we say that $\mathcal{H}$ is *separable* if it can be decomposed into two additively separable functions $T(\mathbf{p})$ and $V(\mathbf{q})$, where T measures kinetic energy and V measures potential energy.

The time evolution of a physical dynamical system is uniquely defined by the partial derivatives of its Hamiltonian:

$$\frac{d\mathbf{p}}{dt} = -\frac{\partial \mathcal{H}}{\partial \mathbf{q}}, \frac{d\mathbf{q}}{dt} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}} \tag{1}$$

The time evolution of a Hamiltonian system is symplectomorphic, preserving the symplectic two-form $d\mathbf{p} \wedge d\mathbf{q}$. As a result, researchers generally time-evolve Hamiltonians with symplectic integrators, which are designed to preserve this two-form. We employ a fourth-order symplectic integrator (pseudocode in Appendix A) (Forest & Ruth, 1990).

3.2. Coupling

Hamiltonians commonly exhibit *coupling*: they can be written as a series of additively separable functions where each function takes some smaller subset of variables. Consider some arbitrary n-body, m-dimensional separable Hamiltonian $\mathcal{H}(\mathbf{p}, \mathbf{q}) = T(\mathbf{p}) + V(\mathbf{q})$ where the bodies are labeled $1, \ldots, n$ and the dimensions are labeled $1, \ldots, m$. What coupling might occur?

Without loss of generality, consider $T(\mathbf{p})$. Denote the momentum of body i in dimension j as $p_{i,j}$. We say that $T(\mathbf{p})$ exhibits complete decoupling if it may be rewritten as the sum of functions that each take only one variable in the system, i.e. it may be written in the form:

$$\begin{aligned} T(\mathbf{p}) = f_1(p_{1,1}) + \cdots + f_m(p_{1,m}) + \cdots + \\ f_{nm-m}(p_{n,1}) + \cdots + f_{nm}(p_{n,m}) \end{aligned} \tag{2}$$

We say that $T(\mathbf{p})$ exhibits dimensional coupling if it may be rewritten as the sum of functions that each take all dimensional variables for only a single particle in the system, i.e. it may be written in the form:

$$\begin{aligned} T(\mathbf{p}) = g_1(p_{1,1}, \ldots, p_{1,m}) + \cdots + \\ g_m(p_{n,1}, \ldots, p_{n,m}) \end{aligned} \tag{3}$$

Finally, we say that $T(\mathbf{p})$ exhibits pairwise coupling if it may be rewritten as the sum of functions that each take all dimensional variables for a pair of particles in the system, i.e. where $\mathbf{p_i}$ denotes all momenta variables of particle i, T may be written in the form:

$$\begin{aligned} T(\mathbf{p}) = h_1(\mathbf{p}_1, \mathbf{p}_2) + \cdots + h_{m-1}(\mathbf{p}_1, \mathbf{p}_n) + \\ h_m(\mathbf{p}_2, \mathbf{p}_3) + \cdots + h_{2m-2}(\mathbf{p}_2, \mathbf{p}_n) + \ldots \end{aligned} \tag{4}$$

3.3. Coupling Composite Functions

Consider some arbitrary coupling function $f(x_1, \ldots, x_n)$. Often, f may be written as a composition of functions $f(x_1, \ldots, x_n) = g(h(x_1, \ldots, x_n))$ where h performs one of several common reduction operations, such as summing, multiplication, Euclidean distance, or Manhattan distance. In these instances, we would say that particles are coupled via the given inner operation, such as their sum. If h is known, then one may learn f by learning g, which tends to be much simpler than learning the entire f directly.

3.4. Symmetries

Suppose that a Hamiltonian function exhibits pairwise coupling, i.e., it may be written as the sum of many additively separable coupling functions. Often, across all pairs of particles, these coupling functions are identical (symmetric). For instance, in an n-body gravitational system with particles of equal mass, or an n-body electrostatic system with particles of equal charge, all particles are pairwise coupled using the exact same coupling function.

4. Algorithm

Here we present the SISR architecture, which consists of a coupling extraction process followed by a deep symbolic regression-inspired training loop. This training loop can further be decomposed into expression generation (4.2), symplectic optimization (4.3), and RNN training (4.4).

4.1. Extracting Function Properties via Symplectic Neural Networks

As detailed in Sections 3.2, 3.3, and 3.4, Hamiltonians commonly exhibit coupling properties. These properties may be used to simplify the Hamiltonian discovery process by imposing restrictions that dramatically reduce the solution space. Thus, the first step of SISR is to find what coupling is taking place in the Hamiltonian, whether it can be reduced into simpler composite functions, and whether it is symmetric across particles or pairs of particles.

Our approach for doing this is rooted in symplectic neural networks as effective black-box predictors of Hamiltonian dynamical systems. Specifically, we can construct multi-layer neural networks that enforce specific coupling properties. Then, we may train these networks by using them to time-evolve the dynamical system through a symplectic integrator. By seeing how these networks perform with and without imposed coupling properties, we may deduce whether those properties hold for the system. To the best of our knowledge, we are the first work to extract coupling properties in the context of Hamiltonian dynamical systems, and we are not aware of any other work that employs symplectic neural networks in a similar manner.

In order to assess whether a physical dynamical system exhibits a particular variety of coupling, we can construct a model that enforces one of the functional forms described in Equations 2, 3, or 4. We do this by parameterizing the individual coupling functions with neural networks and summing the individual outputs of these networks to get the entire function output for V or T. Note that these models are trained via gradient descent through a symplectic integrator; this process is depicted for a 3-body pairwise-momentum dimensional-position model in Figure 1.

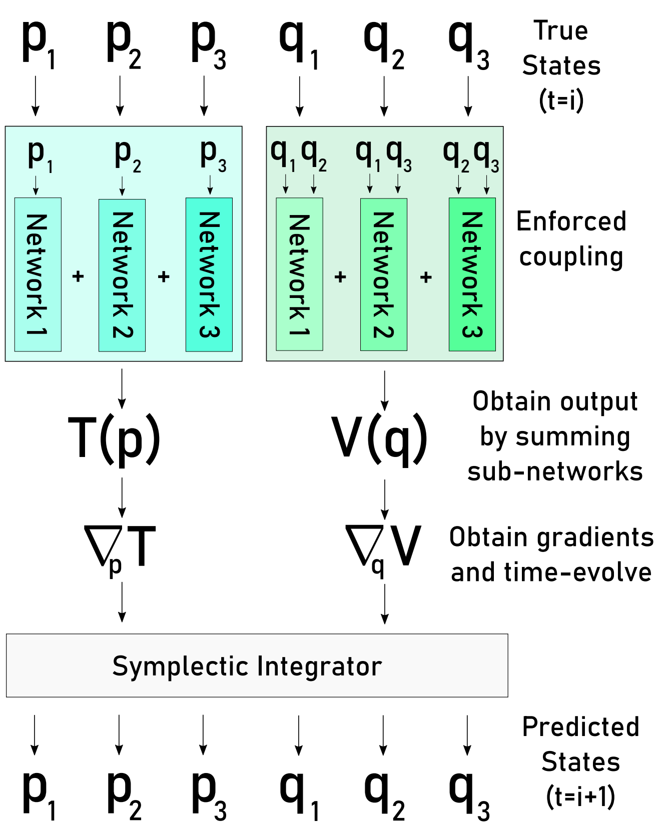
p_1 p_2 p_3 q_1 q_2 q_3
True States (t=i)
p_1 p_2 p_3
$q_1 q_2$ $q_1 q_3$ $q_2 q_3$
Network 1 + Network 2 + Network 3
Network 1 + Network 2 + Network 3
Enforced coupling
T(p)
V(q)
Obtain output by summing sub-networks
$\nabla_p T$
$\nabla_q V$
Obtain gradients and time-evolve
Symplectic Integrator
p_1 p_2 p_3 q_1 q_2 q_3
Predicted States (t=i+1)

Figure 1. Symplectic network constructed with 3 particles exhibiting dimensional position coupling and pairwise momentum coupling. Once the predicted states at $t = i + 1$ are obtained, the model can be trained by taking loss with respect to the actual states at $t = i + 1$.

4.1.1. Coupling

Although this method works for assessing specific instances of coupling, the question still remains of how we construct a search over T and V, as the couplings of these functions are independent. Additionally, even if a Hamiltonian exhibits pairwise coupling, it doesn't mean that *all* particles in the system are necessarily pairwise coupled. Our search approach is simple and works well empirically.

We begin by training a model where T and V are each parameterized by a single network, meaning that there are no coupling properties enforced. The value of the performance metric for this model is used to set a baseline. Then, holding this network constant for T, we enforce complete decoupling in V. If the corresponding performance metric drop is above some pre-defined fixed percentage relative to the baseline, denoted the *maximum tolerable performance decrease*, we say that the V is in fact completely decoupled. Otherwise, we attempt dimensional coupling and possibly pairwise coupling. This process is then repeated for T, with the network in V held constant. Note that, if pairwise coupling is found, we perform a recursive backwards elimination to see which pairs must be present; this elimination process requires an additional maximum tolerable performance decrease parameter; this parameter is generally smaller, as it is easy to overfit to pairs that aren't actually present in the Hamiltonian, especially with little data.

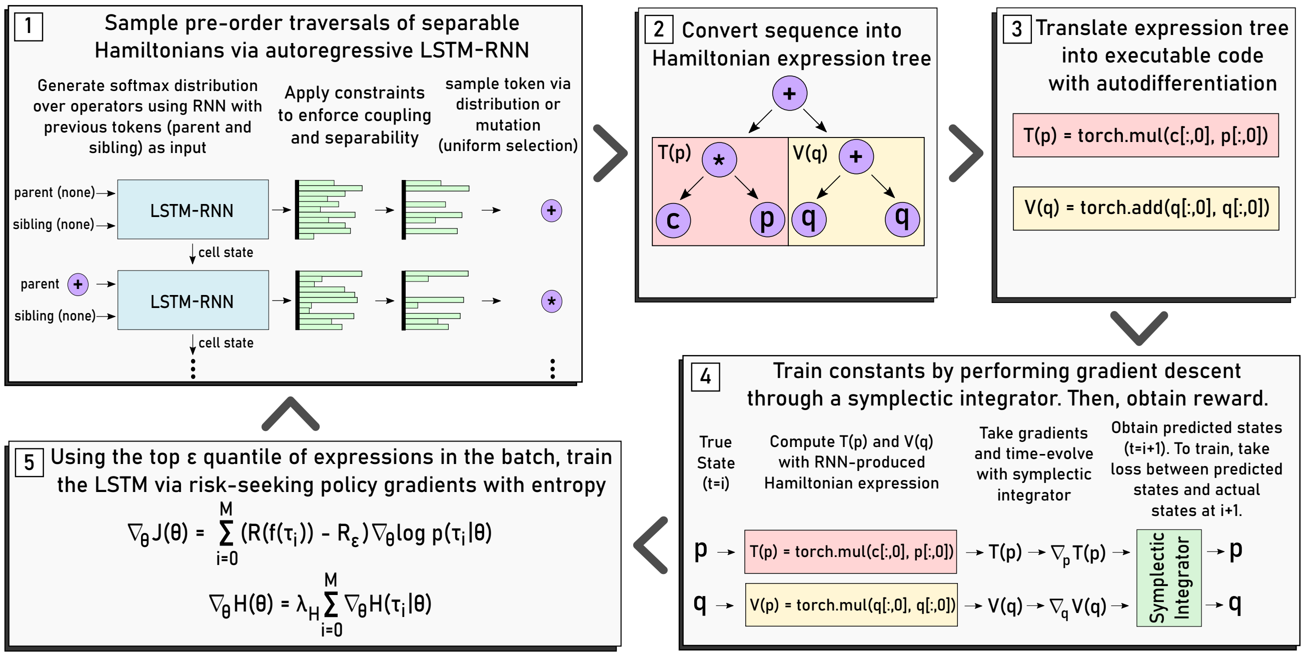
1 Sample pre-order traversals of separable Hamiltonians via autoregressive LSTM-RNN
Generate softmax distribution over operators using RNN with previous tokens (parent and sibling) as input
Apply constraints to enforce coupling and separability
sample token via distribution or mutation (uniform selection)
parent (none)
sibling (none)
LSTM-RNN
cell state
parent
sibling (none)
2 Convert sequence into Hamiltonian expression tree
T(p)
V(q)
3 Translate expression tree into executable code with autodifferentiation
T(p) = torch.mul(c[:,0], p[:,0])
V(q) = torch.add(q[:,0], q[:,0])
4 Train constants by performing gradient descent through a symplectic integrator. Then, obtain reward.
True State (t=i)
Compute T(p) and V(q) with RNN-produced Hamiltonian expression
Take gradients and time-evolve with symplectic integrator
Obtain predicted states (t=i+1). To train, take loss between predicted states and actual states at i+1.
V(p) = torch.mul(q[:,0], q[:,0])
Symplectic Integrator
5 Using the top ε quantile of expressions in the batch, train the LSTM via risk-seeking policy gradients with entropy
$\nabla_\theta J(\theta) = \sum_{i=0}^{M} (R(f(\tau_i)) - R_\varepsilon)\nabla_\theta \log p(\tau_i|\theta)$
$\nabla_\theta H(\theta) = \lambda_H \sum_{i=0}^{M} \nabla_\theta H(\tau_i|\theta)$

Figure 2. Overview of the SISR Hamiltonian generation, symplectic optimization, and RNN training processes. Note that steps 1-4 occur in batches, with many variable-length expressions being sampled, converted, translated, and trained in parallel.

4.1.2. Coupling Composite Functions

Once coupling has been identified, we determine whether the quantities are coupled by their sum, product, distance, or some other operation. To do this, we modify the network so that the operation being tested is performed immediately on the input variables prior to the forward pass. Then, we select the best performing composite function that meets the maximum tolerable performance decrease.

4.1.3. Symmetries

Once we have determined which particles are coupled and by what, if any, composite functions, we assess whether the coupling functions are symmetric across all pairs of particles. To do this, we use the same neural network for all coupled quantities (such as all pairs of positions in an n-body gravitational system). Note that this requires that the domain of input values for each particle are reasonably close. If the performance doesn't degrade below our tolerance, we say that the couplings are symmetric.

4.2. Generating Symbolic Hamiltonians

With our coupling properties determined, we are ready to begin generating candidate symbolic Hamiltonians. We do this via a deep symbolic regression approach, employing an LSTM-RNN.

First, note that free-form symbolic functions may be represented as trees where internal nodes are functional operators and leaves are variables or constants. The pre-order traversal of this expression tree produces a sequence that uniquely defines the symbolic expression. Thus, sequence-generating RNNs can be used in an autoregressive manner to produce symbolic functions. We refer to functional operators, variables, and constants as *tokens* in the sequence.

To generate a symbolic expression, an LSTM-RNN generates a sequence from left-to-right. As input, the network takes a one-hot encoding of the parent and sibling token for the current token being sampled. It then emits a softmax distribution, which is adjusted to eliminate all invalid tokens. This distribution may be sampled to generate the next token in the sequence. We also employ an epsilon-greedy mutation approach where the next token is sampled uniformly from the list of valid tokens with μ probability, where μ is a pre-defined hyperparameter. This process is depicted in Step 1 of Figure 2.

Due to our use of a fourth-order symplectic integrator, we require that any Hamiltonian generated by the LSTM-RNN is of the form $\mathcal{H}(\mathbf{p}, \mathbf{q}) = T(\mathbf{p}) + V(\mathbf{q})$. This constraint is applied *in situ*, meaning that the softmax distribution over the operators is constantly adjusted as expressions are generated so that only Hamiltonians of this form will ever be produced. This same approach is used to incorporate any coupling properties previously discovered: the emitted distribution is adjusted prior to sampling a token so that we will only ever see Hamiltonians that exhibit the desired coupling. To the best of our knowledge, this is the first instance of an LSTM-RNN being used to generate separable Hamiltonians with desired coupling properties.

Once a Hamiltonian sequence is generated, it is translated into an executable model with auto-differentiation (Steps 2 and 3 of Figure 2). The first reason for this is to that constants can be optimized–the sequence contains placeholder "ephemeral" constants that do not refer to specific values. The second purpose is so that the performance of the expression can be assessed. This is done by numerically integrating the Hamiltonian to predict the system, which requires the ability to take its gradients.

4.3. Symplectic Optimization

As mentioned previously, candidate Hamiltonian functions must have their placeholder constants trained. This is performed via a symplectic optimization approach (Step 4 of Figure 2). Using a fourth-order symplectic integrator and a time-series of observation data for the system, we use the candidate Hamiltonian to time-evolve the system and obtain predicted future states. We then take the loss between our predicted future states and the actual future states contained in the observation data, backpropagating through our integrator to train the constants and thus imparting an energy-conserving symplectic prior onto any Hamiltonian generated by the RNN. This approach, which has been applied to black-box system prediction methods with great success, has yet to be employed for any free-form symbolic regression model (Chen et al., 2019; DiPietro et al., 2020).

4.4. Training the RNN

Thus far, we have presented an RNN that can generate separable Hamiltonians (with specific coupling properties) that are then trained via backpropagation through a symplectic integrator. However, in order for a symbolic Hamiltonian search to succeed, the RNN must be trained to produce progressively better-fitting Hamiltonians.

To do this, we employ the risk-seeking policy gradients approach presented by Petersen et al. (2019), which maximizes the best-case expectation of rewards for expressions generated by the policy (in this case the RNN). To do this, we train on the top ϵ quantile in some given batch, where $1 - \epsilon$ is referred to as the *risk factor*. First, let this quantile consist of M expressions where τ_i denotes the i-th expression in the quantile. If f defines a symplectic integration function, R defines a reward function, R_ϵ defines the reward of the top ϵ quantile, and $J(\theta)$ defines the expectation of rewards in the top ϵ quantile conditioned on the weights θ of the RNN, we have:

$$\nabla_\theta J(\theta) = \sum_{i=1}^{M} \left(R(f(\tau_i)) - R_\epsilon\right) \nabla_\theta \log p(\tau_i|\theta) \tag{5}$$

In the event that the expression generation involved mutation, we compute $\log p(\tau_i|\theta)$ using the probabilities produced by the RNN–*not* the uniform probabilities generated to sample the mutated operator.

Further, as in Petersen et al. (2019), we adopt the maximum entropy framework and include an entropy term in the loss (Haarnoja et al., 2018). Where λ_H is the entropy coefficient, we have:

$$\nabla_\theta H(\theta) = \lambda_H \sum_{i=1}^{M} \nabla_\theta H(\tau_i|\theta) \tag{6}$$

Note that H refers to the Shannon entropy function and is *not* a Hamiltonian.

This training process corresponds to Step 5 of Figure 2. We are not aware of any other approach for Hamiltonian dynamical systems that employs these reinforcement learning-oriented training approaches.

5. Experiments

We assess SISR by applying it to four noteworthy Hamiltonian dynamical systems: the harmonic oscillator, the pendulum, the two-body gravitational system (the Kepler problem), and the three-body gravitational system. Aside from expression length, all experiments use the same set of hyperparameters, which are almost entirely replicated from Petersen et al. (2019) and detailed in Appendix B.1.

For each experiment, we generated 3 different time series datasets via fourth-order symplectic integration with different initial conditions, which are available in Appendix B.2. These datasets are exceedingly small–smaller than any other work in this domain that we are aware of. Each experiment was conducted a total of 5 times on each dataset. Note that each dataset consists entirely of training data, as either SISR extracts the correct governing Hamiltonian from the training data or it does not. We say that a SISR-generated Hamiltonian is correct if it is algebraically equivalent to the true Hamiltonian with 10^{-2} precision in its constants. Our LSTM-RNN was optimized with Adam and employed an NRMSE reward function.

In initial experiments, we noticed very high variance in the number of epochs needed to obtain the Hamiltonian. We believe that this may be a consequence of the initial batch being too small to adequately explore the function space. As a result, we set the initial batch size to 4 times the regular batch size; we adjust the risk factor to compensate, meaning that the RNN is trained on the same number of expressions after each batch.

5.1. Harmonic Oscillator

The harmonic oscillator is a one-dimensional, single-particle dynamical system described by the Hamiltonian

$$\mathcal{H}(p, q) = \frac{p^2}{2m} + \frac{1}{2} m\omega^2 q^2 \tag{7}$$

where m is the mass of the particle and ω refers to its oscillator frequency.

We trained SISR on 3 different datasets of harmonic oscillator motion with varying initial conditions, each consisting of only 30 points from $t = 0$ to $t = 3$. In addition to variables and constants, SISR had access to the operators $\{+, -, \div, \cdot, \wedge\}$. The T and V of each generated Hamiltonian were limited to be between 1 and 8 operators long. SISR was applied to each dataset 5 times, recovering the correct Hamiltonian for all 15 trials. This took an average of 4.60 ± 3.25 batches, or 225.83 ± 101.99 seconds. In 4 out of 15 instances, the Hamiltonian was uncovered after a single epoch. Next, we applied SISR to the same datasets with $(\mu = 0, \sigma = 0.001)$ of Gaussian noise. Once again, it uncovered the correct Hamiltonian in all trials, requiring an average of 5.00 ± 3.34 batches, or 252.76 ± 109.04 seconds.

5.2. Pendulum

The pendulum is an angular, single-particle dynamical system described by the Hamiltonian

$$\mathcal{H}(p_\theta, q_\theta) = \frac{p_\theta^2}{2m\ell^2} + mg\ell(1 - \cos(q_\theta)) \tag{8}$$

where m is mass, g is gravity, and ℓ is length.

Using the functional operators $\{+, -, \div, \cdot, \wedge, \cos, \sin\}$, we trained SISR on 3 different datasets of pendulum motion with varying initial conditions, containing 50 points from $t = 0$ to $t = 5$. The T and V of each generated Hamiltonian were limited to be between 1 and 8 operators long. With no noise present, SISR recovered the correct Hamiltonian in all 15 trials, taking an average of 21.80 ± 11.18 batches, or 1049.41 ± 465.10 seconds. When $(\mu = 0, \sigma = 0.005)$ of Gaussian noise was added to each dataset, SISR recovered all correct Hamiltonians in 28.73 ± 13.68 batches, or 1117.95 ± 497.57 seconds. Hamiltonians generated by SISR for the pendulum problem are depicted in Figure 3.

5.3. Two-Body Gravitational System

The two-body gravitational system is a two-dimensional dynamical system described by the Hamiltonian

$$\mathcal{H}(\mathbf{p}, \mathbf{q}) = \frac{Gm_i m_j}{|\mathbf{q}_i - \mathbf{q}_j|} + \frac{\mathbf{p}_i^2}{2m_i} + \frac{\mathbf{p}_j^2}{2m_j} \tag{9}$$

with mass m for particles i and j. Note that we work explicitly with two-body systems where all masses are equal and $G = 1$.

We began by extracting coupling properties via the symplectic neural network procedures described in Section 4.1. Using this process, we first determined that particle positions were pairwise coupled, as well as that particle momenta were completely decoupled. Then, we were able to decompose the position coupling into a composite function of Euclidean distance. Finally, we determined that the functions acting on the momenta of each particle are symmetric. The exact numerical results used to guide this search process are available in Appendix B.3. Once these coupling constraints are imposed on the RNN, the solution space shrinks dramatically. Rather than learning $V(\mathbf{q})$ and $T(\mathbf{p})$ outright as in our previous experiments, we are effectively learning the pairwise position coupling function between particles, which is itself a function of their Euclidean distance, and the decoupled momentum function acting on each particle.

Using the functional operators $\{+, -, \div, \cdot, \wedge\}$, we trained SISR on 3 different datasets of two-body motion with varying initial conditions, containing 200 points from $t = 0$ to $t = 20$. The T and V of each generated Hamiltonian were limited to be between 1 and 12 operators long. With no noise present, SISR recovered the correct Hamiltonian for all 15 trials, taking an average of 1.47 ± 1.30 batches, or 227.42 ± 62.94 seconds. When $(\mu = 0, \sigma = 0.001)$ of Gaussian noise was added to each dataset, SISR recovered all correct Hamiltonians in 1.80 ± 1.15 batches, or 236.79 ± 55.45 seconds. Note how powerful our extracted coupling priors are: with them imposed, the two-body Hamiltonian is learned more quickly than a pendulum Hamiltonian.

5.4. Three-Body Gravitational System

The three-body gravitational system is a two-dimensional dynamical system described by the Hamiltonian

$$\mathcal{H}(\mathbf{p},\mathbf{q}) = \frac{Gm_i m_j}{|\mathbf{q}_i - \mathbf{q}_j|} + \frac{Gm_i m_k}{|\mathbf{q}_i - \mathbf{q}_k|} + \frac{Gm_j m_k}{|\mathbf{q}_j - \mathbf{q}_k|} + \frac{\mathbf{p}_i^2}{2m_i} + \frac{\mathbf{p}_j^2}{2m_j} + \frac{\mathbf{p}_k^2}{2m_k} \tag{10}$$

with mass m for particles i, j, and k. This system exhibits chaotic motion, meaning that small perturbations in initial conditions diverge exponentially in the future (Sprott, 2010).

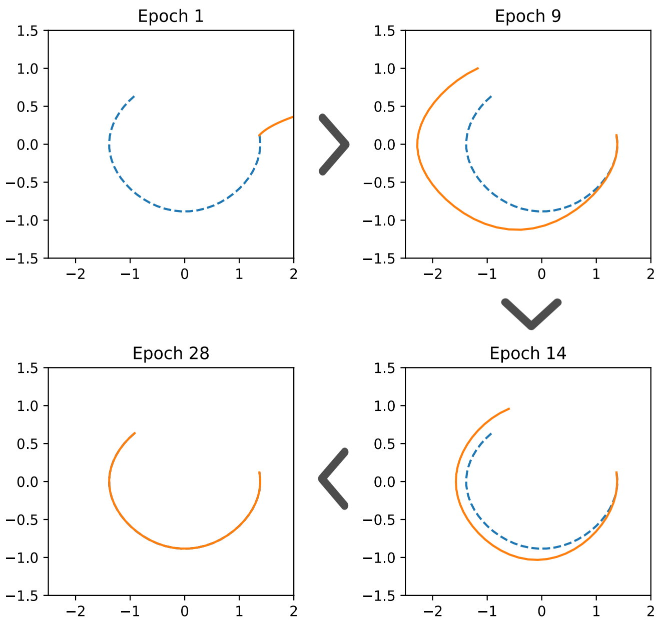

Figure 3. Phase Space ($t = 0$ to $t = 4$) for SISR-extracted pendulum Hamiltonian after 1, 9, 14, and 28 epochs (dataset 2 with Gaussian noise). The dashed blue line indicates the ground truth.

We followed a nearly identical coupling extraction process to our two-body experiments, identifying that all particles are pairwise coupled as a function of their Euclidean distance, as well as all momenta being completely decoupled and symmetric. These results are detailed in Appendix B.3.

Using the functional operators $\{+, -, \div, \cdot, \wedge\}$, we trained SISR on 3 different datasets of three-body motion with 200 points each. The T and V of each generated Hamiltonian were limited to be between 1 and 18 operators long. Five trials were conducted on each dataset. With no noise present, SISR took an average of 2.40 ± 2.32 epochs, or 588.92 ± 238.49 seconds, to recover the Hamiltonian (all were recovered). With $(\mu = 0, \sigma = 0.001)$ of Gaussian noise present, SISR required an average of 2.13 ± 1.77 epochs, or 575.48 ± 201.20 seconds (all were recovered).

5.5. Comparison to Existing Techniques

We benchmarked SISR against two other methods for obtaining physical governing equations for data: SINDy and SSINNs. Both of these methods rely upon sparse regressions in user-defined function spaces. They obtained reasonable NRMSE for many problems but only extracted correct governing equations in 2 of 8 instances. Indeed, with so little data, it is easy for sparse-regression oriented methods to overfit to excess terms. Finally, we compared to the vanilla deep symbolic regression model, which we used to generate systems of differential equations instead of Hamiltonians. These results are summarized in Table 1 with further experimental details available in Appendix B.4.

Table 1. Comparison of four methods for extracting governing equations from time series observations. NRMSE refers to normalized RMSE, a reward function between 0 and 1. NRMSE is reported for single-step predictions on the training dataset. We consider an extracted equation "correct" if it is algebraically equivalent with 10^{-2} precision in its coefficients.

	NRMSE				EXTRACTED CORRECT EQUATION			
PROBLEM	SISR	SINDY	SSINN	DSR	SISR	SINDY	SSINN	DSR
HARM. OSC. (CLEAN)	1.00 ± 0.00	0.99 ± 0.01	1.00 ± 0.00	1.00 ± 0.00	$\surd$	$\surd$	$\surd$	$\surd$
HARM. OSC. (NOISY)	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.00	1.00 ± 0.00	$\surd$	$\surd$	$\surd$	$\surd$
PENDULUM (CLEAN)	1.00 ± 0.00	0.99 ± 0.01	0.92 ± 0.01	0.99 ± 0.00	$\surd$	$\times$	$\times$	$\surd$
PENDULUM (NOISY)	1.00 ± 0.00	0.99 ± 0.01	0.93 ± 0.02	0.99 ± 0.00	$\surd$	$\times$	$\times$	$\surd$
TWO-BODY (CLEAN)	1.00 ± 0.00	0.76 ± 0.18	0.09 ± 0.07	0.65 ± 0.23	$\surd$	$\times$	$\times$	$\times$
TWO-BODY (NOISY)	1.00 ± 0.00	0.65 ± 0.15	0.10 ± 0.03	0.66 ± 0.23	$\surd$	$\times$	$\times$	$\times$
THREE-BODY (CLEAN)	1.00 ± 0.00	0.72 ± 0.07	0.18 ± 0.11	0.77 ± 0.13	$\surd$	$\times$	$\times$	$\times$
THREE-BODY (NOISY)	1.00 ± 0.00	0.69 ± 0.06	0.16 ± 0.08	0.75 ± 0.13	$\surd$	$\times$	$\times$	$\times$

6. Scope and Limitations

Limited to separable Hamiltonians SISR uncovers the dynamics of physical systems by generating separable Hamiltonians, but not all physical dynamical systems can be represented in this way. Real-world systems often exhibit energy dissipation that can violate our formalism. Being robust to such dissipation, or knowing when to switch formalisms, may serve as a basis for future work.

Coupling extraction complexity Our coupling extraction process requires training a large number of black-box symplectic neural networks. Unfortunately, as the physical dynamics of a system become increasingly complicated, these networks must increase in size to maintain predictive accuracy. In doing so, they become more time-consuming to train and further risk overfitting to the system.

Computationally expensive constant optimization Every expression generated by the LSTM-RNN must have its constants optimized. This means that, over the course of the entire Hamiltonian generation process, many thousands of expressions will need to be optimized. Empirically, this operation accounts for approximately 96% of the run-time for SISR. Many of these expressions do not fall in the top quantile, meaning that they are not used to trained the RNN. We leave optimizations to this process for future work.

7. Conclusion

Here we present Symplectically Integrated Symbolic Regression, a new technique for extracting governing Hamiltonians from time series observations of a physical dynamical system. By applying a model-agnostic coupling extraction approach, we are able to generate priors that greatly benefit our model, allowing it to quickly fit oscillator, pendulum, 2-body and 3-body systems from no more than 200 noisy observations. These results are state-of-the-art in their performance and simultaneous use of small data. Future work may include extracting additional Hamiltonian priors besides coupling, as well as allowing for energy dissipation.

References

Breen, P. G., Foley, C. N., Boekhold, T., and Zwart, S. P. Newton versus the machine: solving the chaotic three-body problem using deep neural networks. *Monthly Notices of the Royal Astronomical Society*, 494(2):2465–2470, 2020.

Brunton, S., Noack, B., and Koumoutsakos, P. Machine learning for fluid mechanics, 2020. https://arxiv.org/abs/1905.11075.

Brunton, S. L., Proctor, J. L., and Kutz, J. N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 113(15):3932–3937, 2016.

Chen, Z., Zhang, J., Arjovsky, M., and Bottou, L. Symplectic recurrent neural networks, 2019. https://arxiv.org/abs/1909.13334.

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

Cranmer, M., Sanchez-Gonzalez, A., Battaglia, P., Xu, R., Cranmer, K., Spergel, D., and Ho, S. Discovering Symbolic Models from Deep Learning with Inductive Biases. 2020b.

DiPietro, D. M., Xiong, S., and Zhu, B. Sparse symplectically integrated neural networks. In *Advances in Neural Information Processing Systems 33*, pp. 6074–6085. 2020.

Forest, E. and Ruth, R. D. Fourth-order symplectic integration. *Physica D: Nonlinear Phenomena*, 43(1):105–117, 1990.

Greydanus, S., Dzamba, M., and Yosinski, J. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems 32*, pp. 15379–15389. 2019.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Hand, L. N. and Finch, J. D. *Analytical Mechanics*. Cambridge University Press, Cambridge, England, 1998.

Jin, P., Zhang, Z., Zhu, A., Tang, Y., and Karniadakis, G. E. Sympnets: Intrinsic structure-preserving symplectic networks for identifying hamiltonian systems. *Neural Networks*, 132:166–179, 2020.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. 2012.

Liu, Z. and Tegmark, M. Machine learning conservation laws from trajectories. *Physical Review Letters*, 126(18):180604, 2021.

Long, Z., Lu, Y., Ma, X., and Dong, B. PDE-net: Learning PDEs from data. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3208–3216, 2018.

Long, Z., Lu, Y., and Dong, B. Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399:108925, 2019.

Lu, Q., Ren, J., and Wang, Z. Using genetic programming with prior formula knowledge to solve symbolic regression problem. *Computational intelligence and neuroscience*, 2016, 2016.

Mattheakis, M., Protopapas, P., Sondak, D., Di Giovanni, M., and Kaxiras, E. Physical symmetries embedded in neural networks. *arXiv preprint arXiv:1904.08991*, 2019.

Morin, D. J. *Introduction to Classical Mechanics: With Problems and Solutions*. Cambridge University Press, Cambridge, England, 2008.

Mundhenk, T., Landajuela, M., Glatt, R., Santiago, C., Petersen, B., et al. Symbolic regression via deep reinforcement learning enhanced genetic programming seeding. *Advances in Neural Information Processing Systems*, 34, 2021.

O'Neill, M. Riccardo poli, william b. langdon, nicholas f. mcphee: a field guide to genetic programming, 2009.

Petersen, B. K., Larma, M. L., Mundhenk, T. N., Santiago, C. P., Kim, S. K., and Kim, J. T. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv:1912.04871*, 2019.

Qin, T., Wu, K., and Xiu, D. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 395:620–635, 2019.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Raissi, M. and Karniadakis, G. E. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

Raissi, M., Yazdani, A., and Karniadakis, G. E. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

Reichl, L. E. *A Modern Course in Statistical Physics*. Wiley-VCH, Weinheim, Germany, 2016.

Saemundsson, S., Terenin, A., Hofmann, K., and Deisenroth, M. P. Variational integrator networks for physically structured embeddings, 2020.

Sahoo, S., Lampert, C., and Martius, G. Learning equations for extrapolation and control. In *International Conference on Machine Learning*, pp. 4442–4450. PMLR, 2018.

Sakurai, J. J. and Napolitano, J. J. *Modern Quantum Mechanics (2nd Edition)*. Pearson, London, England, 2010.

Schmidt, M. and Lipson, H. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

Sprott, J. C. *Elegant Chaos: Algebraically Simple Flows*. World Scientific, Singapore, 2010.

Sun, Y., Zhang, L., and Schaeffer, H. Neupde: Neural network based ordinary and partial differential equations for modeling time-dependent data, 2019.

Taylor, J. R. *Classical Mechanics*. University Science Books, Mill Valley, California, 2005.

Tong, Y., Xiong, S., He, X., Pan, G., and Zhu, B. Symplectic neural networks in taylor series form for hamiltonian systems. *Journal of Computational Physics*, 437:110325, 2021.

Udrescu, S.-M. and Tegmark, M. Ai feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16).

Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., and Tang, X. Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3156–3164, 2017.

Xiong, S., Tong, Y., He, X., Yang, S., Yang, C., and Zhu, B. Nonseparable symplectic neural networks. *arXiv preprint arXiv:2010.12636*, 2020.

Zhu, A., Jin, P., and Tang, Y. Deep hamiltonian networks based on symplectic integrators. *arXiv preprint arXiv:2004.13830*, 2020.

A. Symplectic Integration Pseudocode

We employ the fourth-order symplectic integrator introduced by (Forest & Ruth, 1990), defined in Algorithm 1.

Algorithm 1 Fourth-Order Symplectic Integrator

```
Input: $\mathbf{q}_{t_i}, \mathbf{p}_{t_i}, t_i, t_{i+1}, \frac{\partial T}{\partial \mathbf{p}}, \frac{\partial V}{\partial \mathbf{q}}$
Output: $\mathbf{q}_{t_{i+1}}, \mathbf{p}_{t_{i+1}}$
$h \leftarrow t_{i+1} - t_i$
$\mathbf{k}_p \leftarrow \mathbf{p}_{t_i}$
$\mathbf{k}_q \leftarrow \mathbf{q}_{t_i}$
for $j = 1$ to $4$ do
    $\mathbf{t}_p \leftarrow \mathbf{k}_p$
    $\mathbf{t}_q \leftarrow \mathbf{k}_q + c_j \cdot \frac{\partial T}{\partial \mathbf{p}}(\mathbf{k}_p) \cdot h$
    $\mathbf{k}_p \leftarrow \mathbf{t}_p + d_j \cdot \frac{\partial V}{\partial \mathbf{q}}(\mathbf{t}_q) \cdot h$
    $\mathbf{k}_q \leftarrow \mathbf{t}_q$
end for
return $\mathbf{k}_p, \mathbf{k}_q$
```

This algorithm employs the constants $c_1 = c_4 = \frac{1}{2(2-2^{1/3})}$, $c_2 = c_3 = \frac{1-2^{1/3}}{2(2-2^{1/3})}$, $d_1 = d_3 = \frac{1}{2-2^{1/3}}$, $d_2 = \frac{2^{1/3}}{2-2^{1/3}}$, and $d_4 = 0$.

B. Extended Experimental Details

B.1. SISR Hyperparameter Selection

All experiments used the same set of SISR hyperparameters, aside from their expression lengths and functional operators. These hyperparameters were largely obtained from (Petersen et al., 2019), with the exception of mutation rate, number of layers, initial batch size, inner learning rate, inner optimizer, and inner number of epochs. These additional hyperparameters worked well with their initially selected values and did not require any tuning.

All SISR models were trained with a learning rate $\alpha = 0.0005$, entropy coefficient $\lambda_H = 0.005$, mutation rate $\mu = 0.05$, risk factor $\epsilon = 0.95$, batch size of 500, and initial batch size of 2000. The RNN architecture was a two-layer LSTM with 250 nodes per layer and initial state optimization. It was optimized using Adam. Expressions generated by the RNN were trained for 15 epochs with a learning rate of 0.5; they were optimized using RMSProp.

B.2. Initial Conditions

Each experiment used three datasets with different sets of initial conditions. These initial conditions are detailed in Tables 2, 3, 4, and 5.

Table 2. Dataset details for Harmonic Oscillator experiments. These initial conditions were generated randomly.

DATASET	m	ω	q_0	p_0
1	1.23	1.65	-0.05	0.42
2	0.63	1.30	0.27	-0.29
3	1.69	0.83	0.06	-0.54

Table 3. Dataset details for Pendulum experiments. These initial conditions were generated randomly.

DATASET	m	ℓ	g	q_{θ_0}	p_{θ_0}
1	0.47	1.23	1.95	1.32	0.23
2	1.10	0.73	1.02	1.37	0.12
3	0.68	0.33	1.59	0.87	0.15

Table 4. Dataset details for Two-Body Gravitational experiments. These initial conditions were generated by hand to obtain systems exhibiting clear interactions between the bodies and reasonable domains.

DATASET	m_i	m_j	$\mathbf{q}_{i0}$	$\mathbf{q}_{j0}$	$\mathbf{p}_{i0}$	$\mathbf{p}_{j0}$
1	1	1	(0.00, 0.00)	(1.00, 1.00)	(0.00, 1.00)	(1.00, 0.00)
2	1	1	(0.00, 0.00)	(0.50, 1.00)	(0.34, 0.30)	(1.00, 0.21)
3	1	1	(0.00, 0.50)	(0.30, -0.24)	(0.40, -1.00)	(-0.80, -2.10)

Table 5. Dataset details for Three-Body Gravitational experiments. These initial conditions were generated by hand to obtain systems exhibiting clear interactions between the bodies and reasonable domains.

DATASET	m_i	m_j	m_k	$\mathbf{q}_{i0}$	$\mathbf{q}_{j0}$	$\mathbf{q}_{k0}$	$\mathbf{p}_{i0}$	$\mathbf{p}_{j0}$	$\mathbf{p}_{k0}$
1	1	1	1	(0.00, 0.00)	(1.00, 1.00)	(-1.00, -1.00)	(0.50, 0.30)	(1.10, -0.40)	(-0.20, 0.50)
2	1	1	1	(0.00, 0.00)	(-3.00, 0.00)	(3.00, 0.00)	(0.00, -1.00)	(0.50, 0.00)	(0.50, 0.00)
3	1	1	1	(0.25, 0.25)	(2.00, 1.50)	(3.00, -2.00)	(0.00, -1.00)	(0.50, -0.15)	(0.80, 0.25)

B.3. Extracting Function Properties via Symplectic Neural Networks

Using the technique described in Section 4.1, we extracted coupling properties for our two-body and three-body experiments. Both systems used the same symplectic network architecture: each block was 8 hidden layers deep with 128 hidden neurons per layer. We used GeLU as our activation function and trained for 3,000 epochs via Adam with a learning rate of 0.0005. This architecture was largely inspired by the successful three-body prediction network in Breen et al. (2020), with some slight hand-adjustments made to hyperparameters. In order to reduce overfitting, we trained the network for 10 steps of prediction at a time. To do this, the dataset of 200 points was broken into a set of 180 for training and a set of 10 for testing, such that the set of 10 for testing has no overlap with a 10-step prediction made from any training point. The batch size was set to the entire set of 180 training points.

B.3.1. Two-Body Results

Using the architecture described in B.3, we began our experiments by attempting to determine the coupling present in the system. We set our maximum tolerable performance decrease to 10%. Using this approach, our technique correctly deduced for both our clean and noisy datasets that $T(\mathbf{p})$ exhibits pairwise coupling whereas $V(\mathbf{q})$ exhibits complete decoupling; these results are contained in Tables 6 and 7. With these results determined, we began the next search for pairwise position composite functions, again with a maximum tolerable performance decrease of 10%. This tolerance was met by both Manhattan distance and Euclidean distance composite functions. However, Euclidean distance outperformed Manhattan distance, so it was selected as the composite function. These results are depicted for our clean and noisy datasets in Tables 8 and 9. Finally, we assessed whether the completely decoupled momenta functions are symmetric across all quantities; enforcing this symmetry increased performance in both our clean and noisy datasets (Tables 10 and Table11). Thus, we were able to determine that the $V(\mathbf{q})$ exhibits pairwise coupling as a function of Euclidean distance and that $T(\mathbf{p})$ exhibits complete decoupling with symmetric subfunctions for all quantities.

B.3.2. Three-Body Results

Our three-body coupling search preceded similarly to the our two-body search, correctly identifying that $V(\mathbf{q})$ exhibits pairwise coupling as a function of Euclidean distance and that $T(\mathbf{p})$ exhibits complete decoupling with symmetric subfunctions. For this search, however, the tolerable decline was set to 5%. Additionally, since there are 3 pairs of particles, we performed a recursive backwards elimination with a tolerance of 1% to see which pairs were necessary–it was determined that all were. Coupling results are contained in Tables 12 and 13 and coupling composite results are contained in Tables 14 and 15.

B.4. Comparison Details

We compared our method to SINDy, SSINNs, and Vanilla Deep Symbolic Regression. Here we provide some further context on the hyperparameters used for each of these methods.

For SINDy, we employed the standard function library for all tasks except for the pendulum, where we added Fourier terms. The regularization parameter was initially set to 0.10. As SINDy cannot directly learn Hamiltonians, we instead learned a system of differential equations. If all coefficients were removed for any of the given differential equations, we decreased the regularization parameter in increments of 0.01 so that this did not happen. If the equations became too long to numerically integrate within 30 seconds, we raised the regularization parameter in increments of 0.01 so that this did not happen. We planned on converting these differential equations to Hamiltonians so that we could time-evolve with a symplectic integrator. However, many of the systems of differential equations were not separable, so we instead unrolled their predictions using RK4.

For SSINNs, we employed the 3rd degree polynomial function library for all tasks except for the pendulum, where we added a shiftable sine term. We used the hyperparameters described in DiPietro et al. (2020), setting a learning rate of 10^{-4} with decay and a regularization term of 10^{-3}. We trained this architecture for 100 epochs.

Finally, we employed vanilla Deep Symbolic Regression. Although not well-suited for physical dynamical systems, we modified this model to produce a series of differential equations, similar to SINDy. Again, these differential equations did not necessarily convert to a separable Hamiltonian, so we trained the model by numerically integrating them with RK4 to obtain a reward. We employed the same architecture and hyperparameters defined by Petersen et al. (2019), using a 250-node RNN with 0.0005 learning rate, 0.95 risk factor, 0.005 entropy coefficient, and 500 batch size. Then, we trained the model for 200 epochs and optimized with Adam. These hyperparameters worked quite well, as the model achieved reasonable performance for not having any physical priors imposed.

B.5. Hardware Information

All SISR models were trained on an Intel i7-9750H CPU, which was found to be comparable in training time to using GPU. Symplectic Neural Networks were trained on an NVIDIA GeForce GTX 1650 GPU.

B.6. Ablation Analysis

In order to assess the importance of each component of our model, we conducted an ablation analysis (Table 16). These results demonstrate that, for higher degree systems such as two-body and three-body motion, all of our model components are necessary to extract the correct governing equation. Indeed, even with strong coupling priors, SISR fails to obtain the correct governing equations for two-body and three-body motion unless symplectic optimization is used, underscoring the importance of our Hamiltonian formalism and symplectic biases when working with small datasets and complex motion. These priors were less important for simpler systems, such as the harmonic oscillator and pendulum.

Table 6. Coupling detection results for the Two-Body experiment with clean data. Note that pairwise coupling is identical to no coupling being enforced, as there is only one pair.

$T(\mathbf{p})$	$V(\mathbf{q})$	NRMSE	Performance Change
No coupling enforced (baseline)	No coupling enforced (baseline)	0.90 ± 0.10	N/A (baseline)
Complete decoupling enforced	No coupling enforced	0.56 ± 0.23	$-38.00\% \pm 25.44\%$
Dimensional coupling enforced	No coupling enforced	0.65 ± 0.24	$-28.28\% \pm 26.02\%$
Pairwise coupling enforced	No coupling enforced	0.90 ± 0.10	N/A (baseline)
Pairwise coupling enforced	Complete decoupling enforced	0.84 ± 0.21	$-7.17\% \pm 23.12\%$

Table 7. Coupling detection results for the Two-Body experiment with noisy data. Note that pairwise coupling is identical to no coupling being enforced, as there is only one pair.

$T(\mathbf{p})$	$V(\mathbf{q})$	NRMSE	Performance Change
No coupling enforced (baseline)	No coupling enforced (baseline)	0.92 ± 0.08	N/A (baseline)
Complete decoupling enforced	No coupling enforced	0.61 ± 0.22	$-34.46\% \pm 23.69\%$
Dimensional coupling enforced	No coupling enforced	0.64 ± 0.31	$-30.27\% \pm 33.82\%$
Pairwise coupling enforced	No coupling enforced	0.92 ± 0.08	N/A (baseline)
Pairwise coupling enforced	Complete decoupling enforced	0.89 ± 0.11	$-3.45\% \pm 11.97\%$

Table 8. Coupling composition function results for the Two-Body experiment with clean data.

Composite Function Enforced	NRMSE	Performance Change
None (baseline)	0.84 ± 0.21	N/A (baseline)
Sum	0.38 ± 0.28	$-54.72\% \pm 32.99\%$
Product	0.39 ± 0.28	$-54.12\% \pm 33.48\%$
Manhattan Distance	0.93 ± 0.06	$10.91\% \pm 6.73\%$
Euclidean Distance	0.94 ± 0.05	$12.15\% \pm 5.67\%$

Table 9. Coupling composition function results for the Two-Body experiment with noisy data.

Composite Function Enforced	NRMSE	Performance Change
None (baseline)	0.89 ± 0.11	N/A (baseline)
Sum	0.38 ± 0.28	$-57.24\% \pm 31.09\%$
Product	0.39 ± 0.28	$-56.71\% \pm 31.61\%$
Manhattan Distance	0.52 ± 0.33	$-41.81\% \pm 36.95\%$
Euclidean Distance	0.84 ± 0.20	$-5.64\% \pm 22.05\%$

Table 10. Coupling symmetry results for the Two-Body experiment with clean data.

Symmetry Enforced	NRMSE	Performance Change
None (baseline)	0.94 ± 0.05	N/A (baseline)
Complete symmetry	0.99 ± 0.01	$5.17\% \pm 0.75\%$

Table 11. Coupling symmetry results for the Two-Body experiment with noisy data.

Symmetry Enforced	NRMSE	Performance Change
None (baseline)	0.84 ± 0.20	N/A (baseline)
Complete symmetry	0.91 ± 0.10	$8.39\% \pm 11.92\%$

Table 12. Coupling detection results for the Three-Body experiment with clean data. Note that when performing the recursive backwards elimination, the baseline is shifted to the same coupling type being employed, except without the specific pair eliminated.

$T(\mathbf{p})$	$V(\mathbf{q})$	NRMSE	Performance Change
No coupling enforced (baseline)	No coupling enforced (baseline)	0.92 ± 0.08	N/A (baseline)
Complete decoupling enforced	No coupling enforced	0.62 ± 0.20	$-32.65\% \pm 22.07\%$
Dimensional coupling enforced	No coupling enforced	0.76 ± 0.21	$-17.72\% \pm 22.61\%$
Pairwise coupling enforced	No coupling enforced	0.98 ± 0.01	$6.54\% \pm 1.60\%$
Pairwise coupling (i,j and j,k)	No coupling enforced	0.95 ± 0.05	$-2.8\% \pm 5.46\%$
Pairwise coupling (i,j and i,k)	No coupling enforced	0.94 ± 0.07	$-3.8\% \pm 7.16\%$
Pairwise coupling (i,k and j,k)	No coupling enforced	0.89 ± 0.10	$-9.7\% \pm 10.02\%$
Pairwise coupling enforced	Complete decoupling enforced	0.88 ± 0.11	$-4.57\% \pm 11.4\%$

Table 13. Coupling detection results for the Three-Body experiment with noisy data. Note that when performing the recursive backwards elimination, the baseline is shifted to the same coupling type being employed, except without the specific pair eliminated.

$T(\mathbf{p})$	$V(\mathbf{q})$	NRMSE	PERFORMANCE CHANGE
NO COUPLING ENFORCED (BASELINE)	NO COUPLING ENFORCED (BASELINE)	0.92 ± 0.12	N/A (BASELINE)
COMPLETE DECOUPLING ENFORCED	NO COUPLING ENFORCED	0.67 ± 0.15	$-26.58\% \pm 15.96\%$
DIMENSIONAL COUPLING ENFORCED	NO COUPLING ENFORCED	0.83 ± 0.15	$-9.52\% \pm 16.47\%$
PAIRWISE COUPLING ENFORCED	NO COUPLING ENFORCED	0.98 ± 0.01	$7.15\% \pm 1.47\%$
PAIRWISE COUPLING (I,J AND J,K)	NO COUPLING ENFORCED	0.96 ± 0.04	$-1.86\% \pm 3.78\%$
PAIRWISE COUPLING (I,J AND I,K)	NO COUPLING ENFORCED	0.94 ± 0.08	$-4.16\% \pm 7.88\%$
PAIRWISE COUPLING (I,K AND J,K)	NO COUPLING ENFORCED	0.92 ± 0.07	$-6.34\% \pm 6.79\%$
PAIRWISE COUPLING ENFORCED	COMPLETE DECOUPLING ENFORCED	0.98 ± 0.02	$6.4\% \pm 2.35\%$

Table 14. Coupling composition function results for the Three-Body experiment with clean data.

COMPOSITE FUNCTION ENFORCED	NRMSE	PERFORMANCE CHANGE
NONE (BASELINE)	0.88 ± 0.11	N/A (BASELINE)
SUM	0.56 ± 0.17	$-35.92\% \pm 19.74\%$
PRODUCT	0.52 ± 0.21	$-41.01\% \pm 23.78\%$
MANHATTAN DISTANCE	0.70 ± 0.21	$-20.74\% \pm 23.71\%$
EUCLIDEAN DISTANCE	0.99 ± 0.01	$12.51\% \pm 1.06\%$

Table 15. Coupling composition function results for the Three-Body experiment with noisy data.

COMPOSITE FUNCTION ENFORCED	NRMSE	PERFORMANCE CHANGE
NONE (BASELINE)	0.98 ± 0.02	N/A (BASELINE)
SUM	0.56 ± 0.17	$-42.45\% \pm 17.75\%$
PRODUCT	0.52 ± 0.21	$-47.31\% \pm 21.78\%$
MANHATTAN DISTANCE	0.70 ± 0.21	$-28.74\% \pm 21.46\%$
EUCLIDEAN DISTANCE	0.99 ± 0.01	$0.57\% \pm 0.70\%$

Table 16. Ablation analysis for SISR, conducted on noisy datasets. NRMSE is reported for single-step predictions on the training dataset. For each dataset, 3 sets of initial conditions were used, with 5 trials conducted for each set of initial conditions, leading to 15 total trials for each experiment. (*) For our ablation of symplectic optimization, constants were optimized using an RK4 integrator instead.

ABLATION	HARMONIC OSCILLATOR	PENDULUM	TWO-BODY	THREE-BODY
NO ABLATIONS	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00	1.00 ± 0.00
NO SYMMETRY	N/A	N/A	0.76 ± 0.18	0.44 ± 0.12
NO COMPOSITE FUNCTIONS	N/A	N/A	0.68 ± 0.25	0.52 ± 0.20
NO SYMMETRY, NO COMPOSITE FUNCTIONS	N/A	N/A	0.66 ± 0.23	0.41 ± 0.11
NO EXTRACTED PRIORS	N/A	N/A	0.65 ± 0.22	0.38 ± 0.08
NO SYMPLECTIC OPTIMIZATION*	1.00 ± 0.00	1.00 ± 0.00	0.87 ± 0.14	0.55 ± 0.21
NO SCALED INITIAL BATCH	1.00 ± 0.00	1.00 ± 0.00	0.99 ± 0.01	0.93 ± 0.10
ALL ABLATIONS	1.00 ± 0.00	1.00 ± 0.00	0.64 ± 0.22	0.37 ± 0.07