Automatica 164 (2024) 111649

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Neural operators of backstepping controller and observer gain functions for reaction–diffusion PDEs[☆]

Miroslav Krstic[*], Luke Bhan, Yuanyuan Shi

University of California, San Diego, USA

ARTICLE INFO

Article history:
Received 18 March 2023
Received in revised form 18 October 2023
Accepted 3 February 2024
Available online xxxx

ABSTRACT

Unlike ODEs, whose models involve system matrices and whose controllers involve vector or matrix gains, PDE models involve functions in those roles—functional coefficients, dependent on the spatial variables, and gain functions dependent on space as well. The designs of gains for controllers and observers for PDEs, such as PDE backstepping, are mappings of system model functions into gain functions. These infinite-dimensional nonlinear operators are given in an implicit form through PDEs, in spatial variables, which need to be solved to determine the gain function for each new functional coefficient of the PDE. The need for solving such PDEs can be eliminated by learning and approximating the design mapping in the form of a neural operator. Learning the neural operator requires a sufficient number of prior solutions for the design PDEs, offline, as well as the training of the operator. In recent work, we developed the neural operators for PDE backstepping designs for first-order hyperbolic PDEs. Here we extend this framework to the more complex class of parabolic PDEs. The key theoretical question is whether the controllers are still stabilizing, and whether the observers are still convergent, if they employ the approximate functional gains generated by the neural operator. We provide affirmative answers to these questions, namely, we prove stability in closed loop under gains produced by neural operators. We illustrate the theoretical results with numerical tests and publish our code on Github(https://github.com/lukebhan/NeuralOperatorsForAdvectionDiffusionControl). The neural operators are three orders of magnitude faster in generating gain functions than PDE solvers for such gain functions. This opens up the opportunity for the use of this neural operator methodology in adaptive control and in gain scheduling control for nonlinear PDEs.

1. Introduction

ML as a tool for learning PDE control methodologies. In Bhan, Shi, and Krstic (2023a) we introduced a learning-based control *framework* which devises a new role for machine learning (ML): learn an entire control design *methodology*, in the form of a mapping from the plant model to the controller gains, or even to the control inputs. Since the infinite-dimensional state of a PDE is a function of *spatial* variables, in PDE control the controller gain is also a function of spatial variables. Finding the gain typically entails solving a PDE in space (but not in time). It is therefore of interest, in PDE control, to have a capability where producing the control gain functions is just an evaluation of a neural mapping that has already learned the design methodology on a large set of previously offline-solved control design problems for a sample set of PDEs in a certain class.

Neural operators for approximating mappings of functions into functions. Our inspiration for encoding PDE control methodologies into machine learning comes from recent advances in the mathematics of machine learning. Motivated by the tasks of finding solution/flow maps (from the initial conditions into future states) for physical PDEs, several machine learning research teams (Li et al., 2021; Lu, Jin, Pang, Zhang, & Karniadakis, 2021; Seidman, Kissas, Perdikaris, & Pappas, 2022) have developed neural approximation methods, termed "neural operators". These approaches are not simply discretizing PDEs and finding solution maps to the resulting large ODE solution problems. They approximate (non-discretized) function-to-function nonlinear operators and provide provable guarantees of the accuracy of approximation in terms of the required sizes of the training sets and neural networks. Neural operators have demonstrated impressive computational speedups in solving complex PDE systems for weather forecasting (Kurth et al., 2023), state estimation and control (Bhan, Shi,

[☆] The material in this paper was partially presented at The 62nd IEEE Conference on Decision and Control (CDC), December 13–15, 2023, Marina Bay Sands, Singapore. This paper was recommended for publication in revised form by Associate Editor Thomas Meurer under the direction of Editor Luca Zaccarian.

[*] Corresponding author.

E-mail addresses: krstic@ucsd.edu (M. Krstic), lbhan@ucsd.edu (L. Bhan), yyshi@eng.ucsd.edu (Y. Shi).

https://doi.org/10.1016/j.automatica.2024.111649

& Krstic, 2023b; Shi et al., 2022), and producing numerical solutions to previously unsolvable problems such as high-dimensional PDEs (Han, Jentzen, & Weinan, 2018).

With a rigorous and numerically powerful capability like this, specific PDE control methods, for specific classes of PDEs, can be learned once and encoded as neural operators, ready to produce the control gain functions for any new functional coefficients of the same classes of PDEs.

Such an approach is of value if it allows the retention of the theoretical properties proven for the "exact design". This is indeed what we show in the paper (Bhan et al., 2023a) in which we introduce the framework: approximate neural operator representations of a particular PDE control method – PDE backstepping – preserves its stability guarantees in spite of the control gains not being generated by solving the design PDEs but by the gains being generated from the learned "neural model" of PDE backstepping.

Extension of PDE backstepping neural operators from hyperbolic (Bhan et al., 2023a) to parabolic PDEs. Hyperbolic PDEs involve only the first derivatives in space and time. This makes them (all else being equal) the "simplest" PDE class for control. Delay systems combine ODEs with delays—the simplest form of a PDE. While the simplest among PDEs, hyperbolic PDEs are not necessarily easy to control. They can be unstable, with many unstable eigenvalues, and only one input acting at the boundary of a domain. This mix of simplicity within the PDE family, with the non-triviality for control, makes hyperbolic PDEs the ideal entry point for any new study in PDE control, including the introduction of a new framework for learning-based control in our (Bhan et al., 2023a).

Control design problems for hyperbolic PDEs are hyperbolic PDEs themselves, namely, equations with only first derivatives in multiple spatial variables. Parabolic PDEs, with their first derivative in time but second derivatives in space, are the natural next challenge for learning the PDE backstepping methodology using neural operators. This is what we undertake in this paper. The chief difficulty with learning backstepping kernel operators for parabolic PDEs is that the kernels are governed by second-order PDEs, which raises the difficulty for solving such PDEs and for proving the sufficient smoothness of their solutions so that the neural operator (NO) approximations have guarantee of sufficient accuracy for preserving stabilization.

We consider parabolic PDE systems of the form

$$u_t(x,t) = u_{xx}(x,t) + \lambda(x)u(x,t), \qquad x \in (0,1), \tag{1}$$

$$u(0,t) = 0, \tag{2}$$

$$u(1,t) = U(t). \tag{3}$$

Our goal is the design of a PDE backstepping boundary control

$$U(t) = \int_0^1 k(1,y)u(y,t)dy, \tag{4}$$

as well as an observer with the (collocated) boundary sensing of $u_x(1,t)$. By "design" we mean to find the gain function k in the control law (4), namely, to find the output k of the function-to-function mapping $\mathcal{K} : \lambda \mapsto k$, depicted in Fig. 1. This paper's objective is to learn the design operator $\mathcal{K}$ with a neural operator approximation $\hat{\mathcal{K}}$ and to employ the resulting approximate gain $\hat{k}$ in the control law.

Since parabolic PDEs in one dimension have two boundary conditions, and also boundary actuation and boundary sensing can be employed at either boundary, a total of twelve combinations of boundary actuation, boundary sensing, and boundary condition on the unactuated boundary are possible. Taking the symmetry between the boundaries $x = 0$ and $x = 1$ into account, the total number of truly distinct combinations is six. They are listed in Table 1.

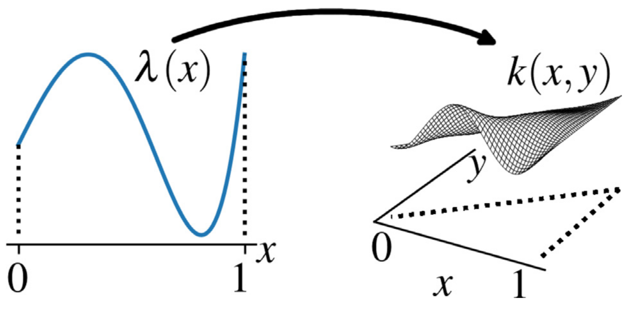


Fig. 1. The PDE backstepping design operator $\mathcal{K} : \lambda \mapsto k$, where $\lambda(x)$ is the spatially-varying reaction coefficient of the PDE, whereas $k(x,y)$ is the kernel function of the backstepping transformation, producing the feedback gain function $k(1,y)$ in the feedback law $U(t) = \int_0^1 k(1,y)u(y,t)dy$.

Table 1
Six possible combinations of boundary actuation, sensing, and boundary condition at the opposite end of [0, 1]. We focus on the simplest combination—in the second row.

Actuation	Opposite boundary	Sensing	
$u(1,t) = U(t)$	$u(0,t) = 0$	$u_x(0,t)$	anti-col
$\boldsymbol{u(1,t) = U(t)}$	$\boldsymbol{u(0,t) = 0}$	$\boldsymbol{u_x(1,t)}$	**col**
$u(1,t) = U(t)$	$u_x(0,t) = 0$	$u(0,t)$	anti-col
$u_x(1,t) = U(t)$	$u(0,t) = 0$	$u_x(0,t)$	anti-col
$u_x(1,t) = U(t)$	$u_x(0,t) = 0$	$u(0,t)$	anti-col
$u_x(1,t) = U(t)$	$u_x(0,t) = 0$	$u(1,t)$	col

We are able to solve all six problems but, in this paper, pursue the simplest of the six combinations for pedagogical reasons. The case with Dirichlet boundary conditions, $u(0,t) = 0, u(1,t) = U(t)$ is, notationally, the simplest case. It allows the reader to most quickly grasp the utility and the technical steps in employing neural operators in the control of parabolic PDEs. We opt to present in the paper the results for the combination in the second row of Table 1 because this combination allows us to "kill two birds with one stone" in our exposition. For this particular actuator–sensor combination, which is collocated (and the simplest of the four collocated combinations), the same kernel is used to obtain the gain functions for both the controller and the observer. Further, using the blueprint presented here, readers interested in the other combinations in Table 1 can develop controllers and observers for any sensing and control application.

All the results in the paper – a full-state controller, an observer, and an output-feedback law (as well as the seven additional combinations not pursued in the paper) – can be extended to the more general class of parabolic PDE systems,

$$\begin{aligned} v_t(x,t) &= \varepsilon(x)v_{xx}(x,t) + b(x)v_x(x,t) + \lambda(x)v(x,t) \\ &\quad + g(x)v(0,t) + \int_0^x f(x,y)v(y,t)dy, \quad x \in (0,L). \end{aligned} \tag{5}$$

PDE backstepping. Even though PDE backstepping was first developed for parabolic systems (Smyshlyaev & Krstic, 2004), it is best to begin its study from the easier, hyperbolic case (Krstic & Smyshlyaev, 2008c). Control of hyperbolic PDEs has grown into a rich area because, in the hyperbolic case, one can stabilize a coupled system with fewer inputs than PDEs. A *pair* of coupled hyperbolic PDEs was stabilized with a single boundary input in Coron, Vazquez, Krstic, and Bastin (2013), an extension to $n+1$ hyperbolic PDEs with a single input was introduced in Di Meglio, Vazquez, and Krstic (2013), an extension to $n+m$ PDEs with boundary actuation on m "homodirectional" PDEs in Hu, Di Meglio, Vazquez, and Krstic (2016), Hu, Vazquez, Di Meglio, and Krstic (2019), an extension to cascades with ODEs in Di Meglio, Argomedo, Hu, and Krstic (2018), and an extension to "sandwiched" ODE-PDE-ODE systems in Wang and Krstic (2020, 2022). Redesigns robust to delays are provided in Auriol, Bribiesca-Argomedo, Saba, Di Loreto, and Di Meglio (2018). PDE backstepping-based output-feedback regulation with disturbances is proposed in Deutscher (2015), Deutscher and Gabriel (2018).

For parabolic PDEs, backstepping for full-state feedback stabilization was developed in Smyshlyaev and Krstic (2004) and for observer design in Smyshlyaev and Krstic (2005). A complex extension from linear to nonlinear parabolic PDEs, using infinite Volterra series, was provided in Vazquez and Krstic (2008a, 2008b). Backstepping was combined with differential flatness in Meurer and Kugi (2009). The first solutions to the null-controllability problem for parabolic PDEs were provided, using backstepping, in Coron and Nguyen (2017), Espitia, Polyakov, Efimov, and Perruquetti (2019). Sampled-data and event-triggered versions of backstepping for parabolic PDEs appeared in Espitia, Karafyllis, and Krstic (2021), Karafyllis, Espitia, and Krstic (2021), Karafyllis and Krstic (2018), Rathnayake, Diagne, Espitia, and Karafyllis (2021). Work on cascades of parabolic PDEs with other systems has included heat-ODE cascades (Bekiaris-Liberis & Krstic, 2010; Krstic, 2009a), delay-parabolic cascades (Krstic, 2009b), and ODE-heat-ODE sandwich systems (Wang & Krstic, 2019). A backstepping design for a moving-boundary PDE-ODE Stefan system was presented in Koga, Diagne, and Krstic (2019). Coupled parabolic PDEs introduce special challenges and have been tackled in Baccoli, Pisano, and Orlov (2015), Orlov, Pisano, Pilloni, and Usai (2017), Vazquez and Krstic (2016a). Extensions from multiple 1D parabolic PDEs to PDEs in 2D and higher dimensions, such as in the book Meurer (2012) are arguably even more challenging and have been pursued for Navier–Stokes and magnetohydrodynamic systems in Vazquez and Krstic (2007), Vazquez, Schuster, and Krstic (2008) on channel domains, as well as for reaction–diffusion systems on balls of arbitrary dimensions (Vazquez & Krstic, 2016b). Adaptive control designs for parabolic PDEs were introduced in Krstic and Smyshlyaev (2008b), Smyshlyaev and Krstic (2007a, 2007b), extended in Karafyllis, Krstic, and Chrysafi (2019), and extended to the hyperbolic case in Bernard and Krstic (2014). For coupled hyperbolic PDEs with unknown parameters, the parabolic designs in Krstic and Smyshlyaev (2008b), Smyshlyaev and Krstic (2007a, 2007b) inspired a comprehensive collection of adaptive control designs in the book (Anfinsen & Aamo, 2019). Applications of backstepping to PDE models of traffic are introduced in Yu and Krstic (2019, 2022). Lastly, it is worth mentioning that all of these backstepping based approaches follow a *late-lumping* approach in which the controller is designed for the PDE and then in the implementation phase it is discretized (Auriol, Morris, & Di Meglio, 2019; Riesmeier & Woittennek, 2022). Further, the advantage of late lumping over early lumping is clearly evident when comparing the results of Smyshlyaev and Krstic (2004) to Balogh and Krstic (2002).

From an implementation perspective, approximation of the kernel PDEs has been explored in Woittennek, Riesmeier, and Ecklebe (2017) and in toolboxes such as Fischer, Gabriel, and Kerschbaum (2022). Furthermore, for observer problems, Grüne and Meurer (2022) decouples the infinite dimensional PDE system into a finite dimensional slow and infinite dimensional fast subsystem for controller implementation. However, a key difference is that this work *learns* a kernel, which can be of high value in adaptive control when the kernel PDE needs to be resolved repeatedly (or numerically approximated).

DeepONet. Using the basic notions and notation for NOs given in Bhan et al. (2023a), we state next the key technical result that enables our use of NOs to learn the PDE backstepping kernel mappings. The result is quoted in its general/abstract form. It is specialized to the PDE control setting in our Theorem 4.

Theorem 1 (*DeepONet Universal Approximation Theorem (Deng, Shin, Lu, Zhang, & Karniadakis, 2022, 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 $\mathbf{u}_m = (u(x_1), u(x_2), \ldots, u(x_m))^{\mathrm{T}}$, such that*

$$|\mathcal{G}(u)(y) - \hat{\mathcal{G}}(\mathbf{u}_m)(y)| < \epsilon \tag{6}$$

for all functions $u \in \mathcal{U}$ and all values $y \in Y$ of $\mathcal{G}(u) \in \mathcal{V}$ and with $\hat{\mathcal{G}}(\mathbf{u}_m)(y) = \sum_{k=1}^{p} g^{\mathcal{N}}(\mathbf{u}_m; \Theta^{(k)}) f^{\mathcal{N}}(y; \theta^{(k)})$.

2. Basic backstepping for reaction–diffusion PDE

We employ the following backstepping transformation,

$$w(x,t) = u(x,t) - \int_0^x k(x,y)u(y,t)dy, \tag{7}$$

to convert (1), (2), (3) into the target system

$$w_t = w_{xx}, \tag{8}$$
$$w(0,t) = 0, \tag{9}$$
$$w(1,t) = 0, \tag{10}$$

with the help of feedback (4). We could as well pursue the target system $w_t = w_{xx} - cw$, $c > 0$, but we forego this design flexibility for the sake of simplicity.

To convert (1), (2), (3) into (8), (9), (10), k needs to satisfy

$$k_{xx}(x,y) - k_{yy}(x,y) = \lambda(y)k(x,y), \quad \forall (x,y) \in \breve{\mathcal{T}}, \tag{11}$$
$$k(x,0) = 0, \tag{12}$$
$$k(x,x) = -\frac{1}{2}\int_0^x \lambda(y)dy, \tag{13}$$

where $\breve{\mathcal{T}} = \{0 < y \leq x < 1\}$ and $\mathcal{T} = \{0 \leq y \leq x \leq 1\}$.

Assumption 2. $\lambda \in C^1([0,1])$.

3. Accuracy of approximation of backstepping kernel operator with DeepONet

Theorem 3 (*Proven in Smyshlyaev and Krstic (2004, 2010)*)**.** *For every $\lambda \in C^1([0,1])$, the PDE system (11), (12), (13) has a unique $C^2(\mathcal{T})$ solution with the property*

$$|k(x,y)| \leq \bar{\lambda}e^{2\bar{\lambda}x}, \tag{14}$$

for all $x \in [0,1]$, where $\bar{\lambda} = \sup_{x\in[0,1]} |\lambda(x)|$.

Next, denote the set of functions

$$\underline{K} = \left\{ k \in C^2(\mathcal{T}) \middle| k(x,0) = 0, \forall x \in [0,1] \right\}, \tag{15}$$

and let the operator $\mathcal{K} : C^1[0,1] \to \underline{K}$ be defined by

$$k(x,y) =: \mathcal{K}(\lambda)(x,y). \tag{16}$$

Additionally, let the operator $\mathcal{M} : C^1[0,1] \to \underline{K} \times C^1[0,1] \times C^2(\mathcal{T})$ be defined by

$$(k(x,y), \kappa_1(x), \kappa_2(x,y)) =: \mathcal{M}(\lambda)(x,y), \tag{17}$$

where

$$\kappa_1(x) = 2\frac{d}{dx}(k(x,x)) + \lambda(x), \tag{18}$$

$$\kappa_2(x,y) = k_{xx}(x,y) - k_{yy}(x,y) - \lambda(y)k(x,y). \tag{19}$$

Continuity of $\mathcal{M}$ is proven by following the same chain of estimates with which boundedness is proven for Theorem 3, except that pairs (λ_1, λ_2) and $(\mathcal{M}(\lambda_1), \mathcal{M}(\lambda_2))$, as well as their respective differences $\lambda_1 - \lambda_2$ and $\mathcal{M}(\lambda_1) - \mathcal{M}(\lambda_2)$, are considered, instead of considering λ and $\mathcal{M}(\lambda)$. By applying Theorem 1, we get the following key result for the approximation of a backstepping kernel by a DeepONet.

Theorem 4. *For all $B_\lambda, B_{\lambda'} > 0$ and $\epsilon > 0$, there exists a neural operator $\hat{\mathcal{M}}$ such that, for all $(x,y) \in \mathcal{T}$,*

$$\left|\mathcal{M}(\lambda)(x,y) - \hat{\mathcal{M}}(\lambda)(x,y)\right| < \epsilon, \tag{20}$$

holds for all Lipschitz λ with the properties that $\|\lambda\|_\infty \le B_\lambda$, $\|\lambda'\|_\infty \le B_{\lambda'}$, namely, there exists a neural operator $\hat{\mathcal{K}}$ such that $\hat{\mathcal{K}}(\lambda)(x,0) \equiv 0$ and

$$\begin{aligned}&\left|\mathcal{K}(\lambda)(x,y) - \hat{\mathcal{K}}(\lambda)(x,y)\right| \\ &+ \left|2\frac{d}{dx}\left(\mathcal{K}(\lambda)(x,x) - \hat{\mathcal{K}}(\lambda)(x,x)\right)\right| \\ &+ \left|\left(\partial_{xx} - \partial_{yy}\right)\left(\mathcal{K}(\lambda)(x,y) - \hat{\mathcal{K}}(\lambda)(x,y)\right)\right. \\ &\left.-\lambda(y)\left(\mathcal{K}(\lambda)(x,y) - \hat{\mathcal{K}}(\lambda)(x,y)\right)\right| < \epsilon.\end{aligned} \tag{21}$$

Note, we refer to the error ϵ between the operator and the neural operator as the *approximation accuracy* throughout the remainder of this paper.

4. Stabilization under DeepONet gain feedback

The following theorem establishes the properties of the feedback system with the approximated kernel.

Theorem 5. *Let $B_\lambda, B_{\lambda'} > 0$ be arbitrarily large and consider the system (1), (2), (3) with any $\lambda \in C^1([0,1])$ whose derivative λ' is Lipschitz and which satisfies $\|\lambda\|_\infty \le B_\lambda$ and $\|\lambda'\|_\infty \le B_{\lambda'}$. There exists a sufficiently small $\epsilon^*(B_\lambda, B_{\lambda'}) > 0$ such that the feedback law*

$$U(t) = \int_0^1 \hat{k}(1,y)u(y,t)dy, \tag{22}$$

with all NO gain kernels $\hat{k} = \hat{\mathcal{K}}(\lambda)$ of approximation accuracy $\epsilon \in (0, \epsilon^*)$ in relation to the exact backstepping kernel $k = \mathcal{K}(\lambda)$ ensures that the closed-loop system satisfies the exponential stability bound

$$\|u(t)\| \le M e^{-(t-t_0)/2}\|u_0\|, \quad \forall t \ge t_0, \tag{23}$$

where

$$M(\epsilon, \bar{\lambda}) = \left(1 + \bar{\lambda}e^{2\bar{\lambda}}\right)\left(1 + \bar{\lambda}e^{2\bar{\lambda}} + \epsilon\right)e^{\bar{\lambda}e^{2\bar{\lambda}}+\epsilon}. \tag{24}$$

Proof. *Approximate backstepping transform and perturbed target system.* Take the backstepping transformation

$$\hat{w}(x,t) = u(x,t) - \int_0^x \hat{k}(x,y)u(y,t)dy, \tag{25}$$

where $\hat{k} = \hat{\mathcal{K}}(\lambda)$. With the control law (22), the target system becomes

$$\begin{aligned}\hat{w}_t(x,t) &= \hat{w}_{xx}(x,t) + \delta_{k0}(x)u(x,t) \\ &\quad + \int_0^x \delta_{k1}(x,y)u(y,t)dy,\end{aligned} \tag{26}$$

$$\hat{w}(0,t) = 0, \tag{27}$$

$$\hat{w}(1,t) = 0, \tag{28}$$

with

$$\begin{aligned}\delta_{k0}(x) &= 2\frac{d}{dx}\left(\hat{k}(x,x)\right) + \lambda(x) \\ &= -2\frac{d}{dx}\left(\tilde{k}(x,x)\right),\end{aligned} \tag{29}$$

$$\begin{aligned}\delta_{k1}(x,y) &= \partial_{xx}\hat{k}(x,y) - \partial_{yy}\hat{k}(x,y) - \lambda(y)\hat{k}(x,y) \\ &= -\partial_{xx}\tilde{k}(x,y) + \partial_{yy}\tilde{k}(x,y) + \lambda(y)\tilde{k}(x,y),\end{aligned} \tag{30}$$

where

$$\tilde{k} = k - \hat{k} = \mathcal{K}(\lambda) - \hat{\mathcal{K}}(\lambda). \tag{31}$$

With (21), we get

$$\|\delta_{k0}\|_\infty \le \epsilon, \tag{32}$$

$$\|\delta_{k1}\|_\infty \le \epsilon. \tag{33}$$

Inverse approximate backstepping transformation. Since the state u appears under the integral in the $\hat{w}$-system (26), in the Lyapunov analysis we need the inverse backstepping transformation

$$u(x,t) = \hat{w}(x,t) + \int_0^x \hat{l}(x,y)\hat{w}(y,t)dy. \tag{34}$$

It is shown in Krstic and Smyshlyaev (2008a) that the direct and inverse backstepping kernels satisfy in general the relationship

$$\hat{l}(x,y) = \hat{k}(x,y) + \int_y^x \hat{k}(x,\xi)\hat{l}(\xi,y)dy. \tag{35}$$

The inverse kernel satisfies the following conservative bound

$$\|\hat{l}\|_\infty \le \|\hat{k}\|_\infty e^{\|\hat{k}\|_\infty}. \tag{36}$$

Since $\|k - \hat{k}\|_\infty < \epsilon$, we have that $\|\hat{k}\|_\infty \le \|k\|_\infty + \epsilon$. With (14) we get

$$\|\hat{k}\|_\infty \le \bar{k} + \epsilon, \tag{37}$$

$$\bar{k}(\bar{\lambda}) := \bar{\lambda}e^{2\bar{\lambda}}, \tag{38}$$

and hence

$$\|\hat{l}\|_\infty \le \left(\bar{\lambda}e^{2\bar{\lambda}} + \epsilon\right)e^{\bar{\lambda}e^{2\bar{\lambda}}+\epsilon}. \tag{39}$$

Lyapunov analysis. The Lyapunov functional

$$V = \frac{1}{2}\|\hat{w}\|^2, \tag{40}$$

has a derivative

$$\dot{V} = -\|\hat{w}_x\|^2 + \Delta_0 + \Delta_1, \tag{41}$$

where

$$\Delta_0(t) = \int_0^1 \hat{w}(x,t)\delta_{k0}(x)u(x,t)dx, \tag{42}$$

$$\Delta_1(t) = \int_0^1 \hat{w}(x,t)\int_0^x \delta_{k1}(x,y)u(y,t)dydx. \tag{43}$$

With several straightforward majorizations, we get

$$\begin{aligned}\Delta_0 &\le \|\delta_{k0}\|_\infty\left(1 + \|\hat{l}\|_\infty\right)\|\hat{w}\|^2 \\ &= \|\delta_{k0}\|_\infty\left(1 + \|\hat{l}\|_\infty\right)2V,\end{aligned} \tag{44}$$

and

$$\begin{aligned}\Delta_1 &= \int_0^1 \hat{w}(x)\int_0^y \hat{w}(y)\int_y^x \delta_k(x,\sigma)\hat{l}(\sigma,y)d\sigma dydx \\ &\quad + \int_0^1 \hat{w}(x)\int_0^x \delta(x,y)\hat{w}(y)dydx\end{aligned}$$

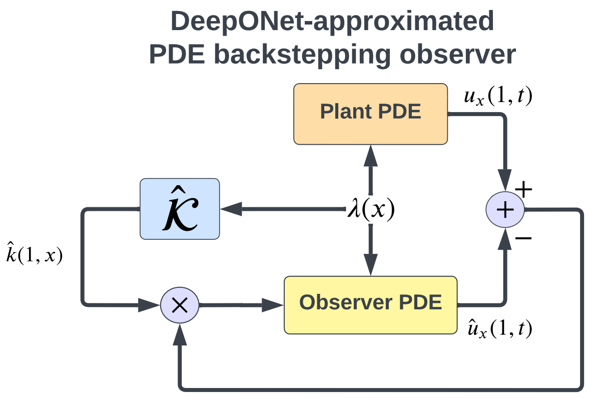


Fig. 2. The PDE backstepping observer (51), (52), (53) uses boundary measurement of the flux $u_x(1,t)$. The gain $\hat k(1,t)$ is produced with the DeepONet $\hat{\mathcal{K}}$.

$$\begin{aligned}&\le \|\delta_{k1}\|_\infty\left(1+\|\hat l\|_\infty\right)\|\tilde w\|^2\\ &= \|\delta_{k1}\|_\infty\left(1+\|\hat l\|_\infty\right)2V. \end{aligned}\tag{45}$$

From (41), (44), (45), (39), and Poincare's inequality, we get

$$\dot V \le -\frac{1}{2}(1-\delta^*)V, \tag{46}$$

where

$$\delta^*(\epsilon,\bar\lambda) = 2\epsilon\left(1+\bar\lambda e^{2\bar\lambda}+\epsilon\right)e^{\bar\lambda e^{2\bar\lambda}+\epsilon}, \tag{47}$$

is an increasing function of $\epsilon, \bar\lambda$, with the property that $\delta^*(0,\bar\lambda)=0$. Hence, there exists $\epsilon^*(\bar\lambda)$ such that, for all $\epsilon\in[0,\epsilon^*]$,

$$\dot V \le -\frac{1}{4}V, \tag{48}$$

namely, $V(t)\le V_0e^{-(t-t_0)/4}$. From the direct and inverse backstepping transformations it follows that

$$\frac{1}{1+\|\hat l\|_\infty}\|u\| \le \sqrt{2V} \le \left(1+\|\hat k\|_\infty\right)\|u\|. \tag{49}$$

In conclusion,

$$\|u(t)\| \le \left(1+\|\hat l\|_\infty\right)\left(1+\|\hat k\|_\infty\right)e^{-(t-t_0)/2}\|u_0\|. \tag{50}$$

With (37), (38), (39), the proof is completed. □

5. Observer design

State estimators (observers) with boundary measurements can be formulated with four measurement choices on the interval $[0,1]$: the measured quantities can be $u(0,t), u_x(0,t), u(1,t), u_x(1,t)$. That leads to many possible problem formulations. Since our goals with observers are twofold – to estimate the unmeasured state but also to use it in output-feedback control for stabilization – our choice of measurement needs to be consistent with the actuation choice we have already pursued in this note, namely, Dirichlet actuation of $u(1,t)=U(t)$. So, we cannot use $u(1,t)$ for measurement but we can use $u(0,t), u_x(0,t), u_x(1,t)$. We let the output $u_x(1,t)$ be measured. Our choice of $u_x(1,t)$ for measurement, as indicated in the observer diagram in Fig. 2, is motivated by the pedagogical fact that, with this measurement, an observer can be built using the same kernel $k(x,y)$ as for the control law.

Theorem 6. *Let $B_\lambda, B_{\lambda'}>0$ be arbitrarily large and consider the system (1), (2), (3) with any $\lambda\in C^1([0,1])$ whose derivative λ' is Lipschitz and which satisfies $\|\lambda\|_\infty\le B_\lambda$ and $\|\lambda'\|_\infty\le B_{\lambda'}$. There exists a sufficiently small $\epsilon^*(B_\lambda,B_{\lambda'})>0$ such that the observer*

$$\begin{aligned}\hat u_t(x,t) &= \hat u_{xx}(x,t)+\lambda(x)\hat u(x,t)\\ &\quad +\hat k(1,x)\left[u_x(1,t)-\hat u_x(1,t)\right], \end{aligned}\tag{51}$$

$$\hat u(0,t) = 0, \tag{52}$$

$$\hat u(1,t) = U(t), \tag{53}$$

with all NO gain kernels $\hat k=\hat{\mathcal{K}}(\lambda)$ of approximation accuracy $\epsilon\in(0,\epsilon^*)$ in relation to the exact backstepping kernel $k=\mathcal{K}(\lambda)$ ensure that the observer error system, for all $u_0,\hat u_0\in L^2[0,1]$, satisfies the exponential stability bound

$$\|u(t)-\hat u(t)\| \le Me^{-(t-t_0)/2}\|u_0-\hat u_0\|, \quad \forall t\ge t_0, \tag{54}$$

where $M(\epsilon,\bar\lambda)$ is defined in (24).

Proof. We start by postulating a PDE backstepping observer in the form

$$\begin{aligned}\hat u_t(x,t) &= \hat u_{xx}(x,t)+\lambda(x)\hat u(x,t)\\ &\quad +p_1(x)\left[u_x(1,t)-\hat u_x(1,t)\right], \end{aligned}\tag{55}$$

$$\hat u(0,t) = 0, \tag{56}$$

$$\hat u(1,t) = U(t). \tag{57}$$

The observer error $\tilde u(x,t)=u(x,t)-\hat u(x,t)$ is governed by the system

$$\begin{aligned}\tilde u_t(x,t) &= \tilde u_{xx}(x,t)+\lambda(x)\tilde u(x,t)\\ &\quad -p_1(x)\tilde u_x(1,t), \end{aligned}\tag{58}$$

$$\tilde u(0,t) = 0, \tag{59}$$

$$\tilde u(1,t) = 0. \tag{60}$$

The backstepping transformation

$$\tilde u(x,t) = \tilde w(x,t)-\int_x^1 p(x,y)\tilde w(y,t)dy, \tag{61}$$

converts (58), (59), (60) into

$$\tilde w_t(x,t) = \tilde w_{xx}(x,t), \tag{62}$$

$$\tilde w(0,t) = 0, \tag{63}$$

$$\tilde w(1,t) = 0, \tag{64}$$

provided $p(x,y)$ satisfies

$$p(x,y) = k(y,x), \tag{65}$$

with k that is governed by (11), (12), (13), and with the observer gain

$$p_1(x) = +k(1,x). \tag{66}$$

It is crucial to note in (65) that the arguments x and y have been commuted in $k(\cdot,\cdot)$. The commuting of the spatial arguments of the backstepping kernel is akin to transposing matrices in going between designs for controllers and observers in finite-dimensional LTI systems. The commuted order of the arguments x and y continues in the rest of the proof. The observer (55), (56), (57) is next rewritten as

$$\begin{aligned}\hat u_t(x,t) &= \hat u_{xx}(x,t)+\lambda(x)\hat u(x,t)\\ &\quad +k(1,x)\left[u_x(1,t)-\hat u_x(1,t)\right], \end{aligned}\tag{67}$$

$$\hat u(0,t) = 0, \tag{68}$$

$$\hat u(1,t) = U(t), \tag{69}$$

and the transformation (61) as

$$\tilde u(x,t) = \tilde w(x,t)-\int_x^1 k(y,x)\tilde w(y,t)dy. \tag{70}$$

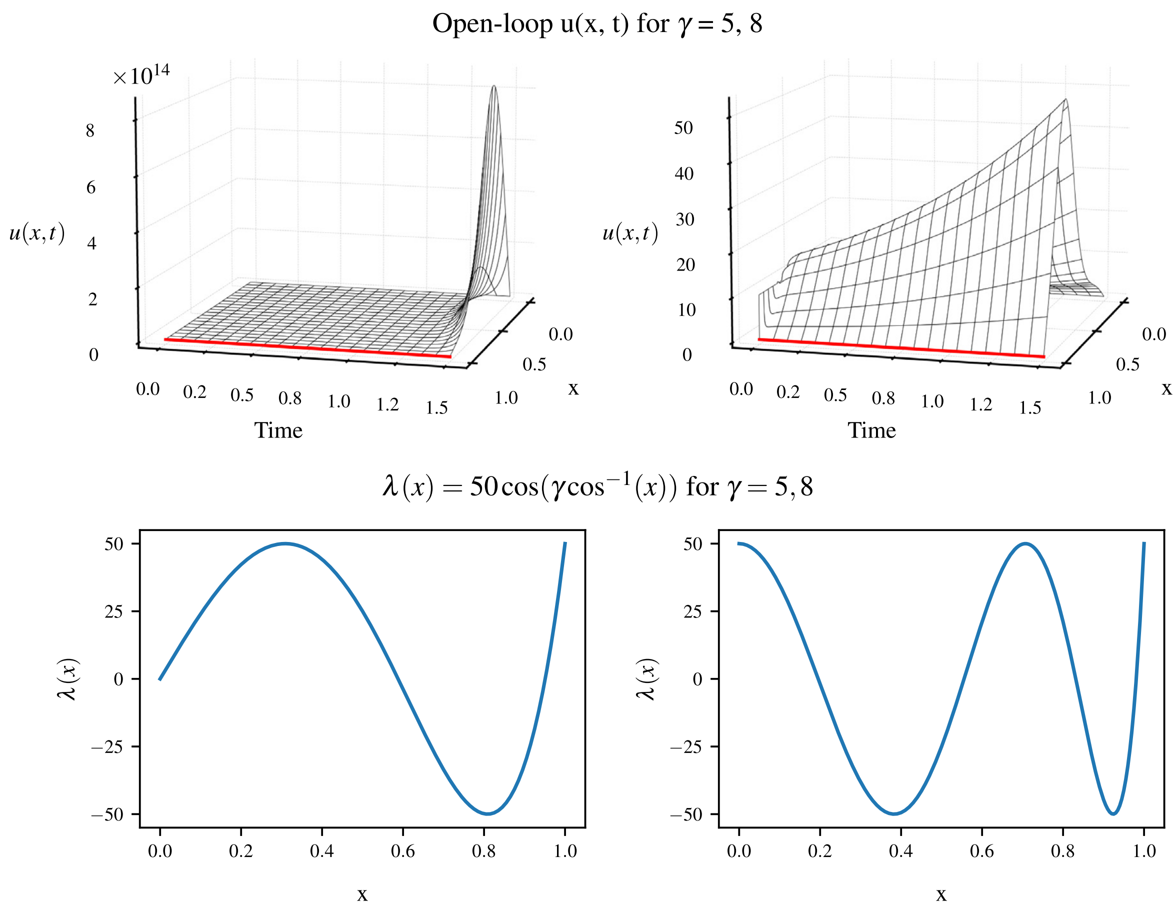


Fig. 3. Open-loop instability (top) for the two respective reaction coefficients $\lambda(x)$ with $\gamma = 5$ on the left and $\gamma = 8$ on the right. The top row shows the open loop stability with the coefficient and bottom row shows the function $\lambda(x)$ with corresponding γ value.

Theorem 4 applies to the kernel $k(y, x)$ of the observer backstepping transformation and the observer gains $-k(1, x)$. The observer (67), (68), (69) is henceforth implemented with the approximate kernel $\hat{k}$ as in (51), (52), (53) whereas the backstepping transformation (70) is applied with $\hat{k}$ as

$$\tilde{u}(x, t) = \omega(x, t) - \int_x^1 \hat{k}(y, x)\omega(y, t)dy. \tag{71}$$

The target system under the approximate kernel $\hat{k}$ becomes

$$\omega_t(x, t) = \omega_{xx}(x, t) + \Omega_0(x, t) + \Omega_1(x, t), \tag{72}$$

$$\omega(0, t) = 0, \tag{73}$$

$$\omega(1, t) = 0, \tag{74}$$

where

$$\Omega_0(x, t) = \delta_{k0}(x)\omega(x, t) + \int_x^1 \hat{l}(y, x)\delta_{k0}(y)\omega(y, t)dy, \tag{75}$$

$$\Omega_1(x, t) = \int_x^1 \left(\delta_{k1}(y, x)\omega(y, t) + \hat{l}(y, x)\int_y^1 \delta_{k1}(s, y)\omega(s, t)ds\right) dy, \tag{76}$$

and δ_{k0}, δ_{k1} are defined in (29), (30), with bounds (32), (33). Note that the arguments in δ_{k1} have been commuted in the integral in (72). Similar as in the proof of Theorem 5, the Lyapunov functional

$$V = \frac{1}{2}\|\omega\|^2, \tag{77}$$

has a derivative

$$\dot{V} \leq -\frac{1}{4}V, \tag{78}$$

namely, $V(t) \leq V_0 e^{-(t-t_0)/4}$, provided $\epsilon \in [0, \epsilon^*]$, with ϵ^* obtained from (47). The result (51) follows from (77), (71), (37), and the inverse backstepping transformation

$$\omega(x, t) = \tilde{u}(x, t) + \int_x^1 \hat{l}(y, x)\tilde{u}(y, t)dy, \tag{79}$$

whose kernel $\hat{l}$ satisfies the bound (39). □

6. Collocated output-feedback stabilization

In this section we put together the observer (51), (52), (53), along with the observer-based controller

$$U(t) = \int_0^1 \hat{k}(1, x)\hat{u}(x, t)dx, \tag{80}$$

to stabilize the system (1), (2), (3) by output feedback.

Furthermore, consider the Neumann-actuated controller with a measured Dirichlet output $u(1, t)$

$$u_t(x, t) = u_{xx}(x, t) + \lambda(x)u(x, t), \qquad x \in [0, 1), \tag{81}$$

$$u(0, t) = 0, \tag{82}$$

$$u_x(1, t) = U(t). \tag{83}$$

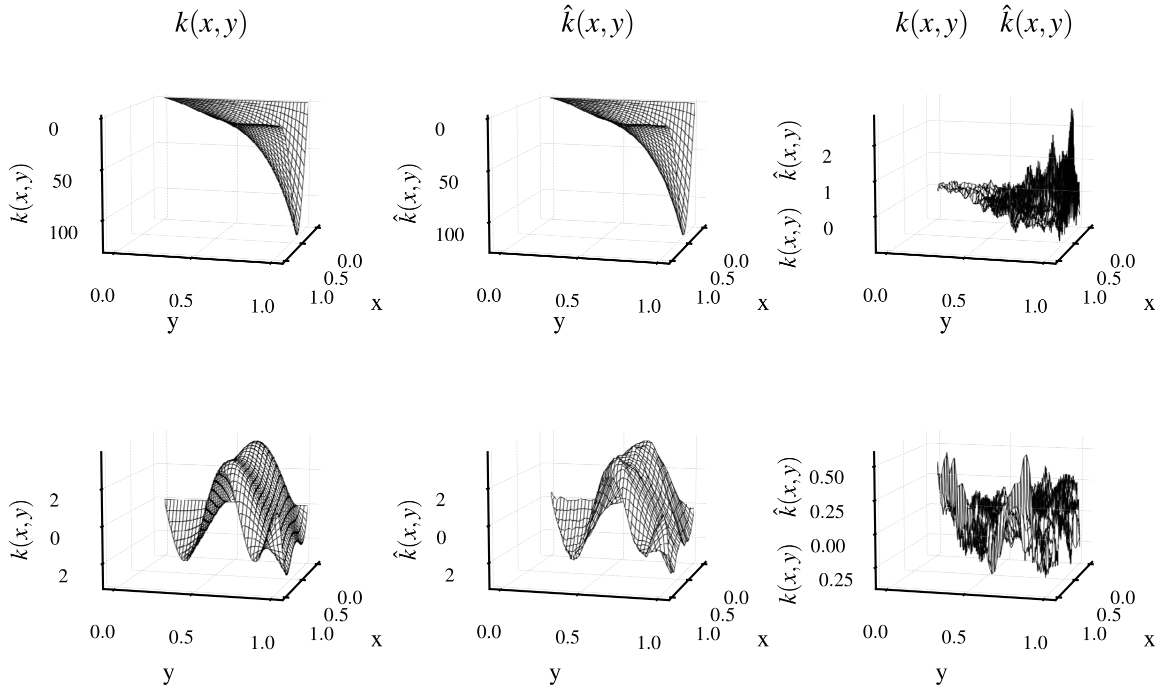


Fig. 4. Examples of the kernel $k(x, y)$ (left column), learned kernel $\hat{k}(x, y)$ (middle column), and the kernel error $k(x, y) - \hat{k}(x, y)$ (right column). The two respective $\lambda(x)$ values correspond to the same respective values as in Fig. 3 with $\gamma = 5$ on the top row and $\gamma = 8$ on the bottom row respectively.

Then, for the plant (81), (82), (83), we suggest the observer-based compensator

$$\hat{u}_t(x,t) = \hat{u}_{xx}(x,t) + \lambda(x)\hat{u}(x,t) + \hat{k}_\xi(\xi,x)\Big|_{\xi=1}\left[u(1,t) - \hat{u}(1,t)\right], \tag{84}$$

$$\hat{u}(0,t) = 0, \tag{85}$$

$$\hat{u}_x(1,t) = U(t) - \hat{k}(1,1)\left(u(1,t) - \hat{u}(1,t)\right), \tag{86}$$

$$U(t) = \hat{k}(1,1)u(1,t) + \int_0^1 \kappa(x)\hat{u}(x,t)dx, \tag{87}$$

to achieve stability. More details are available on arXiv (Krstic, Bhan, & Shi, 2023).

7. Numerical results

In Fig. 3, we show that the system (1), (2), (3) is open-loop unstable for the reaction term $\lambda(x) = 50\cos(\gamma\cos^{-1}(x))$ for $\gamma = 5, 8$. The increased oscillation in larger γ yields a lower rate of instability as shown on the right. We simulate the PDE and its control using the finite difference scheme in the Appendix.

In Fig. 4, we demonstrate both the analytical and learned DeepONet kernels for the two γ values corresponding to Fig. 3. To learn the mapping $\mathcal{K} : \lambda(x) \mapsto k(x,y)$, we construct a dataset of 1000 different $\lambda(x)$ as the Chebyshev polynomials defined in $\lambda(x)$ above with $\gamma \sim$ uniform $(4,9)$ and a 90/10 train test split. We choose λ of this form due to the rich set of kernel functions generated by varying only a single parameter. To effectively utilize the DeepONet without modifying the grid, we stack $\lambda(x)$ repeatedly n_y times over the y axis to make a 2D input to the network. Then, we capitalize on the 2D mapping by implementing a CNN for the DeepONet branch network. In the future, one can explore neural operators on irregular girds along the direction of Li, Huang, Liu, and Anandkumar (2022). For training, the relative L_2 error is $3.5e-2$ and the testing error is $3.6e-2$. With the learned neural operator, we achieve speedups on the magnitude of 10^3 compared to an efficient finite difference implementation. In Fig. 5, we demonstrate closed-loop stability with the neural operator approximated gain function for the control feedback law. Additionally, we see the error is largest at the beginning achieving a maximum in both cases of approximately 10%.

In Fig. 6, we test the observer (52), (52), (53) with a DeepONet-approximated kernel trained as above using $\lambda(x) = 20\cos(5\cos^{-1}(x))$ with $\gamma \sim$ uniform $(4,9)$. Additionally, we apply a boundary signal of $U(t) = 7\sin(16\pi t) + 10\cos(2\pi t)$ to generate a challenging and rich PDE motion for estimation. The true system state begins with initial conditions $u(x,0) = 10$ while the DeepONet observer has initial conditions of $\hat{u}_{NO}(x,0) = 20$. Despite this, the observer approximates the PDE well with a peak error of less than 5% compared to the analytical observer while maintaining the same 10^3x speedup over the finite difference scheme. Lastly, although we chose $\lambda(x)$ to be the Chebyshev polynomials, we emphasize one could chose any set of functions in $C^2[0,1]$ such as polynomials, random Fourier basis, Legendre polynomials, etc. as long as those functions create a sufficiently diverse set of kernels for the neural network to satisfy the ϵ^*. For an exact size of neural network to satisfy ϵ^*, one can utilize the conservative estimates in Deng et al. (2022) although, in many cases, they are practically infeasible due to the extremely difficult challenge of operator approximation.

8. Conclusions

We extend the NO-powered PDE backstepping introduced in Bhan et al. (2023a) from the hyperbolic to parabolic PDEs, which poses additional challenges in advancing from first-order to second-order Goursat PDEs for the kernels, and advancing from the Lyapunov study of hyperbolic target PDEs with relatively

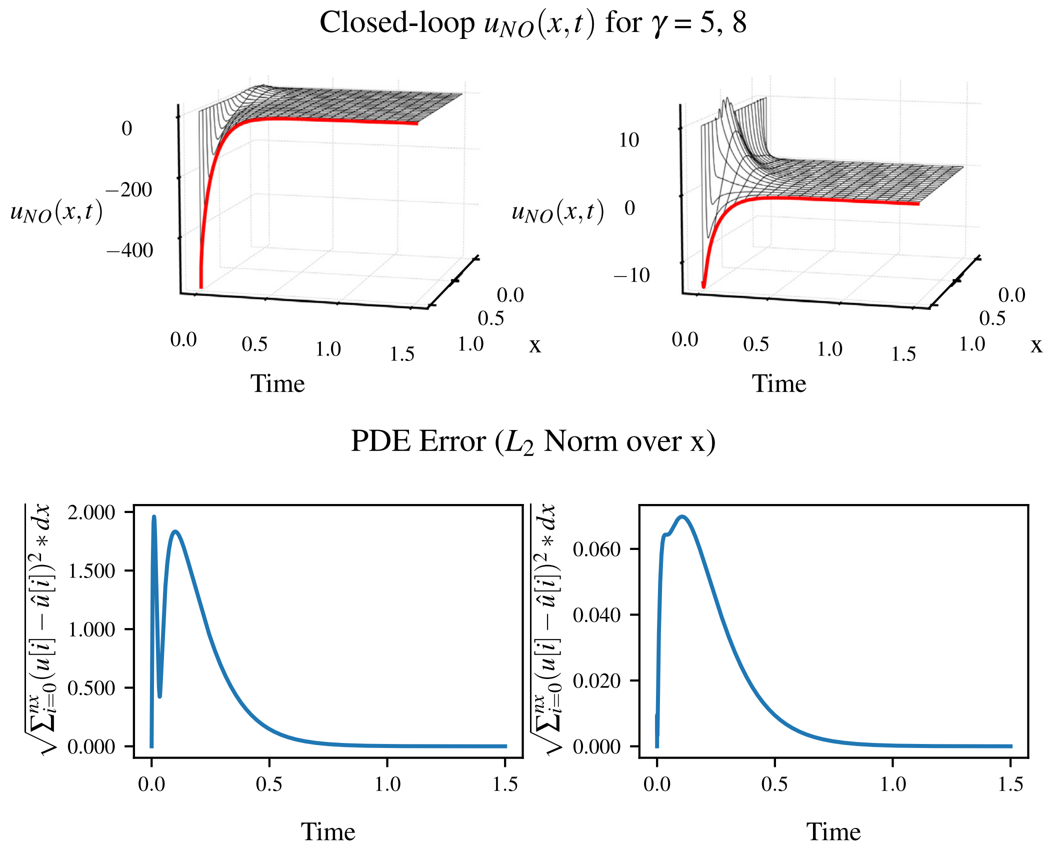


Fig. 5. For the two respective $\lambda(x)$ values as in Fig. 3, the top row showcases closed-loop solutions with the learned kernel $\hat{k}(x, y)$, whereas the bottom row shows the closed-loop PDE error between applying the original kernel $k(x, y)$ and the learned kernel $\hat{k}(x, y)$.

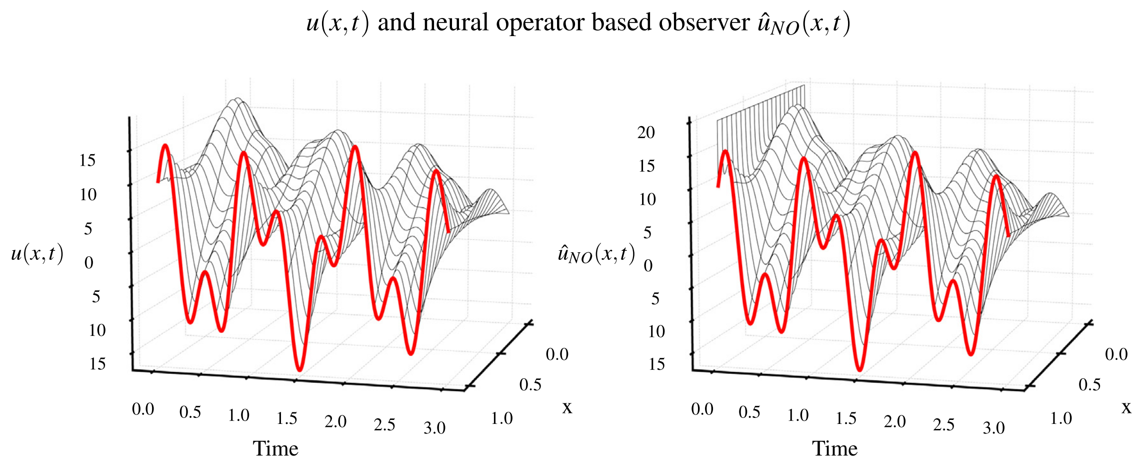


Fig. 6. Left: PDE solution with $\lambda(x) = 20\cos(5\cos^{-1}(x))$ and $U(t) = 7\sin(16\pi t) + 10\cos(2\pi t)$. Right: the observer with the neural operator-learned kernel. Note the difference between the plant initial condition $u(x, 0) = 10$ and the twice as large initial condition of the exact and neural operator observers, $\hat{u}_{\mathrm{NO}}(x, 0) = \hat{u}(0, x) = 20$. The peak error between the analytical observer (not shown) and the neural operator observer is around 0.3.

simple perturbations to parabolic target PDEs with more complex perturbations. Additionally, we present the first extension of DeepONet-backstepping from full-state feedback to observer design.

Acknowledgments

The first author was funded by AFOSR grant FA9550-23-1-0535 and NSF grants ECCS-2151525 and ECCS-2210315.

Appendix. FD scheme for goursat-form kernel PDE

The N-step discretization of (1)–(3) is (Smyshlyaev & Krstic, 2010)

$$k_j^{i+1} = -k_j^{i-1} + k_{j+1}^i + k_{j-1}^i + h^2\lambda_j \frac{k_{j+1}^i + k_{j-1}^i}{2}, \tag{A.1}$$

$$k_i^{i+1} = k_i^i + \frac{h}{2}\lambda_i, \tag{A.2}$$

$$k_{i+1}^{i+1} = k_i^i - \frac{h}{4}(\lambda_i + \lambda_{i+1}), \qquad k_1^{j+1} = 0, \tag{A.3}$$

with $k_i^j = k((i-1)h, (j-1)h)$, $i = 2, \ldots, N$, $j = 2, \ldots, i-1$, $\lambda_i = \bar{\lambda}((i-1)h)$, $h = 1/N$.

References

Anfinsen, H., & Aamo, O. (2019). *Adaptive control of hyperbolic PDEs*. Springer.

Auriol, J., Bribiesca-Argomedo, F., Saba, D., Di Loreto, M., & Di Meglio, F. (2018). Delay-robust stabilization of a hyperbolic PDE-ODE system. *Automatica*, *95*, 494–502.

Auriol, J., Morris, K. A., & Di Meglio, F. (2019). Late-lumping backstepping control of partial differential equations. *Automatica*, *100*, 247–259.

Baccoli, A., Pisano, A., & Orlov, Y. (2015). Boundary control of coupled reaction–diffusion processes with constant parameters. *Automatica*, *54*, 80–90.

Balogh, A., & Krstic, M. (2002). Infinite dimensional backstepping-style feedback transformations for a heat equation with an arbitrary level of instability. *European Journal of Control*, *8*(2), 165–175.

Bekiaris-Liberis, N., & Krstic, M. (2010). Compensating the distributed effect of diffusion and counter-convection in multi-input and multi-output LTI systems. *IEEE Transactions on Automatic Control*, *56*(3), 637–643.

Bernard, P., & Krstic, M. (2014). Adaptive output-feedback stabilization of non-local hyperbolic PDEs. *Automatica*, *50*, 2692–2699.

Bhan, L., Shi, Y., & Krstic, M. (2023a). Neural operators for bypassing gain and control computations in PDE backstepping. *IEEE Transactions on Automatic Control*, 1–16.

Bhan, L., Shi, Y., & Krstic, M. (2023b). Operator learning for nonlinear adaptive control. In *Learning for dynamics and control conference* (pp. 346–357). PMLR.

Coron, J.-M., & Nguyen, H.-M. (2017). Null controllability and finite time stabilization for the heat equations with variable coefficients in space in one dimension via backstepping approach. *Archive for Rational Mechanics and Analysis*, *225*, 993–1023.

Coron, J., Vazquez, R., Krstic, M., & Bastin, G. (2013). Local exponential H^2 stabilization of a 2×2 quasilinear hyperbolic system using backstepping. *SIAM Journal on Control and Optimization*, *51*(3), 2005–2035.

Deng, B., Shin, Y., Lu, L., Zhang, Z., & Karniadakis, G. E. (2022). Approximation rates of DeepONets for learning operators arising from advection–diffusion equations. *Neural Networks*, *153*, 411–426.

Deutscher, J. (2015). A backstepping approach to the output regulation of boundary controlled parabolic PDEs. *Automatica*, *57*, 56–64.

Deutscher, J., & Gabriel, J. (2018). Minimum time output regulation for general linear heterodirectional hyperbolic systems. *International Journal of Control*, *93*, 1826–1838.

Di Meglio, F., Argomedo, F. B., Hu, L., & Krstic, M. (2018). Stabilization of coupled linear heterodirectional hyperbolic PDE–ODE systems. *Automatica*, *87*, 281–289.

Di Meglio, F., Vazquez, R., & Krstic, M. (2013). Stabilization of a system of $n+1$ coupled first-order hyperbolic linear PDEs with a single boundary input. *IEEE Transactions on Automatic Control*, *58*, 3097–3111.

Espitia, N., Karafyllis, I., & Krstic, M. (2021). Event-triggered boundary control of constant-parameter reaction–diffusion PDEs: A small-gain approach. *Automatica*, *128*, Article 109562.

Espitia, N., Polyakov, A., Efimov, D., & Perruquetti, W. (2019). Boundary time-varying feedbacks for fixed-time stabilization of constant-parameter reaction–diffusion systems. *Automatica*, *103*, 398–407.

Fischer, F., Gabriel, J., & Kerschbaum, S. (2022). coni - A matlab toolbox facilitating the solution of control problems. Availabele at https://zenodo.org/records/6420876.

Grüne, L., & Meurer, T. (2022). Finite-dimensional output stabilization for a class of linear distributed parameter systems - A small-gain approach. *Systems & Control Letters*, *164*, Article 105237.

Han, J., Jentzen, A., & Weinan, E. (2018). Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, *115*(34), 8505–8510.

Hu, L., Di Meglio, F., Vazquez, R., & Krstic, M. (2016). Control of homodirectional and general heterodirectional linear coupled hyperbolic PDEs. *IEEE Transactions on Automatic Control*, *61*(11), 3301–3314.

Hu, L., Vazquez, R., Di Meglio, F., & Krstic, M. (2019). Boundary exponential stabilization of 1-dimensional inhomogeneous quasi-linear hyperbolic systems. *SIAM Journal on Control and Optimization*, *57*(2), 963–998.

Karafyllis, I., Espitia, N., & Krstic, M. (2021). Event-triggered gain scheduling of reaction-diffusion PDEs. *SIAM Journal on Control and Optimization*, *59*(3), 2047–2067.

Karafyllis, I., & Krstic, M. (2018). Sampled-data boundary feedback control of 1-D parabolic PDEs. *Automatica*, *87*, 226–237.

Karafyllis, I., Krstic, M., & Chrysafi, K. (2019). Adaptive boundary control of constant-parameter reaction-diffusion PDEs using regulation-triggered finite-time identification. *Automatica*, *103*, 166–179.

Koga, S., Diagne, M., & Krstic, M. (2019). Control and state estimation of the one-phase Stefan problem via backstepping design. *IEEE Transactions on Automatic Control*, *64*(2), 510–525.

Krstic, M. (2009a). Compensating actuator and sensor dynamics governed by diffusion PDEs. *Systems & Control Letters*, *58*(5), 372–377.

Krstic, M. (2009b). Control of an unstable reaction-diffusion PDE with long input delay. *Systems & Control Letters*, *58*, 773–782.

Krstic, M., Bhan, L., & Shi, Y. (2023). Neural operators of backstepping controller and observer gain functions for reaction-diffusion PDEs. arXiv:2303.10506. Available at https://arxiv.org/abs/2303.10506.

Krstic, M., & Smyshlyaev, A. (2008a). *Boundary control of PDEs: A course on backstepping designs*. SIAM.

Krstic, M., & Smyshlyaev, A. (2008b). Adaptive boundary control for unstable parabolic PDEs—Part I: Lyapunov design. *IEEE Transactions on Automatic Control*, *53*(7), 1575–1591.

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

Kurth, T., Subramanian, S., Harrington, P., Pathak, J., Mardani, M., Hall, D., et al. (2023). Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the platform for advanced scientific computing conference* (pp. 1–11).

Li, Z., Huang, D. Z., Liu, B., & Anandkumar, A. (2022). Fourier neural operator with learned deformations for PDEs on general geometries. http://dx.doi.org/10.48550/ARXIV.2207.05209, arXiv:2207.05209. Available at https://arxiv.org/abs/2207.05209.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., liu, B., Bhattacharya, K., Stuart, A., et al. (2021). Fourier neural operator for parametric partial differential equations. In *International conference on learning representations*.

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

Meurer, T. (2012). *Control of higher–dimensional PDEs: Flatness and backstepping designs*. Springer Science & Business Media.

Meurer, T., & Kugi, A. (2009). Tracking control for boundary controlled parabolic PDEs with varying parameters: Combining backstepping and differential flatness. *Automatica*, *45*(5), 1182–1194.

Orlov, Y., Pisano, A., Pilloni, A., & Usai, E. (2017). Output feedback stabilization of coupled reaction-diffusion processes with constant parameters. *SIAM Journal on Control and Optimization*, *55*(6), 4112–4155.

Rathnayake, B., Diagne, M., Espitia, N., & Karafyllis, I. (2021). Observer-based event-triggered boundary control of a class of reaction–diffusion PDEs. *IEEE Transactions on Automatic Control*, *67*(6), 2905–2917.

Riesmeier, M., & Woittennek, F. (2022). Late lumping of observer-based state feedback for boundary control systems. arXiv:2211.13784. Available at https://arxiv.org/abs/2211.13784.

Seidman, J. H., Kissas, G., Perdikaris, P., & Pappas, G. J. (2022). NOMAD: Nonlinear manifold decoders for operator learning. In A. H. Oh, A. Agarwal, D. Belgrave, & K. Cho (Eds.), *Advances in neural information processing systems*.

Shi, Y., Li, Z., Yu, H., Steeves, D., Anandkumar, A., & Krstic, M. (2022). Machine learning accelerated PDE backstepping observers. In *2022 IEEE 61st conference on decision and control* (pp. 5423–5428).

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

Smyshlyaev, A., & Krstic, M. (2005). Backstepping observers for a class of parabolic PDEs. *Systems & Control Letters*, *54*(7), 613–625.

Smyshlyaev, A., & Krstic, M. (2007a). Adaptive boundary control for unstable parabolic PDEs—Part II: Estimation-based designs. *Automatica*, *43*(9), 1543–1556.

Smyshlyaev, A., & Krstic, M. (2007b). Adaptive boundary control for unstable parabolic PDEs—Part III: Output feedback examples with swapping identifiers. *Automatica*, *43*(9), 1557–1564.

Smyshlyaev, A., & Krstic, M. (2010). *Adaptive control of parabolic PDEs*. Princeton University Press.

Vazquez, R., & Krstic, M. (2007). A closed-form feedback controller for stabilization of the linearized 2-D Navier–Stokes poiseuille system. *IEEE Transactions on Automatic Control*, *52*(12), 2298–2312.

Vazquez, R., & Krstic, M. (2008a). Control of 1-D parabolic PDEs with Volterra nonlinearities, Part I: Design. *Automatica*, *44*(11), 2778–2790.

Vazquez, R., & Krstic, M. (2008b). Control of 1D parabolic PDEs with Volterra nonlinearities, Part II: analysis. *Automatica*, *44*(11), 2791–2803.

Vazquez, R., & Krstic, M. (2016a). Boundary control of coupled reaction-advection-diffusion systems with spatially-varying coefficients. *IEEE Transactions on Automatic Control*, *62*(4), 2026–2033.

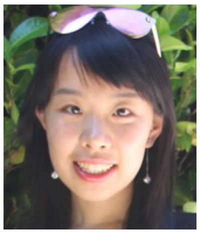
Vazquez, R., & Krstic, M. (2016b). Explicit output-feedback boundary control of reaction-diffusion PDEs on arbitrary-dimensional balls. *ESAIM. Control, Optimisation and Calculus of Variations*, *22*(4), 1078–1096.

Vazquez, R., Schuster, E., & Krstic, M. (2008). Magnetohydrodynamic state estimation with boundary sensors. *Automatica*, *44*(10), 2517–2527.

Wang, J., & Krstic, M. (2019). Output feedback boundary control of a heat PDE sandwiched between two ODEs. *IEEE Transactions on Automatic Control*, *64*(11), 4653–4660.
Wang, J., & Krstic, M. (2020). Delay-compensated control of sandwiched ODE–PDE–ODE hyperbolic systems for oil drilling and disaster relief. *Automatica*, *120*, Article 109131.
Wang, J., & Krstic, M. (2022). Event-triggered output-feedback backstepping control of sandwich hyperbolic PDE systems. *IEEE Transactions on Automatic Control*, *67*(1), 220–235.
Woittennek, F., Riesmeier, M., & Ecklebe, S. (2017). On approximation and implementation of transformation based feedback laws for distributed parameter systems. *IFAC-PapersOnLine*, *50*(1), 6786–6792, 20th IFAC World Congress.
Yu, H., & Krstic, M. (2019). Traffic congestion control for Aw-Rascle-Zhang model. *Automatica*, *100*, 38–51.
Yu, H., & Krstic, M. (2022). *Traffic congestion control by PDE backstepping*. Springer.

Miroslav Krstic is Sr. Assoc. Vice Chancellor for Research at UC San Diego. He is Fellow of IEEE, IFAC, ASME, SIAM, AAAS, and Serbian Academy of Sciences and Arts. He has received the Bode Lecture Prize, Bellman Award, Reid Prize, Oldenburger Medal, Nyquist Lecture Prize, Paynter Award, Ragazzini Award, IFAC Ruth Curtain Distributed Parameter Systems Award, IFAC Nonlinear Control Systems Award, IFAC Adaptive and Learning System Award, Chestnut textbook prize, AV Balakrishnan Award, CSS Distinguished Member Award, the PECASE, NSF Career, and ONR YIP, Schuck ('96 and '19) and Axelby paper prizes.

Luke Bhan received his B.S. and M.S. degrees in Computer Science and Physics from Vanderbilt University in 2022. He is currently pursuing his Ph.D. degree in Electrical and Computer Engineering at the University of California, San Diego. His research interests include neural operators, learning-based control, and control of partial differential equations.



Yuanyuan Shi is an Assistant Professor of Electrical and Computer Engineering at the University of California, San Diego. She received her Ph.D. in Electrical Engineering, masters in Electrical Engineering and Statistics, all from the University of Washington, in 2020. From 2020 to 2021, she was a postdoctoral scholar at the California Institute of Technology. Her research interests include machine learning, dynamical systems, and control, with applications to sustainable power and energy systems.