Physics-informed Machine Learning to Infer Dynamics from Data

Biswadip Dey

Siemens Corporation, Technology

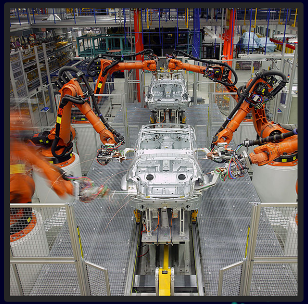
Workshop on *Differentiable Programming for Modeling and Control of Dynamical Systems*

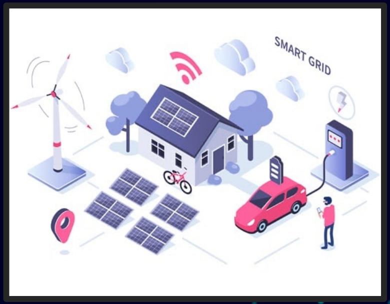
American Control Conference – 2023

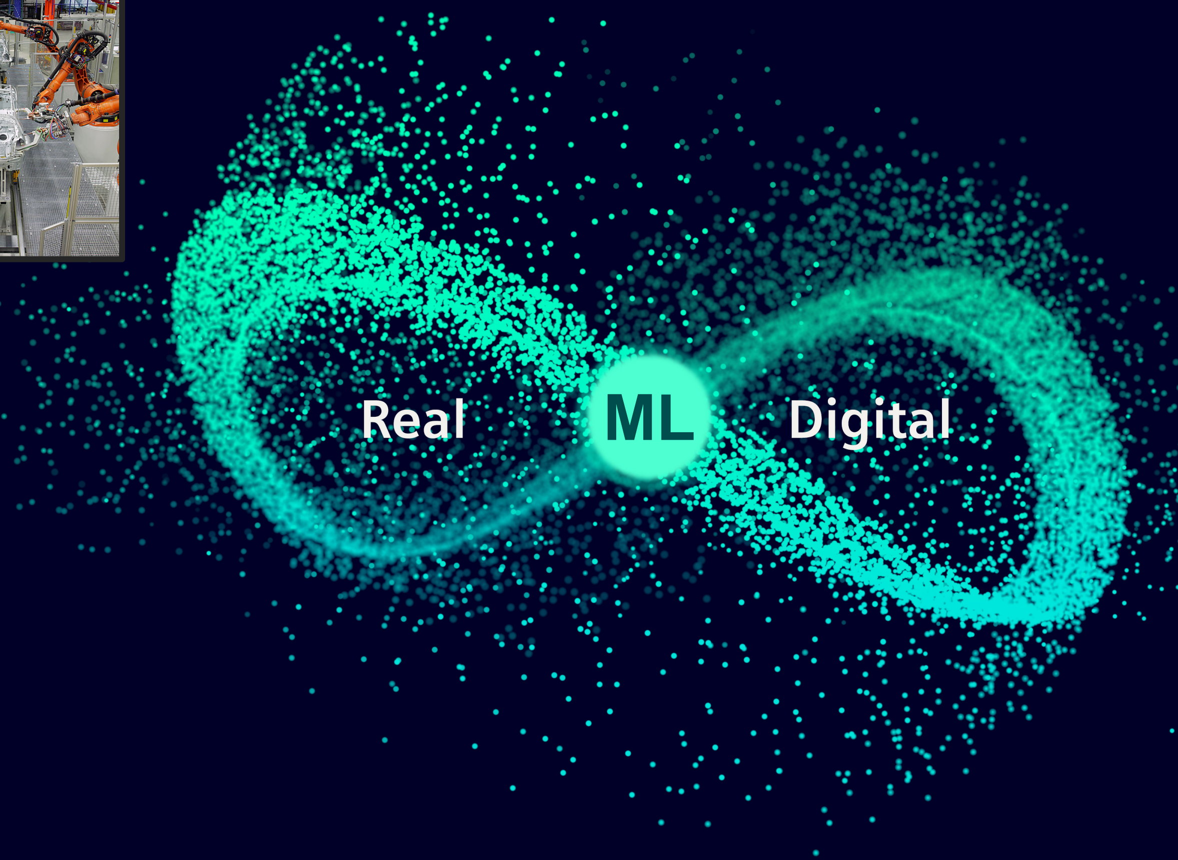
San Diego, CA, USA | May 30, 2023

SIEMENS

Physics-informed ML provides a **bridge between the real and the digital**







Minimize

$$\int_0^T [c_1 y(t) + c_2 x(t)]dt$$

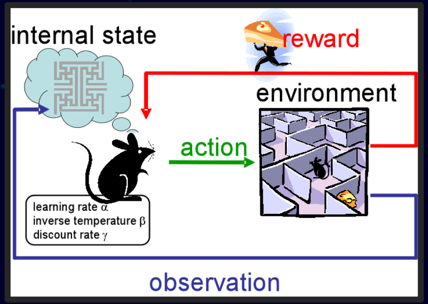
subject to

$$y(t) = y_0 + \int_0^t [x(\tau) - g(\tau)]dt$$

$$y(T) = y_T$$

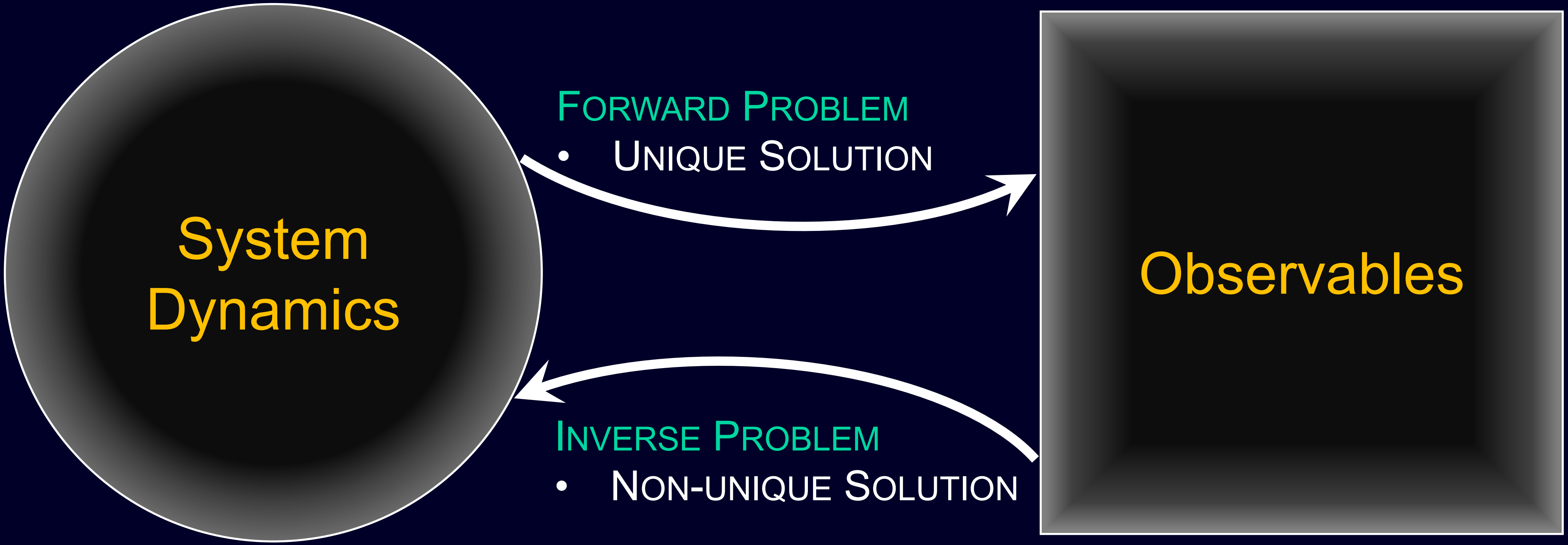
$$0 \le x(t) \le b_1,\ t \in [0, T]$$

$$0 \le y(t) \le b_2,\ t \in [0, T]$$



Real-world systems often **lack good quality data** but come with **lots of domain knowledge**!

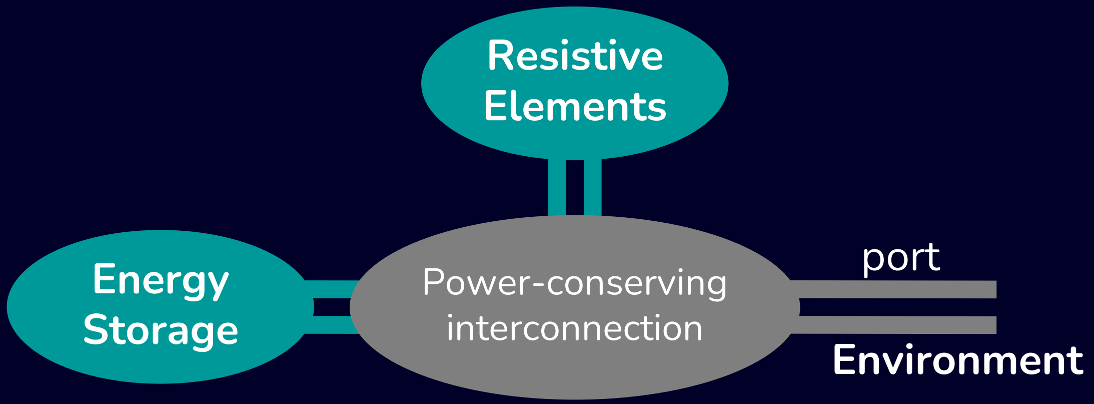
The inverse problem of inferring dynamics from data needs relevant **inductive bias**

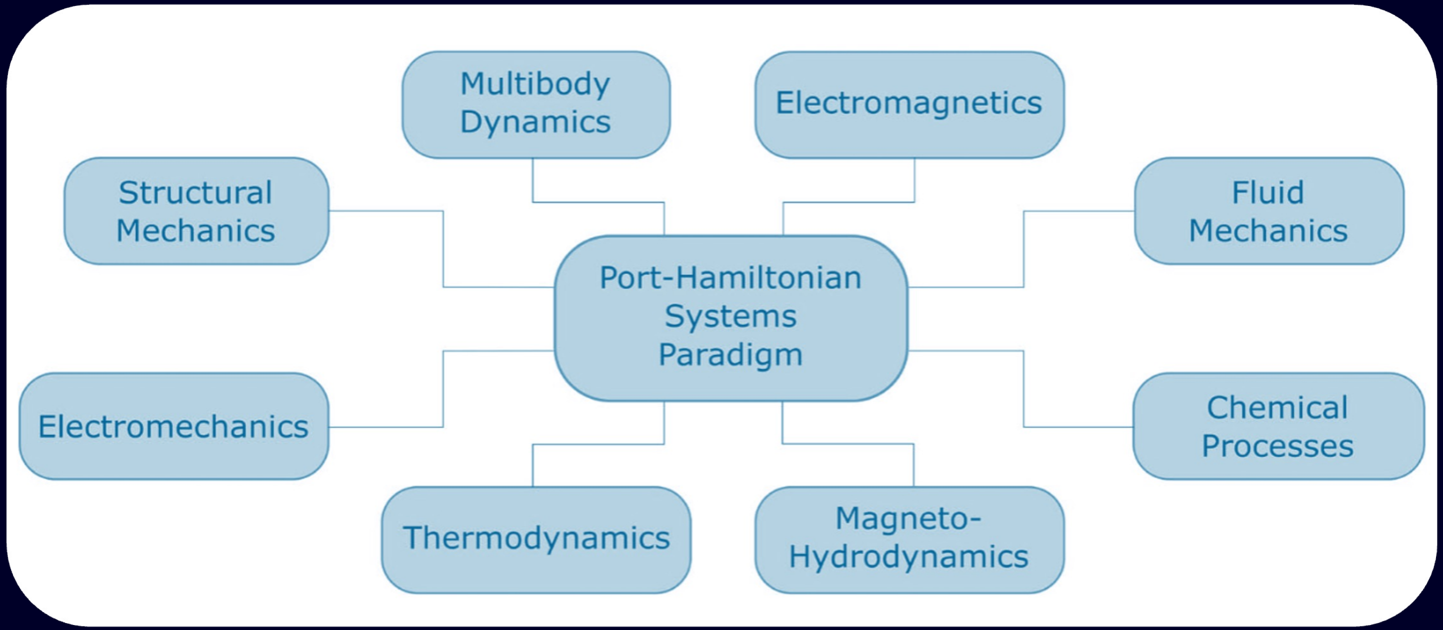


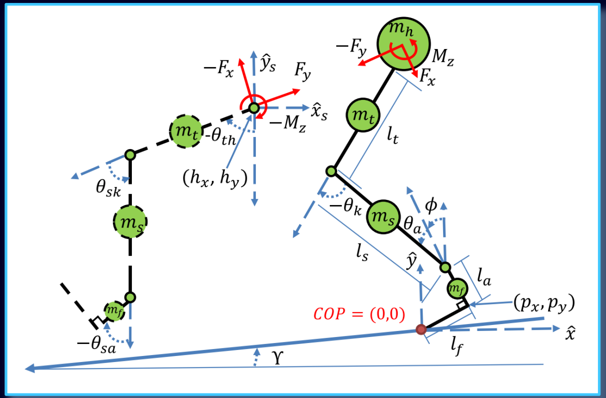
Need to use appropriate **inductive bias**!

Energy-based descriptions!

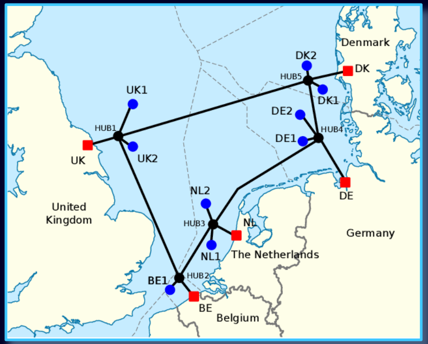
Hamiltonian dynamics and **port-Hamiltonian formulation** provide a relevant indictive bias for a broad class of physical systems



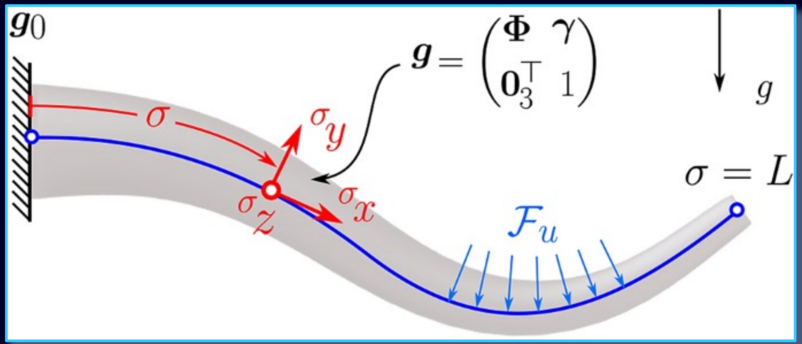




Lv, Gregg | *IEEE Tr. CST* | 2018



Benedito, et al. | *Control Engineering Practice* | 2019



Caasenbrood, et al. | *SN Computer Science* | 2022

Hamiltonian dynamics

- ❑ **Generalized Coordinate** – q
- ❑ **Generalized Momentum** – p
- ❑ **A Conserved Quantity** – H, i.e., the Hamiltonian
  - It usually represents the **total energy**

$$\begin{bmatrix} \dot{q} \\ \dot{p} \end{bmatrix} = \begin{bmatrix} \frac{\partial H}{\partial p} \\ -\frac{\partial H}{\partial q} \end{bmatrix}$$

Symplectic gradient $\Rightarrow \frac{dH}{dt} = \frac{\partial H}{\partial q}\dot{q} + \frac{\partial H}{\partial p}\dot{p} = 0$

Sir William Rowan Hamilton (1833)

- ❑ For physical systems, the total energy is: $H(q,p) = \underbrace{\frac{1}{2} p^T M^{-1}(q) p}_{\text{Kinetic energy}} + \underbrace{V(q)}_{\text{Potential energy}}$

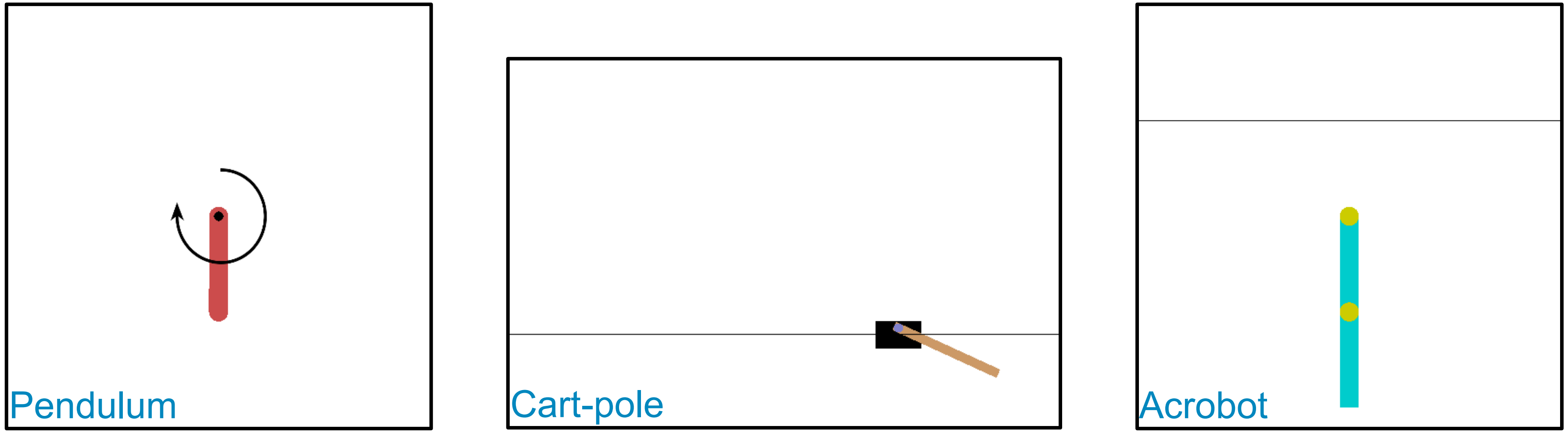
- ❑ An alternative description is provided by the **Lagrangian Dynamics**, in which the system is described in terms of generalized position (q) and generalized velocity ($\dot{q}$). These two sides are related via **Legendre Transformation**, i.e., $p = M(q)\dot{q}$.

SIEMENS

Hamiltonian dynamics with control offer a natural framework for modeling a large class of systems

$$\begin{bmatrix}\dot{q}\\ \dot{p}\end{bmatrix} = \begin{bmatrix}\frac{\partial H}{\partial p}\\ -\frac{\partial H}{\partial q}\end{bmatrix} + \begin{bmatrix}0\\ g(q)\end{bmatrix} u = \begin{bmatrix}0 & I\\ -I & 0\end{bmatrix}\begin{bmatrix}\frac{\partial H}{\partial q}\\ \frac{\partial H}{\partial p}\end{bmatrix} + \begin{bmatrix}0\\ g(q)\end{bmatrix} u$$

External Control:
- Force, Torque, *etc.*



Port-Hamiltonian System:

$$\dot{x} = \big(J(x) - D(x)\big)\nabla_x H + g(x)u$$
$$y = g^T(x)\nabla_x H$$

$J(x)$: *Skew-symmetric*

$D(x)$: *Symmetric, Positive-semidefinite*

$$\dot{H} \leq y^T u$$

SIEMENS

How do we **encode Hamiltonian dynamics** into neural networks for **learning dynamics** from data**?**

Data Driven Approach: Learn a dynamical system governed by a set of differential equations from data

Prior:
Symmetries and Conservation Laws

- Improved model transparency
- Model-based control synthesis
- Better generalization
- Data-efficiency
- Increase in learning speed

Our Solution: *Symplectic ODENet*
Encode Hamiltonian dynamics into the architecture of a neural network

* Zhong, BD, Chakraborty | **Symplectic ODE-Net: Learning Hamiltonian Dynamics with Control** | ICLR 2020.

SIEMENS

Symplectic ODENet encodes Hamiltonian dynamics into neural networks

Available data: $(q, p, u)_{t_0,\cdots,t_n}$

- Leverage **Neural ODE**[9]
  - Consider an ODE – $\dot{x} = f_\theta(x, u)$, where $f_\theta(x)$ is parametrized by a neural network
  - Use *Neural ODE Solvers* to obtain: $\hat{\boldsymbol{x}}_{t_1}, \hat{\boldsymbol{x}}_{t_2}, \dots, \hat{\boldsymbol{x}}_{t_n} = ODESolve(\boldsymbol{x}_{t_0}, \boldsymbol{f}_\theta, u, t_0, \dots, t_n)$
  - Minimize an appropriate penalty function $d(\cdot,\cdot)$ (e.g., MSE, MAE) to find a suitable $f_\theta(\cdot)$

$$L = \sum_{i=1}^{n} d(\boldsymbol{x}_{t_i}, \hat{\boldsymbol{x}}_{t_i})$$

Symplectic ODENet

$$\boldsymbol{f}_\theta(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{u}) = \begin{bmatrix} \dfrac{\partial H_{\theta_1,\theta_2}}{\partial \boldsymbol{p}} \\ -\dfrac{\partial H_{\theta_1,\theta_2}}{\partial \boldsymbol{q}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{g}_{\theta_3}(\boldsymbol{q}) \end{bmatrix} \boldsymbol{u}$$

$$H_{\theta_1,\theta_2}(\boldsymbol{q}, \boldsymbol{p}) = \frac{1}{2} \boldsymbol{p}^T \boldsymbol{M}_{\theta_1}^{-1}(\boldsymbol{q}) \boldsymbol{p} + V_{\theta_2}(\boldsymbol{q})$$

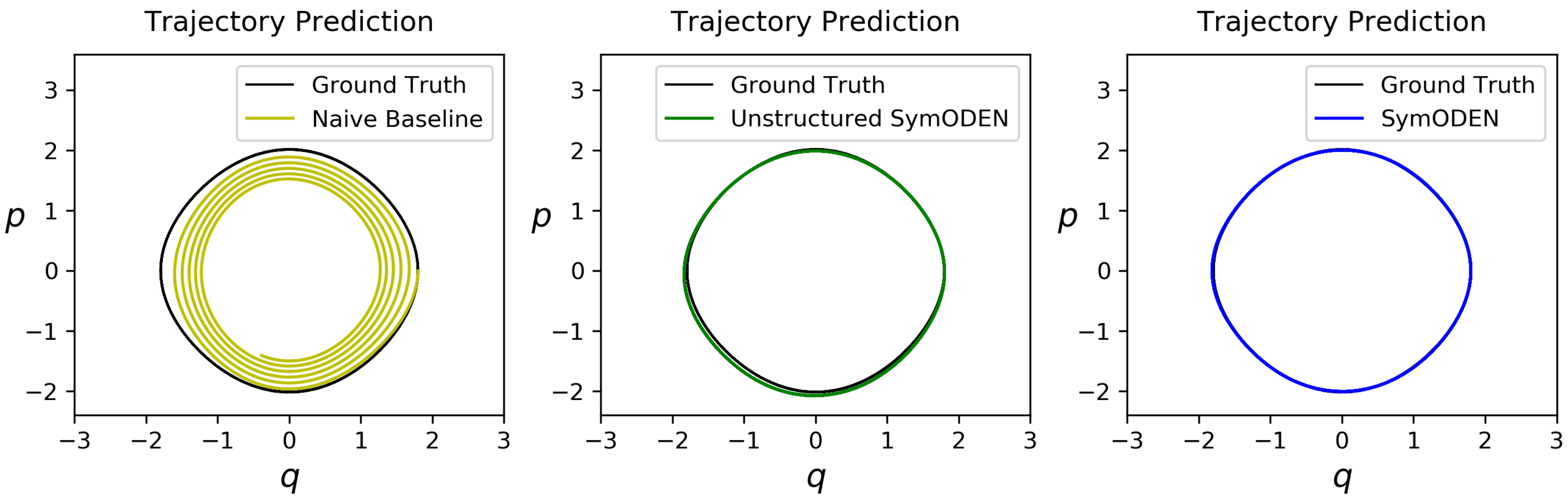
- $M_{\theta_1}^{-1}(q) = L_{\theta_1} L_{\theta_1}^T$ - Fully-connected Feedforward Network
- $V_{\theta_2}(q)$ - Fully-connected Feedforward Network
- $g_{\theta_3}(q)$ - Fully-connected Feedforward Network

We use mean-squared error (MSE) as the penalty function!

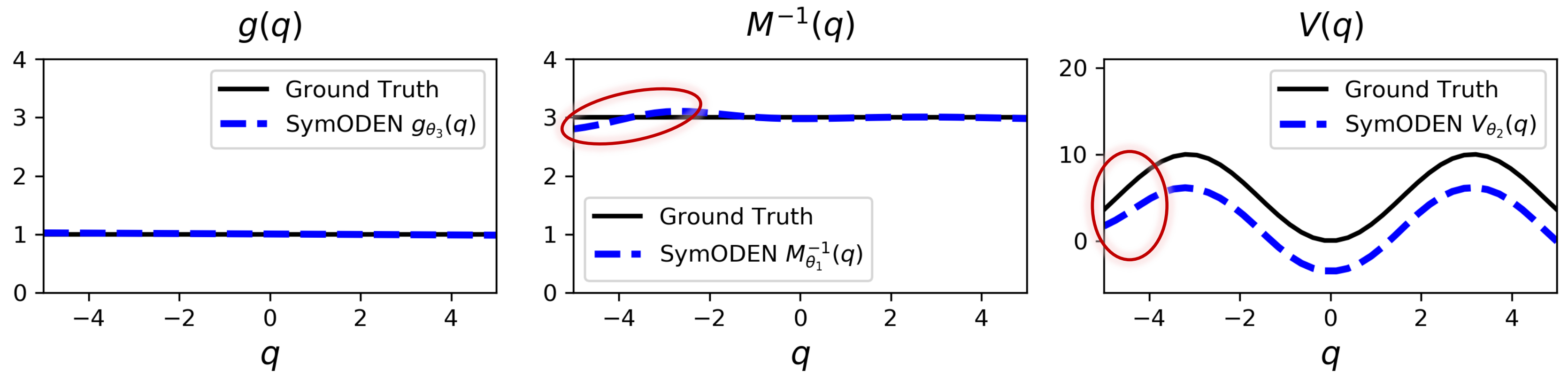
[9] Chen, Rubanova, Bettencourt, Duvenaud | **Neural Ordinary Differential Equations** | NeurIPS 2018.

Can Symplectic ODENet infer the dynamics of a pendulum from data?

Prediction of test trajectories ($u = 0$)



Functions learned by SymODEN



SIEMENS

Bridging this gap through an **angle-aware Design**

- **Theoretical perspective**: Convenient to deal with independent generalized coordinates and momenta, i.e., (q, p).

- **Data-driven perspective**: Angle coordinate – q – is often embedded in $(\cos q\,, \sin q)$ format, since treating q as a variable in $\mathbb{R}^1$ fail to respect the geometry that q lies on the manifold $\mathbb{S}^1$. Also, the velocity data – $\dot{q}$ – is often more readily available than the momentum data p.

  - *Example*: In OpenAI Gym Pendulum-v0 environment, observation data are available in the form $(\cos q\,, \sin q\,, \dot{q})$

Question: *Can we bridge this gap?*

SIEMENS

Symplectic ODENet with **embedded coordinate and momentum Data**

- Define $(x_1, x_2, x_3) = (\sin q\,, \cos q\,, \dot{q})$

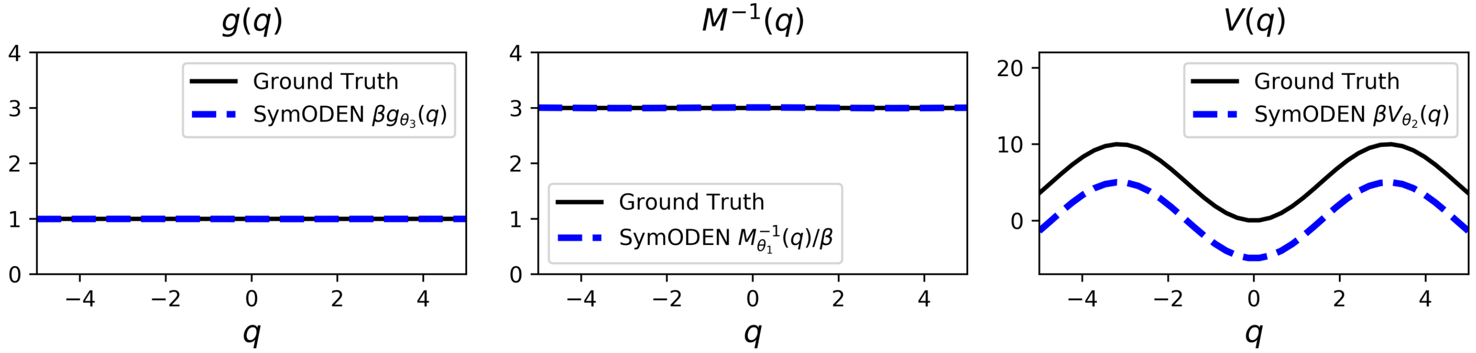
- *Use **chain-rule** and **Hamiltonian dynamics** to express the dynamics of* (x_1, x_2, x_3)

$$\dot{\boldsymbol{x}}_1 = -\sin \boldsymbol{q} \circ \dot{\boldsymbol{q}} = -\boldsymbol{x}_2 \circ \dot{\boldsymbol{q}}$$
$$\dot{\boldsymbol{x}}_2 = \cos \boldsymbol{q} \circ \dot{\boldsymbol{q}} = \boldsymbol{x}_1 \circ \dot{\boldsymbol{q}}$$
$$\dot{\boldsymbol{x}}_3 = \frac{d}{dt}\left(\boldsymbol{M}^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2)\boldsymbol{p}\right) = \frac{d}{dt}\boldsymbol{M}^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2) \cdot \boldsymbol{p} + \boldsymbol{M}^{-1}(\boldsymbol{x}_1, \boldsymbol{x}_2) \cdot \dot{\boldsymbol{p}}$$

where,
$$\boldsymbol{p} = \boldsymbol{M}(\boldsymbol{x}_1, \boldsymbol{x}_2) \cdot \boldsymbol{x}_3$$
$$\dot{\boldsymbol{q}} = \frac{\partial H(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{p})}{\partial \boldsymbol{p}}$$
$$\dot{\boldsymbol{p}} = -\frac{\partial H(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{p})}{\partial \boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{x}_1, \boldsymbol{x}_2)\boldsymbol{u} = -\frac{\partial \boldsymbol{x}_1}{\partial \boldsymbol{q}}\frac{\partial H}{\partial \boldsymbol{x}_1} - \frac{\partial \boldsymbol{x}_2}{\partial \boldsymbol{q}}\frac{\partial H}{\partial \boldsymbol{x}_2} + \boldsymbol{g}(\boldsymbol{x}_1, \boldsymbol{x}_2)\boldsymbol{u} = \boldsymbol{x}_2 \circ \frac{\partial H}{\partial \boldsymbol{x}_1} - \boldsymbol{x}_1 \circ \frac{\partial H}{\partial \boldsymbol{x}_2} + \boldsymbol{g}(\boldsymbol{x}_1, \boldsymbol{x}_2)\boldsymbol{u}$$

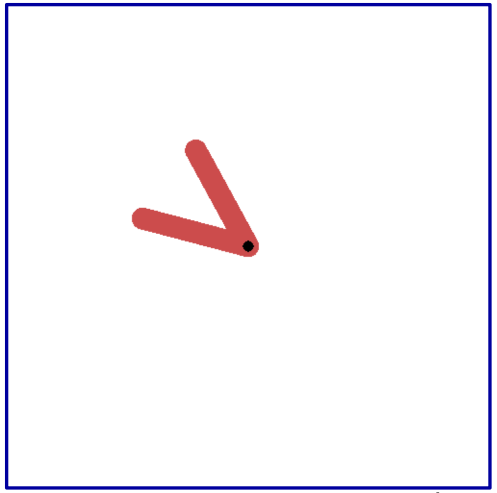
Angle-aware design leads to performance improvement

➢ Learned functions

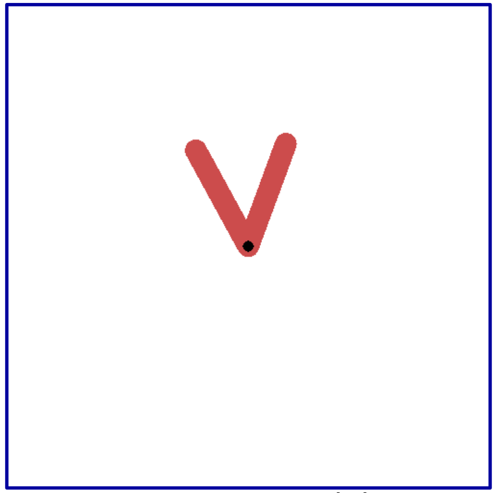


➢ Prediction

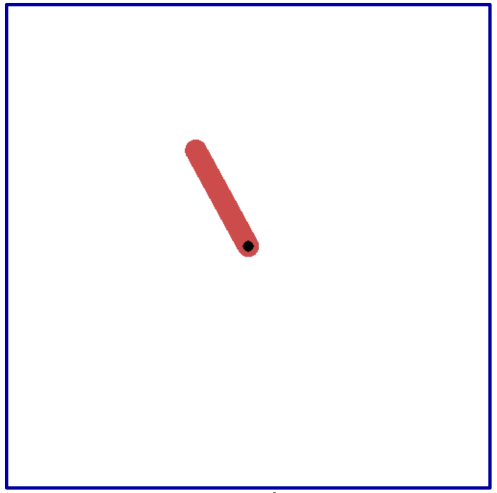
Gray: Ground Truth
Orange: Prediction



Baseline
No energy conservation



Model-variant
Unstructured Hamiltonian



Symplectic ODENet
Structured Hamiltonian

* Zhong, BD, Chakraborty | **Symplectic ODE-Net: Learning Hamiltonian Dynamics with Control** | ICLR 2020.

SIEMENS

Key takeaways

- Symplectic ODENet achieves **better generalization with fewer training samples** by encoding Hamiltonian dynamics into the neural network architecture.
- The **angle-aware design** narrows the gap between model-based and data-driven methods.
- **Integration over longer time-horizon lowers prediction error**, at the cost of increased training time.
- A parallel line of work has investigated similar questions using Lagrangian dynamics!

Unsupervised Learning of Lagrangian Dynamics from Images for Prediction and Control

Yaofeng Desmond Zhong, Princeton University, y.zhong@princeton.edu
Naomi Ehrich Leonard, Princeton University, naomi@princeton.edu

Modeling System Dynamics with Physics-Informed Neural Networks Based on Lagrangian Mechanics ⋆

Manuel A. Roehrl *,** Thomas A. Runkler *,** Veronika Brandtstetter * Michel Tokic * Stefan Obermayer *

* Siemens AG, Corporate Technology, 81739 Munich, Germany (e-mail: manuel.roehrl@siemens.com; thomas.runkler@siemens.com; veronika.brandstetter@siemens.com; michel.tokic@siemens.com)
** Technical University of Munich, 85748 Garching, Germany (e-mail: m.roehrl@tum.de; thomas.runkler@tum.de)

Deep Lagrangian Networks: Using Physics as Model Prior for Deep Learning

Michael Lutter, Christian Ritter & Jan Peters *
Department of Computer Science
Technische Universität Darmstadt
Hochschulstr. 10, 64289 Darmstadt, Germany
{Lutter, Peters}@ias.tu-darmstadt.de

LagNetViP: A Lagrangian Neural Network for Video Prediction

Christine Allen-Blanchette, Sushant Veer, Anirudha Majumdar, Naomi Ehrich Leonard
Department of Mechanical and Aerospace Engineering, Princeton University
{ca15, sveer, ani.majumdar, naomi}@princeton.edu

Deep Lagrangian Networks for end-to-end learning of energy-based control for under-actuated systems

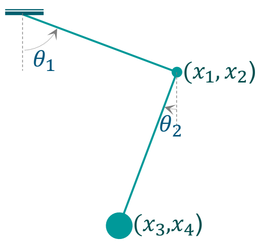
Michael Lutter[1], Kim Listmann[2] and Jan Peters[1,3]

Lagrangian Neural Networks

Miles Cranmer, Princeton University, mcranmer@princeton.edu
Sam Greydanus, Oregon State University, greydanus.17@gmail.com
Stephan Hoyer, Google Research, shoyer@google.com
Peter Battaglia, DeepMind, peterbattaglia@google.com
David Spergel*, Flatiron Institute, davidspergel@flatironinstitute.org
Shirley Ho†, Flatiron Institute, shirleyho@flatironinstitute.org

SIEMENS

The notion of angle-aware design can be extended to accommodate **holonomic constraints in the configuration space**



(θ_1, θ_2) ➔ Independent coordinate, but often results in coordinate dependent mass matrix.

(x_1, x_2, x_3, x_4) ➔ Coordinates are constrained, but admits simplified mass matrix.

Configuration Space with Constraints:

- System configuration is described by Cartesian coordinates $x \in \mathbb{R}^d$.
- Number of degrees of freedom is m.
- There exists $k = d - m$ equality constraints:

$$\Phi_i(x) = 0, \quad i = 1, \cdots, k \quad \Rightarrow \quad \Phi(x) = 0$$

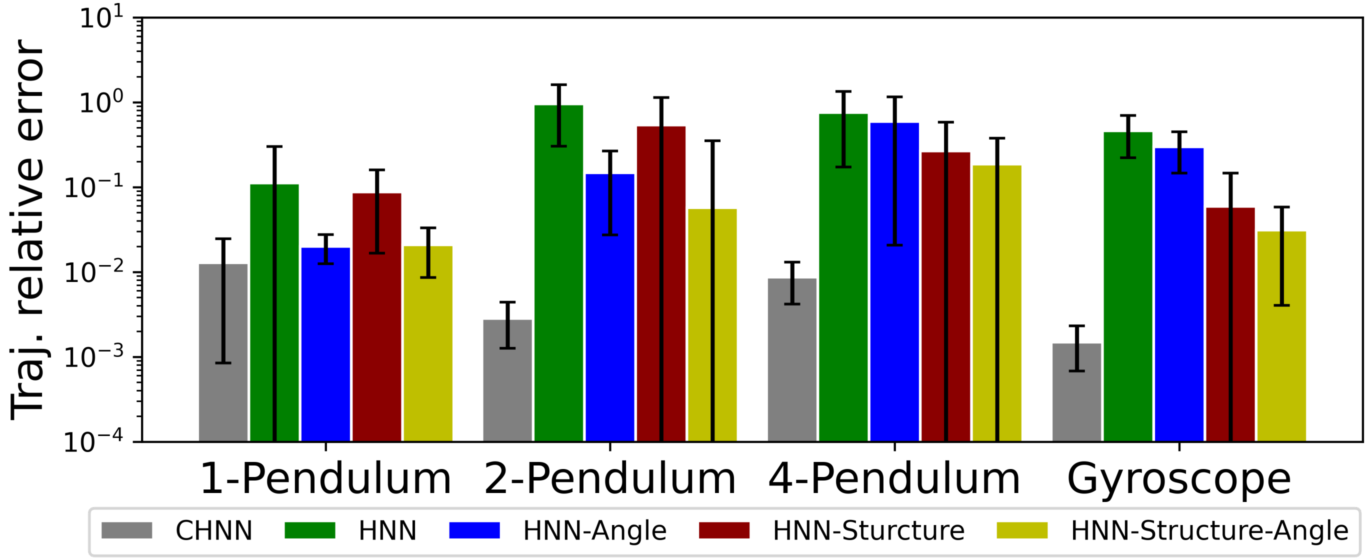
❑ Constrained Dynamics:

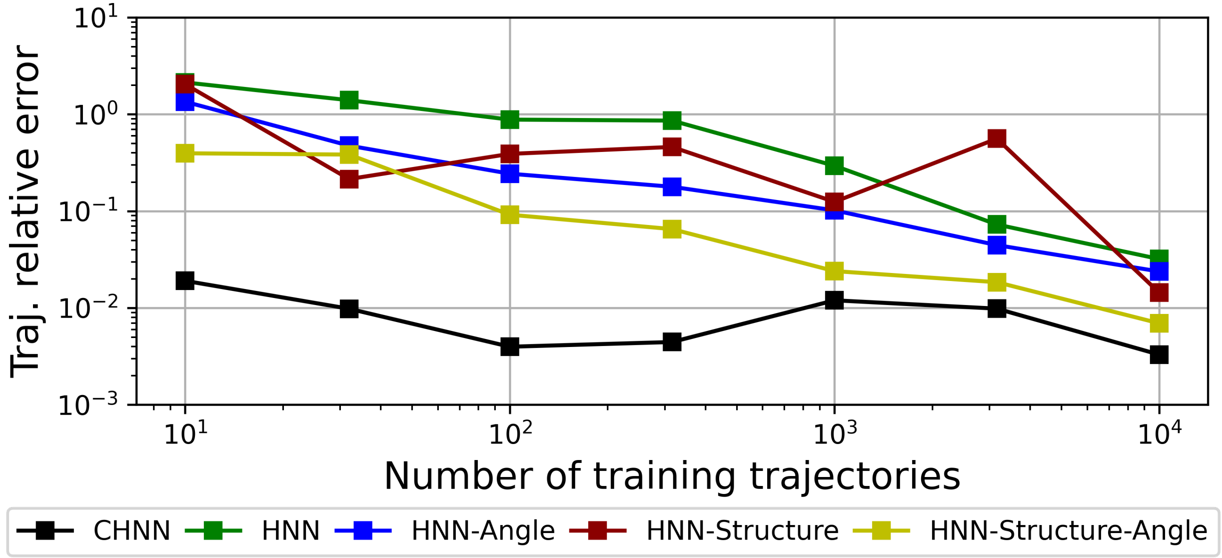
$$\left.\begin{array}{r} \Phi(x) = 0 \quad \Rightarrow \quad (D_x\Phi)\dot{x} = 0 \\ H = \frac{1}{2} p_x^T M^{-1} p_x + V(x) \quad \Rightarrow \quad \dot{x} = M^{-1} p_x \end{array}\right\} \Longrightarrow \quad (D_x\Phi)M^{-1}p_x = 0 \quad \Longrightarrow \quad \Psi(x, p_x) = \begin{bmatrix} \Phi(x) \\ (D_x\Phi)M^{-1}p_x \end{bmatrix}$$

$$\begin{bmatrix} \dot{x} \\ \dot{p}_x \end{bmatrix} = \begin{bmatrix} \frac{\partial H}{\partial p_x} \\ -\frac{\partial H}{\partial x} \end{bmatrix} + \begin{bmatrix} 0 \\ g(x) \end{bmatrix} u - \left[\begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} (D_{[x,p_x]}\Psi)^T \left((D_{[x,p_x]}\Psi) \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix} (D_{[x,p_x]}\Psi)^T \right)^{-1} (D_{[x,p_x]}\Psi) \right] \begin{bmatrix} \frac{\partial H}{\partial p_x} \\ -\frac{\partial H}{\partial x} \end{bmatrix}$$

* Finzi, Wang, Wilson | **Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints** | NeurIPS 2020.
* Zhong, BD, Chakraborty | **Benchmarking Energy-Conserving Neural Networks for Learning Dynamics from Data** | L4DC 2021.

Explicit constraints lead to significant improvement in performance





- Models that enforce **explicit constraints** can generate **predictions that are significantly better** than those from models with implicit constraints.
- On the other hand, models that enforce **implicit constraints** are **easier to implement**.

* Zhong, BD, Chakraborty | **Benchmarking Energy-Conserving Neural Networks for Learning Dynamics from Data** | L4DC 2020.

SIEMENS

Symplectic ODENet can also be **extended to accommodate energy dissipation**

Without Dissipation

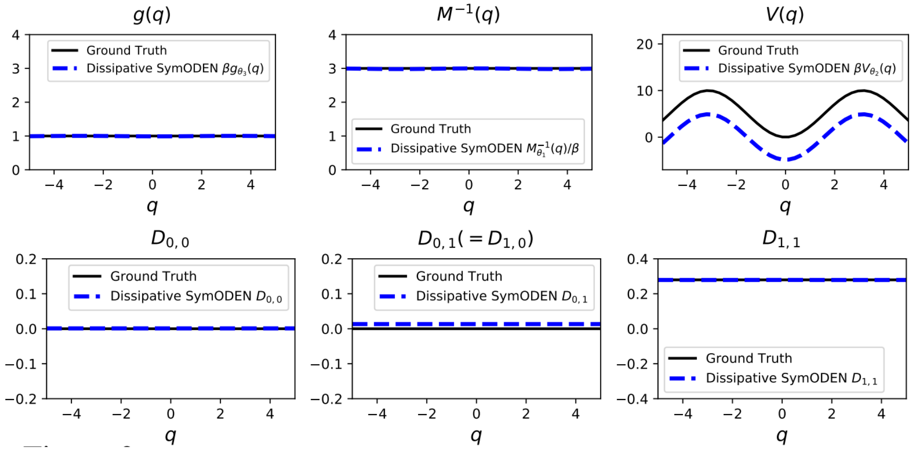
$$\begin{bmatrix}\dot{q}\\ \dot{p}\end{bmatrix} = \begin{bmatrix}\frac{\partial H}{\partial p}\\ -\frac{\partial H}{\partial q}\end{bmatrix} + \begin{bmatrix}0\\ g(q)\end{bmatrix}u$$

With Dissipation

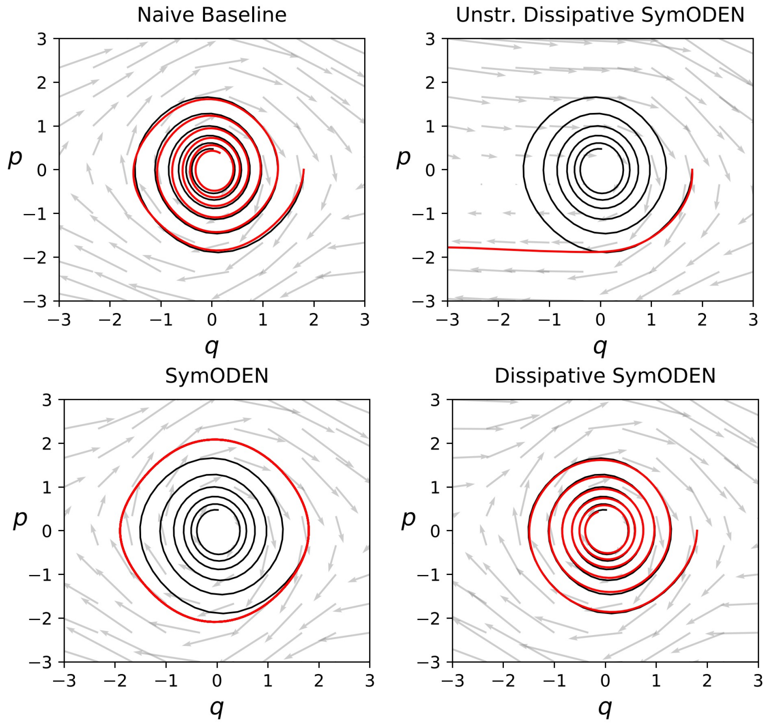
$$\begin{bmatrix}\dot{q}\\ \dot{p}\end{bmatrix} = \left(\begin{bmatrix}0 & I\\ -I & 0\end{bmatrix} - D(q)\right)\begin{bmatrix}\frac{\partial H}{\partial q}\\ \frac{\partial H}{\partial p}\end{bmatrix} + \begin{bmatrix}0\\ g(q)\end{bmatrix}u$$

- ❑ $D(q)$: Positive semi-definite dissipation matrix parametrized via a *Fully-connected Feedforward Network*

Pendulum



Learned functions



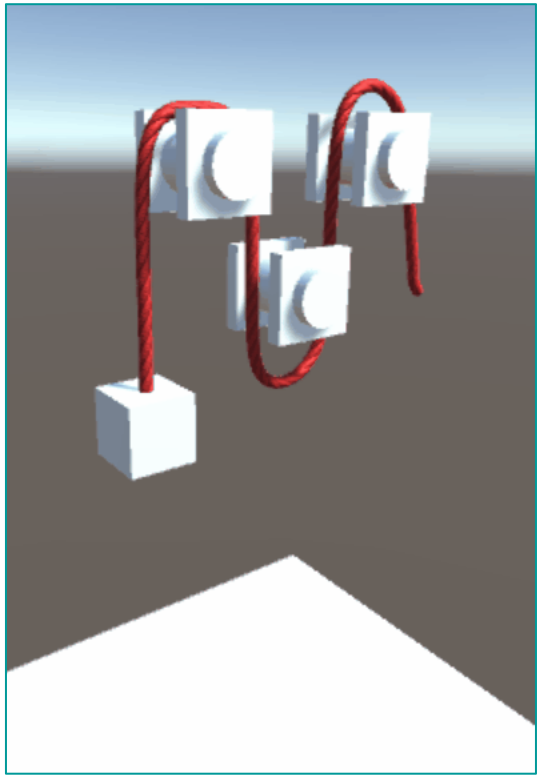
Learned Vector Field

  | Biswadip Dey | ACC Workshop | 05-30-2023

* Zhong, BD, Chakraborty | **Dissipative SymODEN: Encoding Hamiltonian Dynamics with Dissipation and Control into Deep Learning** | DeepDiffEq Workshop, ICLR 2020.

SIEMENS

Can we extend these models to accommodate contacts and collisions?

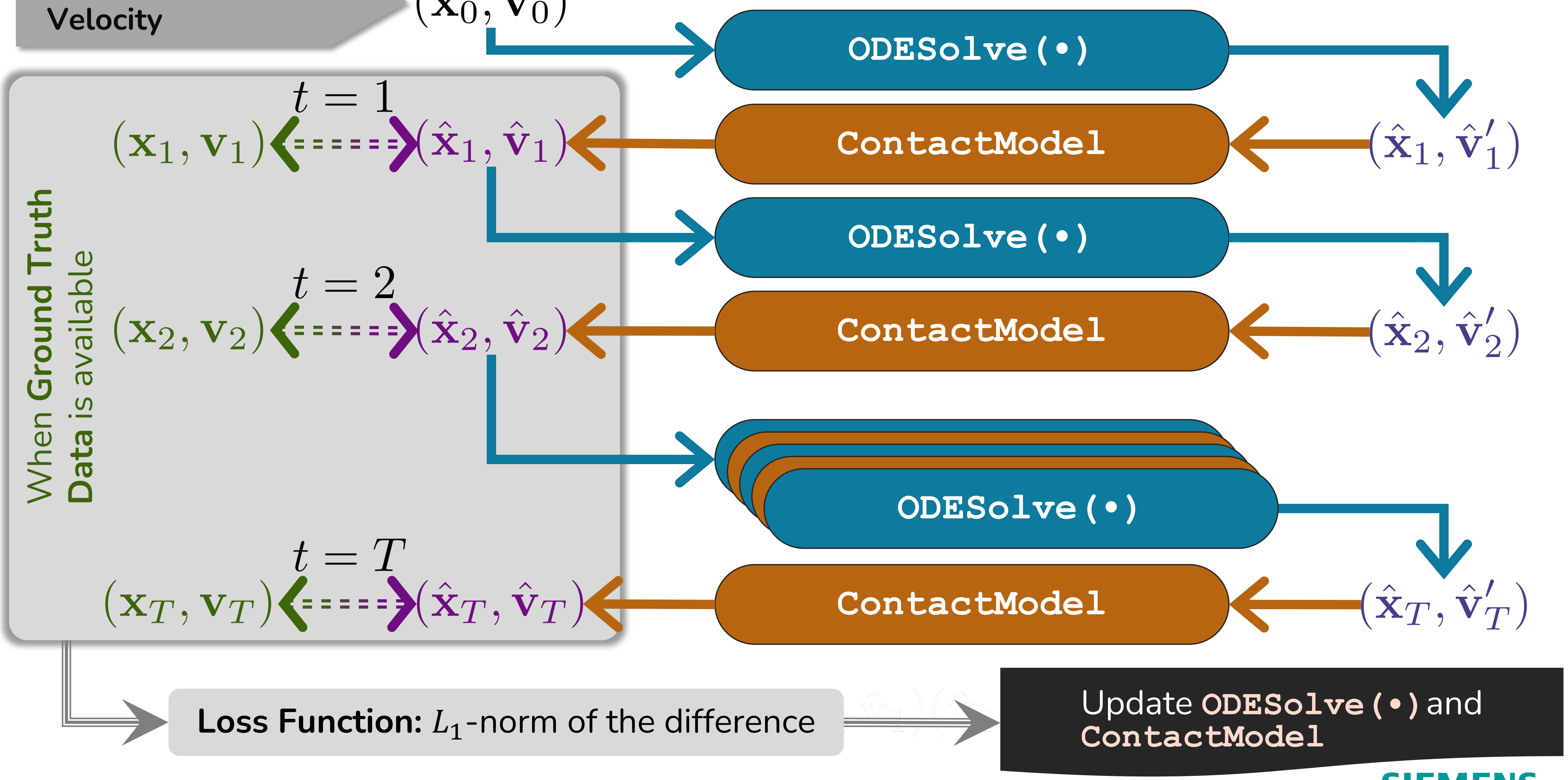


- We utilize maximum dissipation principle to solve post-contact velocities
- We formulate the problem as a two-phase convex optimization problem
  - Compression Phase
  - Restitution Phase
- This formulation allows us to use *differentiable optimization*[9]

* Zhong, BD, Chakraborty | **Extending Lagrangian and Hamiltonian neural networks with differentiable contact model** | NeurIPS 2021.
[9] Agrawal *et. al.* | **Differentiable Convex Optimization Layers** | NeurIPS 2019.

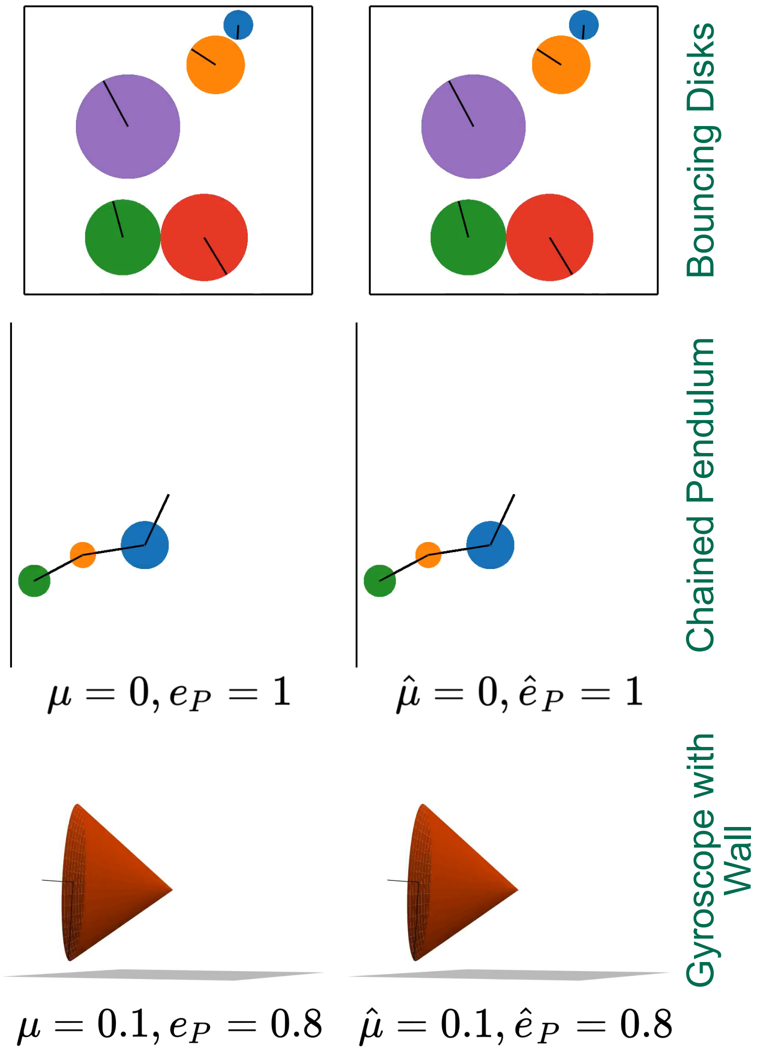
SIEMENS

Dynamics and Parameter Learning



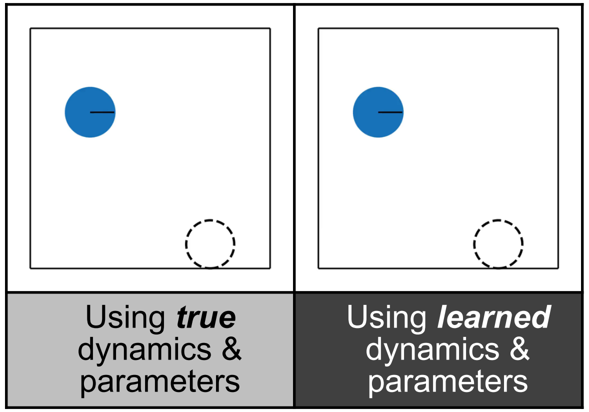
Results

Prediction

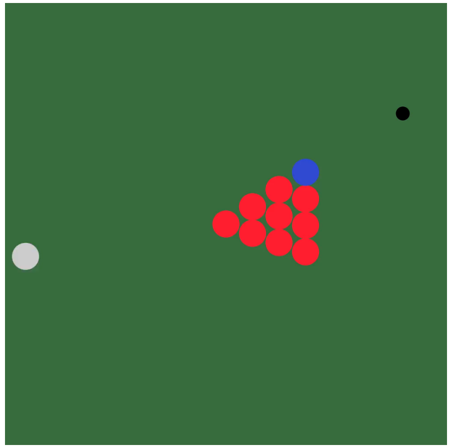


Trajectory Planning

- **Goal**: Hit the target (black) after one bounce off the ground
- **Variable**: Initial *velocity (both linear* and *angular)*



Find the initial position and velocity of the white ball so that the blue ball hits the black target at the 1024th time step



SIEMENS

Key Take-away

- ✓ **Physics-informed ML** exploits the underlying **laws of physics** to define an appropriate **Inductive Bias** (e.g., **ML architecture**, **Loss function**) for the learning framework
- ✓ This **improves** the **model transparency**, **learning speed**, **data efficiency**, and **generalization performance**

The work discussed in this presentation has been done in collaboration with:

Y. D. Zhong

A. Chakraborty

biswa-dey@ieee.org

https://d-biswa.github.io/

@DBiswadip