Neural Operators for Hyperbolic PDE Backstepping Feedback Laws

Luke Bhan, Yuanyuan Shi, and Miroslav Krstic

***Abstract*— We introduce a framework for accelerating the computation of a backstepping controller in PDE control. We learn the nonlinear operator from the plant parameter and PDE solution to the boundary control with a (deep) neural network. We provide closed-loop stability guarantees (semiglobal exponential) under an NN-approximation of the feedback law. While, in the existing PDE backstepping, finding a feedback law requires the solution to multiple integral equations and operations for both the gain and control input value, the neural operator (NO) approach we propose learns the mapping from the functional coefficients of the plant PDE and PDE system state to the boundary control value by employing a sufficiently high number of offline numerical solutions to the analytical feedback control law. We prove the existence of a DeepONet approximation with arbitrarily high accuracy, of the exact nonlinear continuous operator mapping between the PDE coefficient functions and PDE system state into a control feedback law. Once proven to exist, learning of the NO is standard, completed "once and for all" (never online) and the control feedback equation doesn't need to be solved ever again, for both any new functional coefficient and PDE system state that does not exceed the magnitude of the coefficients and states used in training. Simulation illustrations are provided and the code is available on github.**

I. Introduction

ML/AI has been a disruptive force in a wide class of engineering disciplines leading to questions about whether ML/AI will "takeover" model-based sciences such as physics or conventional control theory. Recently, a new framework has emerged [21]–[24] which promises to unite the goals of physics and learning, rather than presenting learning as an alternative or substitute to first-principles physics. This framework, termed neural operators (NO), is formulated as learning mappings from function spaces into function spaces and has achieved success in PDEs with learnable solution/"flow" maps after enough simulations with different initial conditions.

a) Mappings of plant parameters to control gains and learning of those maps: It is worth asking what neural operators can contribute to control theory, namely to the design of controllers, observers, and online parameter estimators. In this work, we explore a first venture in this direction with the capability for future extensions and layout a blueprint to *learn* PDE control design and prove their stability.

Although learning nonlinear maps for various design problems for nonlinear ODEs is worth studying, we focus this initial work one stop beyond, on a benchmark PDE control class. Particularly, we focus on an uncomplicated - but unstable - PDE control class. Our choice of basic PDE control is for pedagogical reasons - combining the operator learning with *PDE backstepping* is complex for even the simplest-looking PDE stabilization problems.

b) PDE backstepping control with the gain computation obviated using neural operators: Consider 1D hyperbolic partial integro-differential equation systems of the general form $v_t(x,t) = v_x(x,t) + \lambda(x)v(x,t) + g(x)v(0,t) + \int_0^x f(x,y)v(y,t)dy$ on the unit interval $x \in [0,1]$, which are transformable, using an invertible backstepping "pre-transformation" introduced in [3] into the simple PDE

$$u_t(x,t) = u_x(x,t) + \beta(x)u(0,t) \tag{1}$$

$$u(1,t) = U(t). \tag{2}$$

Our goal is the design of a PDE backstepping boundary control

$$U(t) = \int_0^1 k(1-y)u(y,t)dy. \tag{3}$$

Physically, (1) is a "transport process (from $x = 1$ towards $x = 0$) with recirculation" of the outlet variable $u(0,t)$. Recirculation causes instability when the coefficient $\beta(x)$ is positive and large. This instability is prevented by the backstepping boundary feedback (3) with the gain function $k(\cdot)$ as a kernel in the spatial integration of the measured state $u(y,t)$. (The full state does not need to be measured, as explained in Remark 1 at the end of Section V.)

Backstepping produces the control law U for a given β, u. We learn the nonlinear continuous mapping $\mathcal{U} : (\beta, u) \mapsto U$ and once $\mathcal{U}$ is learned, the partial differential or integral equation does *not* need to be recomputed for a new β or u. Instead, for a new β or u, finding U is simply a "function evaluation" of the learned mapping $\mathcal{U}$. This benefits traditional control as the integral for u, k does not need to be recomputed at each step. Furthermore, this benefits adaptive control where, at each timestep, both u and the gain estimate $\hat{k}$ needs to be calculated for a new parameter update $\hat{\beta}$ and in gain scheduling for nonlinear PDEs where the gain must be recomputed at each current state.

Traditionally, when ML is applied in the control context (such as RL or other approaches) it is usually a model-free design. However, our approach, summarized in Figure 1 is not model free. It is model-based and only the computation portion of the model-based (PDE backstepping) design is accelerated through ML. Additionally, we emphasize our learning is offline; not as in adaptive control. [1], [3], [10].

Naturally, one can just learn the mapping $\mathcal{U}$ and stop. However, in this work, we extend our analysis to investigate whether the NN approximation of the feedback law U will

This work was supported by NSF Grants ECCS-2151525 and ECCS-2210315 as well as AFOSR FA9550-22-1-0265

The authors are with the University of California, San Diego, USA, lbhan@ucsd.edu, yyshi@eng.ucsd.edu, krstic@ucsd.edu

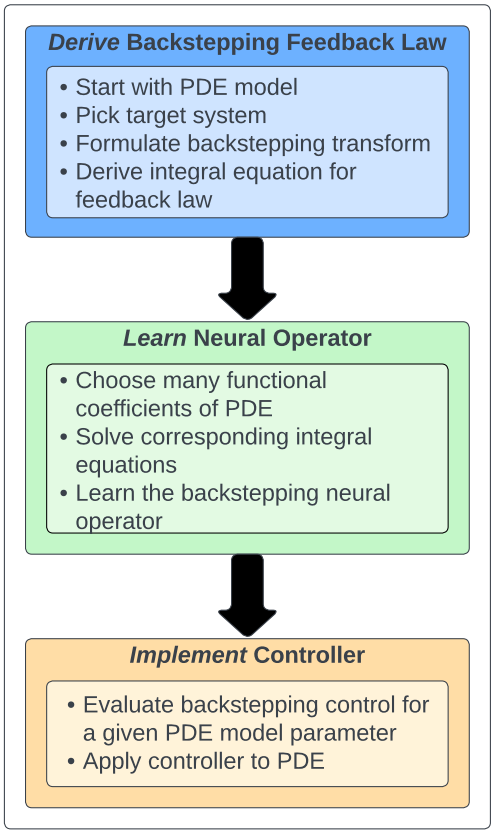

Fig. 1. An algorithmic representation of our design paradigm of employing neural operators in boundary control of PDEs. Three major step clusters are performed: (1) derivation of the integral equations for the backstepping feedback law, performed only once; (2) learning of the mapping from the plant parameter functions and system state into the backstepping control law, also performed only once; and (3) implementation of the controller for specific plant parameters and system states. The task in the top box has been completed in [17]. In this paper, the task in the middle box is introduced and stability guarantees for the task in the bottom box are provided.

result in a stable PDE. We find that with a large enough data set of solved pairs $((\beta_i, u_i), U_i)$, and a large enough trained (deep) NN, closed-loop stability is guaranteed for a new pair (β, u), not in the training set.

c) Neural operator literature—a brief summary: Neural operators are NN-parameterized maps for learning relationships between function spaces. They originally gained popularity due to their success in mapping PDE solutions while remaining discretization-invariant. Generally, nonlinear operators consist of three components: an encoder, an approximator, and a reconstructor [20]. The encoder is an interpolation from an infinite-dimensional function space to a finite-dimensional vector representation. The approximator aims to mimic the infinite map using a finite-dimensional representation of both the domain function space and the target function space. The reconstructor then transforms the approximation output into the infinite-dimensional target function space. The implementation of both the approximator and the reconstructor is generally coupled NNs, but can take many forms as well. More details can be found in [15], [22], [23], [27].

d) Advances in learning-based control: Among the first in demonstrating the stability of learning-based model predictive controllers (MPC) were the papers [2], [26]. This was extended, for nonlinear systems, to deep learning-based approaches consisting of jointly learning the controller and(or) Lyapunov functions via NNs. See [7] for a recent review. In addition, [25], [31] have explored how learning-based control will affect nominal systems with known Lyapunov functions, and [6], [8] studied the problem of learning stability certificates and stable controllers directly from data.

In a separate, but related direction, many reinforcement learning (RL) control approaches have been developed over the past few years. On one side, model-based RL has been studied due to its superior sample efficiency and interpretable guarantees. The main focus has been on learning the system dynamics and providing closed-loop guarantees in *finite-time* for both linear systems [9], [18] (and references within), and nonlinear systems [14], [19]. For model-free RL methods, [12] first proved the convergence of policy optimization, a popular model-free RL method, to the optimal controller for linear time-invariant systems. Since then, convergence results of policy optimization methods have been shown for LQR, H_∞ control, risk-sensitive control, LQG, and output feedback synthesis. See [13] for a recent review.

Our work focus on learning-based control for PDE systems. In our previous work [28], we demonstrate the empirical success of using NOs for accelerating PDE backstepping observers, without theoretical guarantees. The present paper, along with its companion [5] (both submitted to CDC invited sessions), constitutes a larger submission to TAC [4]. This line of works represents the first step towards using neural operators for provably bypassing both the controller gain computations (the companion paper [5] - with exponential stability guarantees) or directly learning the controller (the present paper - with practical stability) in PDE backstepping control.

e) Paper outline and contributions: After a brief introduction to the backstepping design in Section II, for system (1), (2), in Section IV we prove that the backstepping control operator $\mathcal{U}$ is locally Lipschitz, between the spaces of $C^0([0,1]^2)$ into $\mathbb{R}$. In Section V we present our main results: the guarantee of semiglobal practical exponential stability under such a DeepONet approximation. In Section VI we illustrate this feedback law approximation with a theory-confirming simulation.

In summary, the paper's contributions are the PDE stabilization under DeepONet approximations of backstepping control laws (Theorem 2) and practical simulations of a DeepONet approximator for a hyperbolic PDE (Figure 3). Our stabilization results also hold for any other neural operators with a universal approximation property (shown for LOCA [15] and for FNO on the periodic domain [16]).

f) Notation: We denote convolution operations as

$$(a * b)(x) = \int_0^x a(x-y)b(y)dy \tag{4}$$

In the sequel, we suppresses the arguments x and t wherever clear from the context. For instance, we write (1), (2) compactly as $u_t = u_x + \beta u(0)$ and $u(1) = U$, where, from the context, the boundary values $u(0), u(1)$ depend on t as well.

II. Backstepping Design for a Transport PDE with 'Recirculation'

Consider the PDE system (1), (2). We employ the following backstepping transformation:

$$w = u - k * u, \tag{5}$$

i.e., $w(x,t) = u(x,t) - \int_0^x k(x-y)u(y,t)dy$, to convert the plant into the target system

$$\begin{aligned} w_t &= w_x & (6)\\ w(1) &= 0 & (7)\end{aligned}$$

with the help of feedback

$$U(t) = (k * u)(1) = \int_0^1 k(1-y)u(y,t)dy \tag{8}$$

To yield the target system, k must satisfy the integral/convolution equation

$$k(x) = -\beta(x) + \int_0^x \beta(x-y)k(y)dy \tag{9}$$

for $x \in [0,1]$. Note that, the feedback law is clearly nonlinear requiring both the solution for integral equation k and a convolution between k and u.

III. Accuracy of Approximation of Backstepping Kernel Operator with DeepONet

An n-layer NN $f^{\mathcal{N}} : \mathbb{R}^{d_1} \to \mathbb{R}^{d_n}$ is given by

$$f^{\mathcal{N}}(x,\theta) := (l_n \circ l_{n-1} \circ ... \circ l_2 \circ l_1)(x,\theta) \tag{10}$$

where layers l_i start with $l_0 = x \in \mathbb{R}^{d_1}$ and continue as

$$l_{i+1}(l_i, \theta_{i+1}) := \sigma(W_{i+1}l_i + b_{i+1}), \quad i = 1, \ldots, n-1 \tag{11}$$

σ is a nonlinear activation function, and weights $W_{i+1} \in \mathbb{R}^{d_{i+1} \times d_i}$ and biases $b_{i+1} \in \mathbb{R}^{d_{i+1}}$ are parameters to be learned, collected into $\theta_i \in \mathbb{R}^{d_{i+1}(d_i+1)}$, and then into $\theta = [\theta_1^{\mathrm{T}}, \ldots, \theta_n^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{\sum_{i=1}^{n-1} d_{i+1}(d_i+1)}$. Let $\vartheta^{(k)}, \theta^{(k)} \in \mathbb{R}^{\sum_{i=1}^{k-1} d_{k,(i+1)}(d_{k,i}+1)}$ denote a sequence of NN weights.

An neural operator (NO) for approximating a nonlinear operator $\mathcal{G} : \mathcal{U} \mapsto \mathcal{V}$ is defined as

$$\mathcal{G}_{\mathbb{N}}(\mathbf{u}_m)(y) = \sum_{k=1}^{p} g^{\mathcal{N}}(\mathbf{u}_m; \vartheta^{(k)}) f^{\mathcal{N}}(y; \theta^{(k)}) \tag{12}$$

where $\mathcal{U}, \mathcal{V}$ are function spaces of continuous functions $u \in \mathcal{U}, v \in \mathcal{V}$. $\mathbf{u}_m$ is the evaluation of function u at points $x_i = x_1, ..., x_m$, p is the number of chosen basis components in the target space, $y \in Y$ is the location of the output function $v(y)$ evaluations, and $g^{\mathcal{N}}$, $f^{\mathcal{N}}$ are NNs termed branch and trunk networks. Note, $g^{\mathcal{N}}$ and $f^{\mathcal{N}}$ are not limited to feedforward NNs 10, but can also be of convolutional or recurrent.

Theorem 1: (DeepONet universal approximation theorem [11, Theorem 2.1]). Let $X \subset \mathbb{R}^{d_x}$ and $Y \subset \mathbb{R}^{d_y}$ be compact sets of vectors $x \in X$ and $y \in Y$, respectively. Let $\mathcal{U} : X \to U \subset \mathbb{R}^{d_u}$ and $\mathcal{V} : Y \to V \subset \mathbb{R}^{d_v}$ be sets of continuous functions $u(x)$ and $v(y)$, respectively. Let $\mathcal{U}$ be also compact. Assume the operator $\mathcal{G} : \mathcal{U} \to \mathcal{V}$ is continuous. Then, for all $\epsilon > 0$, there exist $m^*, p^* \in \mathbb{N}$ such that for each $m \geq m^*$, $p \geq p^*$, there exist $\theta^{(k)}, \vartheta^{(k)}$, neural networks $f^{\mathcal{N}}(\cdot;\theta^{(k)}), g^{\mathcal{N}}(\cdot;\vartheta^{(k)}), k = 1, \ldots, p$, and $x_j \in X, j = 1, \ldots, m$, with corresponding $\mathbf{u}_m = (u(x_1), u(x_2), \cdots, u(x_m))^{\mathrm{T}}$, such that

$$|\mathcal{G}(u)(y) - \mathcal{G}_{\mathbb{N}}(\mathbf{u}_m)(y)| < \epsilon \tag{13}$$

for all functions $u \in \mathcal{U}$ and all values $y \in Y$ of $\mathcal{G}(u) \in \mathcal{V}$.

IV. Approximating the Full Feedback Law Map $(\beta, u) \mapsto U$

We start by establishing the Lipschitzness of the backstepping feedback map.

Lemma 1: Consider the feedback (8), namely,

$$U = (\mathcal{K}(\beta) * u)(1), \tag{14}$$

and the associated map $\mathcal{U} : (\beta, u) \mapsto U$ from $C^0([0,1]^2)$ into $\mathbb{R}$. For arbitrary $B_\beta, B_u > 0$, the mapping $\mathcal{U}$ is Lipschitz on any set of x-dependent Lipschitz functions (β, u) such that $\|\beta\|_\infty \leq B_\beta, \|u\|_\infty \leq B_u$, with a Lipschitz constant

$$C_{\mathcal{U}} = B_\beta \mathrm{e}^{B_\beta} + B_u \mathrm{e}^{3B_\beta}. \tag{15}$$

Proof: Let $U_1 = \mathcal{U}(\beta_1, u_1) = (\mathcal{K}(\beta_1) * u_1)(1)$ and $U_2 = \mathcal{U}(\beta_2, u_2) = (\mathcal{K}(\beta_2) * u_2)(1)$. A calculation gives

$$\begin{aligned} &|U_1 - U_2| = |(\mathcal{K}(\beta_1) * u_1)(1) - (\mathcal{K}(\beta_2) * u_2)(1)| \\ &\leq \|\mathcal{K}(\beta_1)\|_\infty \|u_1 - u_2\|_\infty + \|u_2\|_\infty \|\mathcal{K}(\beta_1) - \mathcal{K}(\beta_2)\|_\infty. \end{aligned} \tag{16}$$

Let $\|\beta_1\|_\infty, \|\beta_2\|_\infty \leq B_\beta$ and $\|u_1\|_\infty, \|u_2\|_\infty \leq B_u$. Recall that $\|\mathcal{K}(\beta)\|_\infty \leq B_\beta \mathrm{e}^{B_\beta}$ and $\|\mathcal{K}(\beta_1) - \mathcal{K}(\beta_2)\|_\infty \leq \mathrm{e}^{3B_\beta} \|\beta_1 - \beta_2\|_\infty$. Then we get

$$\begin{aligned} &|\mathcal{U}(\beta_1, u_1) - \mathcal{U}(\beta_2, u_2)| \\ &\leq \left(B_\beta \mathrm{e}^{B_\beta} + B_u \mathrm{e}^{3B_\beta}\right) \|(\beta_1 - \beta_2, u_1 - u_2)\|_\infty. \end{aligned} \tag{17}$$

■

Taking the backstepping transformation $w = u - k * u$, where $k = \mathcal{K}(\beta)$ is the exact backstepping kernel for β, we get

$$\begin{aligned} w_t &= w_x & (18)\\ w(1) &= U - (\mathcal{K}(\beta) * u)(1) & (19)\end{aligned}$$

Let now $\hat{\mathcal{U}}$ be the NO version of the mapping $\mathcal{U}(\beta, u) = (\mathcal{K}(\beta) * u)(1)$. Taking the NO control $U = \hat{\mathcal{U}}(\beta, u)$, we obtain the boundary condition $w(1) = \hat{\mathcal{U}}(\beta, u) - (\mathcal{K}(\beta) * u)(1)$, namely, the target system

$$\begin{aligned} w_t &= w_x & (20)\\ w(1) &= \hat{\mathcal{U}}(\beta, u) - \mathcal{U}(\beta, u) & (21)\end{aligned}$$

Due to the Lipschitzness of $\mathcal{U}$, based on the DeepONet approximation accuracy theorem, we get the following.

Lemma 2: For all $B_\beta, B_u > 0$ and ϵ, there exists an NO $\hat{\mathcal{U}}$ such that

$$|\mathcal{U}(\beta,u) - \hat{\mathcal{U}}(\beta,u)| < \epsilon \tag{22}$$

for all $\beta, u \in C^0[0,1]$ that are Lipschitz in x and such that $\|\beta\|_\infty \leq B_\beta, \|u\|_\infty \leq B_u$.

V. Stability under feedback law approximation with DeepONet

Theorem 2: (Semiglobal practical stability under DeepONet approximation of backstepping feedback law). If $\epsilon < \epsilon^*$, where

$$\epsilon^*(B_\beta, B_u, c) := \frac{\sqrt{c} B_u}{\mathrm{e}^{c/2}\left(1+B_\beta\right)} > 0, \tag{23}$$

and $\|u(0)\| \leq B_u^0$, where

$$B_u^0(\epsilon, B_\beta, B_u, c) := \frac{1}{1+B_\beta \mathrm{e}^{B_\beta}} \left(\frac{B_u}{\mathrm{e}^{c/2}\left(1+B_\beta\right)} - \frac{\epsilon}{\sqrt{c}} \right) > 0, \tag{24}$$

the closed-loop solutions under the NO approximation of the PDE backstepping feedback law, i.e.,

$$\begin{aligned} u_t(x,t) &= u_x(x,t) + \beta(x) u(0,t) && (25)\\ u(1,t) &= \hat{\mathcal{U}}(\beta,u)(t) && (26) \end{aligned}$$

satisfy the *semiglobal practical exponential stability* estimate

$$\begin{aligned} \|u(t)\| \leq\ & \left(1+B_\beta\right)\left(1+B_\beta \mathrm{e}^{B_\beta}\right) \mathrm{e}^{c/2} \mathrm{e}^{-ct/2} \|u(0)\| \\ & + \left(1+B_\beta\right) \frac{\mathrm{e}^{c/2}}{\sqrt{c}} \epsilon, \qquad \forall t \geq 0. \end{aligned} \tag{27}$$

The estimate (27) is semiglobal because the radius B_u^0 of the ball of initial conditions in $L^2[0,1]$ is made arbitrarily large by increasing B_u, and by increasing, in accordance with the increase of B_u, the training set size and the number of NN nodes. Nevertheless, though semiglobal, the attraction radius B_u^0 in (24) is much smaller than the magnitude B_u of the samples of u in the training set.

The residual value,

$$\limsup_{t\to} \|u(t)\| \leq \left(1+B_\beta\right) \frac{\mathrm{e}^{c/2}}{\sqrt{c}} \epsilon \tag{28}$$

is made arbitrarily small by decreasing ϵ, and by increasing, in accordance with the decrease of ϵ, the training set size and the number of NN nodes. As the magnitude B_β of the (potentially destabilizing) gain samples β used for training grows, the residual error grows.

Proof: (of Theorem 2) To make the notation concise, denote $\tilde{\mathcal{U}} = \mathcal{U} - \hat{\mathcal{U}}$ and note that this mapping satisfies $|\tilde{\mathcal{U}}(\beta,u)| = |w(1)| \leq \epsilon$ for all $\|\beta\|_\infty \leq B_\beta, \|u\|_\infty \leq B_u$. Note also that $\tilde{\mathcal{U}}$ depends on ϵ, B_β, B_u through the number of training data and NO size. Consider now the Lyapunov functional $V(t) = \int_0^1 \mathrm{e}^{cx} w^2(x,t) dx$. Its derivative is

$$\begin{aligned} \dot{V} &= \mathrm{e}^c w^2(1) - w^2(0) - c\int_0^1 \mathrm{e}^{cx} w^2(x,t) dx \\ &\leq -cV + \mathrm{e}^c w^2(1) \end{aligned} \tag{29}$$

which yields

$$\begin{aligned} V(t) &\leq V(0)\mathrm{e}^{-ct} + \frac{\mathrm{e}^c}{c} \sup_{0\leq\tau\leq t} w^2(1,\tau) \\ &\leq V(0)\mathrm{e}^{-ct} + \frac{\mathrm{e}^c}{c} \sup_{0\leq\tau\leq t} \left(\tilde{\mathcal{U}}(\beta,u)(\tau)\right)^2. \end{aligned} \tag{30}$$

Using the facts that

$$\frac{1}{\left(1+\|l\|_\infty\right)^2} \|u\|^2 \leq V \leq \mathrm{e}^c \left(1+\|k\|_\infty\right)^2 \|u\|^2. \tag{31}$$

and $\|k\|_\infty, \|l\|_\infty \leq B_\beta \mathrm{e}^{B_\beta}$, $\|l\|_\infty \leq B_\beta$ we get

$$\begin{aligned} \|u(t)\| \leq\ & \left(1+B_\beta\right)\left(1+B_\beta \mathrm{e}^{B_\beta}\right) \mathrm{e}^{c/2} \mathrm{e}^{-ct/2} \|u(0)\| \\ & + \left(1+B_\beta\right) \frac{\mathrm{e}^{c/2}}{\sqrt{c}} \sup_{0\leq\tau\leq t} \left|\tilde{\mathcal{U}}(\beta,u)(\tau)\right|. \end{aligned} \tag{32}$$

The conclusions of the theorem are directly deduced from this estimate and the bound $|\tilde{\mathcal{U}}| < \epsilon$ in Lemma 2. ■

Remark 1: Full-state measurement $u(x,t)$ is employed in the feedback law (14) but can be avoided by employing only the measurement of the outlet signal, $u(0,t)$, from which the full state $u(x,t)$ is observable, the observer

$$\begin{aligned} \breve{u}_t &= \breve{u}_x + \beta u(0) && (33)\\ \hat{u}(1) &= U && (34) \end{aligned}$$

and the observer-based controller

$$\hat{U} = (k * \breve{u})(1), \tag{35}$$

which can avoid solving the PDE (33), (34) online by employing its explicit solution as an arbitrary function $\breve{u}(x,t) = \breve{u}_0(x)$ for $t + x \in [0,1)$ and

$$\breve{u}(x,t) = \hat{U}(t+x-1) + \int_{t+x-1}^t \beta(t+x-\tau) u(0,\tau) d\tau \tag{36}$$

for $t + x \geq 1$. A closed-loop stability result as in Theorem 2 can be established for this observer-based controller.

VI. Simulations: Practical Stabilization with NO-Approximated Feedback Law $(\beta, u) \to U$

Learning the map $(\beta, u) \mapsto U$ is challenging due to the combination of two functions, β, and u. The network architecture, as presented in Figure 2 requires significant enhancement over a traditional DeepONet. To learn this mapping, we emulate the operator structure where the map (β, u) requires two DeepONet layers for the integral operators adjoined with linear layers for the multiplicative operation.

We can learn the mapping using a training set defined by β as in Figure 3 (Chebyshev Polynomials $\beta = 6\cos(\gamma \cos^{-1}(x))$ with $\gamma \in$ uniform$(2,6)$ and completely random values of u. We present results with the learned mapping in Figure 3 of the open-loop unstable system. We can see that the learned control contains a significant error. Due to this, the PDE in the right of Figure 3 contains a significant ripple past the time $T = 1$ whereas the analytically controlled PDE is stabilized, as stipulated by the target system, by $T = 1$. Additionally, to make the network feasible, we use a spatial resolution of $dx = 0.01$ and a

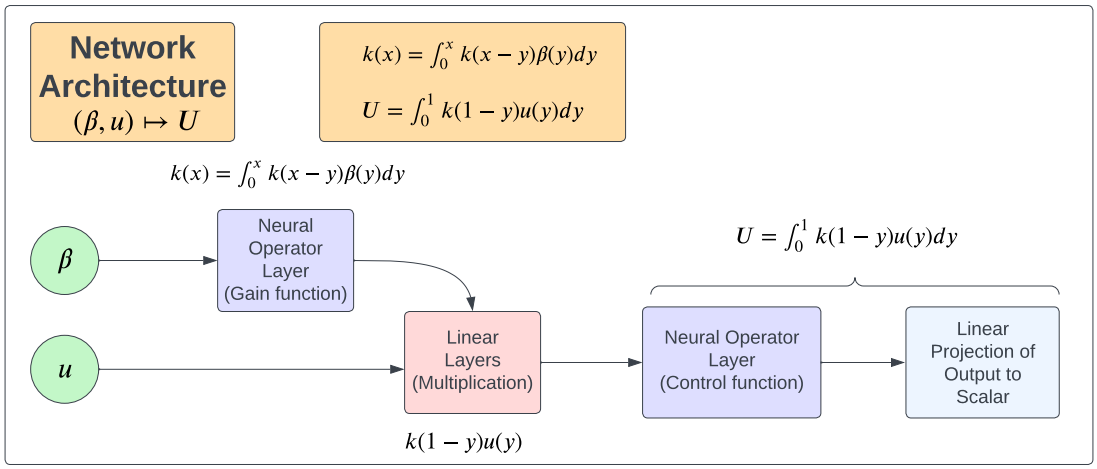

Fig. 2. Network architecture for the map $(\beta, u) \mapsto U$ presented in Section V. The network first solves the kernel function using a DeepONet layer, then utilizes linear layers to multiply k with the PDE state u, and concludes by learning a second neural operator layer for the nonlinear integral operation yielding the final control output U.

larger dataset. The dataset requires a combination of both β and u and thus consists of 50000 instances. Therefore a network of approximately 415 thousand parameters takes approximately 20 minutes to train. We achieved a training relative L_2 error of $7.2e-3$ and a testing relative L_2 error of $3.3e-2$. This demonstrates, to the practical user, that although the map (β, u) requires more training data and significant architectural enhancements, it is possible to learn the entire control feedback law. In the future, we plan to study network architecture and data construction enhancements to improve the learning of this challenging feedback map.

VII. Conclusions

We introduce a novel framework for approximating solution maps for the feedback law U (9) in *control of PDEs*. We provide the guarantees that (i) any desired level of accuracy of NO approximation of the backstepping feedback law is achieved for any (β, u) that satisfies $\|\beta\|_\infty \leq B_\beta$, $\|u\|_\infty \leq B_u$ for arbitrarily large given $B_\beta, B_u > 0$, and (ii) the PDE is stabilized with an NO-approximated control for any $\|\beta\|_\infty \leq B_\beta$, $\|u\|_\infty \leq B_u$.

For any given pair $B_\beta, B_u > 0$ and any chosen positive $\epsilon < \epsilon^*(B_\beta, B_u)$, the determination of the NO approximate operator $\mathcal{U}(\cdot)$ is done offline, once only, and such a $\hat{\mathcal{U}}(\cdot)$, which depends on B_β, B_u and ϵ, is usable "forever," so to speak, for any recirculation kernel and system state that does not violate $\|\beta\|_\infty \leq B_\beta$, $\|u\|_\infty \leq B_u$.

Our achievement of semi-global exponential stability (not "practical"/approximate, but with an actual convergence of the state to zero) relies crucially—in each of the lemmas and theorems that we state—on the theoretical steps from the PDE backstepping toolkit (backstepping transform, target system, integral equation for kernel, successive infinite-series approximation, Lyapunov analysis). It is only by assigning the NO a service role in an otherwise model-based design that stability is assured. As future work, we plan to extend the results to feedback laws for parabolic PDEs in [29] as well as to observers [30] with guarantees of observer convergence and with observer-based stabilization (separation principle).

References

[1] H. Anfinsen and O. Aamo. *Adaptive Control of Hyperbolic PDEs*. Springer, 2019.

[2] A. Aswani, H. Gonzalez, S. S. Sastry, and C. Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, 2013.

[3] P. Bernard and M. Krstic. Adaptive output-feedback stabilization of non-local hyperbolic PDEs. *Automatica*, 50:2692–2699, 2014.

[4] L. Bhan, Y. Shi, and M. Krstic. Neural operators for bypassing gain and control computations in pde backstepping. *arXiv preprint arXiv:2302.14265*, 2023.

[5] L. Bhan, Y. Shi, and M. Krstic. Neural operators for hyperbolic pde backstepping kernels. Invited paper submitted to IEEE Conference on Decision and Control, 2023.

[6] N. Boffi, S. Tu, N. Matni, J.-J. Slotine, and V. Sindhwani. Learning stability certificates from data. In *Conference on Robot Learning*, pages 1341–1350. PMLR, 2021.

[7] C. Dawson, S. Gao, and C. Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods. *arXiv preprint arXiv:2202.11762*, 2022.

[8] C. De Persis, M. Rotulo, and P. Tesi. Learning controllers from data via approximate nonlinearity cancellation. *IEEE Transactions on Automatic Control*, pages 1–16, 2023.

[9] S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. *Advances in Neural Information Processing Systems*, 31, 2018.

[10] M. Demetriou, I. Rosen, and P. Ioannou. Adaptive parameter estimation for a class of distributed parameter systems with persistence of excitation. In *[1992] Proceedings of the 31st IEEE Conference on Decision and Control*, pages 1742–1743 vol.2, 1992.

[11] B. Deng, Y. Shin, L. Lu, Z. Zhang, and G. E. Karniadakis. Approximation rates of deeponets for learning operators arising from advection–diffusion equations. *Neural Networks*, 153:411–426, 2022.

[12] M. Fazel, R. Ge, S. Kakade, and M. Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *International conference on machine learning*, pages 1467–1476. PMLR, 2018.

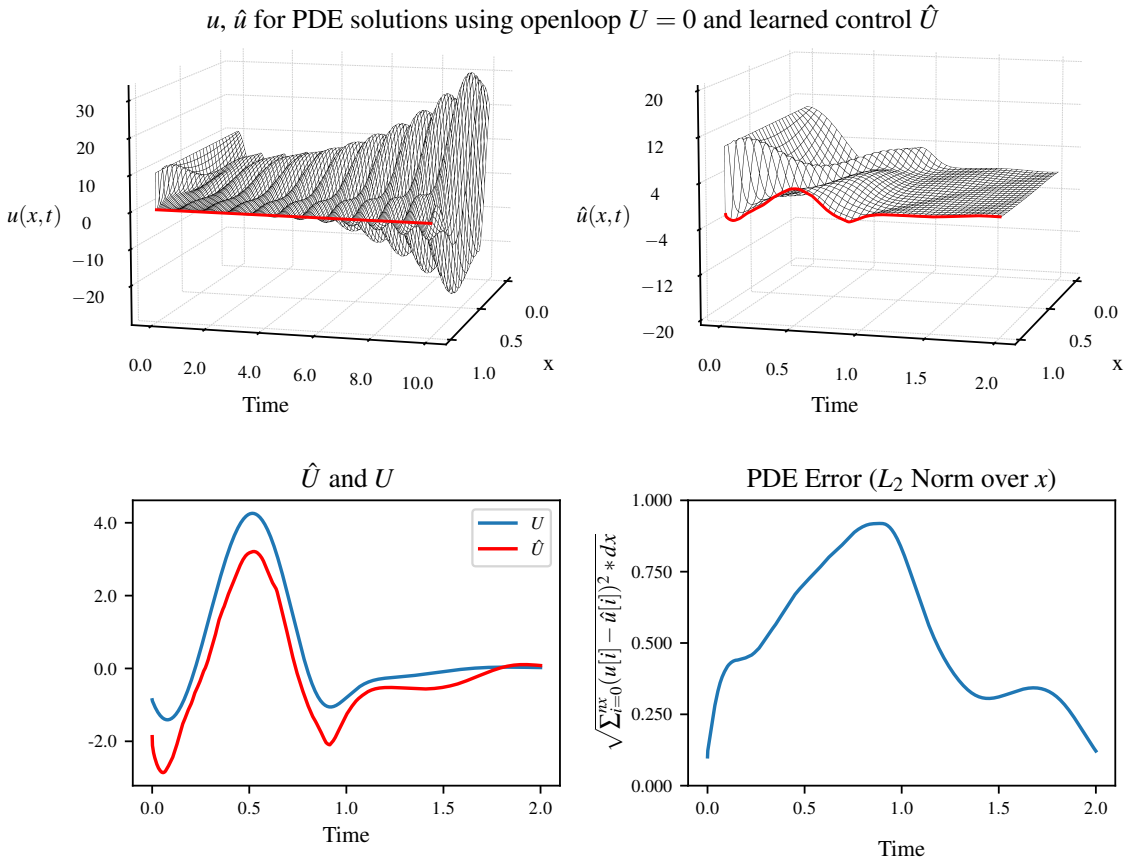

Fig. 3. Examples of PDE open-loop state instability, closed-loop state response with learned feedback law and errors between the response with "perfect control" U and "approximate control" $\hat{U}$. β is constructed with the chebyshev polynomials $\beta = 6\cos(\gamma \cos^{-1}(x))$ and the $\gamma = 3$ instance is shown.

[13] B. Hu, K. Zhang, N. Li, M. Mesbahi, M. Fazel, and T. Başar. Towards a theoretical foundation of policy optimization for learning control policies. *arXiv preprint arXiv:2210.04810*, 2022.

[14] S. Kakade, A. Krishnamurthy, K. Lowrey, M. Ohnishi, and W. Sun. Information theoretic regret bounds for online nonlinear control. *arXiv preprint*, 2020.

[15] G. Kissas, J. H. Seidman, L. F. Guilhoto, V. M. Preciado, G. J. Pappas, and P. Perdikaris. Learning operators with coupled attention. *Journal of Machine Learning Research*, 23(215):1–63, 2022.

[16] N. Kovachki, S. Lanthaler, and S. Mishra. On universal approximation and error bounds for fourier neural operators. *The Journal of Machine Learning Research*, 22(1):13237–13312, 2021.

[17] M. Krstic and A. Smyshlyaev. Backstepping boundary control for first-order hyperbolic PDEs and application to systems with actuator and sensor delays. *Systems & Control Letters*, 57(9):750–758, 2008.

[18] S. Lale, K. Azizzadenesheli, B. Hassibi, and A. Anandkumar. Reinforcement learning with fast stabilization in linear dynamical systems. In *International Conference on Artificial Intelligence and Statistics*, pages 5354–5390. PMLR, 2022.

[19] S. Lale, Y. Shi, G. Qu, K. Azizzadenesheli, A. Wierman, and A. Anandkumar. Kcrl: Krasovskii-constrained reinforcement learning with guaranteed stability in nonlinear dynamical systems. *arXiv preprint arXiv:2206.01704*, 2022.

[20] S. Lanthaler, S. Mishra, and G. E. Karniadakis. Error estimates for DeepONets: a deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1), 03 2022. tnac001.

[21] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.

[22] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

[23] L. Lu, P. Jin, and G. E. Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv:1910.03193*, 2019.

[24] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

[25] H. H. Nguyen, T. Zieger, S. C. Wells, A. Nikolakopoulou, R. D. Braatz, and R. Findeisen. Stability certificates for neural network learning-based controllers using robust control theory. In *2021 American Control Conference (ACC)*, pages 3564–3569, 2021.

[26] U. Rosolia and F. Borrelli. Learning model predictive control for iterative tasks. a data-driven control framework. *IEEE Transactions on Automatic Control*, 63(7):1883–1896, 2017.

[27] J. H. Seidman, G. Kissas, P. Perdikaris, and G. J. Pappas. NOMAD: Nonlinear manifold decoders for operator learning. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[28] Y. Shi, Z. Li, H. Yu, D. Steeves, A. Anandkumar, and M. Krstic. Machine learning accelerated pde backstepping observers. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 5423–5428, 2022.

[29] A. Smyshlyaev and M. Krstic. Closed-form boundary state feedbacks for a class of 1-d partial integro-differential equations. *IEEE Transactions on Automatic Control*, 49(12):2185–2202, 2004.

[30] A. Smyshlyaev and M. Krstic. Backstepping observers for a class of parabolic pdes. *Systems & Control Letters*, 54(7):613–625, 2005.

[31] A. J. Taylor, V. D. Dorobantu, M. Krishnamoorthy, H. M. Le, Y. Yue, and A. D. Ames. A control lyapunov perspective on episodic learning via projection to state stability. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 1448–1455. IEEE, 2019.