Symplectic Integration through Unsupervised Learning

Cheng Tang

February 7, 2020

1 Problem Setting

Consider an N degrees-of-freedom conservative dynamical system with its initial state in the canonical coordinates $(\vec{q}_0, \vec{p}_0) \in \mathbb{R}^{2N}$, we'd like to learn a function $\hat{F}$, which is an approximation to the mapping F:

$$F(\vec{q}_0, \vec{p}_0, t) = (\vec{q}_t, \vec{p}_t), \forall t \in [0, T] \tag{1}$$

Assume only that we are given its Hamiltonian $\mathcal{H}(\vec{q}, \vec{p})$. Denoting $\vec{x}_0 = (\vec{q}_0, \vec{p}_0)$, $\vec{x}_t = (\vec{q}_t, \vec{p}_t)$, assume also that $\vec{x}_0$ is sampled from some distribution $\mathbb{P}$. For a conservative system, consider for example the uniform distribution in the phase-space with the constraint $\mathcal{H} < |E|$.

Such a mapping can be used for numerical integration, where $T \in \mathbb{R}$ is not necessarily small(but if it is, it should converge to learning a finite difference scheme). The output of $\hat{F}$ can be used as an input again to generate long-time dynamics.

2 Unsupervised Learning

The mapping is learned through unsupervised learning, assuming only the analytic form of the Hamiltonian is handed to us. In this setting, we do not need trajectories generated beforehand(or 'ground truth' data during training).

It can be easily tweaked to a supervised learning setting where we have instead have an oracle(e.g. a high order numerical integrator) to produce $(\vec{q}_t, \vec{p}_t)$ the dynamics as training data and we learn to approximate the Hamiltonian itself, and use existing numerical integrators to integrate the dynamics.

3 Symplectic Layers

Since the Hamiltonian flow is symplectic, we parametrize the mapping generated by the Hamiltonian dynamics as:

$$\hat{F} = \Phi_1^h \circ \Phi_2^h \circ \ldots \circ \Phi_{N_{blocks}}^h \circ \Phi_{N_{blocks}}^h \circ \ldots \circ \Phi_2^h \circ \Phi_1^h \tag{2}$$

3.1 Symplectic Blocks

The determinant of the Jacobian of symplectic maps satisfy:

$$(J^T)\Omega(J) = \Omega, \tag{3}$$

Where

$$\Omega = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} \tag{4}$$

Each Φ_k in (2) is a symplectic transformation. A symplectic nonlinear layer can be build using the additive coupling layer introduced in [Din], which was used in [Bon] for building symplectic transformation.

The idea is to use these layers as building blocks to approximate the symplectic Hamiltonian flow. In its original form, The additive coupling layer already has the useful properties that:

- it is reversible
- the determinant of its Jacobian is 1

More restrictions are needed for it to be symplectic(see below).

Concretely, define:

$$\Phi_k(q, p, \tau) = \Phi_a^\tau \circ \Phi_b^\tau \circ \Phi_c^\tau \tag{5}$$

where

$$\Phi_a^\tau : (q, p) \to (q + NN_A(p, \frac{\tau}{2})\frac{\tau}{2}, p) \tag{6}$$

$$\Phi_b^\tau : (q, p) \to (q, p + NN_B(q, \tau)\tau) \tag{7}$$

$$\Phi_c^\tau = \Phi_a^\tau \tag{8}$$

$\tau = \frac{t}{2N_{blocks}}$ gives a scale of the mapping and it is how t enters the transformation. Note the importance of the time dependence of $NN_{A,B}$. This makes the transformation powerful. Otherwise for any t at a given starting $\vec{x}_0$, $\vec{x}_t$ depends linearly in dt, which degrades into something similar to a first order Euler step.

The transform $\hat{F}(q, p, t)$ is symplectic, reversible, symmetric in time(achieved by weight sharing), which are all properties of the Hamiltonian dynamic for conservative system. (See P149 of [Hai] for symmetric composition of symmetric methods.)

For systems with more than 1 degree of freedom, for (3) to be true NN_A and NN_B need to satisfy:

$$\partial_i NN = \partial_j NN \tag{9}$$

This has been addressed addressed using what is called the 1-layer(or possibly more) irrotational MLP in [Bon]. I added a symmetric skip connection for these

network that also satisfy the above condition. Explicitly, $NN_{A,B}$ are 1-hidden-layer networks with 1-hidden layer with number of hidden units n_h chosen to be N or $2N$.

Concretely, we have:

$$NN_{A,B}(X,\tau) = \sigma(\sigma(\tau W_0 + XW_1 + b_1)W_2 + b_2)W_3 + b3 + \frac{X}{2}(W_4 + W_4^T) \quad (10)$$

Where the activation function is chosen to be tanh.

The weight parameters $W_0 \in \mathbb{R}^{1\times n_h}$, $W_1 = W_3^T \in \mathbb{R}^{N\times n_h}$, $W_2 = diag(D), D \in \mathbb{R}^{n_h}$ is a diagonal matrix and $W_4 \in \mathbb{R}^{N\times N}$

With the above, we have the symplectic transformation by construction, and we wish to optimize these parameters and minimize the loss function that I will define below.

4 Loss function

Denote $(\hat{q}_t, \hat{p}_t) = \hat{F}(q_0, p_0, t; \theta)$ The loss function we need to minimize encodes the Hamilton's equation:

$$\mathcal{L}(q_0, p_0, t; \theta) = (\frac{\partial \mathcal{H}(\hat{q}_t, \hat{p}_t)}{\partial \hat{q}_t} + \frac{d\hat{p}_t}{dt})^2 + (\frac{\partial \mathcal{H}(\hat{q}_t, \hat{p}_t)}{\partial \hat{p}_t} - \frac{d\hat{q}_t}{dt})^2 \quad (11)$$

Trivial solution: I conjecture that the trivial solution where $\hat{F} = 0$ is prevented due to the symplecticity of the transformation. If we sample $\vec{x}_0$ uniformly(so that we are just solving a initial value problem with fixed bounday condition) from the phase space and symplectic transformation preserves the volume form, then $F(\vec{x}_0)$ would have the same 'volume' in the phase space as $\vec{x}_0$ and a trivial solution is thus prevented when we minimize the loss which is a sum of the above quantity with $\vec{x}_0$ sampled from some distribution.

5 Training

Training is done by sampling (q_0, p_0, t) and minimizing the Loss. No labeled data is needed during training. Using L-BFGS with learning-rate = 0.2, memory = 100 to train for 2000 iterations.

6 Experiment

Set $N = 10$, the number of hidden units in each $NN_{A,B}$ to be 20, and sampling 2048 pairs of $(\vec{q}, \vec{p})$ are drawed from a normal distribution around $\vec{Q}, \vec{P}$, which is also randomly selected. T is sampled uniformly from $\mathcal{U}(0, 10)$.

A better approach to sample $\vec{x}_0$ would be restricting the maximum energy of the system and sample uniformly.

Results are shown in below.

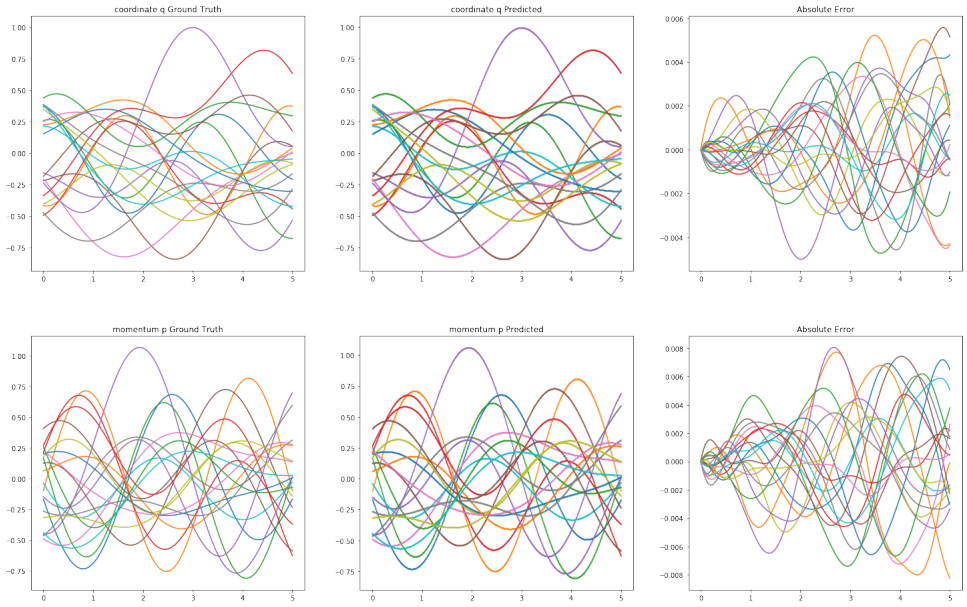

Figure 1: Testing: Integrated Dynamics for a chosen fixed starting position. Ground Truth is generated using Leap-frog integration with a tiny step.

7 Comparison with related work

The source of error of the learned dynamic comes from how well the network is able to learn the function, and no truncation error since it does not make use of any finite difference approximations like ([Gre], [Che])

This framework applies regardless of the separability of the Hamiltonian, since we assume only that we approximate a symplectic transformation, and it does not rely on an oracle for generating ground truth data(usually numerical integrators; these easy to construct for separable Hamiltonians but hard for non-separable ones) when making predictions like many others ([Gre], [Che], [Tot]).

Since the dynamic is learned in an unsupervised learning setting, we do not require alternative integrators to generate the labeled data like most other work and the network is essentially integrating the dynamics.

Comparing to [Mat], which also learns the dynamis through unsupervised learning, this setting is able to embed the symplecticity in the network architecture and relax the fixed initial condition by sampling from different starting points.

Improvements Needed:

- autograd returns aggregated gradients, which makes backpropgating $\frac{d\hat{q}}{dt}$ and $\frac{d\hat{p}}{dt}$ slow. Pytorch doesn't seem to support this yet but there is someone working on this. Tensorflow seems to already have support for this.

- Optimization error could be reduced using algorithms that address the saddle point.

8 Learn the Hamiltonian through Supervised Learning

Another orthogonal setting would be: assuming we don't have the analytic form of the Hamiltonian system, and we are given an oracle(a very good approximation to F itself; can be a high-order symplectic integrator) that generates training data. Then we can change the loss to be:

$$\mathcal{L}(q_0, p_0, t; \theta) = (\frac{\partial \hat{\mathcal{H}}(\hat{q}_t, \hat{p}_t)}{\partial \hat{q}_t} + \frac{d\hat{p}_t}{dt})^2 + (\frac{\partial \hat{\mathcal{H}}(\hat{q}_t, \hat{p}_t)}{\partial \hat{p}_t} - \frac{d\hat{q}_t}{dt})^2 + (\hat{q}_t - q_t)^2 + (\hat{p}_t - p_t)^2 \tag{12}$$

Where $\hat{H}$ can be parametrized by a simple MLP, and (q_t, p_t) is the data generated by the oracle. After learning the dynamics, we could use the learned $\hat{H}$ to generate the dynamics. This is in line with the Hamiltonian neural network paper and a lot of others that use an existing integrator to generate the dynamics after learning the Hamiltonian itself. The HNN paper did not seem logically sound to me since it is learning directly $(\frac{\partial H}{\partial p}, \frac{\partial H}{\partial q})$ from $(\dot{q}, \dot{p})$. But assuming you know $(\dot{q}, \dot{p})$ you already know how to generate the dynamics using integrators. Zhengdao's paper resolved this perfectly because obviously we need to learn this information$((\dot{q}, \dot{p}))$ from data.

For general Hamiltonian though, getting the oracle may be itself a non-trivial task already. Consider 1D non-separable Hamiltonian $\frac{(Q^2+1)(P^2+1)}{2}$, which is usually integrated implicitly and [Tao] proposed a trick that integrates it explicitly and symplectically. In the unsupervised learning setting, this can be done with much ease. We don't need to generate data for learning, and we don't need an existing integrator to make prediction.

References

[Bon] Roberto Bondesan. *Learning Symmetries of Classical Integrable Systems*. URL: `https://arxiv.org/pdf/1906.04645.pdf`.

[Che] Zhengdao Chen. *Symplectic Recurrent Neural Networks*. URL: `https://arxiv.org/pdf/1909.13334.pdf`.

[Din] Laurent Dinh. *NICE: NON-LINEAR INDEPENDENT COMPONENTS ESTIMATION*. URL: `https://arxiv.org/pdf/1410.8516.pdf`.

[Gre] Sam Greydanus. *Hamiltonian Neural Networks*. URL: `https://arxiv.org/pdf/1906.01563.pdf`.

[Hai] Ernst Hairer. *Geometric Numerical Integration*.

[Mat] M. Mattheakis. *Physical Symmetries Embedded in Neural Networks*. URL: https://arxiv.org/pdf/1904.08991.pdf.

[Tao] Molei Tao. *Explicit symplectic approximation of nonseparable Hamiltonians: algorithm and long time performance*. URL: https://arxiv.org/abs/1609.02212.

[Tot] Peter Toth. *HAMILTONIAN GENERATIVE NETWORKS*. URL: https://arxiv.org/pdf/1909.13789.pdf.