Structure-Preserving Neural Networks for the N-Body Problem

ECCOMAS Congress 2022

Philipp Horn and Barry Koren (Eindhoven University of Technology)
Veronica Saz Ulibarrena and Simon Portegies Zwart (Leiden Observatory)

Mathematics and Computer Science
Motivation to use NNs for the N-body problem

- 3-body problem is chaotic, without an analytical solution
- Requires small time steps to solve it numerically
- Main bottleneck in large $N$-body simulations
- An NN surrogate could speed up calculations
Motivation to use NNs for the N-body problem

- 3-body problem is chaotic, without an analytical solution
- Requires small time steps to solve it numerically
- Main bottleneck in large $N$-body simulations
- An NN surrogate could speed up calculations

Figure: First proof of concept [1]

What is a Hamiltonian System?

The ODE describing a Hamiltonian system can be written as:

\[
\begin{pmatrix}
\dot{p} \\
\dot{q}
\end{pmatrix} = -J \nabla H(p, q), \quad \text{with} \quad J = \begin{pmatrix}
0 & I_d \\
-I_d & 0
\end{pmatrix}
\]

\[
= \begin{pmatrix}
-\nabla_q H(p, q) \\
\nabla_p H(p, q)
\end{pmatrix}, \quad p, q \in \mathbb{R}^d.
\]
What is a Hamiltonian System?

The ODE describing a Hamiltonian system can be written as:

\[
\begin{pmatrix}
\dot{p} \\
\dot{q}
\end{pmatrix} = - J \nabla H(p, q), \quad \text{with } J = \begin{pmatrix}
0 & I_d \\
-I_d & 0
\end{pmatrix}
\]

\[
= \begin{pmatrix}
-\nabla_q H(p, q) \\
\nabla_p H(p, q)
\end{pmatrix}, \quad p, q \in \mathbb{R}^d.
\]

In some Hamiltonian systems the Hamiltonian \( H : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R} \) is separable, meaning:

\[
H(p, q) = T(p) + U(q).
\]

Then, \( T : \mathbb{R}^d \to \mathbb{R} \) is referred to as the kinetic energy and \( U : \mathbb{R}^d \to \mathbb{R} \) as the potential energy.
Two key properties of Hamiltonian systems

1. Total energy is conserved along trajectories:

\[ H(p_0, q_0) = H(p(t), q(t)), \quad t \in \mathbb{R}_+ \]  \hspace{1cm} (3)

where \((p(t), q(t))^T\) is the solution of ODE (1) with initial condition \((p_0, q_0)^T\)
Two key properties of Hamiltonian systems

1. Total energy is conserved along trajectories:

\[ H(p_0, q_0) = H(p(t), q(t)), \quad t \in \mathbb{R}_+ \]  \hspace{1cm} (3)

where \((p(t), q(t))^T\) is the solution of ODE (1) with initial condition \((p_0, q_0)^T\).

2. Symplecticity of the flow map \(\varphi_h : x(t) \mapsto x(t + h)\) meaning:

\[
\left( \frac{\partial \varphi_h}{\partial x} \right)^T J \frac{\partial \varphi_h}{\partial x} = J, \quad x = \begin{pmatrix} p \\ q \end{pmatrix} \in \mathbb{R}^{2d}. \]  \hspace{1cm} (4)
How to solve a Hamiltonian system

Classically one knows $H$ and, given the initial conditions $(p_0, q_0)^T$, the objective is to calculate trajectories $(p(t), q(t))^T$ or only $(p(t_{\text{end}}), q(t_{\text{end}}))^T$ at a final time $t_{\text{end}}$. This is done using numerical integrators.

Figure: Iterative scheme to solve an ODE using a numerical integrator.
Semi-implicit (symplectic) Euler method

The symplectic Euler method for Hamiltonian systems given $p_n$ and $q_n$ is:

\[
\begin{align*}
    p_{n+1} &= p_n - h \nabla_q H(p_{n+1}, q_n) \\
    q_{n+1} &= q_n + h \nabla_p H(p_{n+1}, q_n)
\end{align*}
\]
Semi-implicit (symplectic) Euler method

The symplectic Euler method for Hamiltonian systems given $p_n$ and $q_n$ is:

$$
\begin{align*}
    p_{n+1} &= p_n - h \nabla_q H(p_{n+1}, q_n) \\
    q_{n+1} &= q_n + h \nabla_p H(p_{n+1}, q_n)
\end{align*}
$$

For separable Hamiltonian systems it becomes an explicit method:

$$
\begin{align*}
    x_n := \begin{pmatrix} p_n \\ q_n \end{pmatrix} &\rightarrow \begin{pmatrix} p_n - \nabla U(q_n)h \\ q_n \end{pmatrix} =: \begin{pmatrix} p_{n+1} \\ q_n \end{pmatrix} \rightarrow \\
    &\rightarrow \begin{pmatrix} q_n + \nabla T(p_{n+1})h \\ p_{n+1} \end{pmatrix} =: \begin{pmatrix} p_{n+1} \\ q_{n+1} \end{pmatrix} =: x_{n+1}
\end{align*}
$$

(5)
How to solve a Hamiltonian system using NNs

To solve a Hamiltonian system without knowledge of $H$ but from data with the use of NNs there are two different approaches. The first simply replaces the integrator with an NN. The second one does not work in an iterative manor. Instead it also receives a time $t$ as input and predicts $(p(t), q(t))^T$ directly from $(p_0, q_0)^T$.

![Diagram]

Figure: Iterative scheme to solve an ODE using an NN compared to a scheme requiring $t$ as an input.
Hamiltonian neural network by Greydanus et al. [2]

\[
\begin{align*}
H &= \nabla p H \quad \nabla q H \\
\dot{q}_n &= -\nabla q H \\
\dot{p}_n &= \nabla p H \\
\end{align*}
\]

Hamiltonian neural network by Chen et al. [3]

A standard SympNet by Jin et al. [4]

- Each layer: $\mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$ and is a symplectic map
- A concatenation of symplectic maps is again symplectic $\Rightarrow$ A SympNet is a symplectic map.

\[ p_n \rightarrow \hat{p}_1 \rightarrow \hat{p}_2 \rightarrow \hat{p}_3 \rightarrow \ldots \rightarrow \hat{p}_{2M-1} \rightarrow p_{n+1} \]

\[ q_n \rightarrow \hat{q}_1 \rightarrow \hat{q}_2 \rightarrow \hat{q}_3 \rightarrow \ldots \rightarrow \hat{q}_{2M-1} \rightarrow q_{n+1} \]

SympNets - Gradient layers

A gradient layer has the trainable parameters: $K \in \mathbb{R}^{n \times d}$, $b \in \mathbb{R}^n$ and $a \in \mathbb{R}^n$. Also, an activation function $\sigma$ has to be chosen beforehand. Usually the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ is used. With this (and a slight abuse of matrix vector multiplication notation) the upper and lower gradient layers are defined as:

$$G_{up}(p, q) := \begin{bmatrix} I & \hat{\sigma}_{K,a,b} \\ 0 & I \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} p + K^T \text{diag}(a) \sigma(Kq + b) \\ q \end{bmatrix}$$ (6)

$$G_{low}(p, q) := \begin{bmatrix} I & 0 \\ \hat{\sigma}_{K,a,b} & I \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} = \begin{bmatrix} p \\ q + K^T \text{diag}(a) \sigma(Kp + b) \end{bmatrix}$$ (7)

The name of this layer comes from the fact that $\hat{\sigma}_{K,a,b}$ can approximate any gradient of a function $V : \mathbb{R}^d \rightarrow \mathbb{R}$ (so any $\nabla V$).
Two symplectic gradient layers in detail

Focusing on two alternating gradient layers and using

\[ \nabla_q a^T \Sigma (Kq + b) = K^T \text{diag}(a) \sigma (Kq + b) \]  

(8)

\[ \nabla_p \tilde{a}^T \Sigma (\tilde{K}p + \tilde{b}) = \tilde{K}^T \text{diag}(\tilde{a}) \sigma (\tilde{K}p + \tilde{b}) \]  

(9)
Two gradient layers learn a Hamiltonian

Two layers, starting with an upper and followed by a lower gradient layer can be written as:

\[
\begin{pmatrix}
    p_i \\
    q_i
\end{pmatrix}
\rightarrow
\begin{pmatrix}
    p_i + \nabla_q a^T \Sigma (K q_i + b) \\
    q_i
\end{pmatrix}
=:
\begin{pmatrix}
    p_{i+1} \\
    q_i
\end{pmatrix}
\rightarrow
\begin{pmatrix}
    p_{i+1} \\
    q_i + \nabla p \tilde{a}^T \Sigma (\tilde{K} p_{i+1} + \tilde{b})
\end{pmatrix}
=:
\begin{pmatrix}
    p_{i+1} \\
    q_{i+1}
\end{pmatrix}
\]  

(10)
Two gradient layers learn a Hamiltonian

Two layers, starting with an upper and followed by a lower gradient layer can be written as:

\[
\begin{align*}
(p_i) \rightarrow (p_i + \nabla_q a^T \Sigma (Kq_i + b)) &=: (p_{i+1}) \rightarrow \\
(q_i) \rightarrow (q_i + \nabla p \tilde{a}^T \Sigma (\tilde{K}p_{i+1} + \tilde{b})) &=: (q_{i+1})
\end{align*}
\] (10)

These are the updates of the symplectic Euler method for the Hamiltonian

\[
H(p, q) = \frac{1}{h} \tilde{a}^T \Sigma (\tilde{K}p + \tilde{b}) - \frac{1}{h} a^T \Sigma (Kq + b)
\] (11)
Two gradient layers as an HNN

\[ \hat{\mathbf{p}}_i, \hat{\mathbf{q}}_i \xrightarrow{T} \hat{\mathbf{p}}_i, \hat{\mathbf{q}}_i + 1 \xrightarrow{U} H \xrightarrow{\nabla_q H} - \nabla q H \xrightarrow{\nabla p H} \text{Symplectic Euler} \xrightarrow{\hat{\mathbf{p}}_{i+1}} \hat{\mathbf{q}}_{i+1} \]
SympNet as multiple HNNs in series

\[ \hat{p}_n, \hat{q}_n \rightarrow \hat{p}_1, \hat{q}_1 \rightarrow \hat{p}_2, \hat{q}_2 \rightarrow \cdots \rightarrow \hat{p}_{n+1}, \hat{q}_{n+1} \]
Data for first numerical experiment

- Three bodies with equal masses
- Disturbed circular orbit

- In the $x$-$y$-plane
- 5000 trajectories until $t_{end} = 10$
Data for first numerical experiment

- Three bodies with equal masses
- Disturbed circular orbit
- In the $x$-$y$-plane
- 5000 trajectories until $t_{\text{end}} = 10$
SympNets vs physics-unaware neural networks

Figure: Relative mean square error (MSE) for a physics-unaware multilayer perceptron (MLP) and a SympNets trained on data until $t_{\text{end}} = 4.5$ (vertical line).
SympNets vs the symplectic Euler method

Figure: Trajectories of a planet orbiting a heavy central body. Predicted by the symplectic Euler method and a SympNet.
Conclusion

• We have shown a new connection between SympNets and HNNs
• SympNets generalize better than physics-unaware MLPs
• NNs for Hamiltonian systems can outperform numerical integrators
Conclusion

• We have shown a new connection between SympNets and HNNs
• SympNets generalize better than physics-unaware MLPs
• NNs for Hamiltonian systems can outperform numerical integrators

Outlook

• Improved SympNet topology to be more exact, also inside the range of training data
• Tackle the more chaotic 3-body problem using a non-iterative approach
Conclusion

• We have shown a new connection between SympNets and HNNs
• SympNets generalize better than physics-unaware MLPs
• NNs for Hamiltonian systems can outperform numerical integrators

Outlook

• Improved SympNet topology to be more exact, also inside the range of training data
• Tackle the more chaotic 3-body problem using a non-iterative approach

Thank you for your attention!