Physics-informed Machine Learning for Inverse Problems

Biswadip Dey

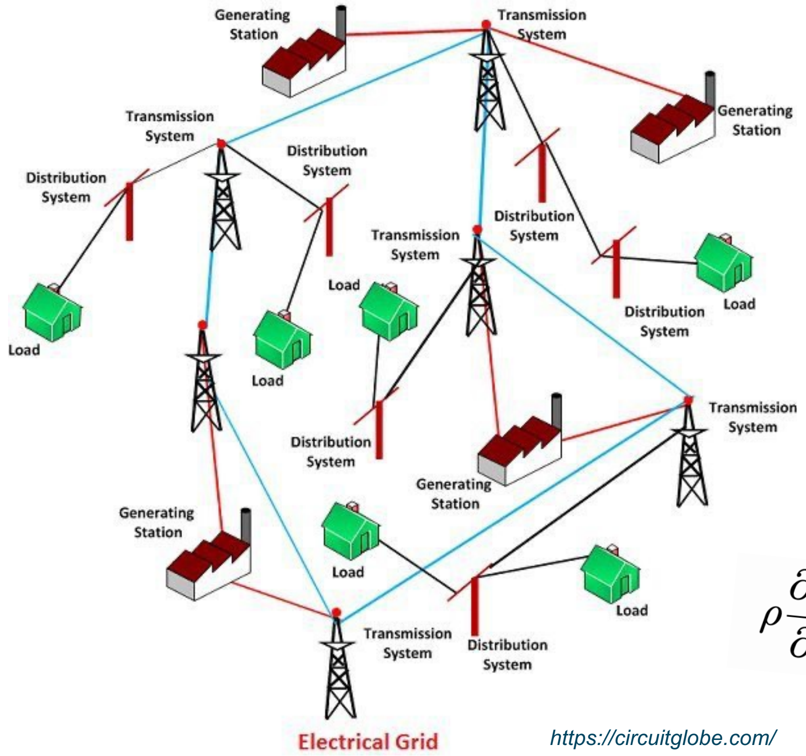
Siemens Corporation, Technology

American Control Conference – 2023

San Diego, CA, USA | June 2, 2023

SIEMENS

Inverse Problems are **Ubiquitous** in Systems and Control



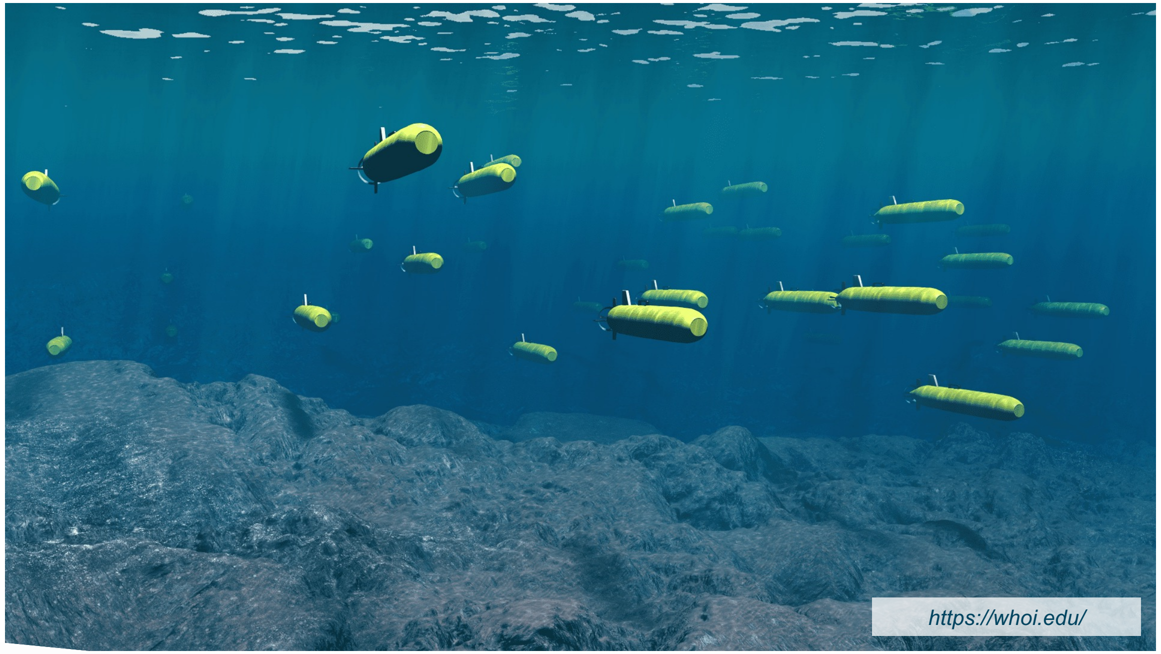
https://circuitglobe.com/

$$\dot{x}(t) = Ax(t) + Bu(t)$$
$$r(t) = Cx(t)$$

$$\dot{g} = T_e L_g \cdot \xi_u$$

$$x_{n+1} = f(x_n)$$

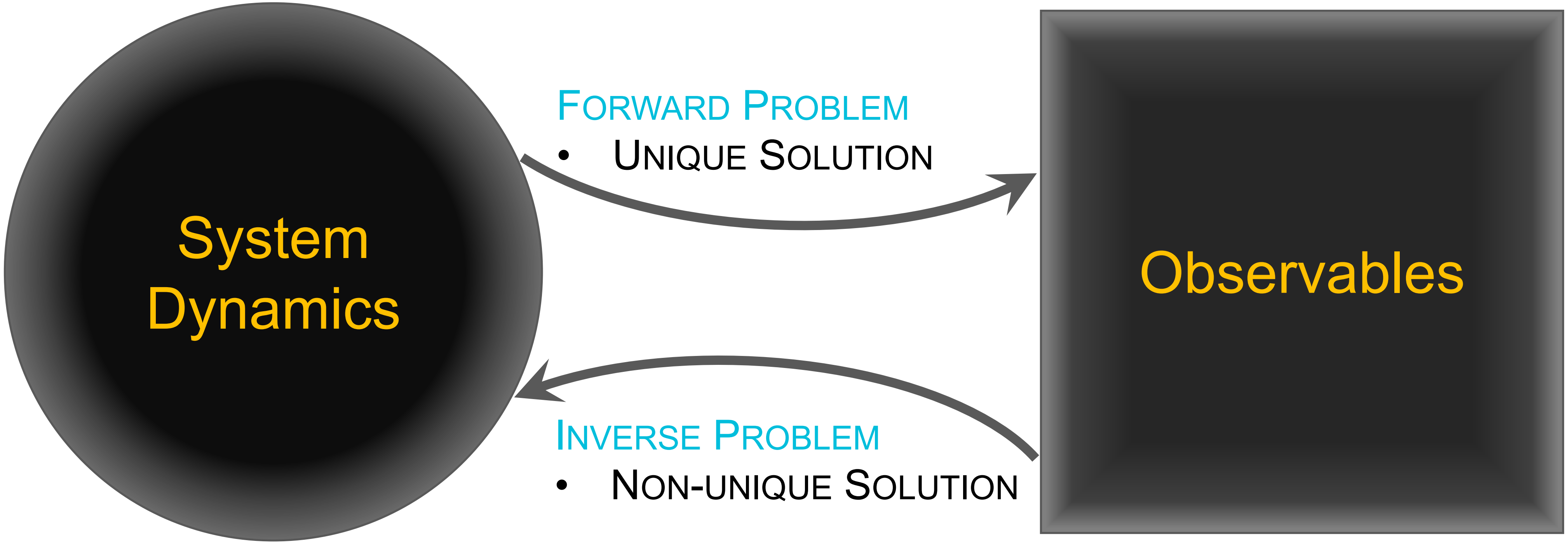
$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}_{ext}$$



System Identification
Parameter Estimation
Data Assimilation

SIEMENS

An inverse problem can be made well-posed via a relevant **inductive bias**



Need to use appropriate **<u>inductive bias</u>** while inferring functional relationships from data!

Real-world systems often **lack good quality data**, but **underlying physics is fairly known**

PIML: *ML approaches with inductive bias defined explicitly by the laws of physics*

"PHYSICS-INFORMED ML" exploits **"the laws of physics"** *(e.g., Mechanics, Thermodynamics)* in its design!

Architecture of ML Pipeline	Learning Objective	Dataset

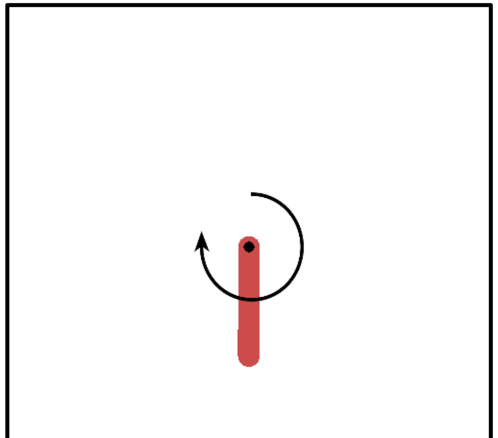
AI/ML SYSTEMS

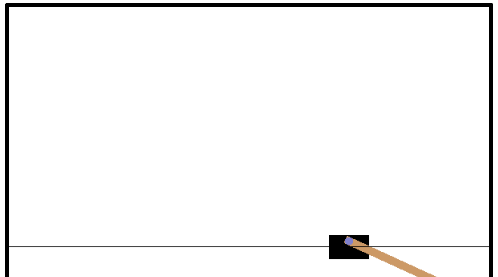
Value Proposition

- ✓ Better generalization
- ✓ Data-efficiency
- ✓ Transparency / Grey-box models
- ✓ Increase in learning speed

SIEMENS

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


Pendulum


Cart-pole

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

- Generalized Coordinate – q
- Generalized Momentum – p
- Hamiltonian – H – which usually represents the *total energy*

$$H(q,p) = \underbrace{\frac{1}{2}\, p^T M^{-1}(q)p}_{\text{Kinetic energy}} + \underbrace{V(q)}_{\text{Potential energy}}$$

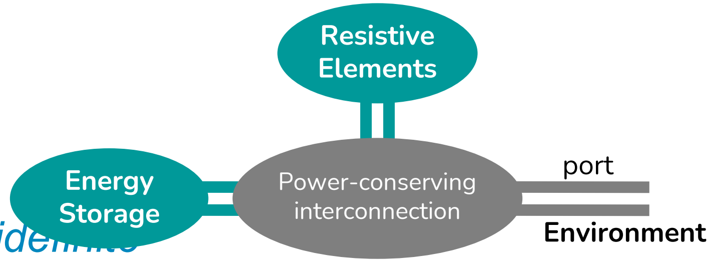
- External Control – u

Port-Hamiltonian System:

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

Symmetric, Positive-semidefinite

Skew-symmetric



SIEMENS

How do we **encode Hamiltonian dynamics** into neural network architecture?

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

Consider an ODE – $\dot{x} = f_\theta(x, u)$, where $f_\theta(x)$ is parametrized by a neural network

Use *Neural ODE Solvers* [5] 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!

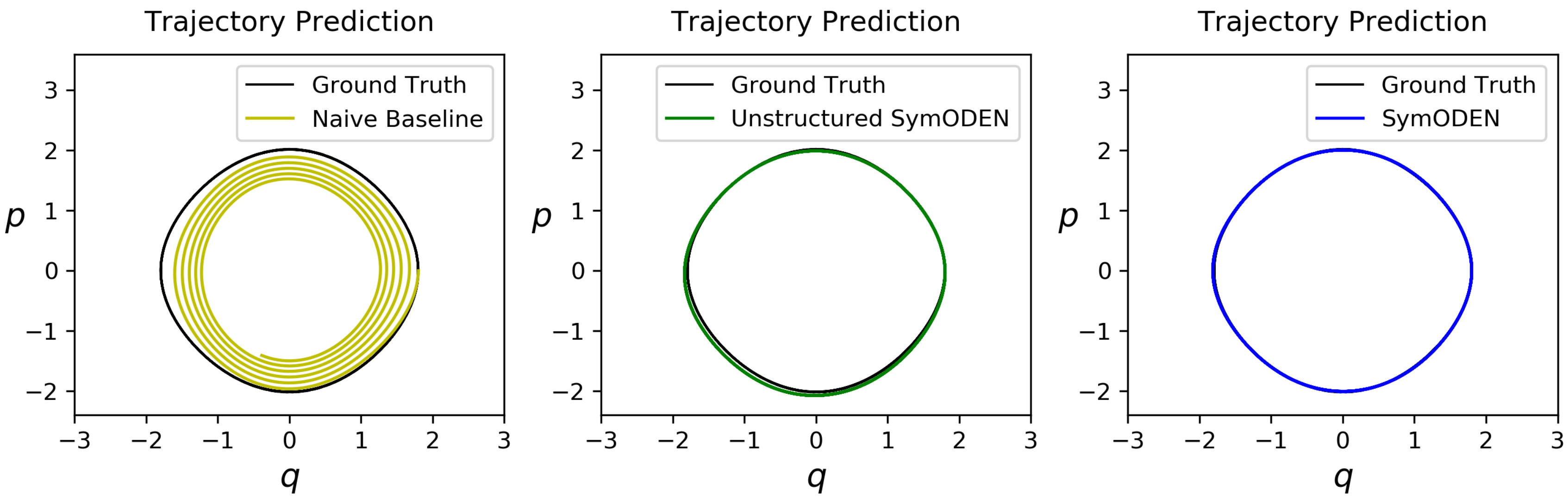
[5] Chen, Rubanova, Bettencourt, Duvenaud | **Neural Ordinary Differential Equations** | NeurIPS 2018.

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

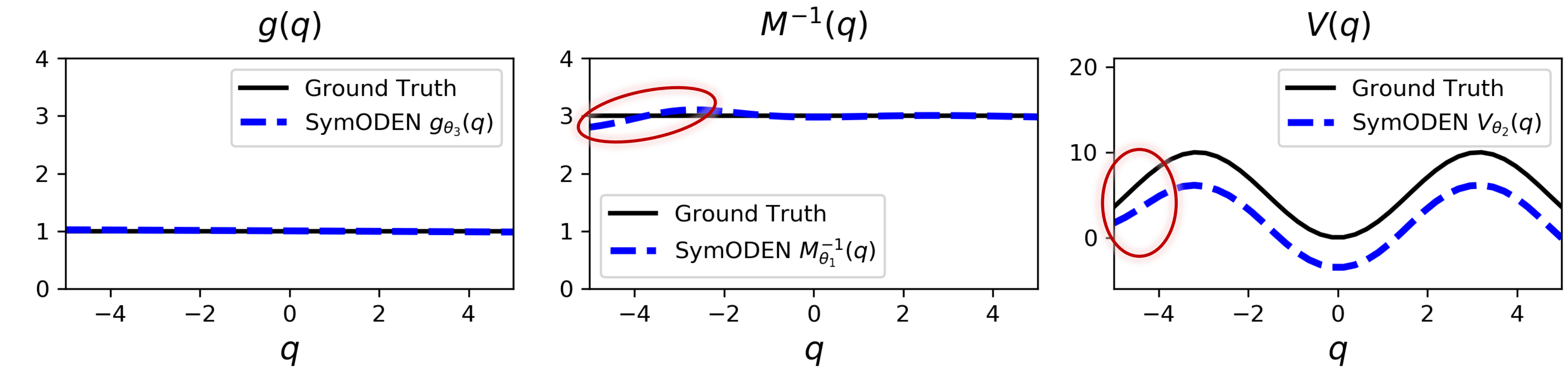
SIEMENS

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.

Question: *Can we bridge this gap?*

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)

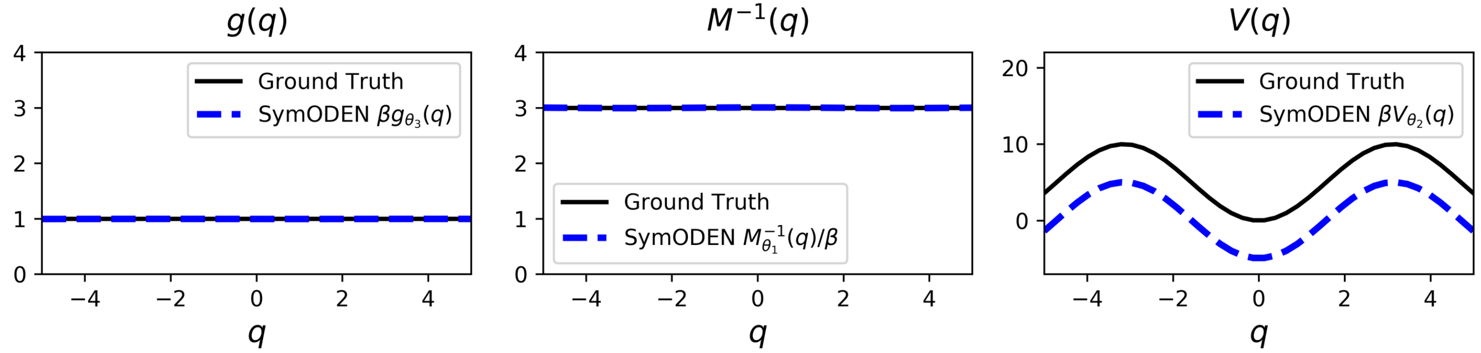
$$\begin{aligned}
\dot{\boldsymbol{x}}_1 &= -\sin \boldsymbol{q} \circ \dot{\boldsymbol{q}} = -\boldsymbol{x}_2 \circ \dot{\boldsymbol{q}} \\
\dot{\boldsymbol{x}}_2 &= \cos \boldsymbol{q} \circ \dot{\boldsymbol{q}} = \boldsymbol{x}_1 \circ \dot{\boldsymbol{q}} \\
\dot{\boldsymbol{x}}_3 &= \frac{d}{dt}\left(\boldsymbol{M}^{-1}(\boldsymbol{x}_1,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

where,

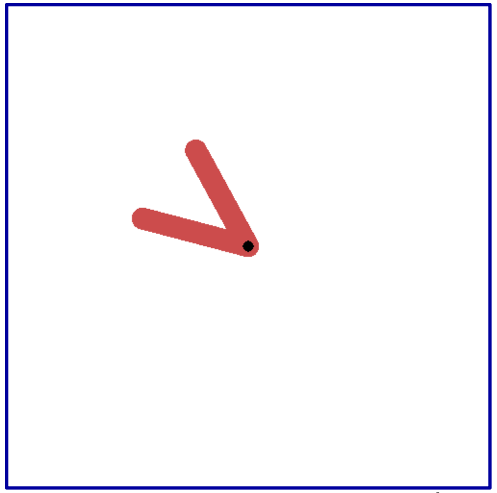
$$\begin{aligned}
\boldsymbol{p} &= \boldsymbol{M}(\boldsymbol{x}_1, \boldsymbol{x}_2) \cdot \boldsymbol{x}_3 \\
\dot{\boldsymbol{q}} &= \frac{\partial H(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{p})}{\partial \boldsymbol{p}} \\
\dot{\boldsymbol{p}} &= -\frac{\partial H(\boldsymbol{x}_1, \boldsymbol{x}_2,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

Angle-aware Design leads to performance improvement

➢ Learned functions

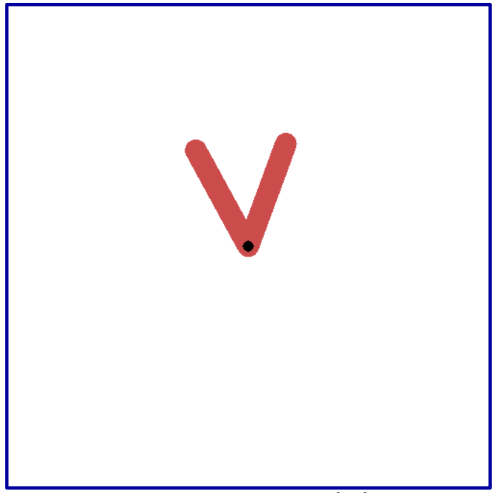


➢ Prediction

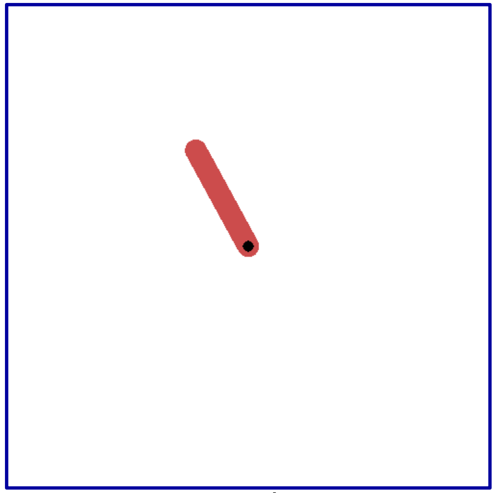


Gray: Ground Truth
Orange: Prediction

Baseline
No energy conservation



Model-variant
Unstructured Hamiltonian



Symplectic ODENet
Structured Hamiltonian

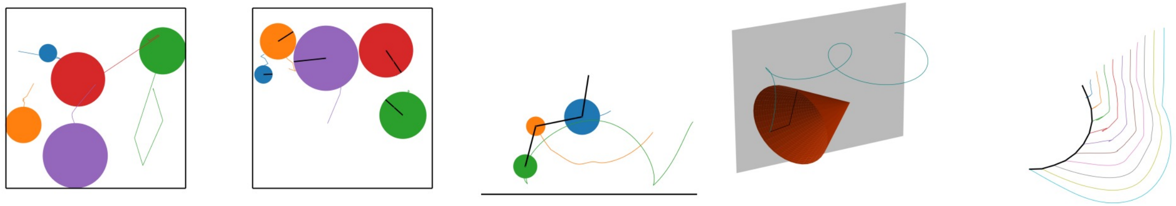
Extending the scope to accommodate **dissipation**, **embedded representation**, and **contacts/collisions**

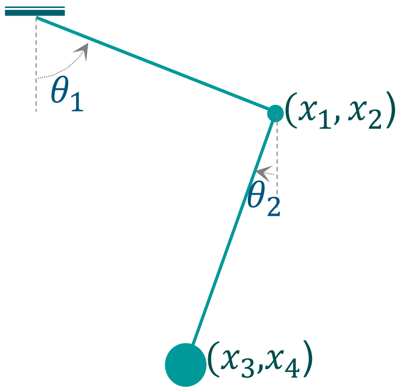
Simplifying Hamiltonian and Lagrangian Neural Networks via Explicit Constraints

Marc Finzi* New York University

Ke Alexander Wang* Cornell University

Andrew Gordon Wilson New York University

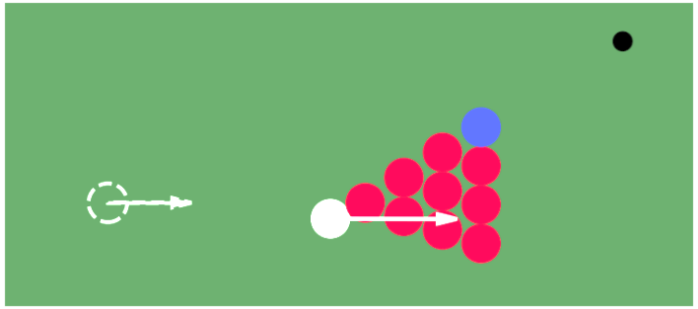


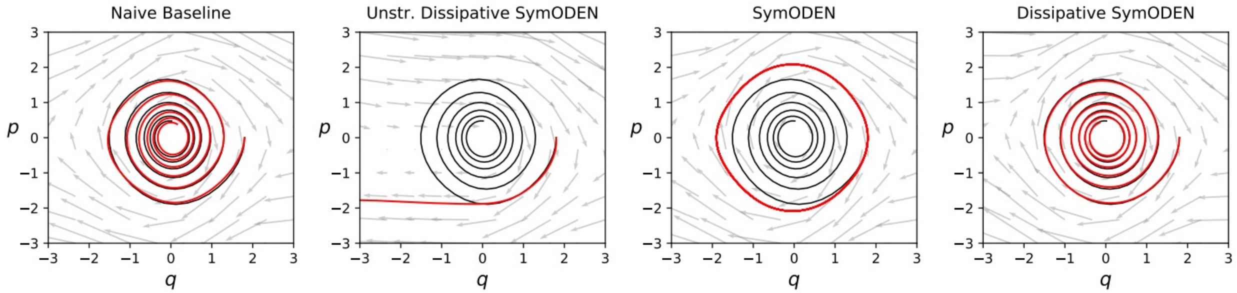


ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations (DeepDiffEq)

DISSIPATIVE SYMODEN: ENCODING HAMILTONIAN DYNAMICS WITH DISSIPATION AND CONTROL INTO DEEP LEARNING

Yaofeng Desmond Zhong[†]*, Biswadip Dey[‡], and Amit Chakraborty[‡]
[†]Princeton University , [‡]Siemens Corporation, Corporate Technology
y.zhong@princeton.edu, (biswadip.dey,amit.chakraborty)@siemens.com





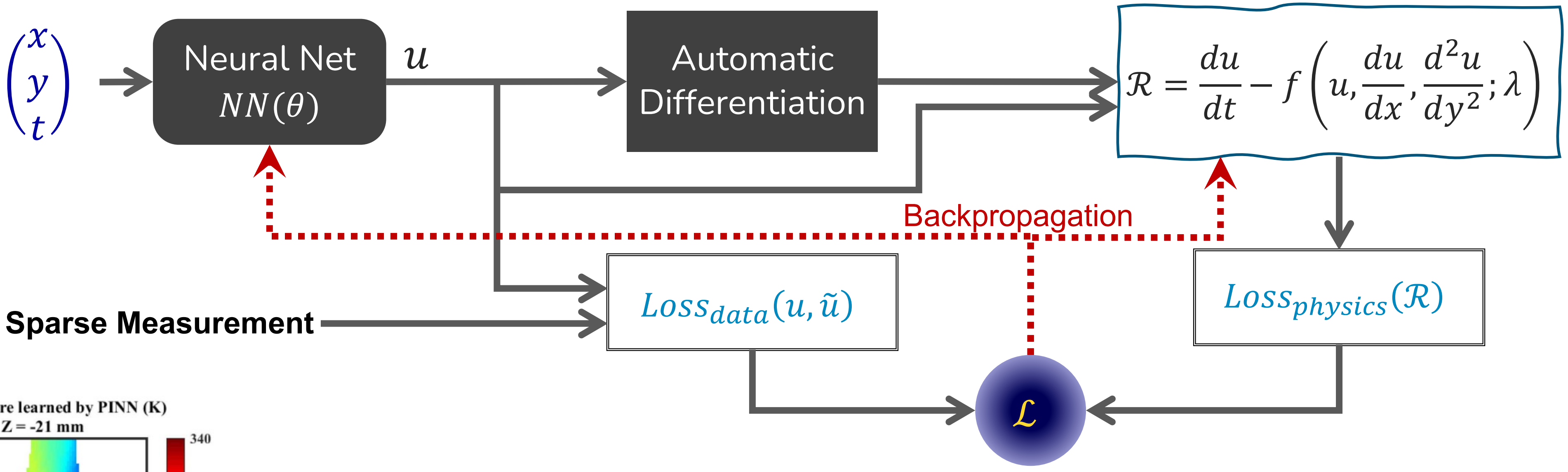
Extending Lagrangian and Hamiltonian Neural Networks with Differentiable Contact Models

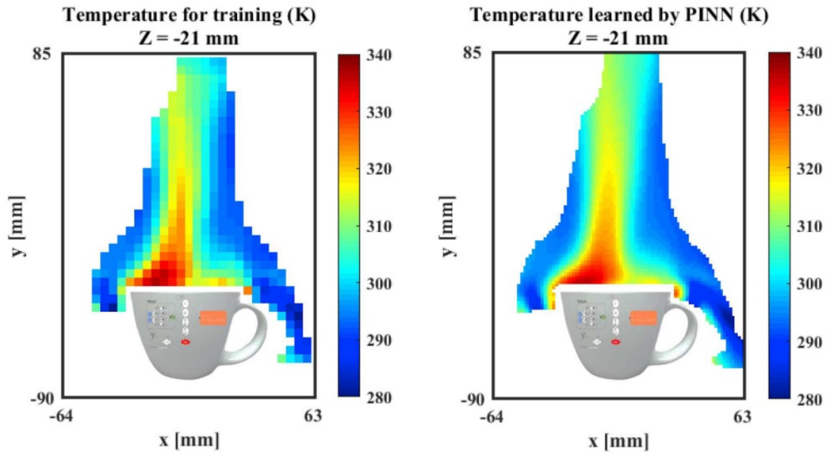
Yaofeng Desmond Zhong[†], Biswadip Dey[†], Amit Chakraborty[†]
[†]Siemens Technology, Princeton, NJ 08536, USA.
{yaofeng.zhong, biswadip.dey, amit.chakraborty}@siemens.com

An alternative approach: **Encoding physics into the learning objective**

Differential Equation

$$\frac{du}{dt} = f\left(u, \frac{du}{dx}, \frac{d^2u}{dy^2}; \lambda\right)$$





Cai et al. | **Flow over an espresso cup** | J. Fluid Mech.

Correct PDE	$u_t + (uu_x + vu_y) = -p_x + 0.01(u_{xx} + u_{yy})$ $v_t + (uv_x + vv_y) = -p_y + 0.01(v_{xx} + v_{yy})$
Identified PDE (clean data)	$u_t + 0.999(uu_x + vu_y) = -p_x + 0.01047(u_{xx} + u_{yy})$ $v_t + 0.999(uv_x + vv_y) = -p_y + 0.01047(v_{xx} + v_{yy})$
Identified PDE (1% noise)	$u_t + 0.998(uu_x + vu_y) = -p_x + 0.01057(u_{xx} + u_{yy})$ $v_t + 0.998(uv_x + vv_y) = -p_y + 0.01057(v_{xx} + v_{yy})$

Raissi, Perdikaris, Karniadakis | **PINN: A DL framework for solving forward and inverse problems involving nonlinear PDEs** | J. Comp. Physics

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 solving the inverse problem
- ✓ This leads to **improvement** in **model transparency**, **learning speed**, **data efficiency**, and **generalization performance**

biswa-dey@ieee.org

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

@DBiswadip

