Hamiltonian Neural Networks

Seongok Ryu
KAIST Chemistry

Preliminaries on Physics Principles

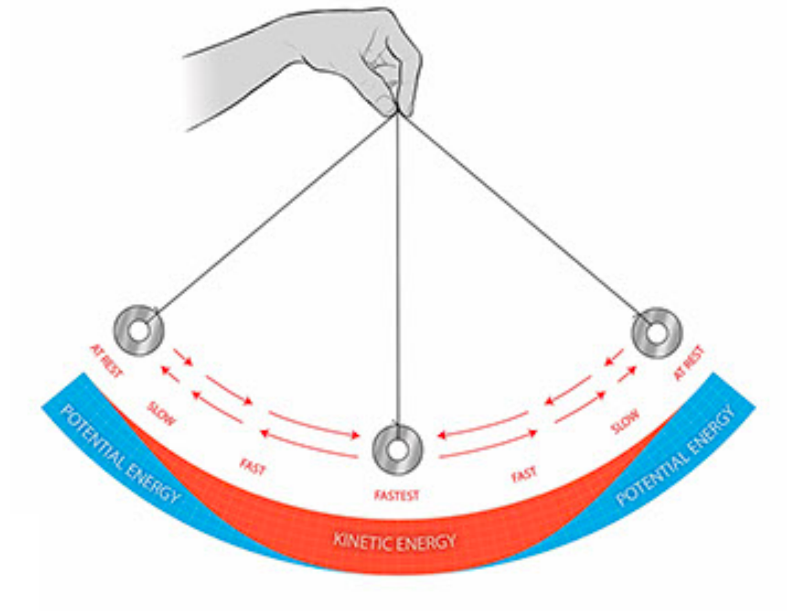
Preliminaries on Physics Principles

Hamiltonian system and Hamilton's equations

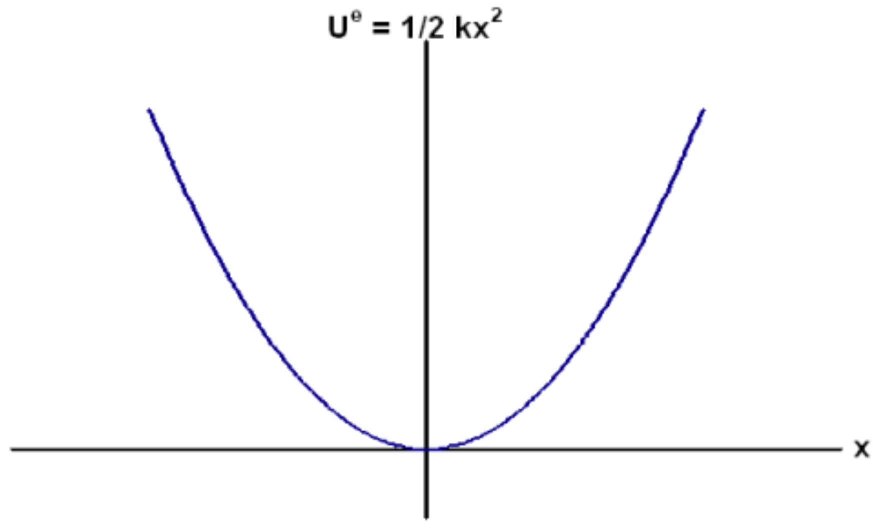
- Hamiltonian

$$H(\mathbf{p},\mathbf{q}) = K(\mathbf{p}) + V(\mathbf{q})$$

  - ✓ $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}\mathbf{M}\mathbf{p}^T$: Kinetic energy of the system
  - ✓ $V(\mathbf{q})$: Potential energy of the system



- Equation of motions
  - ✓ $\mathbf{F} = \frac{d\mathbf{p}}{dt} = m\frac{d\mathbf{v}}{dt} = m\mathbf{a}$
  - ✓ $\mathbf{F} = \frac{\partial V}{\partial \mathbf{q}} = \frac{\partial H}{\partial \mathbf{q}}$



Preliminaries on Physics Principles

Hamiltonian system and Hamilton's equations

- Hamiltonian system

  "A **Hamiltonian system** is a dynamical system governed by **Hamilton's equation**.

  Formally, a Hamiltonian system is a dynamical system completely described by the scalar function $H(\mathbf{q}, \mathbf{p}, t)$"

- Hamilton's equations

  The evolution of system is governed by **Hamilton's equation**

$$\frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}, \qquad \frac{d\mathbf{q}}{dt} = +\frac{\partial H}{\partial \mathbf{p}}$$

  The trajectory of system $\mathbf{r}(t) = (\mathbf{q}(t), \mathbf{r}(t)) \in \mathbb{R}^{2N}$ is the solution of initial value problem defined by the Hamilton's equations and the initial condition $\mathbf{r}(0)$.

Preliminaries on Physics Principles

Physics simulations – Molecular Dynamics

- Molecular Dynamics (MD)

  MD is a computer simulation method for **analyzing the physical movement of atoms and molecules**. The atoms and molecules are allowed to interact for a fixed period of time, giving a view of the dynamic evolution of the system.

  → Solving Hamilton's equation (or equation of motions) to estimate the trajectory of the system

- Ergodicity

  The evolution of one molecular dynamics simulation may be used to determine macroscopic thermodynamic properties of the system; **the time average of an ergodic system correspond to (micro-) canonical ensemble averages**.

$$\langle A \rangle_t = \langle A \rangle_{\mathbf{r}}$$

https://www.youtube.com/watch?v=sD6vyfTtE4U

Hamiltonian Neural Networks

Hamiltonian Neural Networks

Two key challenges in physics modelling

- Discrete time steps
  - ✓ The evolution of system is ran over continuous time dynamics:

$$(\mathbf{q}_1, \mathbf{p}_1) = (\mathbf{q}_0, \mathbf{p}_0) + \int_{t_0}^{t_1} \mathbf{S}(\mathbf{p}, \mathbf{q})dt$$

    where $\mathbf{S}(\mathbf{p}, \mathbf{q})$ is the time-derivatives of the coordinates of the system.
  - ✓ Numerical integrators relax the above integration by taking discrete time steps, which relies on small enough time step. ($error(\mathbf{p}, \mathbf{q}) = \mathcal{O}\left((\Delta t)^2\right)$ for Verlet algorithms)

- Energy conservative property
  - ✓ Hamilton's equation governs the dynamics of the energy conservative system:

$$\frac{d\mathbf{p}}{dt} = -\frac{\partial H}{\partial \mathbf{q}}, \qquad \frac{d\mathbf{q}}{dt} = +\frac{\partial H}{\partial \mathbf{p}}$$

Greydanus, Sam, Misko Dzamba, and Jason Yosinski. "Hamiltonian Neural Networks." *arXiv preprint arXiv:1906.01563* (2019).

Hamiltonian Neural Networks

Overall scheme

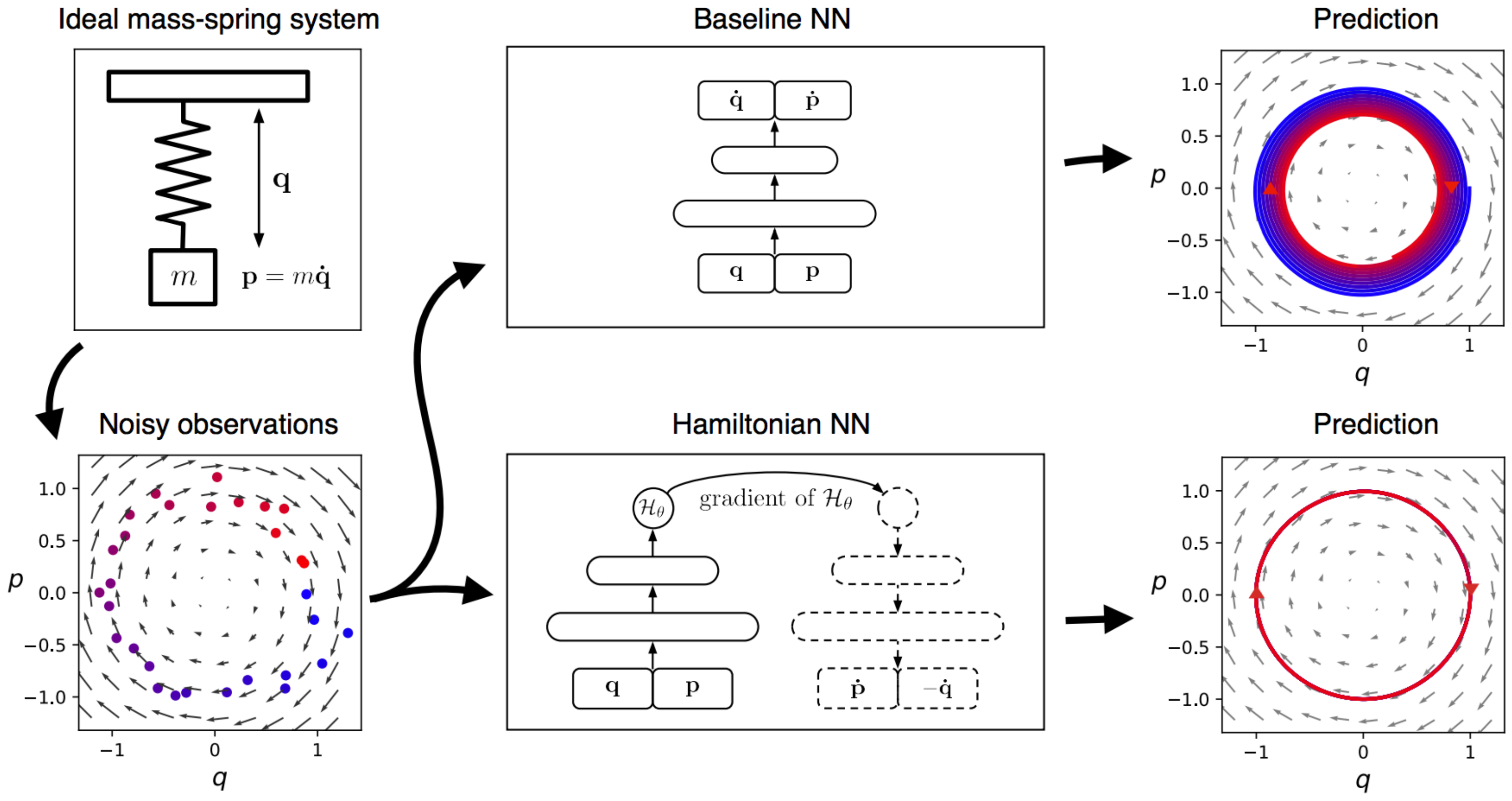


Figure 1: Learning the Hamiltonian of a mass-spring system. The variables q and p correspond to position and momentum coordinates. As there is no friction, the baseline's inner spiral is due to model errors. By comparison, the Hamiltonian Neural Network learns to *exactly* conserve a quantity that is analogous to total energy.

Hamiltonian Neural Networks

Baseline NN

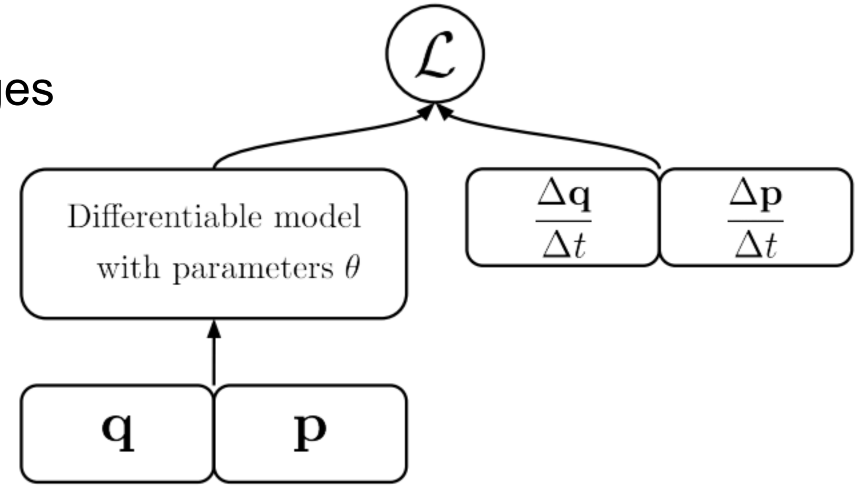
- Set its learning objective as to minimize the error of coordinate changes

$$\mathcal{L}(\theta) = \left\| f_{\theta,\mathbf{q}}(\mathbf{q},\mathbf{p}) - \frac{\Delta\mathbf{q}}{\Delta t} \right\|^2 + \left\| f_{\theta,\mathbf{p}}(\mathbf{q},\mathbf{p}) - \frac{\Delta\mathbf{p}}{\Delta t} \right\|^2$$

- This approaches involves the following two problems

  i) discretization error in predictions

  ii) does not guarantee the energy conservation, i.e. Hamilton's equation


$\mathcal{L}$
Differentiable model with parameters θ
$\frac{\Delta\mathbf{q}}{\Delta t}$
$\frac{\Delta\mathbf{p}}{\Delta t}$
$\mathbf{q}$
$\mathbf{p}$

(a) Baseline NN

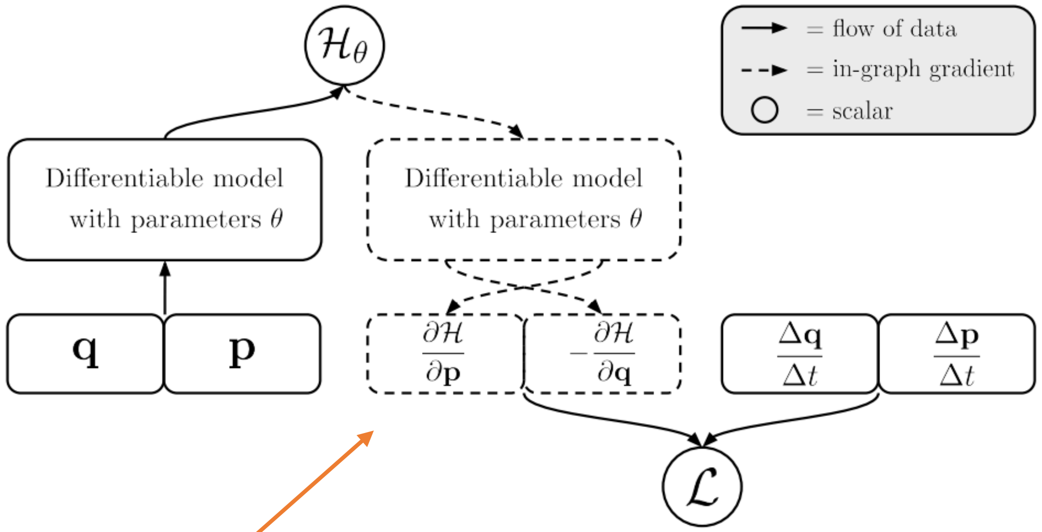
Hamiltonian Neural Networks

Hamiltonian NN

- Hamiltonian NN aims to learn the Hamiltonian of the given system.
- The output given the coordinates is the Hamiltonian of the system H_θ.
- Then, its learning objective is to force the prediction system follows the Hamilton's equation

$$\mathcal{L}_{\mathrm{HNN}}(\theta) = \left\|\frac{\partial H}{\partial \mathbf{p}} - \frac{\Delta \mathbf{q}}{\Delta t}\right\|^2 + \left\|\frac{\partial H}{\partial \mathbf{q}} + \frac{\Delta \boldsymbol{p}}{\Delta t}\right\|^2$$

- We can understand that its learning objective impose strong inductive bias which governs physics law on the prediction system.



$\mathcal{H}_\theta$
= flow of data
= in-graph gradient
= scalar
Differentiable model with parameters θ
Differentiable model with parameters θ
$\mathbf{q}$
$\mathbf{p}$
$\frac{\partial \mathcal{H}}{\partial \mathbf{p}}$
$-\frac{\partial \mathcal{H}}{\partial \mathbf{q}}$
$\frac{\Delta \mathbf{q}}{\Delta t}$
$\frac{\Delta \mathbf{p}}{\Delta t}$
$\mathcal{L}$

(b) Hamiltonian NN

- Model input : $\mathbf{p}, \mathbf{q}$,
- Model output : H
- Thus, we can apply autograd, tf.gradient to estimate $\frac{\partial H}{\partial \mathbf{p}}$ and $\frac{\partial H}{\partial \mathbf{q}}$.

Greydanus, Sam, Misko Dzamba, and Jason Yosinski. "Hamiltonian Neural Networks." *arXiv preprint arXiv:1906.01563* (2019).

Hamiltonian Neural Networks

Experiment 1

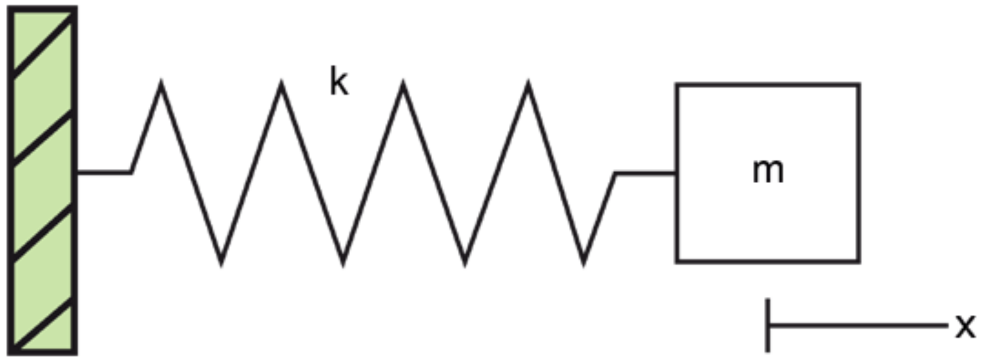
1. Ideal mass-spring system

$$H = \frac{\mathbf{p}^2}{2m} + \frac{1}{2}k\mathbf{q}^2$$

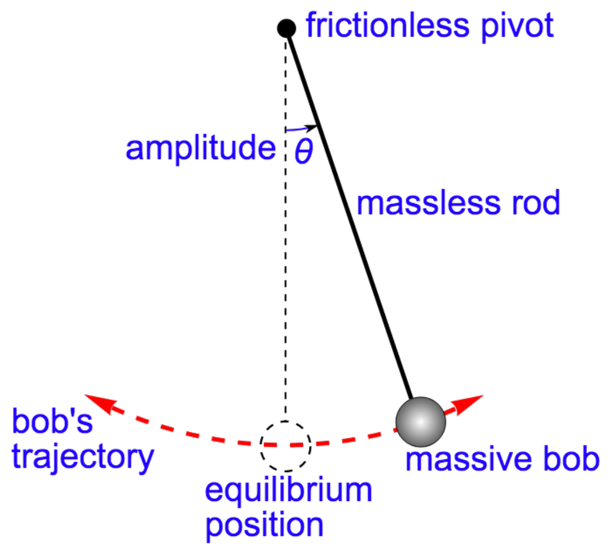
2. Ideal pendulum system

$$H = \frac{l^2\mathbf{p}^2}{2m} + 2mgl(1 - \cos\mathbf{q})$$

3. Real pendulum system
   - ✓ Unknown Hamiltonian
   - ✓ Observations from noisy and biased real-world



Ideal mass-spring system



Ideal pendulum system

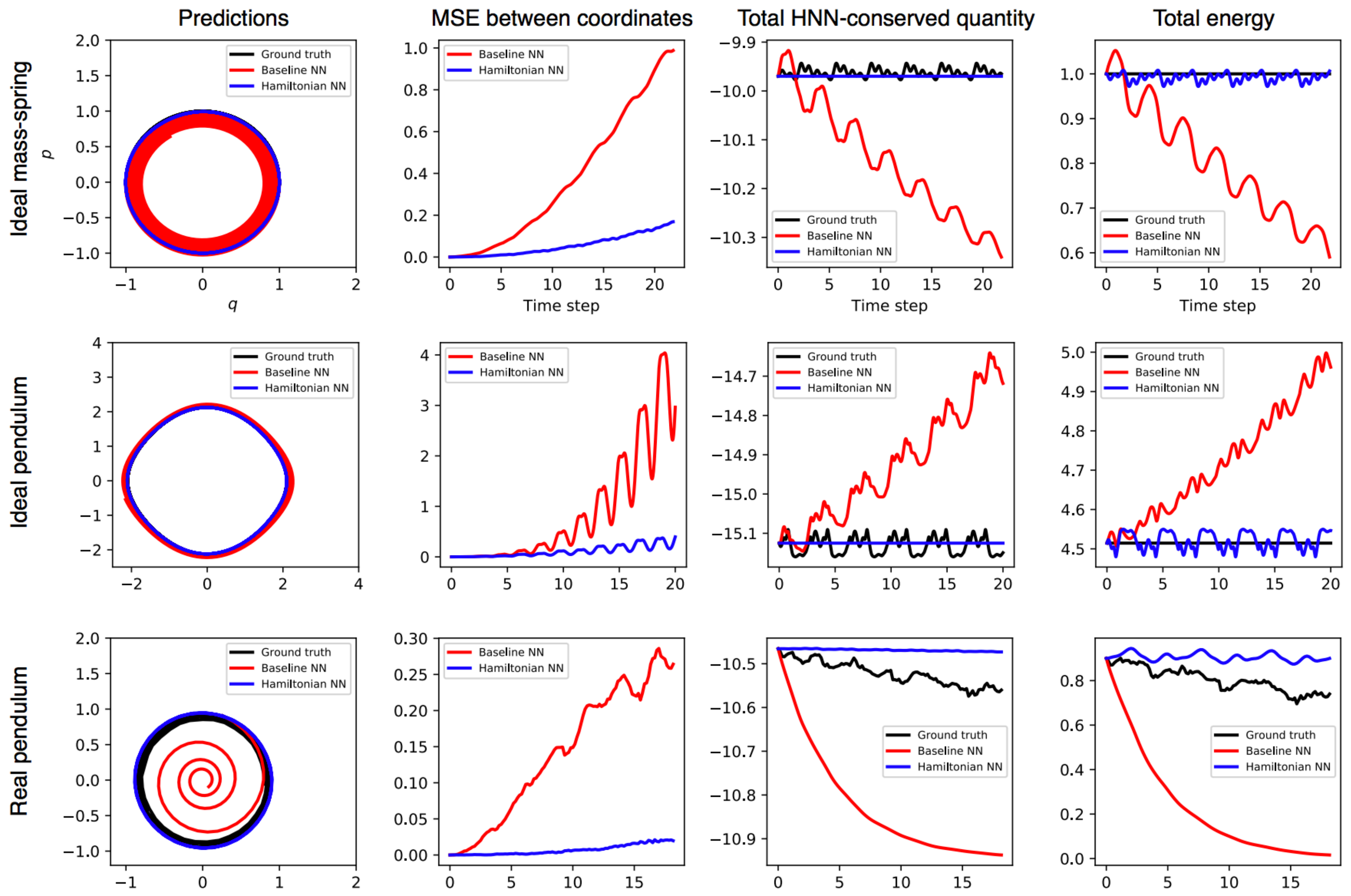
Schmidt, Michael, and Hod Lipson. "Distilling free-form natural laws from experimental data." *Science* 324.5923 (2009): 81-85.

Hamiltonian Neural Networks

Experimental results

$$H = \frac{\mathbf{p}^2}{2m} + \frac{1}{2}k\mathbf{q}^2$$

$$H = \frac{l^2\mathbf{p}^2}{2m} + 2mgl(1 - \cos\mathbf{q})$$



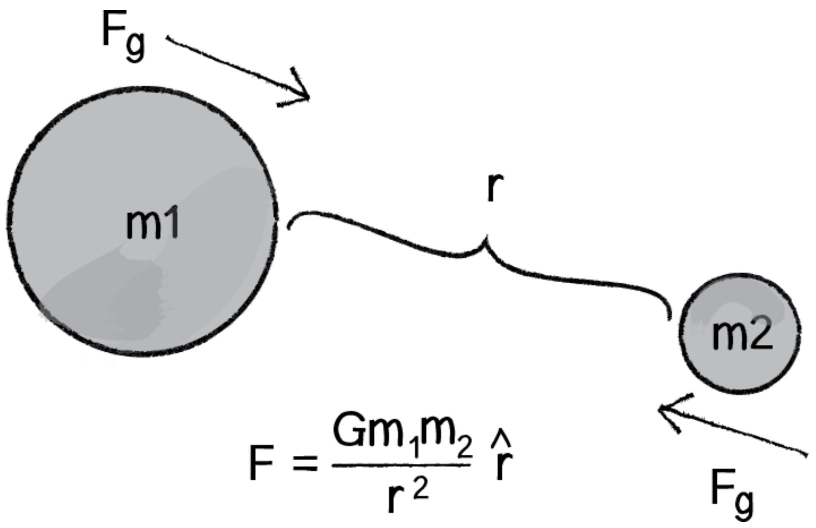
Hamiltonian Neural Networks

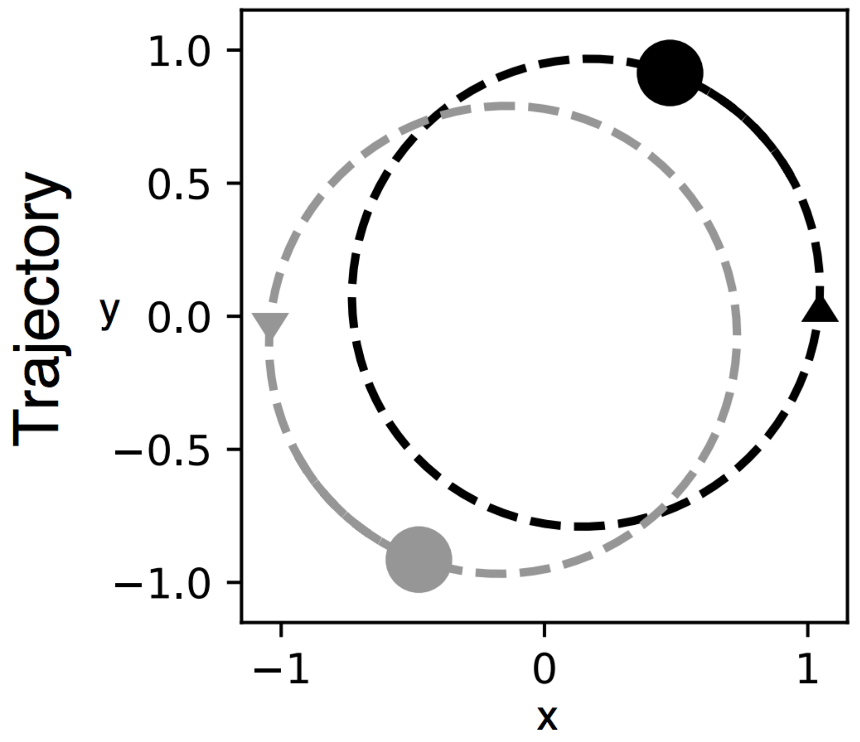
Experiment 2

- Two-particle gravitational system

$$H = \frac{\mathbf{p}_{\mathrm{CM}}^2}{m_1 + m_2} + \frac{\mathbf{p}_1^2 + \mathbf{p}_2^2}{2\mu} + g\,\frac{m_1 m_2}{|\mathbf{q}_1 - \mathbf{q}_2|^2}$$

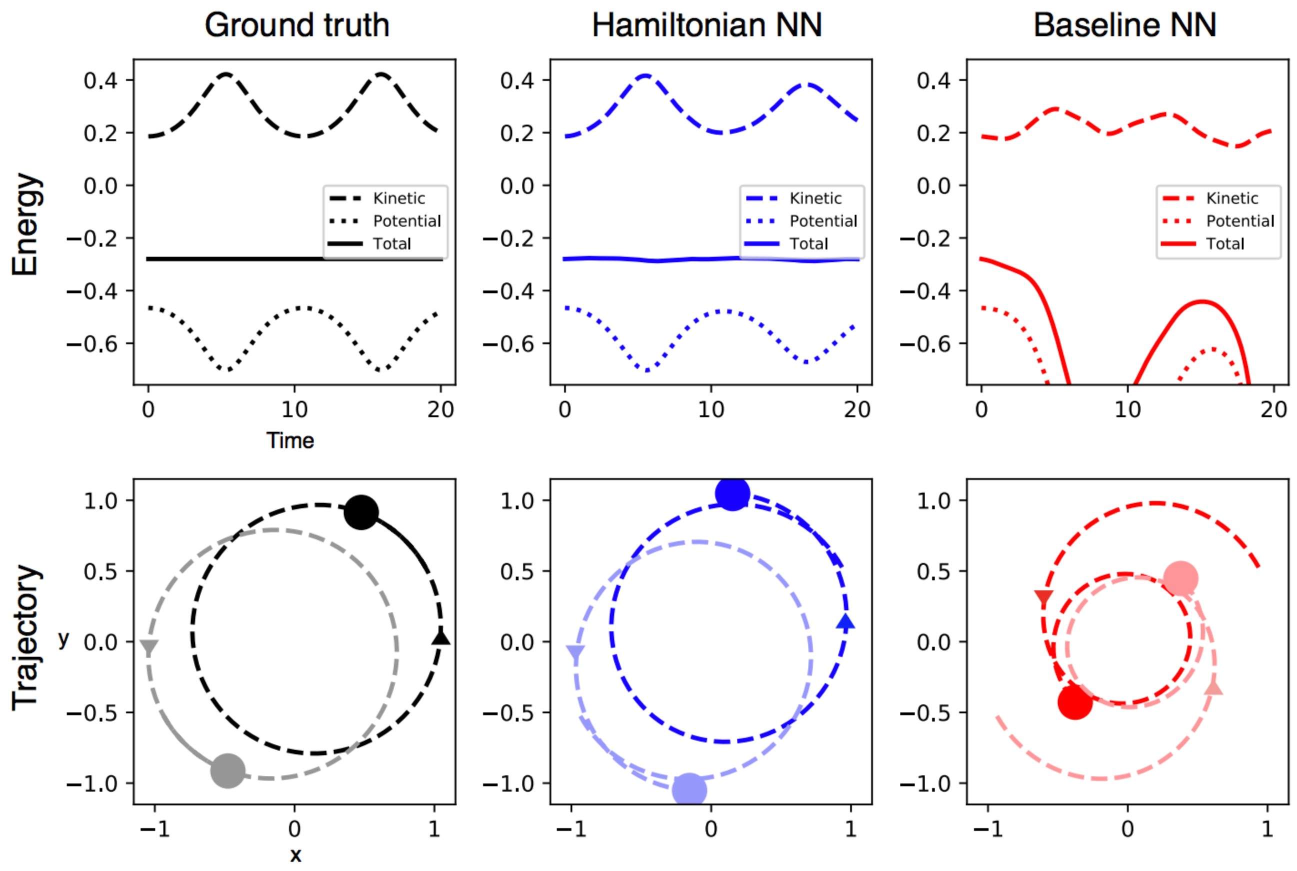
✓ $\mu = \left(\frac{1}{m_1} + \frac{1}{m_2}\right)^{-1}$: effective mass





Hamiltonian Neural Networks

Experimental results



- The Baseline NN shows energy-dissipation behaviour (broken energy-conservation).
- The Hamiltonian NN doesn’t.

Greydanus, Sam, Misko Dzamba, and Jason Yosinski. "Hamiltonian Neural Networks." *arXiv preprint arXiv:1906.01563* (2019).

Hamiltonian Neural Networks

Experiment 3

- Learning from pixels
  - ✓ OpenAI Gym's *Pendulum-v0* environment
  - ✓ Using fully-connected layers instead of convolutional layers

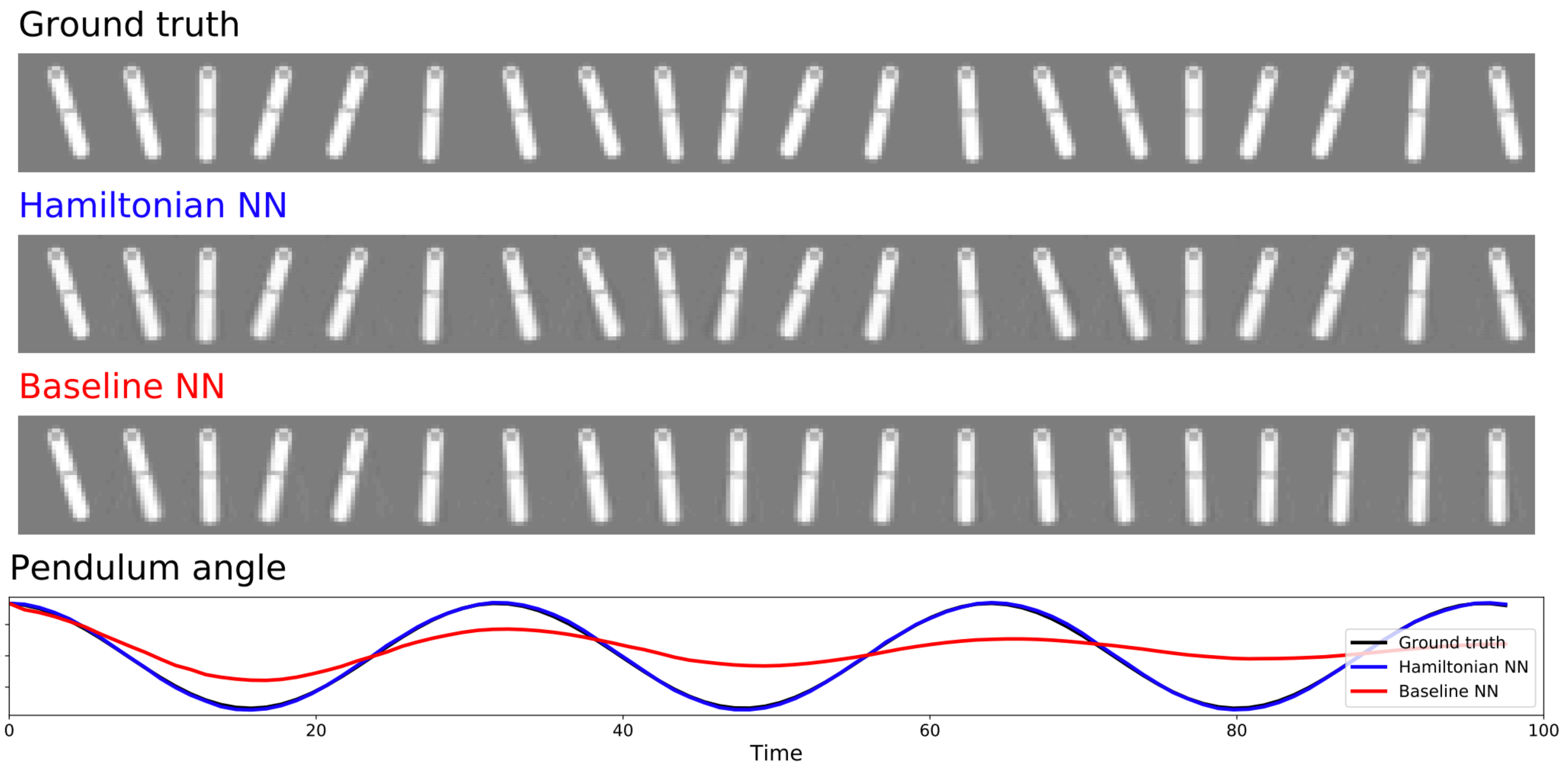
    → CNNs sometimes struggle to extract even simple position information
  - ✓ Total loss function:

$$\mathcal{L} = \mathcal{L}_{HNN} + \mathcal{L}_{AE} + \mathcal{L}_{CC}$$

  - ▪ $\mathcal{L}_{HNN}(\mathbf{z}_p, \mathbf{z}_q)$: Hamiltonian loss (or trajectory prediction loss)
  - ▪ $\mathcal{L}_{AE}(\mathbf{x}, \hat{\mathbf{x}}) = \|\hat{\mathbf{x}} - \mathbf{x}\|_2$: auto-encoder reconstruction loss
  - ▪ $\mathcal{L}_{CC} = \left\|\mathbf{z}_p^t - \left(\mathbf{z}_q^t - \mathbf{z}_q^{t+1}\right)\right\|_2$: to encourage latent vector $(\mathbf{z}_p^t, \mathbf{z}_q^t)$ have roughly same properties as canonical coordinates $(\mathbf{q}, \mathbf{p})$.

Hamiltonian Neural Networks

Experimental results



- The Baseline NN shows wrong pendulum motion.
- The Hamiltonian NN doesn't.

Hamiltonian Neural Networks

Limitations

- They used simple MLPs for Hamiltonian modelling.

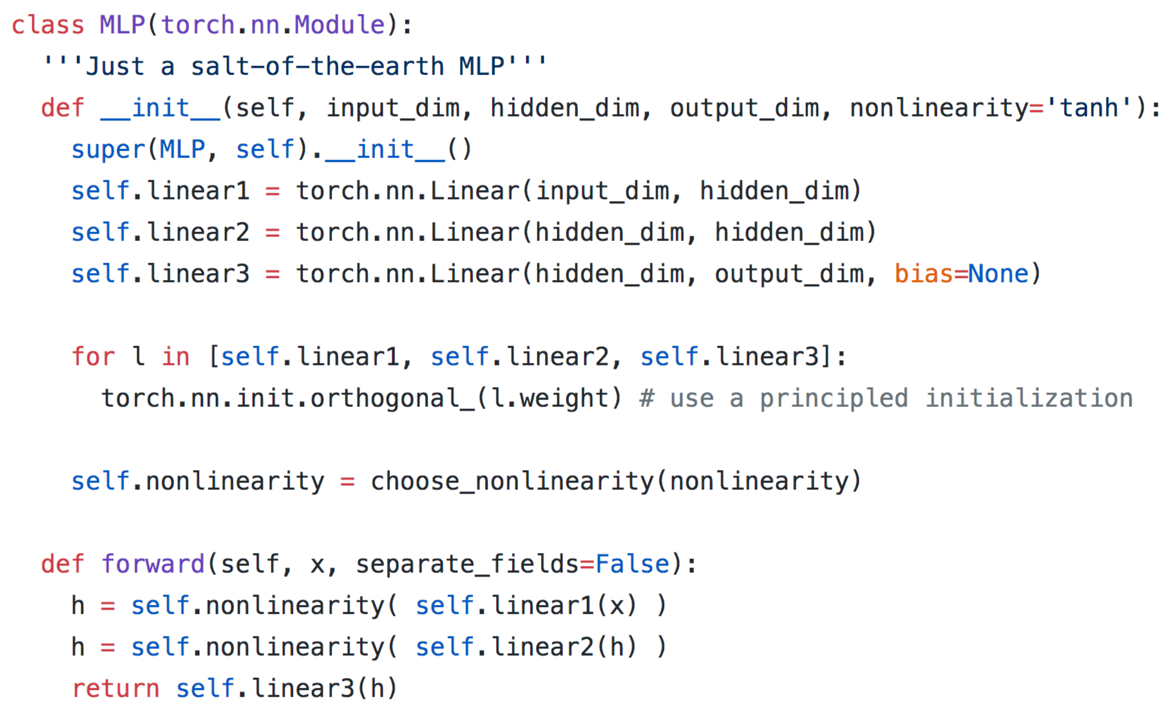
  Q) Can this generalize to unseen environment?

  ex) Two spring system → Three spring system

  A) Maybe not.

  Q) To do so, what modelling should we employ?

  A) Graph Networks!



```python
class MLP(torch.nn.Module):
  '''Just a salt-of-the-earth MLP'''
  def __init__(self, input_dim, hidden_dim, output_dim, nonlinearity='tanh'):
    super(MLP, self).__init__()
    self.linear1 = torch.nn.Linear(input_dim, hidden_dim)
    self.linear2 = torch.nn.Linear(hidden_dim, hidden_dim)
    self.linear3 = torch.nn.Linear(hidden_dim, output_dim, bias=None)

    for l in [self.linear1, self.linear2, self.linear3]:
      torch.nn.init.orthogonal_(l.weight) # use a principled initialization

    self.nonlinearity = choose_nonlinearity(nonlinearity)

  def forward(self, x, separate_fields=False):
    h = self.nonlinearity( self.linear1(x) )
    h = self.nonlinearity( self.linear2(h) )
    return self.linear3(h)
```

Hamiltonian Neural Networks

Lessons

- They demonstrated that the intervention of Hamiltonian principle (physics law) can be good inductive biases on neural models for physics modeling:
  "Instead of crafting the Hamiltonian by hand, we propose parameterizing it with a neural network and then learning it directly from data."

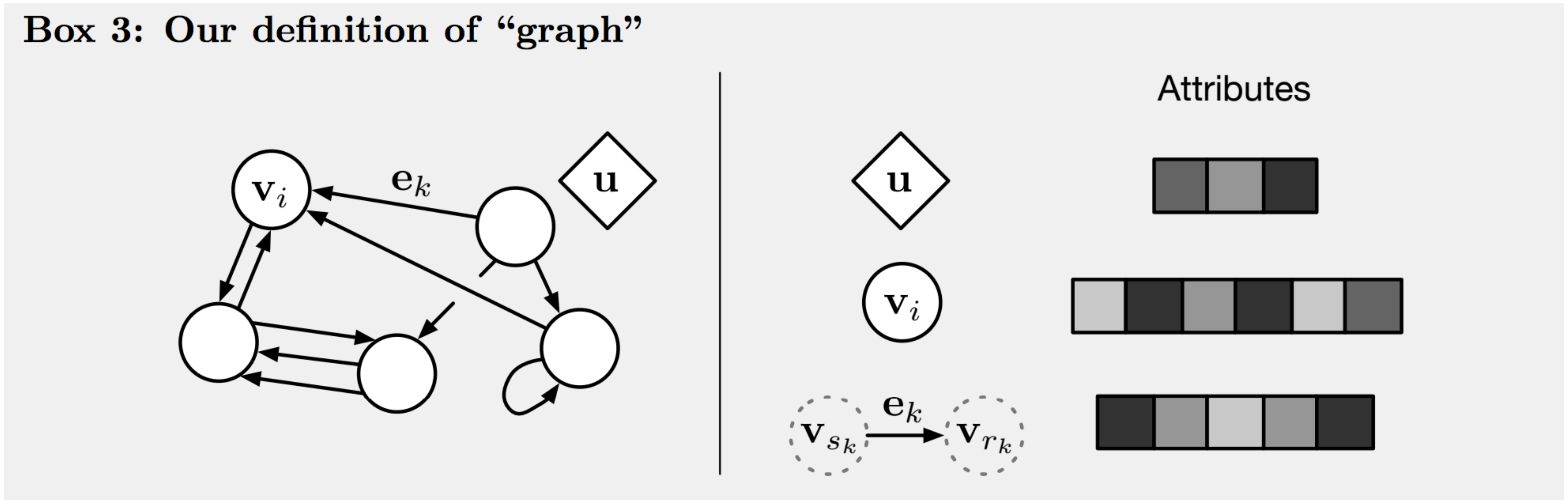
- 'Do we still need models or just more data and compute?', Max Welling:
  "There is no predictions without assumptions, no generalization without inductive bias."
  https://staff.fnwi.uva.nl/m.welling/wp-content/uploads/Model-versus-Data-AI.pdf

- As a result, they can learn relevant physics law from data even from sensory observations and predict the trajectory of various systems.

Greydanus, Sam, Misko Dzamba, and Jason Yosinski. "Hamiltonian Neural Networks." *arXiv preprint arXiv:1906.01563* (2019).

Hamiltonian Neural Networks

Graph Networks



Graph is defined as a 3-tuple $G(\mathbf{u}, V, E)$

- Global attributes: ex) gravitational field, energy function
- Node attributes: ex) position, velocity, mass of particles
- Edge attributes: ex) the presence of springs between balls and their corresponding spring constants

Hamiltonian Neural Networks

Related works

- Learning physics with graph networks
  - ✓ Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." *Advances in neural information processing systems*. 2016.
  - ✓ Kipf, Thomas, et al. "Neural relational inference for interacting systems." *arXiv preprint arXiv:1802.04687* (2018).
  - ✓ Cranmer, Miles D., et al. "Learning Symbolic Physics with Graph Networks." *arXiv preprint arXiv:1909.05862* (2019).

- Learning with ODE integrators
  - Sanchez-Gonzalez, Alvaro, et al. "Hamiltonian graph networks with ode integrators." *arXiv preprint arXiv:1909.12790* (2019).
  - Symplectic recurrent neural networks, https://arxiv.org/abs/1909.13334

Learning Symbolic Physics with Graph Networks

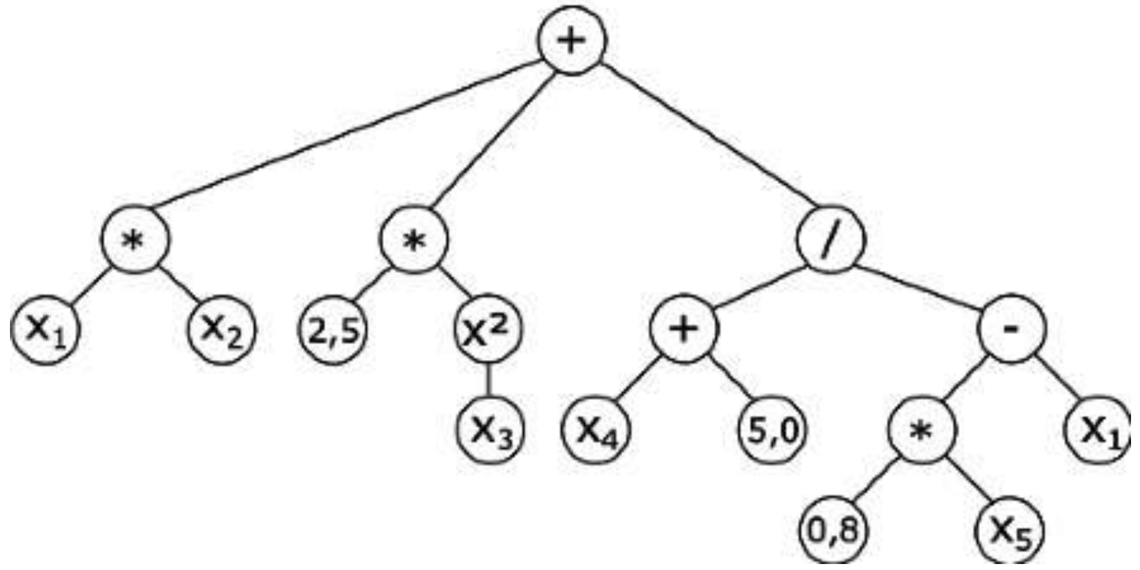
Learning Symbolic Physics with Graph Networks

Symbolic regression

- Symbolic regression is a type of regression analysis that searches the space of mathematical expressions to find the model that best fits a given dataset.

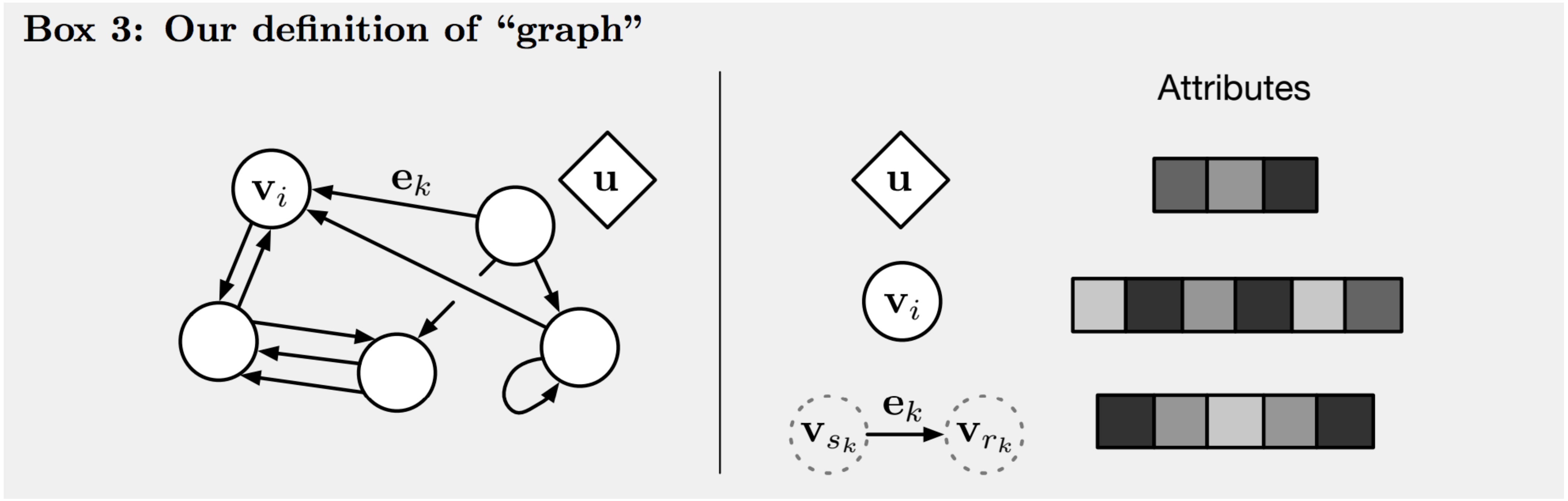
- The fitness function that derives the evolution of the models take into account not only error metrics, but also special complexity measures, thus ensuring that the resulting models reveal the data's underlying structure in a way that's understandable from a human perspective.

$$f(x) = x_1 x_2 + 2.5\, x_3^2 + \frac{x_4 + 5.0}{0.8\, x_5 - x_1}$$



Learning Symbolic Physics with Graph Networks

Graph Networks



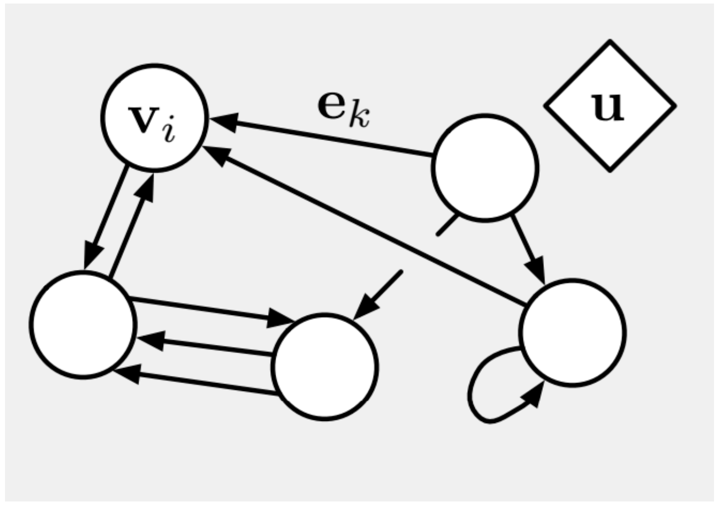
Graph is defined as a 3-tuple $G(\mathbf{u}, V, E)$

- Global attributes: ex) gravitational field
- Node attributes: ex) position, velocity, mass of particles
- Edge attributes: ex) the presence of springs between balls and their corresponding spring constants

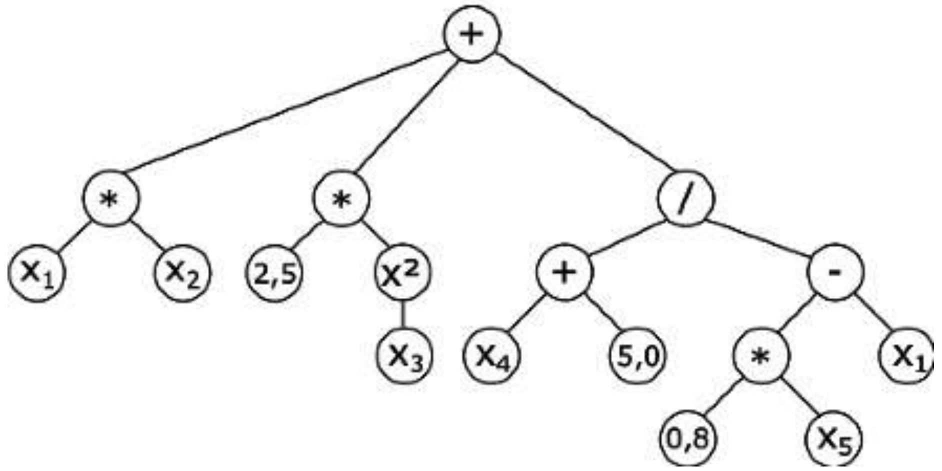
Learning Symbolic Physics with Graph Networks

Paper contribution

- **A modified GN with inductive biases** that promote learning general-purpose physical law.
- **Using symbolic regression** to extract analytical physical laws from trained neural networks
- Improved **zero-shot generalization** to larger systems than those in training.



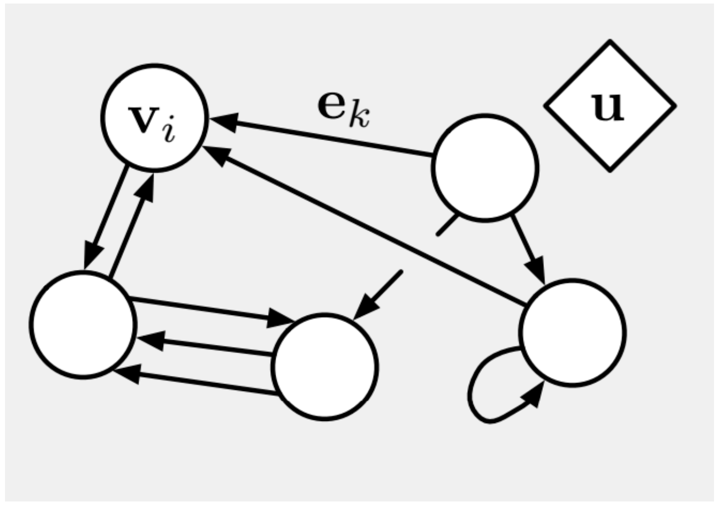
$$f(x) = x_1 x_2 + 2.5\, x_3^2 + \frac{x_4 + 5.0}{0.8\, x_5 - x_1}$$



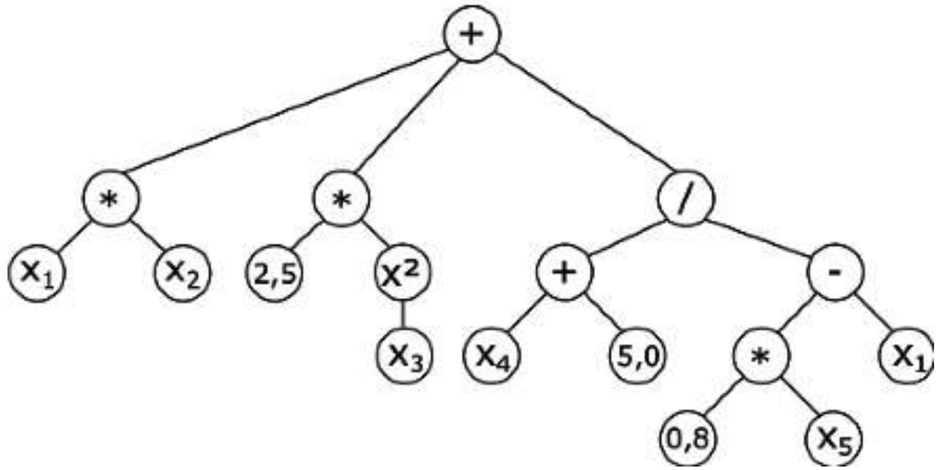
Learning Symbolic Physics with Graph Networks

Paper contribution

- **A modified GN with inductive biases** that promote learning general-purpose physical law.
- **Using symbolic regression** to extract analytical physical laws from trained neural networks
- Improved **zero-shot generalization** to larger systems than those in training.



$$f(x) = x_1x_2 + 2.5\,x_3^2 + \frac{x_4 + 5.0}{0.8\,x_5 - x_1}$$



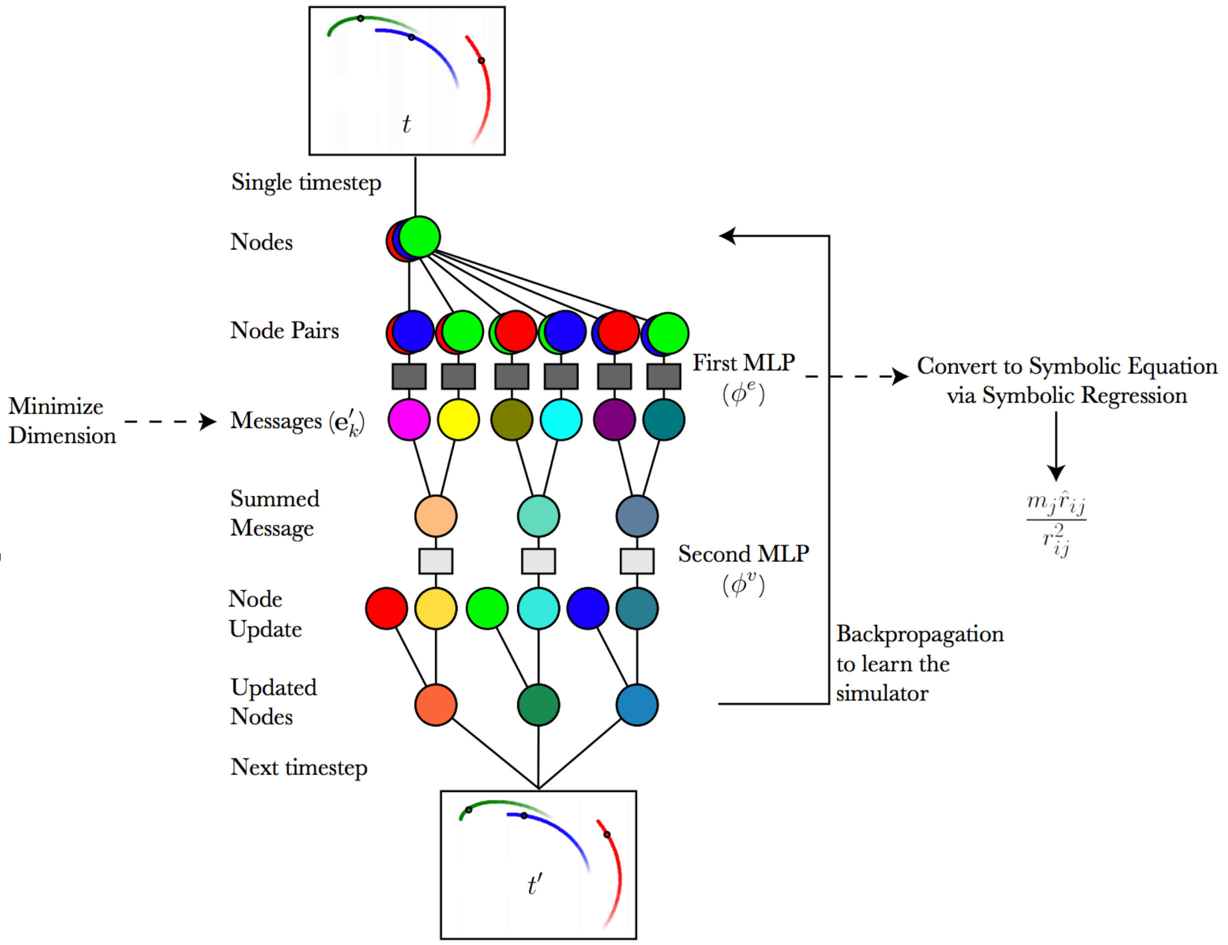
Learning Symbolic Physics with Graph Networks

Model architecture

1. Message function
   - $\phi^e : \mathbb{R}^{L^v} \times \mathbb{R}^{L^v} \times \mathbb{R}^{L^e} \rightarrow \mathbb{R}^{L^{e'}}$
   - Computing the pairwise interactions

2. Message pooling
   - $\bar{\mathbf{e}}_i = \rho^{e \rightarrow v}\left(\{\mathbf{e}'_k\}_{r_k = i, k=1:N^e}\right)$
   - $\rho^{e \rightarrow v}$ must be a permutation-invariant operation, such as elementwise-summation.

3. Node update function
   - $\phi^v : \mathbb{R}^{L^v} \times \mathbb{R}^{L^{e'}} \rightarrow \mathbb{R}^{L^{v'}}$
   - Computing the node updates



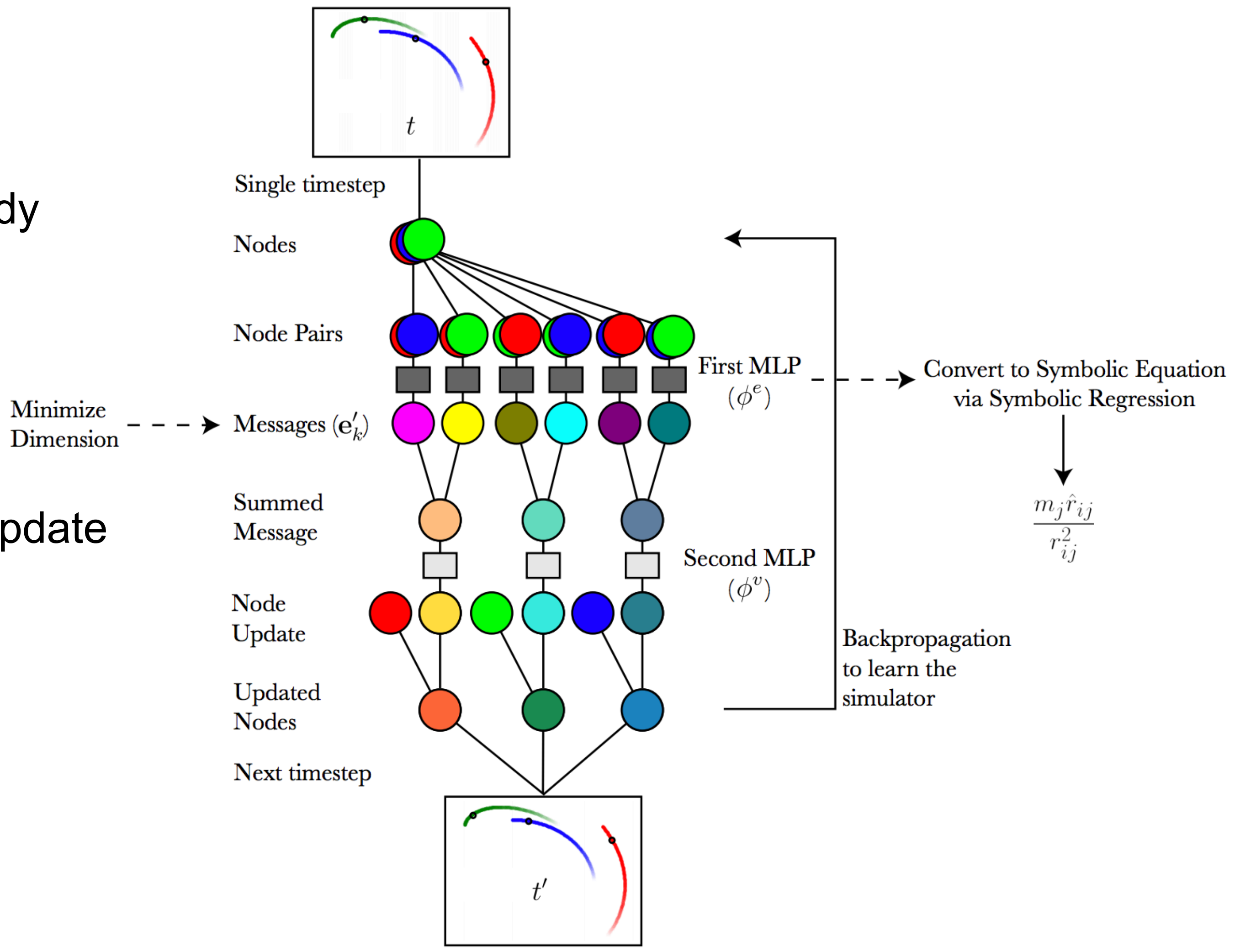
t
Single timestep
Nodes
Node Pairs
First MLP (ϕ^e)
Convert to Symbolic Equation via Symbolic Regression
$\frac{m_j \hat{r}_{ij}}{r_{ij}^2}$
Minimize Dimension
Messages ($\mathbf{e}'_k$)
Summed Message
Second MLP (ϕ^v)
Node Update
Backpropagation to learn the simulator
Updated Nodes
Next timestep
t'

Learning Symbolic Physics with Graph Networks

Idea

1. Dimensionality of $\mathbf{e}'_k$, i.e. $L^{e'}$
   - Encouraging $\mathbf{e}'_k$ be the force
   - Thus, let $L^{e'} = D$ for D-dimensional n-body problem. (Non-rigorous proof is given)

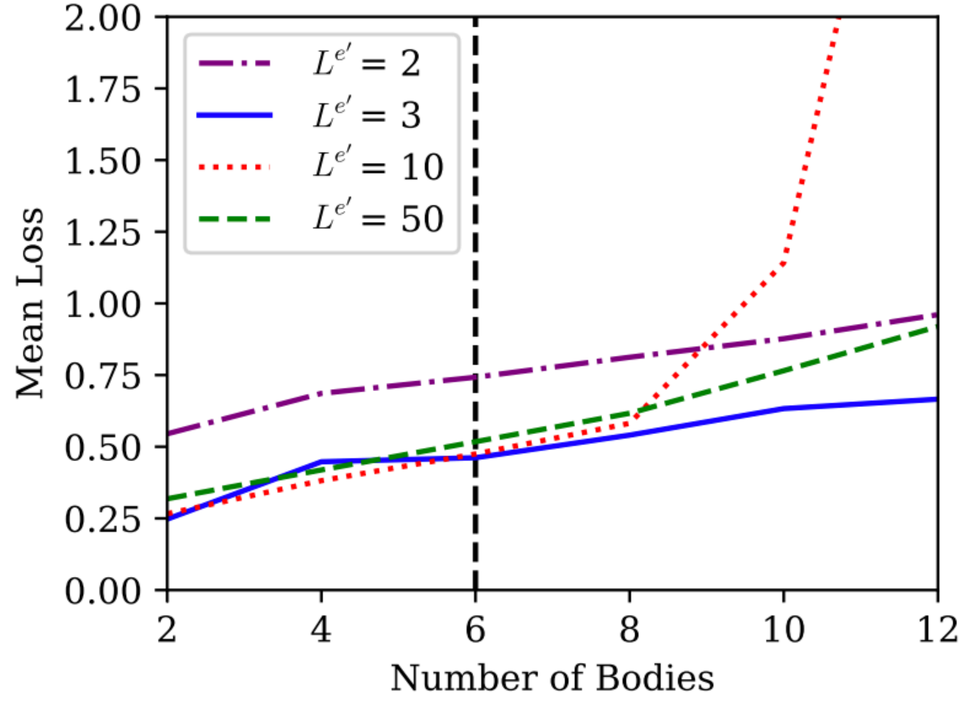
2. Learning objective
   - L1 loss between the predictive velocity update and the true velocity update
   - $\mathcal{L}(\theta) = \left|\widehat{\Delta \mathbf{v}} - \Delta \mathbf{v}\right|$

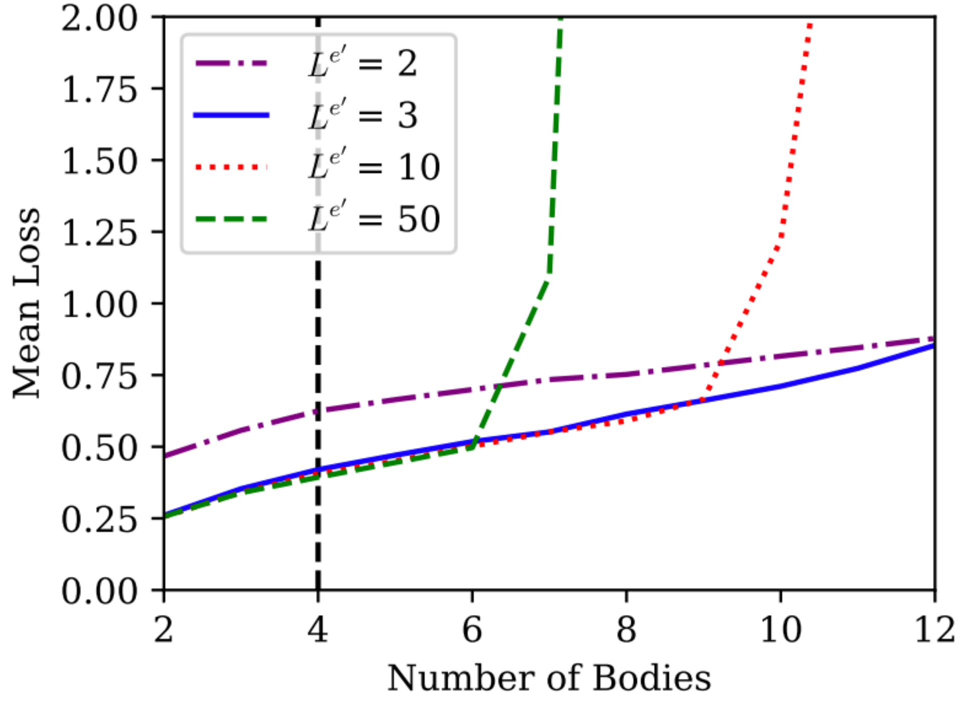


t
Single timestep
Nodes
Node Pairs
Minimize Dimension
Messages ($\mathbf{e}'_k$)
First MLP (ϕ^e)
Convert to Symbolic Equation via Symbolic Regression
$\frac{m_j \hat{r}_{ij}}{r_{ij}^2}$
Summed Message
Second MLP (ϕ^v)
Node Update
Backpropagation to learn the simulator
Updated Nodes
Next timestep
t'

Learning Symbolic Physics with Graph Networks

Experimental results

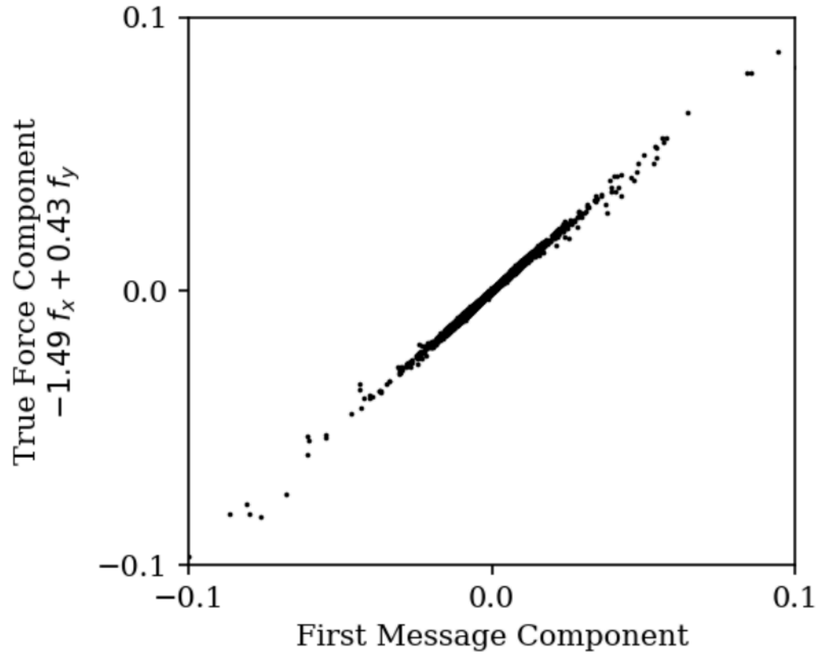


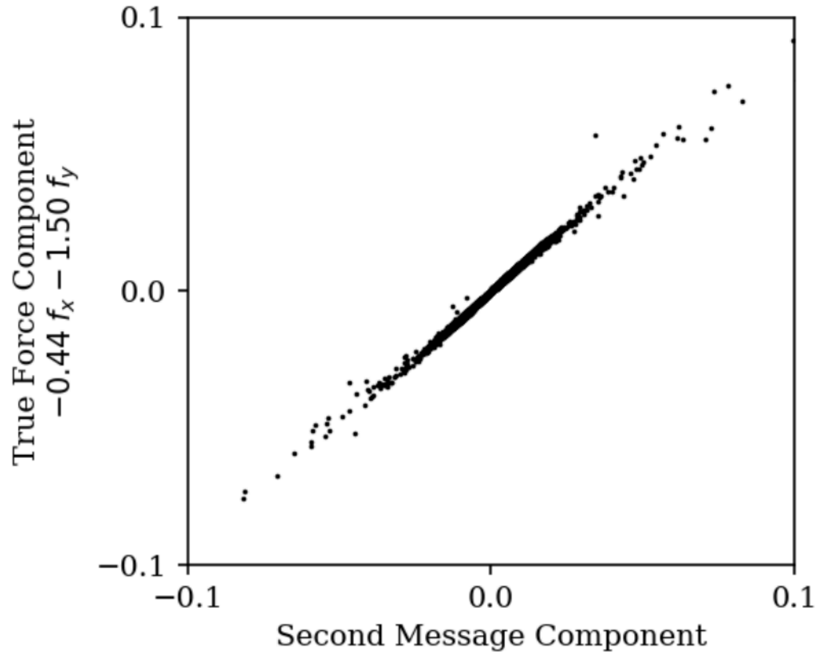


- Trained on six-body (left) and four-body (right)
- $L^{e'} = 3$ works best for three-body problems.
- Demonstrating the zero-shot generalizations

Learning Symbolic Physics with Graph Networks

Experimental results





- To find the force law by using symbolic regression to fit an algebraic function approximates ϕ^e on the trained two-dimensional model.
- Using ***eureqa*** to fit algebraic function with operators $+, -, \times, /$ as well as input variables (Δx and Δy)

Thank you!