List 4: Reinforcement Learning.
Deep learning control of physics-informed neural networks for fast actuation in uncertain environments

Jostein Barry-Straume

February 8, 2022

Introduction: Motivation

- Scientific Machine Learning (SciML) can replace traditional numerical discretization methods
  1. Mesh-free approach via automatic differentiation
  2. Breaks the curse of dimensionality

- SciML offers potential to improve predictions beyond state-of-art physical models
  1. Smaller number of samples
  2. More generalizable in out-of-sample scenarios

- Physics-Informed Neural Networks (PINNs) solve supervised learning tasks while respecting properties of physical laws
  1. Inverse problems, fractional differential equations, and stochastic differential equations
  2. Recent applications to control problems

Overview of Research Problem

The novel results presented in this work include

- 1-stage Control PINN framework that solves the learning tasks of
  1. a dynamical system state
  2. and its respective optimal control
- Non-reliance on a priori controller data
- Non-reliance on an external controller

Why is this problem important to solve?

- Control PINNs can find optimal control solutions to complex computational scientific problems more efficiently
- Control PINNs can be utilized as agents in deep reinforcement learning (DRL)

Physics-Informed Neural Networks

Algorithm 1: The PINN algorithm for solving differential equations

Result: Minimize loss function $L(\theta; T)$

1. Construct neural network $\hat{u}(x; \theta)$ with parameters θ.
2. Specify two training sets T_f and T_b for the equation and boundary/initial conditions.
3. Specify a loss function by summing the weighted L^2 norm of both the PDE equation and boundary condition residuals.
4. Train the neural network to find the best parameters θ^* by minimizing the loss function $L(\theta; T)$.

Prior Approaches

- Hwang *et* al. propose a two stage framework for solving PDE-constrained control problems using operator learning.
  - Strength: the ability to apply their framework to both data-driven and data-free cases.
  - Weakness: the two stage nature of the framework, as the control is found only after a surrogate model has been trained.
- Chen *et* al. train an input convex recurrent neural network. Subsequently, they then solve a convex model predictive control (MPC) problem.
  - Strength: the guarantee of an optimal solution, thanks to the convex nature of the trained model.
  - Weakness: two stage framework of system identification and controller design.

Prior Approaches Continued

- Antonelo *et* al. introduce a new framework called Physics-Informed Neural Nets for Control (PINC). PINC uses data from the control action u, and initial state $y(0)$, to solve an optimal control problem.
  - Strength: the ability to "run for an indefinite time horizon [...] without significant deterioration of network prediction"
  - Weakness: PINC is essentially a PINN that is amenable to being trained on data of the controller data, instead of learning the optimal control itself.
- Mowlavi and Nabi conduct an evaluation of the comparative performance between traditional PINNs and classic direct-adjoint-looping (DAL) to solve optimal control problems
  - Strength: Good comparison paper to benchmark PINNs with classic methods
  - Weaknesses: Steady state Navier Stokes, manual derivation in DAL, dampened control over time

Addressing Limitations of Prior Work

- Control PINN goes beyond the prior approaches because it does not rely on data from the controller
- Not a comparison paper, but a new taxonomical form of PINNs
- The main contribution of Control PINNs is that an unknown controller u can be solved at the same time as the complex dynamical state of y.

Evaluation of New Approach

1. Architect will be designed and then trained as a neural network while monitoring the loss function

2. Comparison of model results and reference solutions will be carried out for a problem with a known analytical solution

3. For problems with no known solution, the model results will be compared to simulation results using numerical analysis techniques

Intuition behind Approach

Combining both DRL and PINNs, when both have successful track records solving scientific computing problems, is an intuitive and interesting research avenue.

Q-learning can learn a policy based on the Control PINN. The quality of the both the controller, and state of the system, provided by the Control PINN, can be systematized such that the best quality of action-state can be carried out.

Before leveraging a Control PINN as an agent in a DRL framework, it first must be demonstrated that it can solve optimal/open loop control problems.

Open Loop $\longrightarrow$ Closed Loop $\longrightarrow$ Agent in DRL

Methodology

We denote by $y(t)$ the solution of the ODE, and by $u(t)$ the control function. We seek to solve the control problem:

$$\begin{aligned} &\min_{u(t)} \Psi(y) = \int_0^{t_f} g\big(y(t), u(t)\big)\, dt + w\big(y(t_f)\big) \\ &\text{subject to} \quad y' = f(y,u), \ \forall t \in [0, t_f], \quad y(0) = y_0. \end{aligned} \tag{1}$$

The optimality conditions are:

$$y'(t) = f\big(y(t), u(t)\big), \ \forall t \in [0, t_f]; \quad y(0) = y_0^*; \tag{2a}$$

$$\lambda'(t) = -f_y^T\big(y(t), u(t)\big)\, \lambda(t) - g_y^T\big(y(t), u(t)\big), \ \forall t \in [t_f, 0]; \tag{2b}$$

$$\lambda(t_f) = w_y^T\big(y(t_f)\big);$$

$$0 = -f_u\big(y(t), u(t)\big)\, \lambda(t) - g_u^T\big(y(t), u(t)\big), \ \forall t \in [0, t_f]. \tag{2c}$$

Methodology Continued

Algorithm 2: The procedure to train a Control PINN model

Result: Training of a Control PINN that learns the optimal solution and the optimal control function for the given problem in (1)

1. Construct a network with inputs t, x (time and space), and outputs y, u, and λ (system state, system control, and system push back on control
2. Via auto differentiation and back-propagation, compute the following derivatives of the output w.r.t the input: $\frac{\delta y}{\delta x}$, $\frac{\delta y}{\delta \delta x}$, $\frac{\delta f}{\delta y}$, $\frac{\delta y}{\delta t}$, $\frac{\delta \lambda}{\delta t}$, $\frac{\delta f}{\delta u}$, $\frac{\delta g}{\delta y}$, $\frac{\delta g}{\delta u}$
3. With snapshots of the exact solution denoted by $y^*(t)$, minimize the loss function:

$$
\begin{aligned}
L = & \sum_i (t_{i+1} - t_i) \left\| y(t_i) - y^*(t_i) \right\|^2 \\
& + \sum_i (t_{i+1} - t_i) \left\| D_t y(t_i) - f\big(y(t_i), u(t_i)\big) \right\|^2 \\
& + \sum_i (t_{i+1} - t_i) \left\| D_t \lambda(t_i) + f_y^T\big(y(t_i), u(t_i)\big)\, \lambda(t_i) + g_y^T\big(y(t_i), u(t_i)\big) \right\|^2 \\
& + \sum_i (t_{i+1} - t_i) \left\| f_u\big(y(t_i), u(t_i)\big)\, \lambda(t_i) + g_u^T\big(y(t_i), u(t_i)\big) \right\|^2 .
\end{aligned}
\tag{3}
$$

Methodology Continued

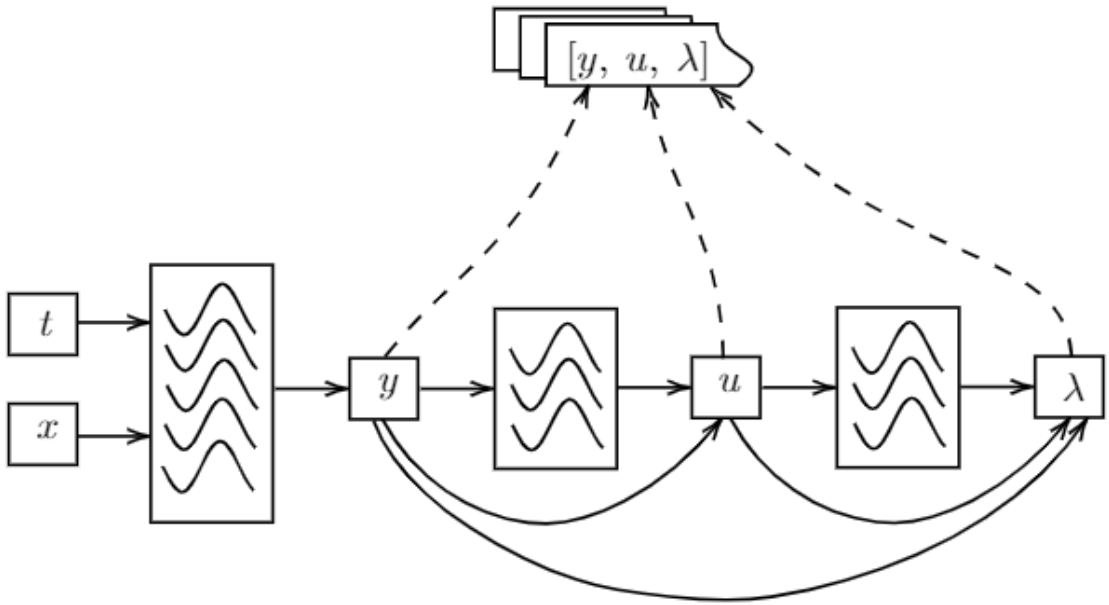


Figure 2: Visual Representation of Control PINN Architecture

Analytical Toy Problem

As a proof of concept, and to provide a foundation for methodology validation, let us consider the below test problem from:

$$t_f = 1, \quad w\big(y(t_f)\big) = 0, \quad y(0) = y_0 = 1 \tag{3a}$$

$$f\big(y(t), u(t)\big) = \frac{1}{2} y(t) + u(t) \tag{3b}$$

$$f_y\big(y(t), u(t)\big) = \frac{1}{2}, \quad f_u\big(y(t), u(t)\big) = 1 \tag{3c}$$

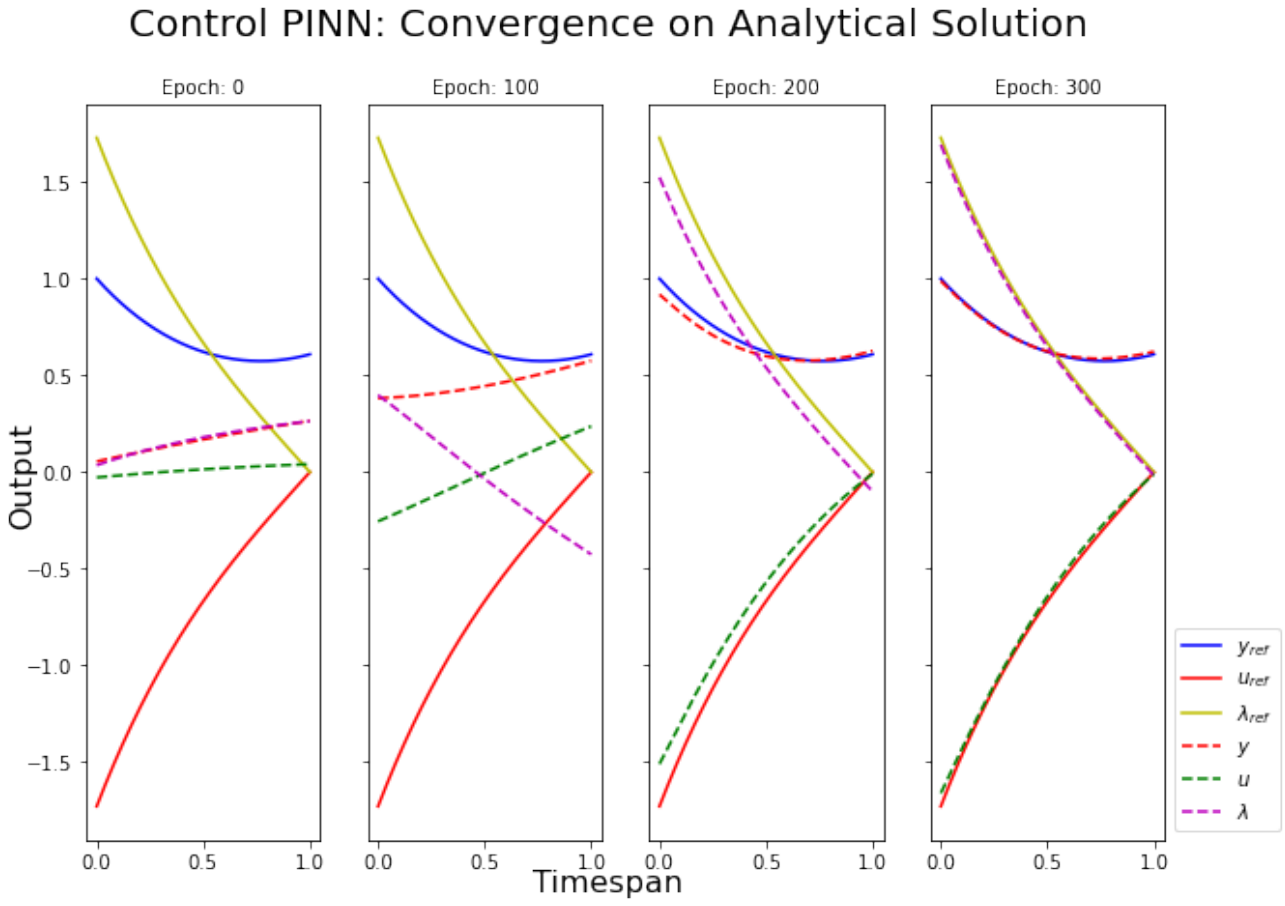
$$g\big(y(t), u(t)\big) = y^2(t) + \frac{1}{2}\, u^2(t) \tag{3d}$$

$$g_y\big(y(t), u(t)\big) = 2\, y(t), \quad g_u\big(y(t), u(t)\big) = u(t) \tag{3e}$$

$$\lambda'(t) = -\frac{1}{2}\, \lambda(t) - 2y(t), \ \forall t \in [0, 1]; \quad \lambda(t_f) = 0 \tag{3f}$$

$$0 = -\lambda(t) - u(t), \ \forall t \in [0, 1] \tag{3g}$$

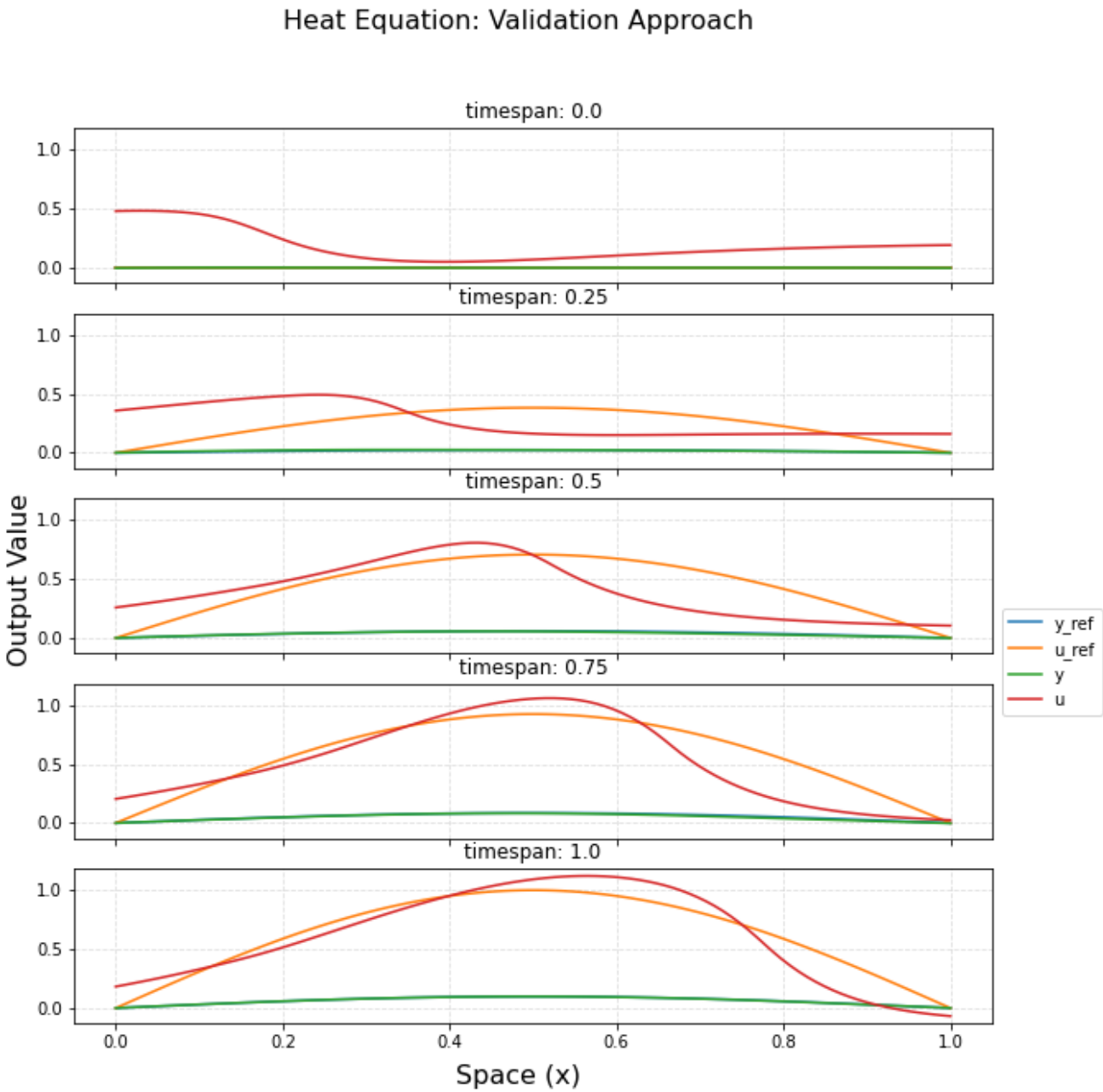
Validation of Toy Problem



The optimal solution is:

$$y_{ref}(t) = \frac{2e^{3t}+e^3}{e^{3t/2}(2+e^3)}, \quad u_{ref}(t) = \frac{2(e^{3t}-e^3)}{e^{3t/2}(2+e^3)}, \quad \lambda_{ref}(t) = -u_{ref}(t).$$

Future Steps: Preliminary Results



Future Steps: Preliminary Results Continued