

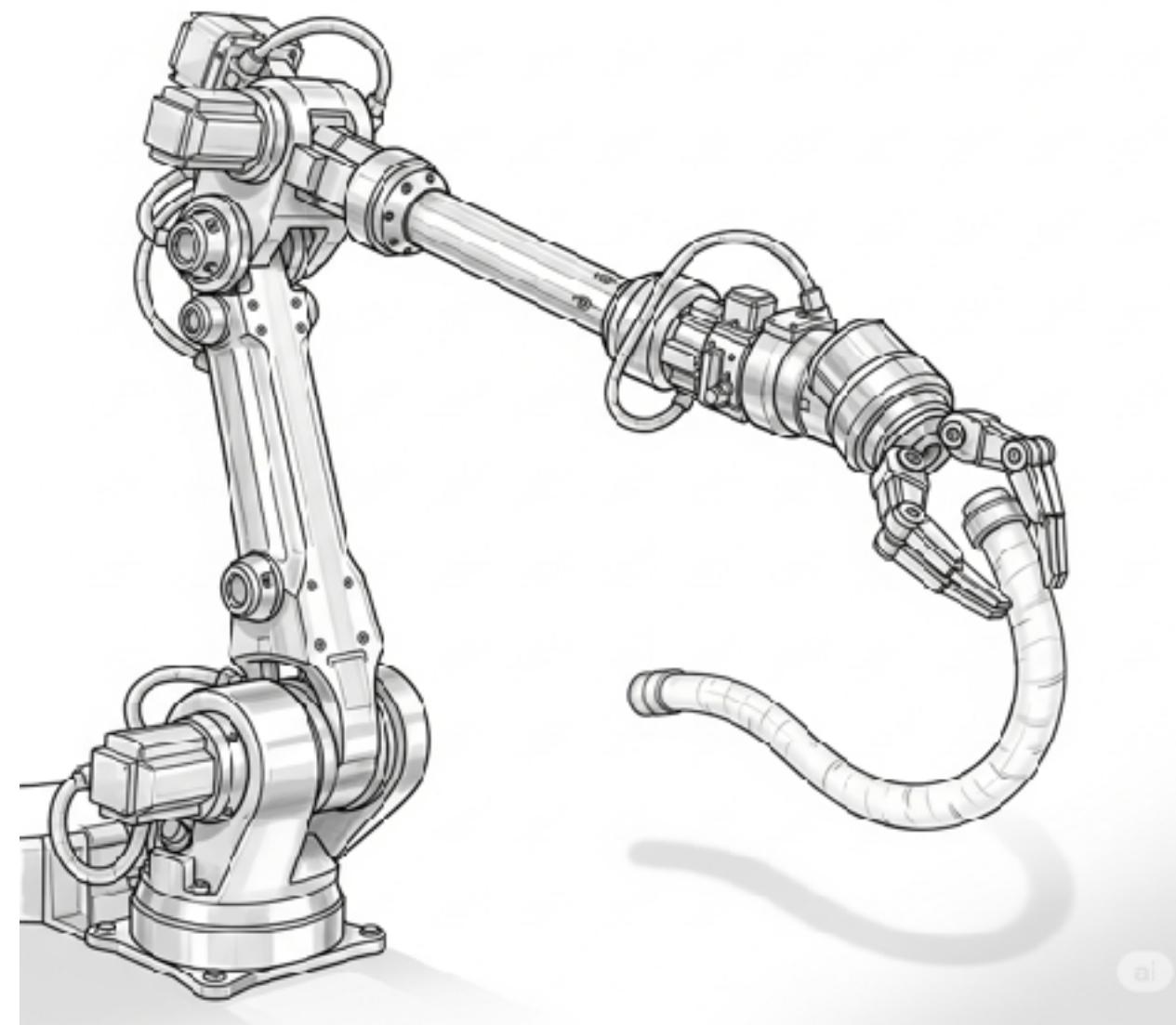


Structure- and Stability-Preserved Learning with Port-Hamiltonian Systems

Dr.-Ing. Thomas Beckers
Vanderbilt University

Tutorial Session on Safe Physics-Informed Machine Learning for Dynamics and Control
July 8th, 2025

Modeling of complex systems



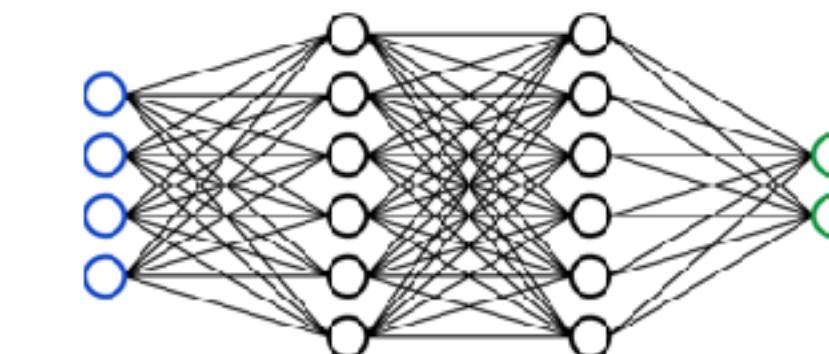
Physics-based

Robot $\dot{x} = f(x, u) = \dots$

Tube $0 = f\left(\frac{\partial x}{\partial t}, \dots, \frac{\partial x}{\partial z}, \dots\right)$

- ✓ Physical consistent
- ✓ Generalization
- ✗ Model selection (nonlinear)
- ✗ Time-consuming

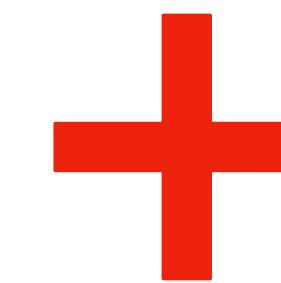
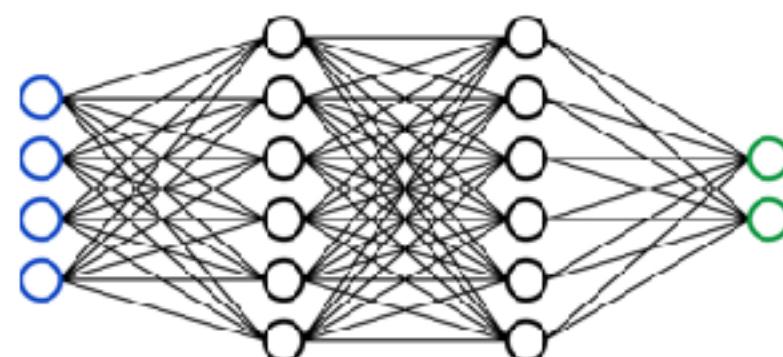
Learning-based



- ✓ Expressive models
- ✓ Minimal expert knowledge
- ✗ Physical correctness (structure)
- ✗ Stability, passivity?

Combining the best of both worlds

Learning-based

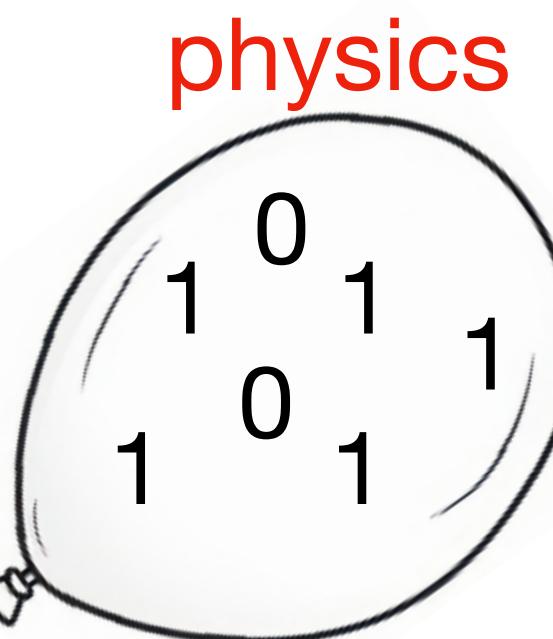


- Structure to constrain/inform the model output to be **physically consistent**
- Data-driven methods to fill the physics structure

Structure / Physics-based

$$\begin{aligned}\dot{x} &= f(x, u) = \dots \\ 0 &= f\left(\frac{\partial x}{\partial t}, \dots, \frac{\partial x}{\partial z}, \dots\right)\end{aligned}$$

Data...101011011



“Best” Learning technique?

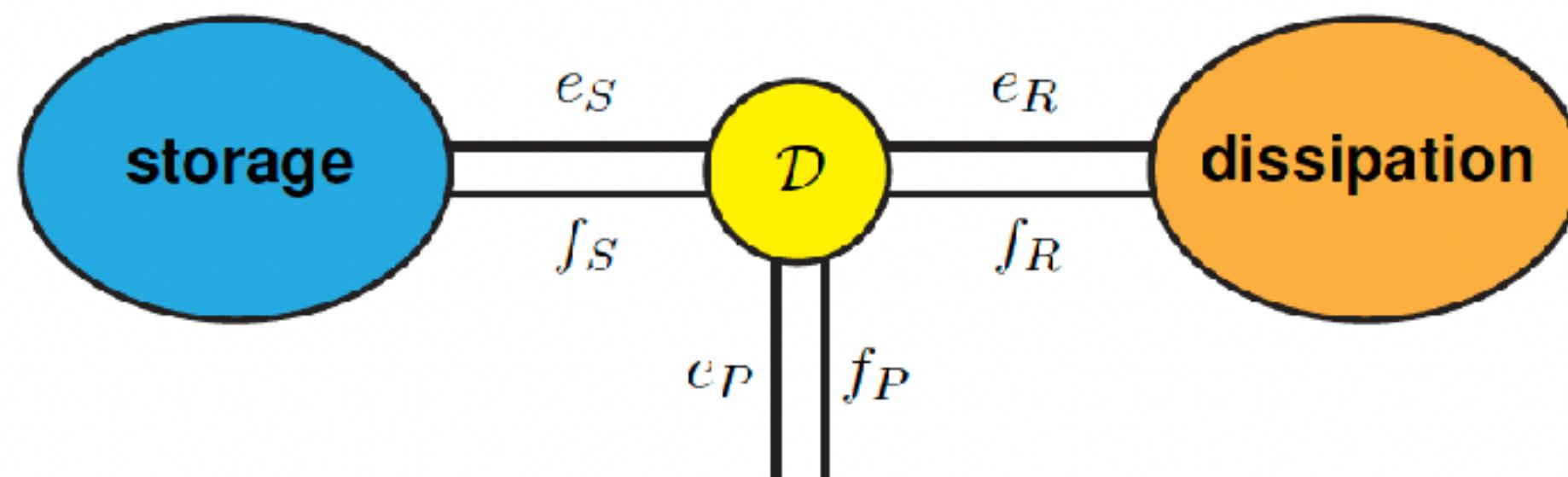
- Neural Networks
- GNNs, RNNs, LSTMs
- Gaussian Processes
- Operator Learning

“Best” Structure?

- Parametric equation
- Hamiltonian Mechanics
- Lagrangian Mechanics
- Port-Hamiltonian Systems

Choice depends on the use-case!

Physics model



[A. Van Der Schaft and D. Jeltsema]

Port-Hamiltonian system

Interconnection Dissipation Hamiltonian

$$\dot{x} = [J(x) - R(x)] \frac{\partial H}{\partial x} + G(x)u$$
$$y = G(x)^T \frac{\partial H}{\partial x}$$

Input
Output

The equations define the dynamics of a Port-Hamiltonian system. The first equation, $\dot{x} = [J(x) - R(x)] \frac{\partial H}{\partial x} + G(x)u$, represents the state evolution. The second equation, $y = G(x)^T \frac{\partial H}{\partial x}$, represents the output. Red arrows point from the labels "Interconnection", "Dissipation", and "Hamiltonian" to the corresponding terms in the first equation. A red arrow points from the label "Input" to the term $G(x)u$. A red arrow points from the label "Output" to the term $G(x)^T \frac{\partial H}{\partial x}$.

- Skew-symmetric J ensures **energy conservation**
- Input and output defines a **passive systems**
- Connection with another PHS **preserves the structure**
- Suitable for **multi-domain applications** via energy flows

Port-Hamiltonian use-cases

2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Port-Ha

Abstract—In this paper we propose a framework for human commands to control robots. Such a human-robot interaction framework is suitable for complex systems. In this paper we model the interaction between a human and a robot manipulating an object, a car, and a truck. Furthermore, we propose a framework for the interaction between a human and a robot.



Review article

Port-Hamiltonian

Maris Tõnsu, V

Department of Software S

ARTICLE I

Keywords:

Port-Hamiltonian system

Power systems

Microgrids

Passivity based control

Interconnection and dan



The port H
of Continu

H. Hoang^{a,b}, F

^a LAGEP, Université de L

^b Université Lyon 1, Ville

^c FEMTO-ST AS2M, ENS

ARTICLE I

Article history:

Received 29 November

Received in revised for

Accepted 20 June 2011

Available online 31 July

Contents lists available at ScienceDirect

Energy Reports



Contents lists available at ScienceDirect



52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy



Port

of Continu

H. Hoang^{a,b}, F

^a LAGEP, Université de L

^b Université Lyon 1, Ville

^c FEMTO-ST AS2M, ENS

ARTICLE I

Article history:

Received 29 November

Received in revised for

Accepted 20 June 2011

Available online 31 July

Available online at www.sciencedirect.com

IOP Publishing

J. Phys. A: Math. Theor. 57 (2024) 295203 (25pp)

Journal of Physics A: Mathematical and Theoretical

<https://doi.org/10.1088/1751-8121/ad5d2f>

Stability analysis of a stochastic port-Hamiltonian car-following model

Barbara Rüdiger¹, Antoine Tordeux^{2,*}  and Baris E Ugurcan¹ 

¹ Group for Stochastic, IMACM, University of Wuppertal, Wuppertal, Germany

² Group for Traffic Safety and Reliability, IMACM, University of Wuppertal, Wuppertal, Germany

E-mail: tordeux@uni-wuppertal.de, ruediger@uni-wuppertal.de, beu4@cornell.edu and ugurcan@uni-wuppertal.de

Structured-learning approach

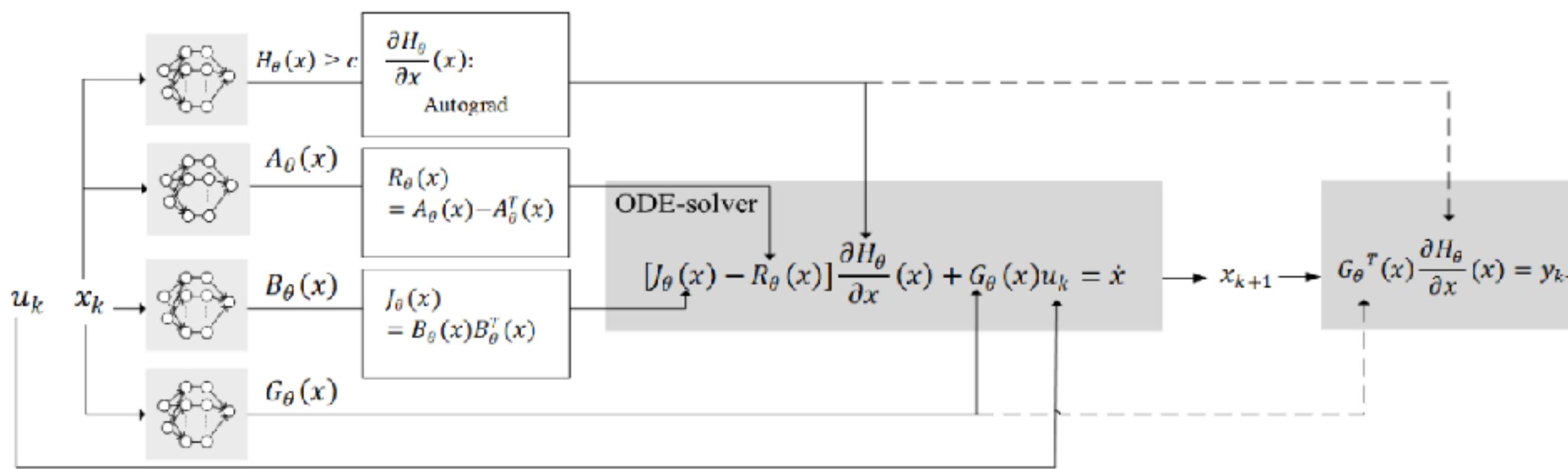
Port-Hamiltonian system

$$\dot{x} = [J(x) - R(x)] \frac{\partial H}{\partial x} + G(x)u$$

$$y = G(x)^\top \frac{\partial H}{\partial x}$$

Port-Hamiltonian Neural Networks

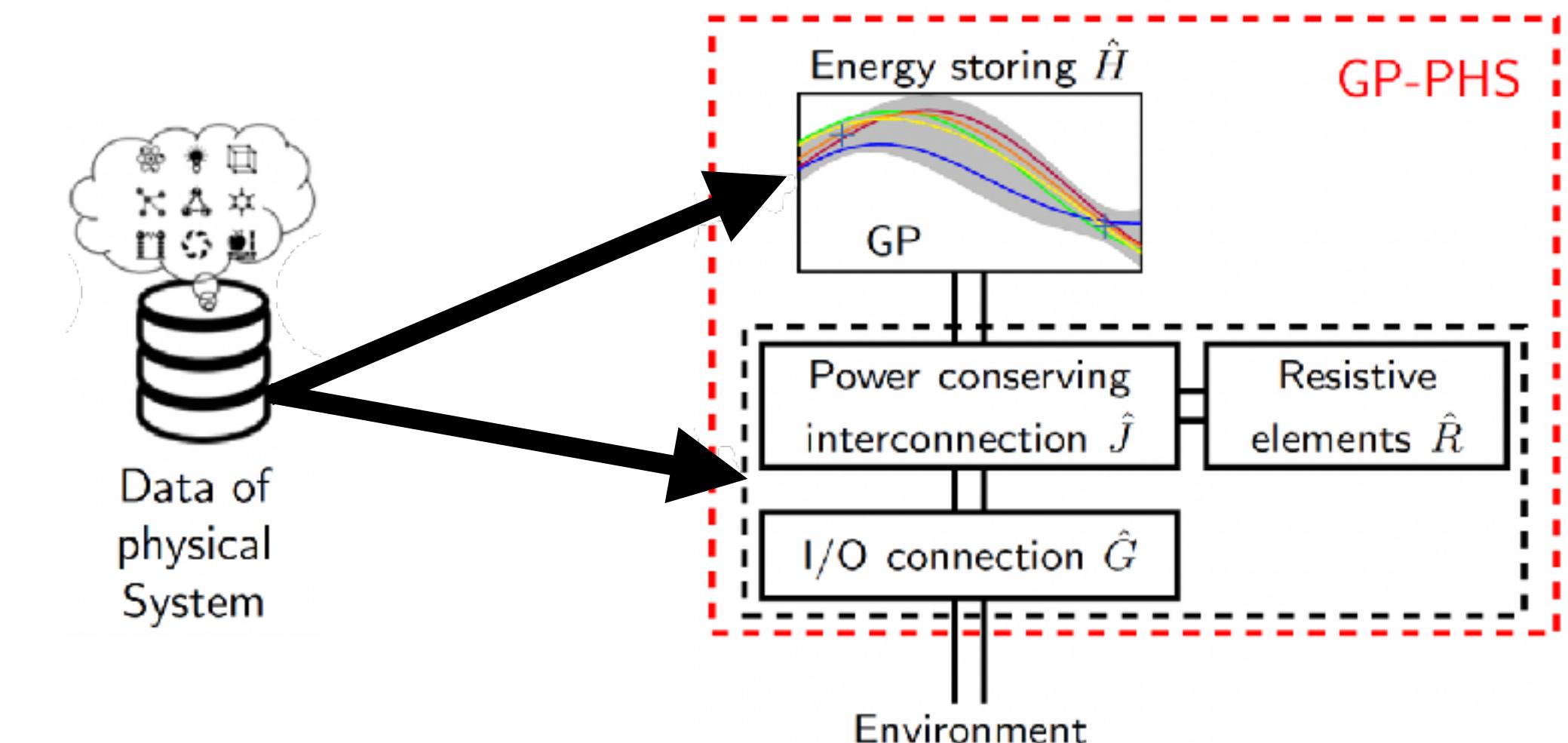
- Elements are approximated by a NN



[S. Moradi, et al.]

Gaussian Process Port-Hamiltonian systems

- Hamiltonian approximated by a GP



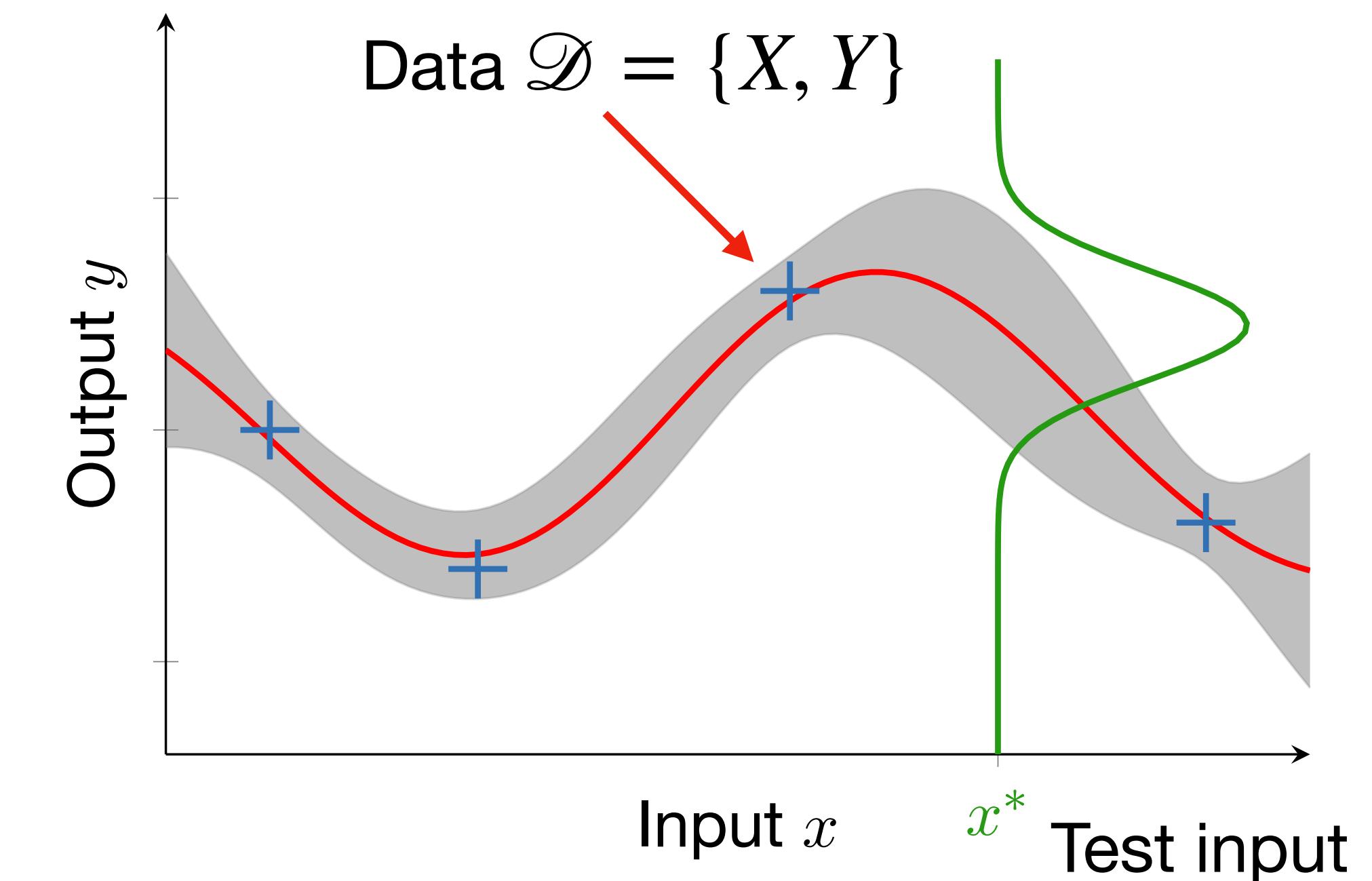
Gaussian process

Prior: Gaussian distribution over function space

$$f(x) \sim GP(m(x), k(x, x'))$$

Mean function Covariance

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}}$$



Posterior mean

$$\mu(y | \mathbf{x}, \mathcal{D}) = m(\mathbf{x}) + \sum_i^{N_{\text{Data points}}} w_i k(\mathbf{x}, X_i)$$

Prior mean function

Covariance

Weighting factor includes X, Y

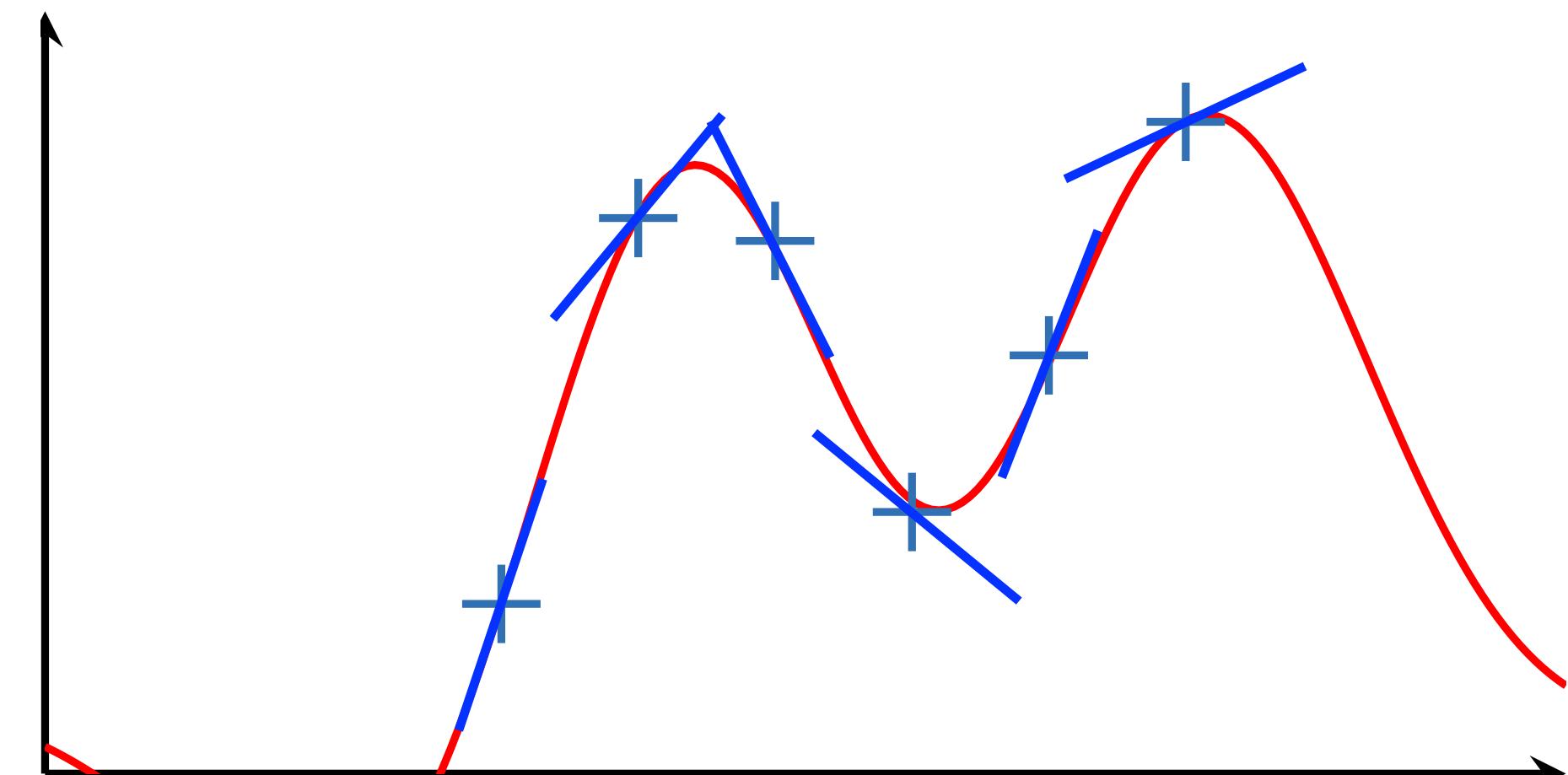
Linear operators

GPs are closed under linear operators

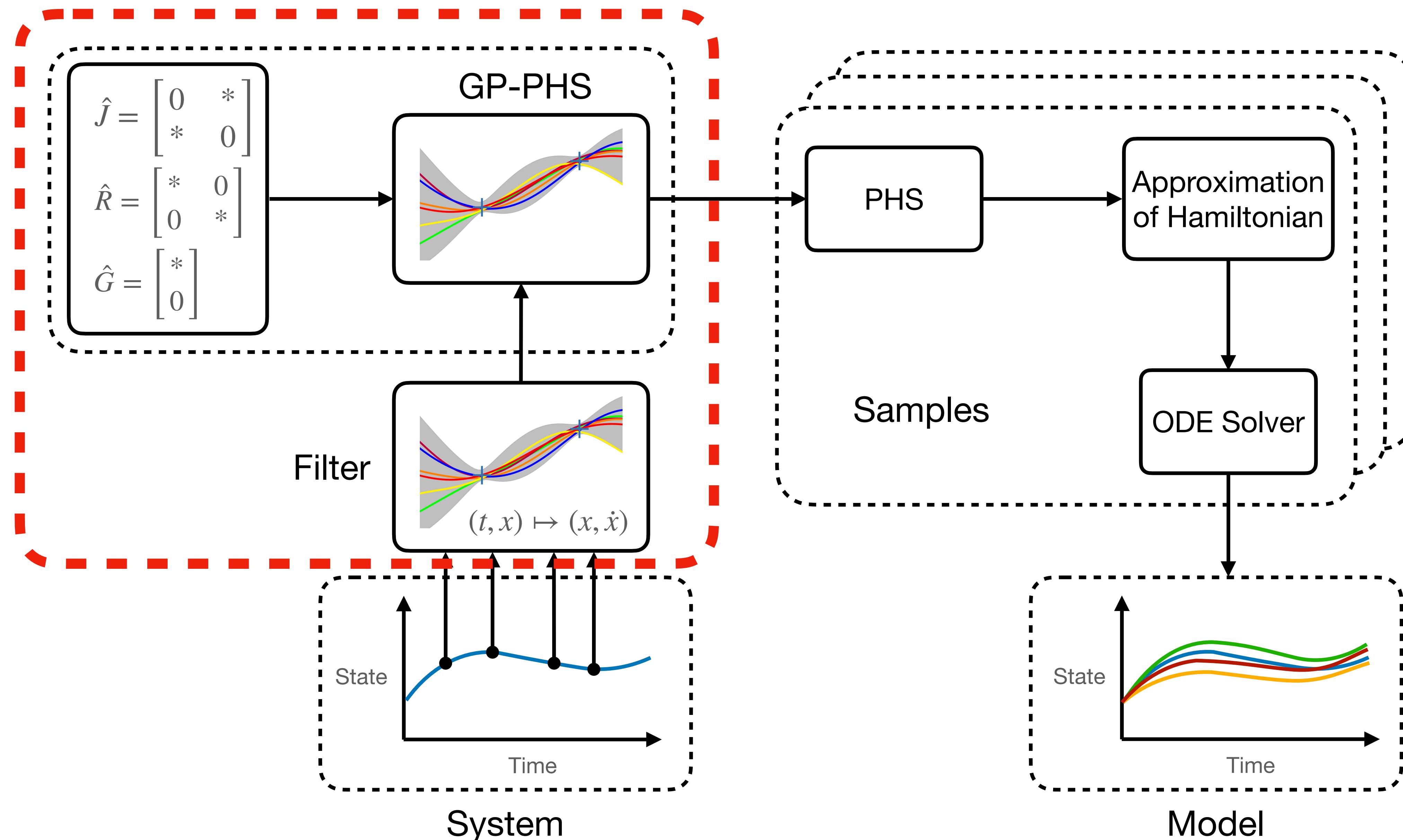
$$f(x) = \mathcal{L}_x g(x) \sim GP(\mathcal{L}_x \mu, \mathcal{L}_x k \mathcal{L}_x^\top)$$

Example: Differential operator $\mathcal{L}_x = \frac{\partial}{\partial x}$

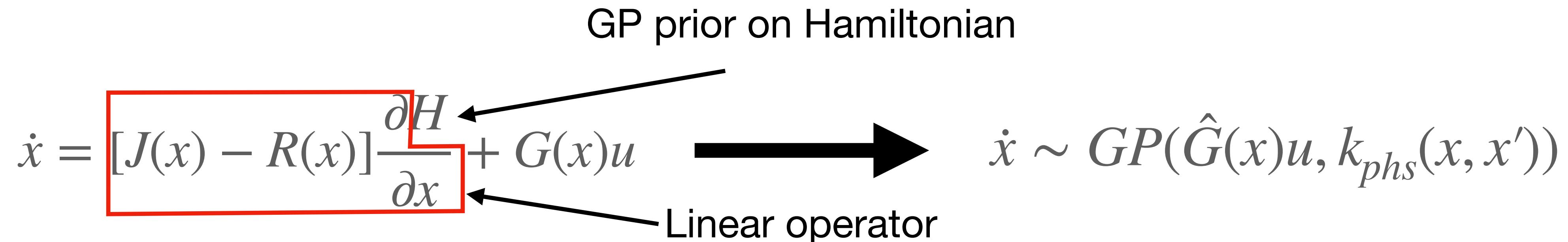
$$\begin{bmatrix} \mathbf{y} \\ \mathbf{Y}' \end{bmatrix} = \mathcal{N} \left(0, \begin{bmatrix} k(\mathbf{x}, \mathbf{x}) & \frac{\partial}{\partial x} k(\mathbf{x}, \mathbf{X}) \\ \frac{\partial}{\partial x} k(\mathbf{x}, \mathbf{X})^\top & \frac{\partial}{\partial x} k(\mathbf{X}, \mathbf{X}) \frac{\partial}{\partial x} \end{bmatrix} \right)$$



GP-PHS



GP-PHS training



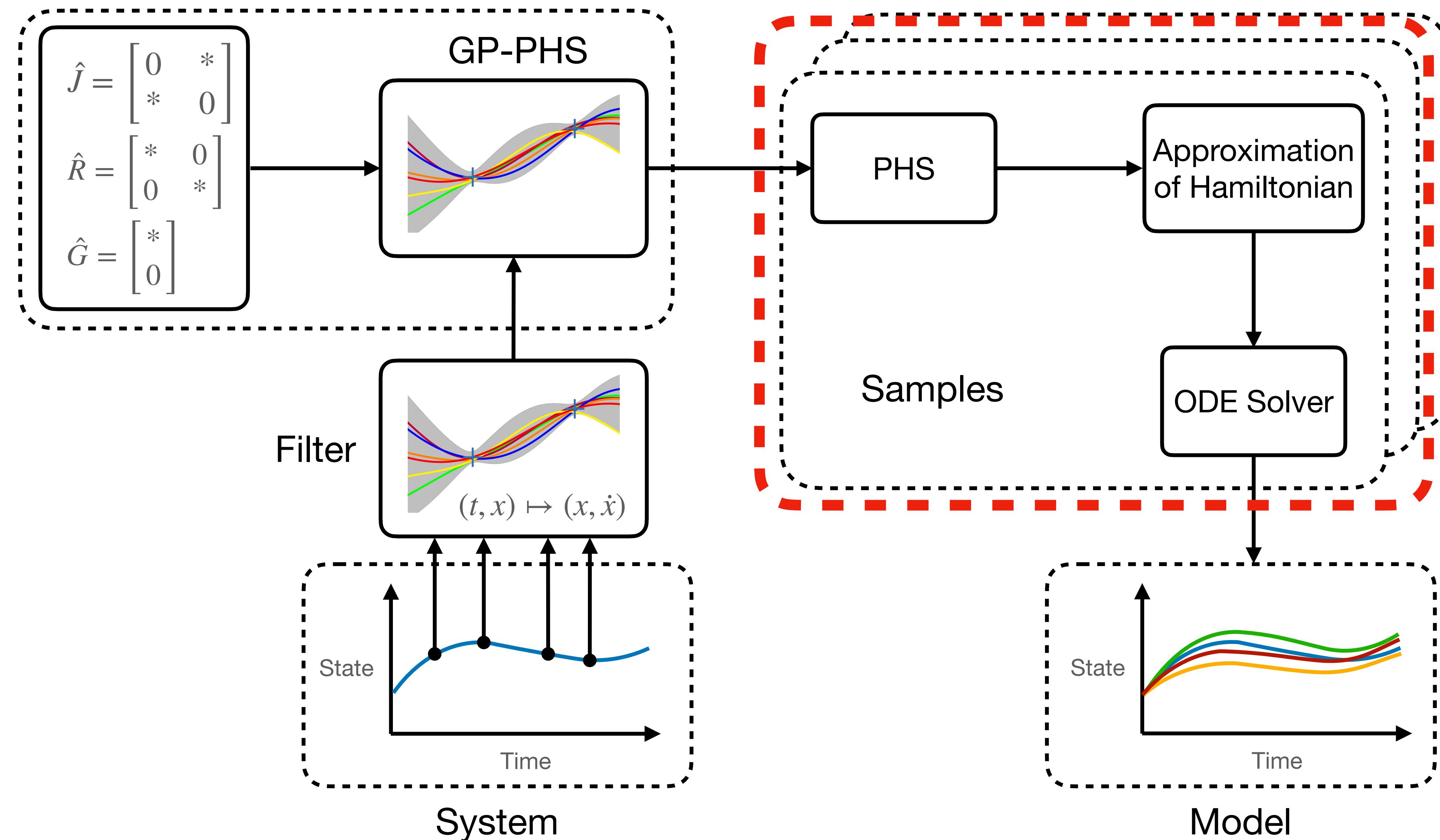
Port-Hamiltonian Kernel

$$k_{phs}(x, x') = \sigma_f^2 [\hat{J}(x) - \hat{R}(x)] \left[\underbrace{\frac{\partial}{\partial x \partial x'} \exp(-\|x - x'\|_\Lambda^2)}_{\text{Squared exp. kernel}} \right] \overbrace{[\hat{J}(x') - \hat{R}(x')]}^{\text{Parametrized estimates}}^\top$$

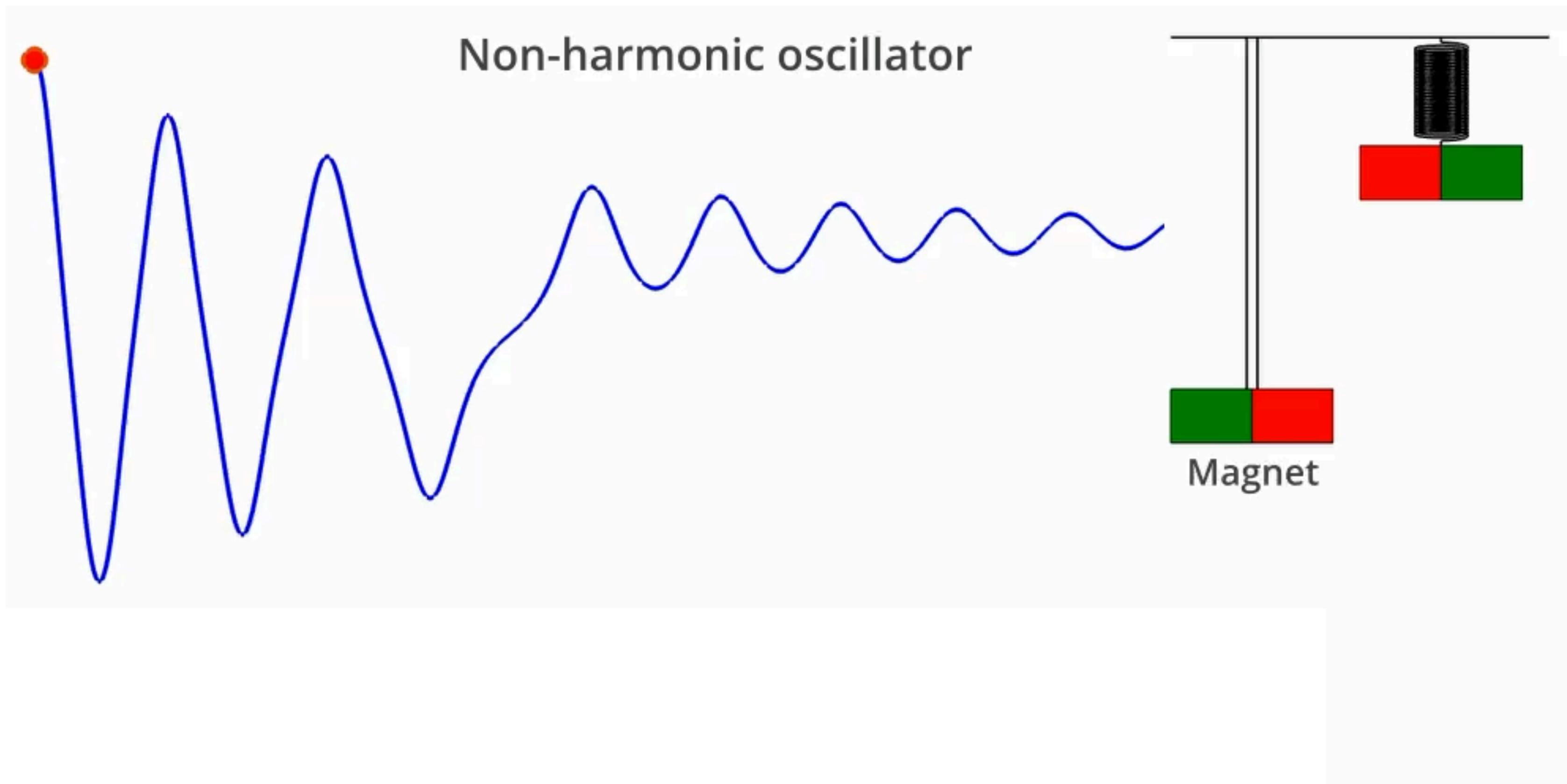
Hyperparameters φ = Kernel parameters + unknown PHS parameters

Minimizing NLML $-\log p(\dot{X} | \varphi, X) \sim \underbrace{\dot{X}^\top [K_{phs} + \Delta]^{-1} \dot{X} + \log |K_{phs} + \Delta|}_{\text{Uncertainty of filter}}$

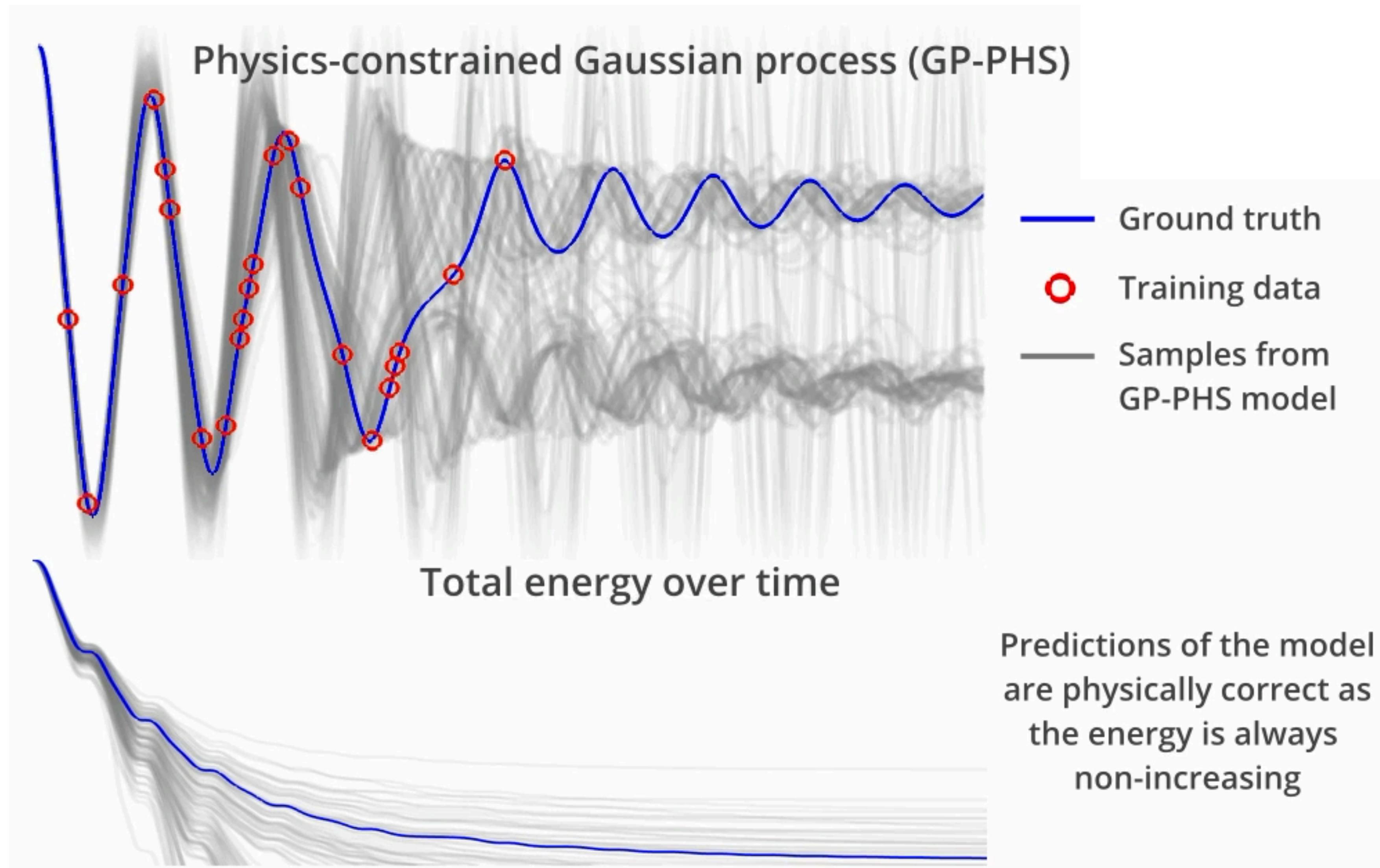
GP-PHS



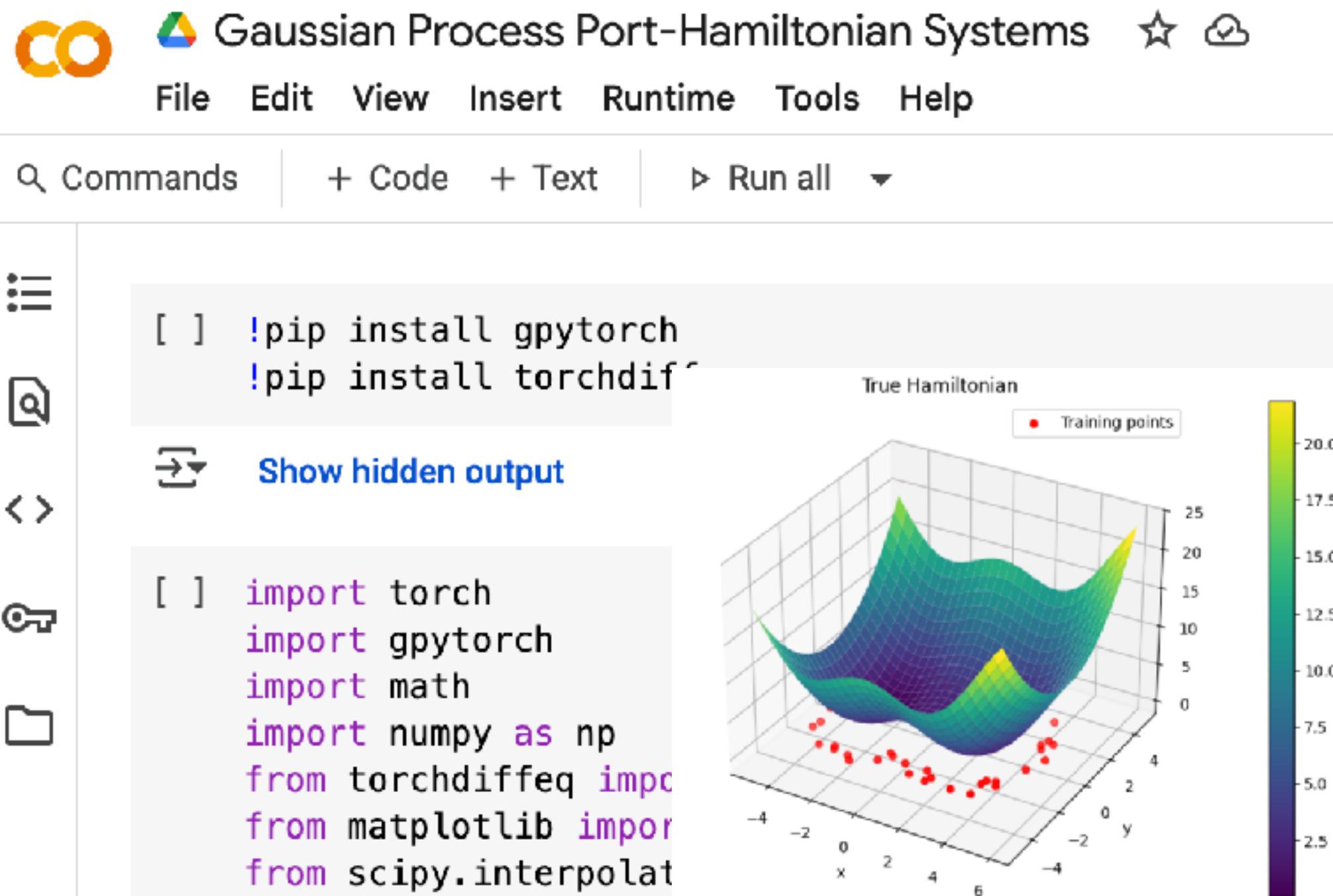
Example



Example



Python Implementation



The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with a logo, file navigation, and a search bar labeled "Commands". Below the toolbar, there are tabs for "Code" and "Text", and a button to "Run all". On the left, there are several icons: a file folder, a key, a magnifying glass, a list, and a refresh symbol. In the main area, there are two code cells. The first cell contains commands to install gpytorch and torchdiffeq. The second cell contains Python code for importing torch, gpytorch, math, numpy, and other libraries, along with some plotting code. To the right of the code, there is a 3D surface plot titled "True Hamiltonian" showing a complex, multi-peaked function. A color bar on the right indicates the value of the function, ranging from approximately -2.5 to 20.0. Red dots on the plot represent "Training points".

Implementation based on Torch and GPyTorch

- Use of Torch features such as GPU learning
- Tools from GPyTorch (speed up computation, etc.)
- PHS-Kernel implemented based on

RBFKernelGradGrad

```
class gpytorch.kernels.RBFKernelGradGrad(ard_num_dims=None, batch_shape=None, active_dims=None, lengthscale_prior=None, lengthscale_constraint=None, eps=1e-06, **kwargs) [source]
```

Computes a covariance matrix of the RBF kernel that models the covariance between the values and first and second (non-mixed) partial derivatives for inputs \mathbf{x}_1 and \mathbf{x}_2 .

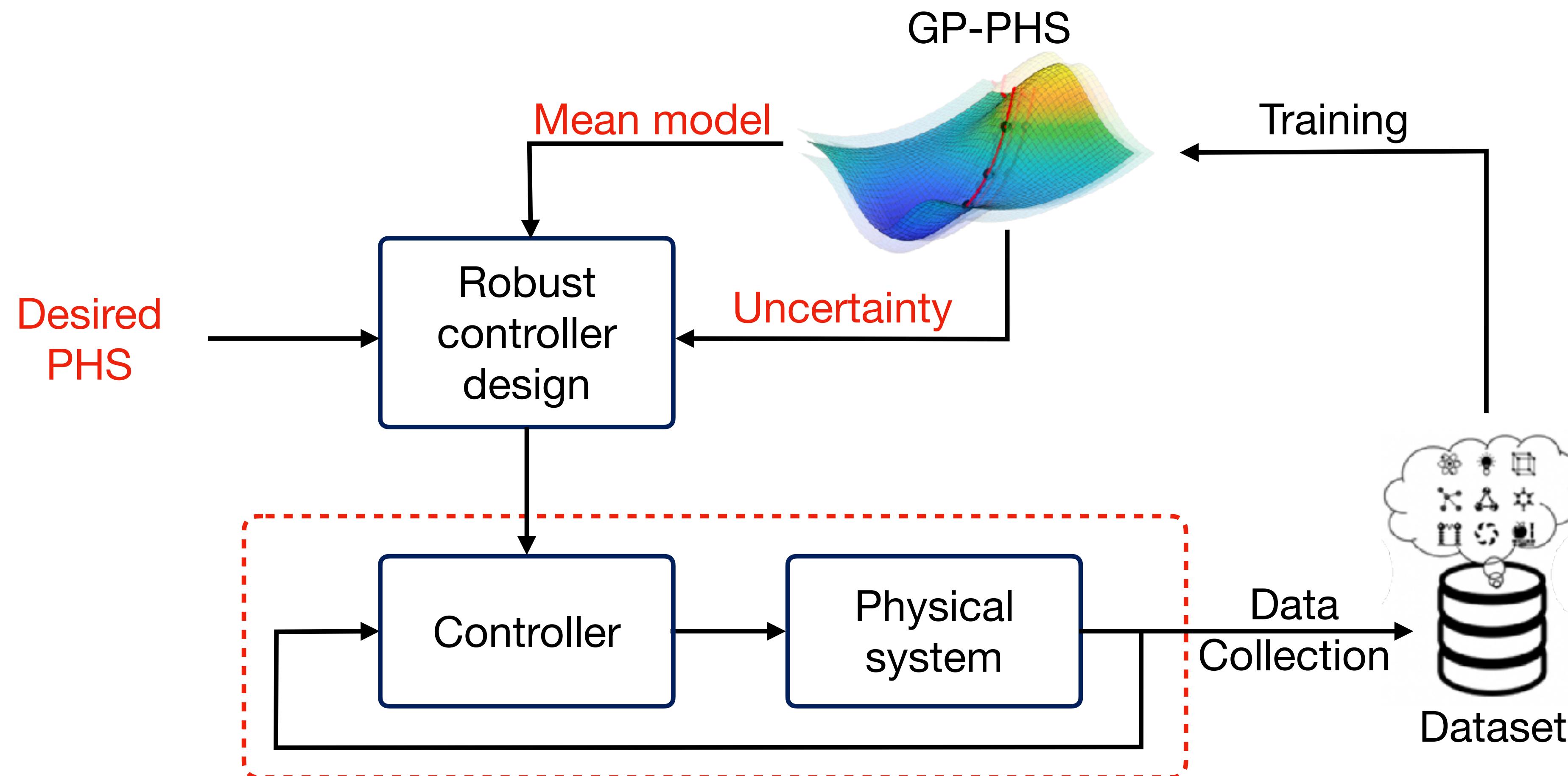
See [gpytorch.kernels.Kernel](#) for descriptions of the lengthscale options.

<https://www.tbeckers.com/research>

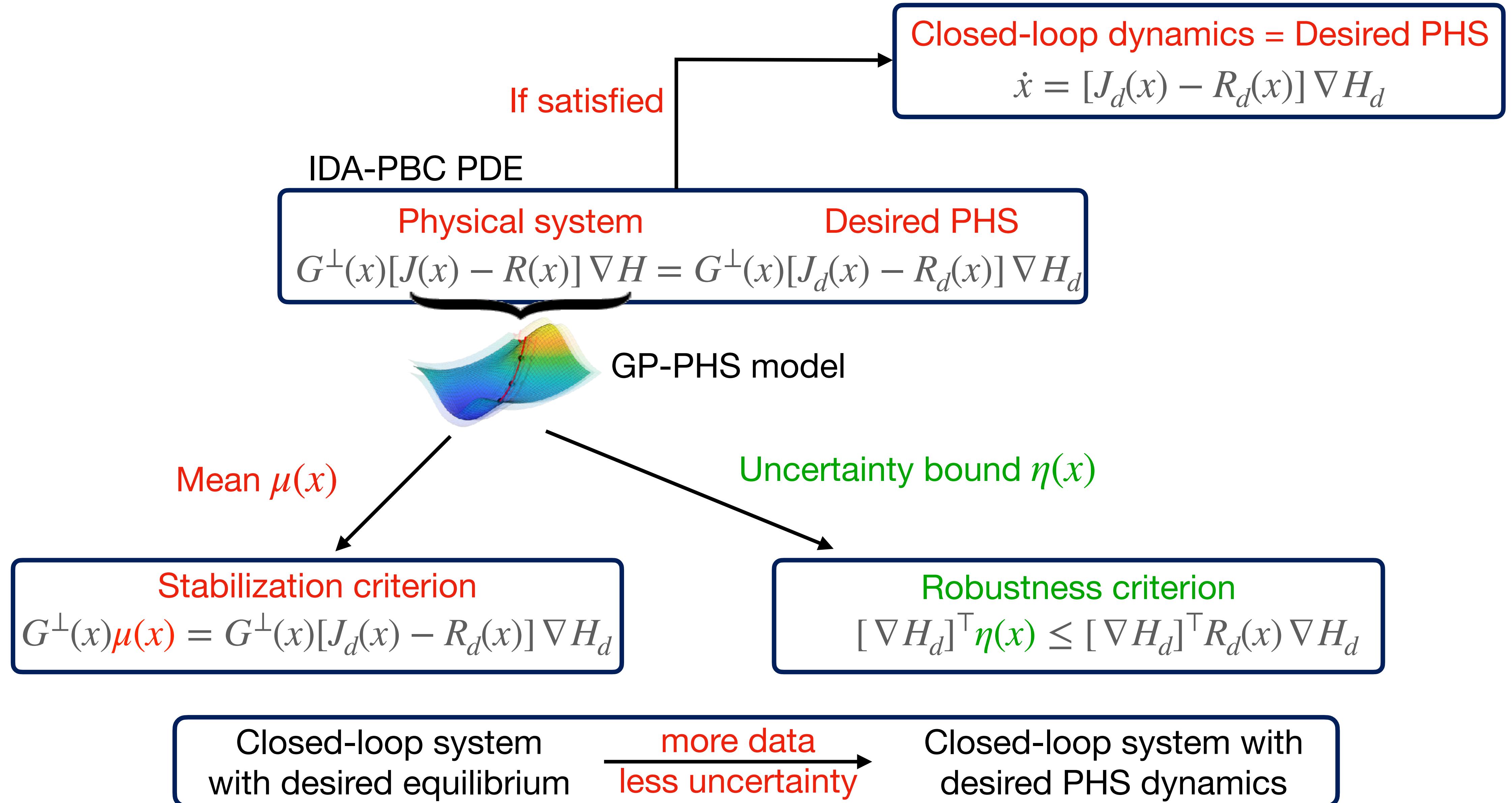
$$\begin{bmatrix} k(\mathbf{x}, \mathbf{x}) & \frac{\partial}{\partial \mathbf{x}} k(\mathbf{x}, \mathbf{X}) \\ \frac{\partial}{\partial \mathbf{x}} k(\mathbf{x}, \mathbf{X})^\top & \frac{\partial}{\partial \mathbf{x}}^\top k(\mathbf{X}, \mathbf{X}) \frac{\partial}{\partial \mathbf{x}} \end{bmatrix}$$

Control

Design of a **stabilizing controller** for the physical system with unknown dynamics



Controller design



GP-PHS for PDEs

Goal: Learning the dynamics of a vibrating string with unknown stress-strain curve

$$\frac{\partial^2 x}{\partial t^2} - f\left(\frac{\partial x}{\partial z}\right) \frac{\partial^2 x}{\partial z^2} = 0,$$



In PHS form

$$\frac{\partial}{\partial t} \begin{bmatrix} p(t, z) \\ q(t, z) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial z} & 0 \end{bmatrix}}_{J-R} \delta_{pq} \mathcal{H}(p, q)$$

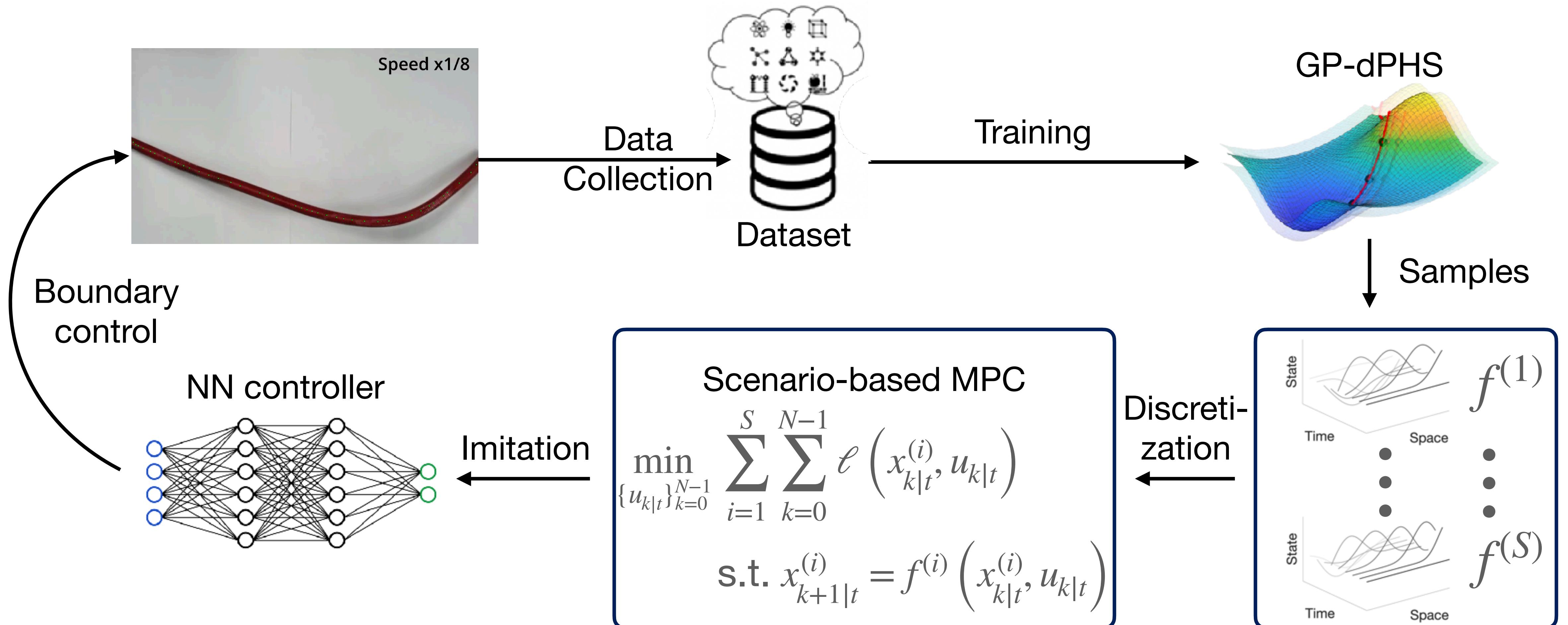
Training data

16 spatial points every 4ms



Control of PDEs

Design of an **MPC controller** for a physical PDE system with unknown dynamics



Conclusion

Structure-preserved learning

- Using **data to “fill” a physics structure such as PHS**
- GP-PHS model is **physically consistent** and enables UQ
- Preserve the **structure and passivity characteristic**
- **Robust control** of physical system based on GP-PHS

