Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 404 (2023) 115810
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Kolmogorov n–width and Lagrangian physics-informed neural networks: A causality-conforming manifold for convection-dominated PDEs

Rambod Mojgani[a,*], Maciej Balajewicz[b], Pedram Hassanzadeh[a,c]

[a] *Department of Mechanical Engineering, Rice University, Houston, TX, United States of America*
[b] *Independent researcher, Boulder, CO, United States of America*
[c] *Department of Earth, Environmental and Planetary Sciences, Rice University, Houston, TX, United States of America*

Received 4 May 2022; received in revised form 21 November 2022; accepted 21 November 2022
Available online 19 December 2022

Dataset link: https://github.com/rmojgani/LPINNs/

Abstract

We make connections between complexity of training of physics-informed neural networks (PINNs) and Kolmogorov n-width of the solution. Leveraging this connection, we then propose Lagrangian PINNs (LPINNs) as a partial differential equation (PDE)-informed solution for convection-dominated problems. PINNs employ neural-networks to find the solutions of PDE-constrained optimization problems with initial conditions and boundary conditions as soft or hard constraints. These soft constraints are often blamed to be the sources of the complexity in the training phase of PINNs. Here, we demonstrate that the complexity of training (i) is closely related to the Kolmogorov n-width associated with problems demonstrating transport, convection, traveling waves, or moving fronts, and therefore becomes apparent in convection-dominated flows, and (ii) persists even when the boundary conditions are strictly enforced. Given this realization, we describe the mechanism underlying the training schemes such as those used in eXtended PINNs (XPINN), curriculum learning, and sequence-to-sequence learning. For an important category of PDEs, i.e., governed by non-linear convection–diffusion equation, we propose reformulating PINNs on a Lagrangian frame of reference, i.e., LPINNs, as a PDE-informed solution. A parallel architecture with two branches is proposed. One branch solves for the state variables on the characteristics, and the second branch solves for the low-dimensional characteristics curves. The proposed architecture conforms to the causality innate to the convection, and leverages the direction of travel of the information in the domain, i.e., on the characteristics. This approach is unique as it reduces the complexity of convection-dominated PINNs at the PDE level, instead of optimization strategies and/or schedulers. Finally, we demonstrate that the loss landscapes of LPINNs are less sensitive to the so-called "complexity" of the problems, i.e., convection, compared to those in the traditional PINNs in the Eulerian framework.

Keywords: Deep learning; Kolmogorov n-width; Partial differential equations; Method of characteristics; Lagrangian frame of reference

* Corresponding author.
E-mail address: rm99@rice.edu (R. Mojgani).

https://doi.org/10.1016/j.cma.2022.115810
0045-7825/

1. Introduction

The evolution of many physical phenomena and engineering systems can be derived from first principles leading to governing equations in the form of partial differential equations (PDEs). Although analytical solutions of many of non-linear PDEs are seldom known, development of numerical methods has made approximation of the solution possible. One of the paradigms of solving for a state of the system is through optimization. Consider a PDE

$$R\left(w\left(x,t\right)\right)=0, \tag{1}$$

where $R\left(\right)$ is a differential operator representing the residual of the PDE, $w\left(x,t\right)$ is the state parameter on a spatial domain of $x\in\Omega$ and $t\in[0,T]$ with appropriate boundary and initial conditions. Therefore, solving the PDE of Eq. (1), is equivalent to finding a minimizer, w^{*}, in

$$w^{*}=\underset{w}{arg\min}\ R\left(w\left(x,t\right)\right), \tag{2}$$

subject to boundary and initial conditions as constraints. Iterative methods are classically used to find minimizers of high-dimensional non-linear residual equations.

In the absence of the governing equations, where the phenomena/task cannot be described using first principles, machine learning (ML) methods such as artificial neural networks (ANNs) have been revolutionary. Moreover, the application of ANNs to solve the systems with known or partially known governing equations, inverse problems and data assimilation (DA) shows great promise [1,2]. While the compromise between incorporating *a priori* knowledge of the system and learning from data (experimental or observation) remains a problem dependent endeavor, setting the known governing equations as a loss component in ANNs nudges the network to a solution informed by the physics. This paradigm is matured in physics–informed neural networks (PINNs) [2], such that ANNs are trained to find the minimizer of (2), as well as the observational/experimental data to estimate the state parameter (direct problem) or the unknown parameters (inverse problem).

Although many of the traditional numerical solvers outperform PINNs in well-posed forward problems without DA, there are many critical applications where the use of PINNs is particularly advantageous [3]. PINNs provide flexible and scalable implementations [4], and readily provide adjoints via automatic differentiation. The low inference cost of PINNs and simultaneous estimation of the state parameters and independent parameters are particularly appealing. These advantages lead to successful use of PINNs, especially in inverse problems and inverse design [5], ill-posed/conditioned problems [6], and control [7].

However, the training phase of PINNs, which is equivalent to solving ordinary differential equations (ODEs) or PDEs, faces some practical challenges [8–16]. The innovations and attempts to improve the accuracy of the PINNs can be classified into two categories. In the first category, the ANN architecture and loss are targeted to improve the training behavior. In [10], it is shown that the eigenvalues of the Neural Tangent Kernels (NTKs) of different loss components explains the training behavior. Accordingly, penalty weights in the loss function are adaptability determined at each iteration of the training. Similarly in [11], the unbalanced gradients of the components of the loss is associated with training failure, and annealing the learning rate is proposed. It is also demonstrated that the architecture of the network can meaningfully change the stiffness of the gradients in the learning phase, and therefore it is suggested that a specialized architecture can be beneficial to specific problems [11]. Subsequently, reformulating the constraints using Augmented Lagrangian method (ALM) [17] demonstrates a flatter/smoother loss landscape, and therefore leads to a more favorable training behavior [15], compared to the originally proposed penalty terms [2]. Notably, the loss landscape of PINNs are less smooth compared to purely data-driven ANNs [12,15]. The raggedness of loss landscape explains why PINNs are more prone to converge to an unfavorable local minima. Such challenge depends on both the governing equations, and system parameters [8,12,15].

In the second category, prior knowledge/property of the system is leveraged. For instance, in hyperbolic systems, e.g., inviscid Burgers' equation, total variation diminishing (TVD), and entropy inequalities can be imposed in addition to artificial viscosity [18]. In [8], the flux term of non-linear convection diffusion had to be modified to help with the accuracy of the solution. Both of these remedies violate the consistency between the solution and the governing equations. However, such solutions are PDE specific, and are not applicable to all the challenging test cases. Approaches such as curriculum learning (training on a simple problem and transferring the learned weights to the harder problems) [12], adaptive sampling (in both space and time) [16], parallel in time decomposition [13] or the very similar sequence-to-sequence learning[1] (decomposing the temporal domain) [12] can be applied to a

[1] Different than the seminal work of Venugopalan et al. [19] in the context of video to text caption generation.

wider range of problems without any prior assumptions. Note that sequence-to-sequence learning [12] or parallel in time decomposition [13] here refer to the same method and should not be confused with PPINNs [20]. In many of the aforementioned studies, the same network is shown to be capable of expressing a more accurate solution, given additional attention is paid in defining the loss and the training phase. Such experiments demonstrate that the used architectures are expressive enough, and therefore, the challenge lies in the training. Unfortunately, many of the remedies require solving many intermediate subproblems [12], or training multiple networks [13]. It is argued that all such challenges can be more naturally overcome by respecting the underlying spatio-temporal *causality* [14]. One proposed approach is to impose such causality is by prioritizing the earlier time steps in the training phase by addition of temporal weights[14]. Causality training of [14] is successfully applied to Allen–Cahn equation and as an additional component of temporal decomposition to Kuramoto–Sivashinsky and Navier–Stokes equation ($Re = 100$). However, the proposed approach does not take into account the direction of travel of information.

While chaotic systems, e.g., turbulent flow, and Lorenz 63, and higher order derivatives in governing equations, has been recognized to be challenging since the earliest formulation of PINNs [14], they merely cannot explain the difficulty in training of regular non-chaotic systems, e.g., advection/convection, reaction, reaction–diffusion, Poisson's, or wave equations. Table 1 summarizes challenging test cases for training of accurate PINNs, the proposed remedies and their advantages and shortcomings. There main aspects of these methods are compared, i.e.,

1. Is the method applicable in a no-data regime? That is, can the method be applied to PINN when no training data is available?
2. Can a single ANN deliver a global solution in the full training regime, and without requiring to solve for intermediate problems, i.e., in one shot?
3. Does the solution of the PINN satisfy the governing equation of interest, i.e., does the method preserve consistency between the equations and the model?

Moreover, the type of the differential operators, i.e., parabolic, hyperbolic, or elliptic, is believed to describe the difficulty in discovery of the minima. In the case of elliptic and parabolic PDEs, the generalization error, i.e., the difference between a global minimizer of the loss and the solution to the PDEs, converges to zero under certain conditions, given enough number of data points [25]. However, similar results are lacking for hyperbolic equations. More importantly, the optimization error, i.e., the difference between the global and local minima given some data points, is poorly understood. This is especially important as many of the non-chaotic, yet challenging cases in Table 1 are either hyperbolic or hyperbolic–parabolic (where hyperbolic traits are dominant), the main focus of the present paper.

Although the previous studies have described some of the difficulties and dynamics of the training phase, there is no universal theory on convergence rate or *a priori* measure of success of the training [26]. In this paper, we provide some evidence that connects the dimensionality of solution, in the Kolmogorov n–width sense, to the difficulties in the training phase. Moreover, we explain how the previous remedies connects to the presented description of complexity. Subsequently, we add an unrecognized complex test case, to the existing list of Table 1, i.e., the Burgers' equation in the presence of a shock sweeping the domain (traveling shock). Specifically, we demonstrate that in the cases where the shock sweeps long distances, the training phase has a similar complexity of training as in convection equation. We emphasize that PINNs can easily solve the viscous Burgers' equation with stationary shocks [2]. Finally, we emphasize that most of the methods in Table 1, e.g., [12,13,23,24], are not demonstrated for the problem of our interest, i.e., convection-dominated problems. Moreover, the proposed methods are algorithmically more complex than the original PINN [2], and require multiple ANNs or training passes (solution of subproblems). We propose an approach that can reach the solution in one shot of training and without requiring solutions of intermediate problems.

In this paper, we focus on an important category of the recognized challenging cases, i.e., convection–diffusion problems. We propose LPINNs which conforms to the "causality" in the system. LPINNs' architecture is informed by the direction of travel of information in the domain, i.e., along the characteristic curves. In this architecture the solution is to be learned with an inherently reduced dimensionality, a simpler task for any of ML architectures. The proposed approach successfully learns the solution with a single ANN, in a no-data regime, in one shot and without requiring to learn any intermediate solutions. The solution is consistent with the governing PDEs on the Eulerian frame, giving PINNs an advantage over methods introducing artificial viscosity.

Table 1
Challenging problems in the training of PINNs and the suggested remedies. Only some of these methods have been demonstrated to improve the training of convection-dominated problems (marked by ‡).

Study	PDEs	Domain	Test case	Proposed remedy	No-data regime?	Trained in one shot?	Consistent?
[8]	Convection–diffusion	1D	Buckley–Leverett	Introducing viscosity	✓	✗	✗
[16]	Reaction–diffusion Adsorption/ desorption – surface diffusion	1D 2D 3D	Allen–Cahn Cahn–Hilliard	Time adaptive (sampling/marching), and mini-batching, and regularizing the loss components	✗ ✓ ✓	✗ ✓ ✓	✓ ✓ ✓
[9]	Euler	1D	Sod's shock tube	Characteristic form, Oversampling the shock	✓ ✗	✓ ✓	✓ ✓
[21]	Convection–diffusion	1D	Buckley–Leverett	Tuning the flux term	✓	✗	✗
[11]	Helmholtz Klein–Gordon Navier–Stokes	2D 1D 2D	N/A N/A Lid-driven cavity ($Re = 100$)	Annealing the learning rate	✓	✓	✓
[13]‡	Advection Shallow-water	Spherical	Traveling feature	Parallel in time decomposition	✓	✗	✓
[12]‡	Convection Reaction Reaction–diffusion	1D	Moving interfaces, traveling features	Curriculum learning, Sequence-to-sequence learning	✓ ✓	✗ ✗	✓ ✓
[22]	Inviscid Burgers' Convection–diffusion	1D	Shock Rarefaction and shock	Enriching the data-set, artificial viscosity	✗ ✓	✓ ✗	✓ ✗
[10]	Wave	1D	Traveling wave	Adaptive penalty	✓	✓	✓
[14]	Chaotic ODE Reaction–diffusion Kuramoto–Sivashinsky Navier–Stokes	$\mathbb{R}^3$ 1D 1D 2D	Lorenz 63 Allen–Cahn Decaying turbulence ($Re = 100$)	Causal training of loss, and/or transformer architecture	✓	✓	✓
[15]	Poisson's	1D 2D	High wave–number forcing	Enforcing the constraints by ALM	✓	✓	✓
[18]	Euler Convection–diffusion	1D	Sod's shock tube Buckley–Leverett	TVD, and entropy inequalities	✓	✓	✓
[23]‡	Convection Reaction Reaction–diffusion	1D	Moving interfaces, traveling features	Ensemble training	✓	✗	✓
[24]‡	Convection Reaction–diffusion	1D	Traveling features, Allen Cahn	Evolutionary sampling (Evo)	✓	✗	✓
Ours‡	Convection–diffusion Viscous Burgers'	1D 2D	Traveling feature, Traveling shock	Lagrangian PINN	✓	✓	✓

The paper is organized as follows. We first summarize PINNs in Section 2. In Section 3, some of the challenging cases of training of PINNs are discussed. More importantly, we for the first time, explain and demonstrate the mechanism leading to the complexity in training through the lens of approximation theory. Our explanation motivates the proposed LPINNs in Section 4, where we focus on an important canonical set of problems governed by non-linear convection–diffusion equation, in one-dimensional (1D) and 2D domains. The proposed approach is numerically investigated in Section 5. The properties and possible foreseeable challenges of the proposed LPINNs are discussed in Section 6, followed by our conclusions in Section 7.

2. PINNs

A PINN architecture is composed of a densely connected ANN that minimizes a composite loss comprised of the PDE of Eq. (1) evaluated at the spatiotemporal collocation points, data, and the initial and boundary points [2]. The output of the ANN given ith coordinate of a spatiotemporal grid, $\mathcal{N}(\boldsymbol{u})$, is the local and instantaneous state parameter, $\boldsymbol{w}(\boldsymbol{x}_i, t_i)$, i.e.,

$$\boldsymbol{w}(\boldsymbol{x}_i, t_i) = \mathcal{N}(\boldsymbol{u}) = \phi_m\left(\boldsymbol{W}_m \phi_{m-1}\left(\boldsymbol{W}_{m-1} \cdots \phi_1\left(\boldsymbol{W}_1 \boldsymbol{u} + \boldsymbol{b}_1\right) \cdots + \boldsymbol{b}_{m-1}\right) + \boldsymbol{b}_m\right), \tag{3}$$

where the input vector is the concatenation of the spatial and temporal location, i.e., $\boldsymbol{u} = [\boldsymbol{x}_i, t_i]^\top \in \mathbb{R}^{d+1}$, d is the dimension of physical space, $\phi_i(.)$ is the activation function at the ith-layer, $\boldsymbol{W}_1 \in \mathbb{R}^{w \times (d+1)}$, $\boldsymbol{W}_i \in \mathbb{R}^{w \times w}, \forall i \in \{2, \ldots, m-1\}$, and $\boldsymbol{W}_w \in \mathbb{R}^{d \times w}$ are the weights, and $\boldsymbol{b}_i \in \mathbb{R}^w, \forall i \in \{1, \ldots, m-1\}$, and $\boldsymbol{b}_m \in \mathbb{R}^d$ are biases. The weights and biases are learned to minimize the physics-informed loss, minimizing the governing equation and the appropriate boundary and initial conditions, i.e.,

$$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_{bc} + \mathcal{L}_{ic}, \tag{4}$$

where

$$\mathcal{L}_r = \lambda_r \frac{1}{N_r} \sum_{i=1}^{N_r} \left| R\left(\boldsymbol{x}_i, t_i\right) \right|_2, \tag{5a}$$

$$\mathcal{L}_{bc} = \lambda_{bc} \frac{1}{N_{bc}} \sum_{i=1}^{N_{bc}} \left| \mathcal{B}[w]\left(\boldsymbol{x}_{bc}^i, t_{bc}^i\right) \right|_2, \tag{5b}$$

$$\mathcal{L}_{ic} = \lambda_{ic} \frac{1}{N_{ic}} \sum_{i=1}^{N_{ic}} \left| \boldsymbol{w}(\boldsymbol{x}, 0) - g\left(\boldsymbol{x}_{ic}^i\right) \right|_2, \tag{5c}$$

and $\left\{t_r^i, \boldsymbol{x}_r^i\right\}_{i=1}^{N_r}$ is the set of temporal and spatial coordinates of the collocation points where the residual is evaluated, $\left\{t_{bc}^i, \boldsymbol{x}_{bc}^i\right\}_{i=1}^{N_{bc}}$ is a set of temporal and spatial coordinates of the boundary points ($\mathcal{B}[w]\left(\boldsymbol{x}_{bc}^i, t_{bc}^i\right)$), and $\left\{\boldsymbol{x}_{ic}^i\right\}_{i=1}^{N_{ic}}$, is the set of coordinates where the initial condition is known ($g\left(\boldsymbol{x}_{ic}^i\right)$ at $t = 0$). The partial derivatives in the residual operator, $R\left(\boldsymbol{x}_i, t_i\right)$, are calculated using automatic differentiation (AD). The hyper-parameters λ_r, λ_{bc}, and λ_{ic} are scalars tuned to improve the convergence. Augmenting the loss using more data points leads to faster convergence, especially in convection-dominated problems [22]. However, we intentionally refrain from using any data points to evaluate the convergence of PINNs as a solver (no-data regime).

In this paper, without loss in generality, we limit the spatial domain to 1D and 2D problems. The periodic boundary condition is strictly enforced using a custom layer [13], i.e., hard constraint, and therefore $\lambda_{bc} = 0$. The custom layer in $x \in [0, 2\pi]$ transforms the domain to a polar coordinate, i.e.,

$$\gamma(x) = [\cos(x), \sin(x)]^\top, \tag{6}$$

and for 2D domains, i.e., $(x_1, x_2) \in [0, 2\pi] \times [0, 2\pi]$ transforms the domain to a doubly periodic polar coordinate, i.e.,

$$\gamma(x_1, x_2) = [\cos(x_1), \sin(x_1), \cos(x_2), \sin(x_2)]^\top, \tag{7}$$

and out-put of this layer is fed into the traditional ANN as described in Eq. (3) with appropriate adjustment of the dimension of the weight of the first layer, i.e., $\boldsymbol{W}_1 \in \mathbb{R}^{w \times (2d+1)}$, increasing the number of network variables by only $w \times d$. Further discussion of strictly enforcing the boundary conditions can be found in [27]. The described architecture is illustrated in Fig. 1.

3. Kolmogorov n–width and convergence of PINNs

In this section, we connect the dimensionality of the problem to the training difficulties in PINNs. Specifically, we provide numerical evidence to connect the decay of singular values of the snapshots to the difficulties in the training phase, where the slow decay of singular values corresponds to transport phenomena, convection, traveling waves, and moving fronts. Although limited attempts were made to quantify connection of dimensionality and slow

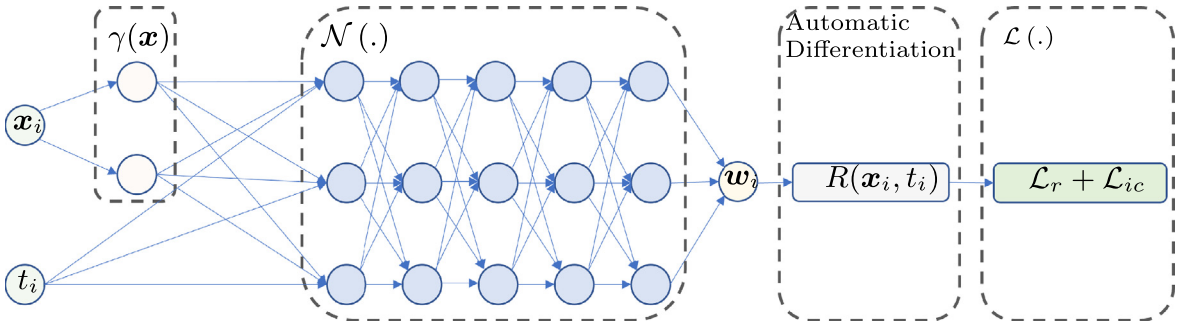


Fig. 1. The traditional PINNs architecture with periodic boundary condition satisfied via a custom layer (hard constraint).

convergence of some specific classes of ML architectures [28,29], many of the questions regarding the choice of activation functions, norms, and architecture remains open [28]. Formal investigations of such questions are out of the scope of the present paper.

In approximation theory, Kolmogorov n–width is a measure of how close n-dimensional subspaces can approximate the solution manifold, $\mathcal{M}$ [30]. The following definitions briefly explains this measure [31].

Definition. Let $\mathcal{M}$ be a normed linear space and $\widetilde{\mathcal{M}}_n$ any n-dimensional subspace of $\mathcal{M}$. For each $x \in \mathcal{M}$, $\delta\left(x, \widetilde{\mathcal{M}}_n\right)$ shall denote the distance of the n-dimensional subspace $\widetilde{\mathcal{M}}_n$ from x, defined by

$$\delta\left(x; \widetilde{\mathcal{M}}_n\right) = \inf\left\{\|x - y\|_X : y \in \widetilde{\mathcal{M}}_n\right\}, \tag{8}$$

where $\|.\|_X$ denotes any arbitrary choice of norms. If there exists a $y^* \in \widetilde{\mathcal{M}}_n$ for which $\delta\left(x, \widetilde{\mathcal{M}}_n\right) = \|x - y^*\|$, then y^* is the best approximation of x from $\widetilde{\mathcal{M}}_n$. Extending the concept from a single element of x to $\mathcal{S}$, a given subset of $\mathcal{M}$, the deviation of $\mathcal{S}$ from $\widetilde{\mathcal{M}}_n$ is defined as

$$\delta\left(\mathcal{S}; \widetilde{\mathcal{M}}_n\right) = \sup_{x \in \mathcal{S}} \inf_{y \in \widetilde{\mathcal{M}}_n} \|x - y\|, \tag{9}$$

representing the worst element of $x \in \mathcal{S}$ approximated in $\widetilde{\mathcal{M}}_n$.

Definition. Kolmogorov n–width of $\mathcal{M}$, $d_n(\mathcal{M})$, is defined as

$$d_n(\mathcal{S}; \mathcal{M}) := \inf_{\widetilde{\mathcal{M}}_n} \delta\left(\mathcal{S}; \widetilde{\mathcal{M}}_n\right), \tag{10}$$

where the infimum is taken over all n-dimensional subspaces $(\widetilde{\mathcal{M}}_n)$ of the state space, $\mathcal{M}$.

In the context of Petrov–Galerkin projection schemes, n–width correlates with the best achievable rate of convergence for a given set of snapshots [32]. The connection of rate of decay of singular values of the snapshots to accuracy of approximation in linear subspaces are established [33–35]. The extension of such results to non-linear manifolds, such as those discovered through training of PINNs, has not been identified yet. Here, we numerically evaluate whether the rate of decay of singular values could be used as a simple *a priori* guideline to the complexity of the training phase of PINNs.

Krishnapriyan et al. [12] have characterized complexities of training of PINNs for some canonical problems. They argue that the condition number of the governing equation as the regularization term is one of the origins of the difficulties in the training of PINNs in specific regimes. [2] Here, we revisit these cases and introduce the rate of decay of singular values of the solution as a robust and predictive indicator of difficulties in training of PINNs. Based on our findings, we predict Burgers' equation, in convection-dominated regimes, as an additional challenging test case for PINNs.

[2] We show that the so-called "failure modes" are more probable for optimization using L-BFGS and are less frequent with Adam optimizer. Regardless of the choice of optimizer, it is harder to train PINNs for higher convection speeds. A detailed and probabilistic investigation of convergence of PINNs for convection and its sensitivity to optimization scheme, sampling of collocation points, hyper-parameters of the composite loss, and soft versus hard enforcing of the periodic boundary condition is presented in Appendix A. Subsequently, we demonstrate that traditional PINNs are slower to train in the presence of high convection, moving features, and surfaces.

Table 2
Canonical problems with high Kolmogorov n–width complexity. In all cases, the domain is defined on $[0, 2\pi] \times [0, 1]$. $\mathcal{F}(.)$, $\mathcal{F}^{-1}(.)$, and κ respectively denote Fourier transform, its inverse, and wave number in the Fourier space.

	Canonical problem	PDE	Initial condition, $w(x,0)$	Boundary condition	Solution, $w(x,t)$
1	Convection (Introduced in [12])	$\frac{\partial w}{\partial t} + c\frac{\partial w}{\partial x} = 0$	$\sin(x)$	$\boldsymbol{w}(0,t) = \boldsymbol{w}(2\pi,t)$	$\mathcal{F}^{-1}\left(\mathcal{F}(w(x,0))\, e^{-ick t}\right)$[12]
2	Reaction (Introduced in [12])	$\frac{\partial w}{\partial t} - \rho w(1-w) = 0$	$\exp\left(-\frac{(x-\pi)^2}{2(\pi/4)^2}\right)$	$\boldsymbol{w}(0,t) = \boldsymbol{w}(2\pi,t)$	$\frac{w(x,0)\, e^{\rho t}}{w(x,0)\, e^{\rho t} + 1 - w(x,0)}$ [12]
3	Reaction–diffusion (Introduced in [12])	$\frac{\partial w}{\partial t} - \nu\frac{\partial^2 w}{\partial x^2} - \rho w(1-w) = 0$	$\exp\left(-\frac{(x-\pi)^2}{2(\pi/4)^2}\right)$	$\boldsymbol{w}(0,t) = \boldsymbol{w}(2\pi,t)$	Strang splitting [12], i.e., i. $\frac{w(x,0)e^{\rho t}}{w(x,0)e^{\rho t} + 1 - w(x,0)}$ ii. $\mathcal{F}^{-1}\left(\mathcal{F}(w(x,0))\, e^{-\nu\kappa^2 t}\right)$
4	Convection–diffusion (Similar to 1)	$\frac{\partial w}{\partial t} + c\frac{\partial w}{\partial x} = \nu\frac{\partial^2 w}{\partial x^2}$	$\sin(x)$	$\boldsymbol{w}(0,t) = \boldsymbol{w}(2\pi,t)$	$\mathcal{F}^{-1}\left(\mathcal{F}(w(x,0))\, e^{-\nu\kappa^2 t} e^{-ick t}\right)$
5	Burgers' (Our contribution)	$\frac{\partial w}{\partial t} + w\frac{\partial w}{\partial x} = \nu\frac{\partial^2 w}{\partial x^2}$	$\sin(x) + c$	$\boldsymbol{w}(0,t) = \boldsymbol{w}(2\pi,t)$	Fourier Pseudo–Spectral

In Figs. 2 to 5, the snapshots of the solution and the corresponding singular value decays are plotted for given snapshots of convection equation, reaction equation, reaction–diffusion equation, and Burgers' equation where the state variable $w = w(x,t)$ is in the domain $(x,t) \in [0, 2\pi] \times [0, 1]$, and the PDEs are equipped with initial conditions $w(x,0) = w_0(x)$, and periodic boundary conditions, as summarized in Table 2. Convection speed, viscosity, and reaction coefficient are denoted by c, ν, and ρ, respectively. Note that to introduce *traveling shocks* in the Burgers' equation, the initial condition is offset by c.

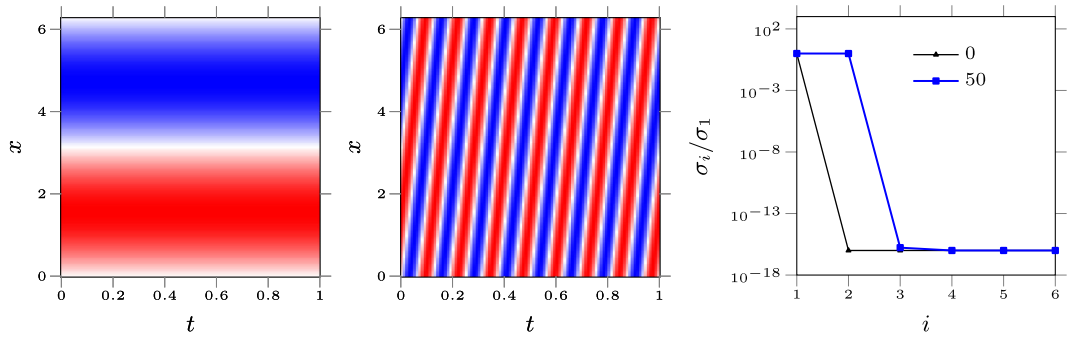
In each of these experiments, the solution of the governing equation is depicted in the space–time domain, and the corresponding singular value spectra are also compared, where σ_i is the ith singular value of the snapshots of the solution.

Krishnapriyan et al. [12] showed that PINNs are harder to train for higher convection speeds (so-called "failure modes"). Subsequently, the cases with $c = 0$ and $c = 50$ are compared in Fig. 2.It is clear that σ_2/σ_1 increases quickly as c increases. For the reaction equation, the decay of singular values becomes slower as ρ increases, in a similar trend, the PINNs become more difficult to train. For the cases of reaction–diffusion in [12], all cases show similar slow rate of decay of singular values, and in a similar trend, training in all the cases encounters difficulties. Finally, we consider Burgers' equation, a case successfully solved in early studies of PINNs [2]. However, in that case, a viscous shock forms and collapses on its place, without *traveling* in the domain. In this paper, to impose the shock moving through the domain, the initial condition is offset from zero. Similar to the convection case, the rate of decay of singular values decreases as the speed of the shock is increased. In such cases, the shock sweeps the domain before collapsing/diffusing. Subsequently, similar challenges in training of PINNs for Burgers' equation in convection-dominated regimes is observed.

These experiments suggest that although ANNs can express any solutions on non-linear manifolds, the dimensionality of the solution on the linear optimal subspace of singular vectors can still inform the convergence behavior of PINNs, for the same system of governing equations. However, the singular value spectra between two different governing equations do not explain difficulty of the training, e.g., collapsing shock has a slower rate of decay of singular values compared to convection with high speed and yet the training phase is well-behaved. Nevertheless, in all cases the network is hard to train at the presence of traveling features, such as shock, fronts, and gradients.

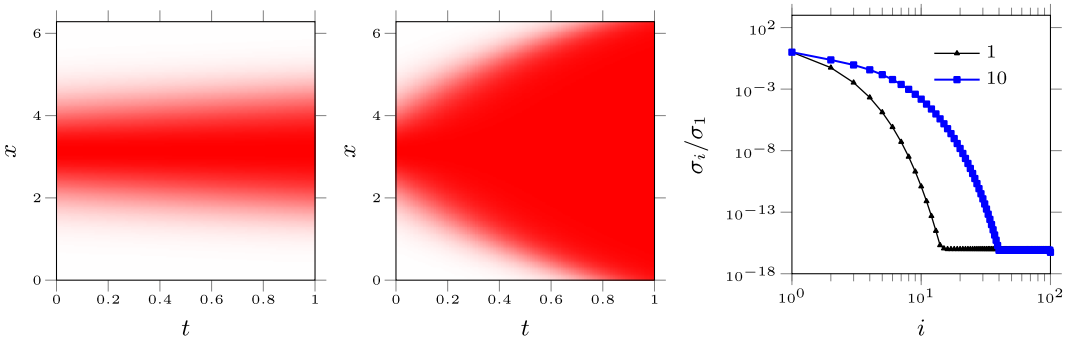
Although the connection of Kolmogorov n–width and training of PINNs is lacking from the literature, some of the successful remedies in training of PINNs [9,12,13] can be explained by reducing the Kolmogorov n–width, an active research subject in finite element method of solving PDEs [36,37], DA [38], neural networks (NNs)–based reduced order models (ROMs) [39–41], projection-based ROMs [37,42–50], flexDeepONet [51], Gaussian Process Hydrodynamics [52], and projection–based ROMs on NN–based manifolds [53,54].

Here, we apply some of the proposed remedies to enhance convergence of traditional PINNs [4,12,13] to synthetic snapshots, and demonstrate the connection between the proposed remedies and the rate of decay of (normalized) singular values. We compare (i) random and uniformly sampling of the domain, (ii) random but weighted sampling



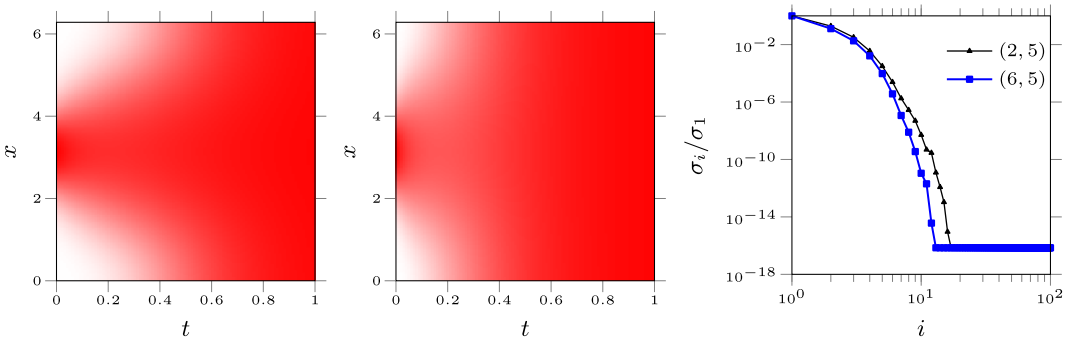
(a) $c = 0$ (b) $c = 50$ (c) Singular value spectra

Fig. 2. Convection equation. In 2(a) and 2(b), dark red and dark blue represent $+1$ and -1, respectively. In 2(c), the black triangle and blue rectangle markers represent $c = 0$ (fast convergence), and $c = 50$ (slow convergence), respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



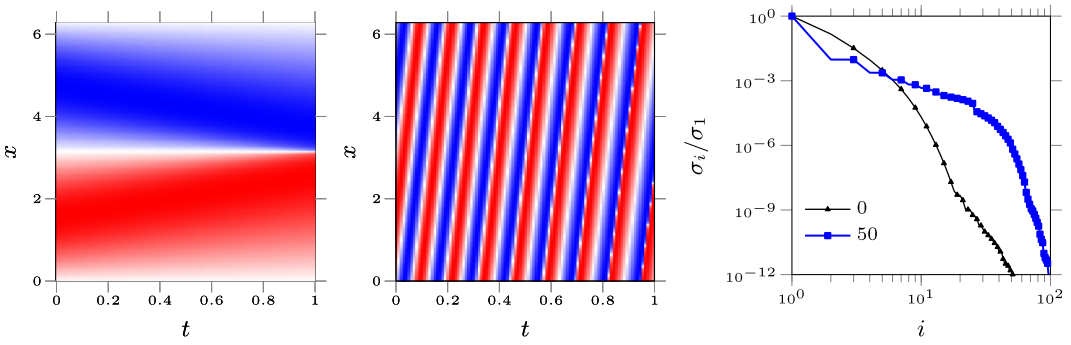
(a) $\rho = 1$ (b) $\rho = 10$ (c) Singular value spectra

Fig. 3. Reaction equation. In 3(a) and 3(b), dark red and white represent $+1$ and 0, respectively. In 3(c), the black triangle and blue rectangle markers represent $\rho = 1$ (fast convergence), and $\rho = 10$ (slow convergence), respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $(\nu, \rho) = (2, 5)$ (b) $(\nu, \rho) = (6, 5)$ (c) Singular value spectra

Fig. 4. Reaction–diffusion equation. In 4(a) and 4(b), dark red and white represent $+1$ and 0, respectively. In 4(c), the black triangle and blue rectangle markers represent $(\nu, \rho) = (5, 2)$ (slow convergence), and $(\nu, \rho) = (6, 5)$ (slow convergence), respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $c = 0$ (b) $c = 50$ (c) Singular value spectra

Fig. 5. Burgers' equation. In 5(a) and 5(b), dark red and dark blue represent $c+1$ and $c-1$, respectively. In 2(c), the black triangle and blue rectangle markers represent $c = 0$ (fast convergence), and $c = 50$ (slow convergence), respectively. In all cases, $\nu = 0.01$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

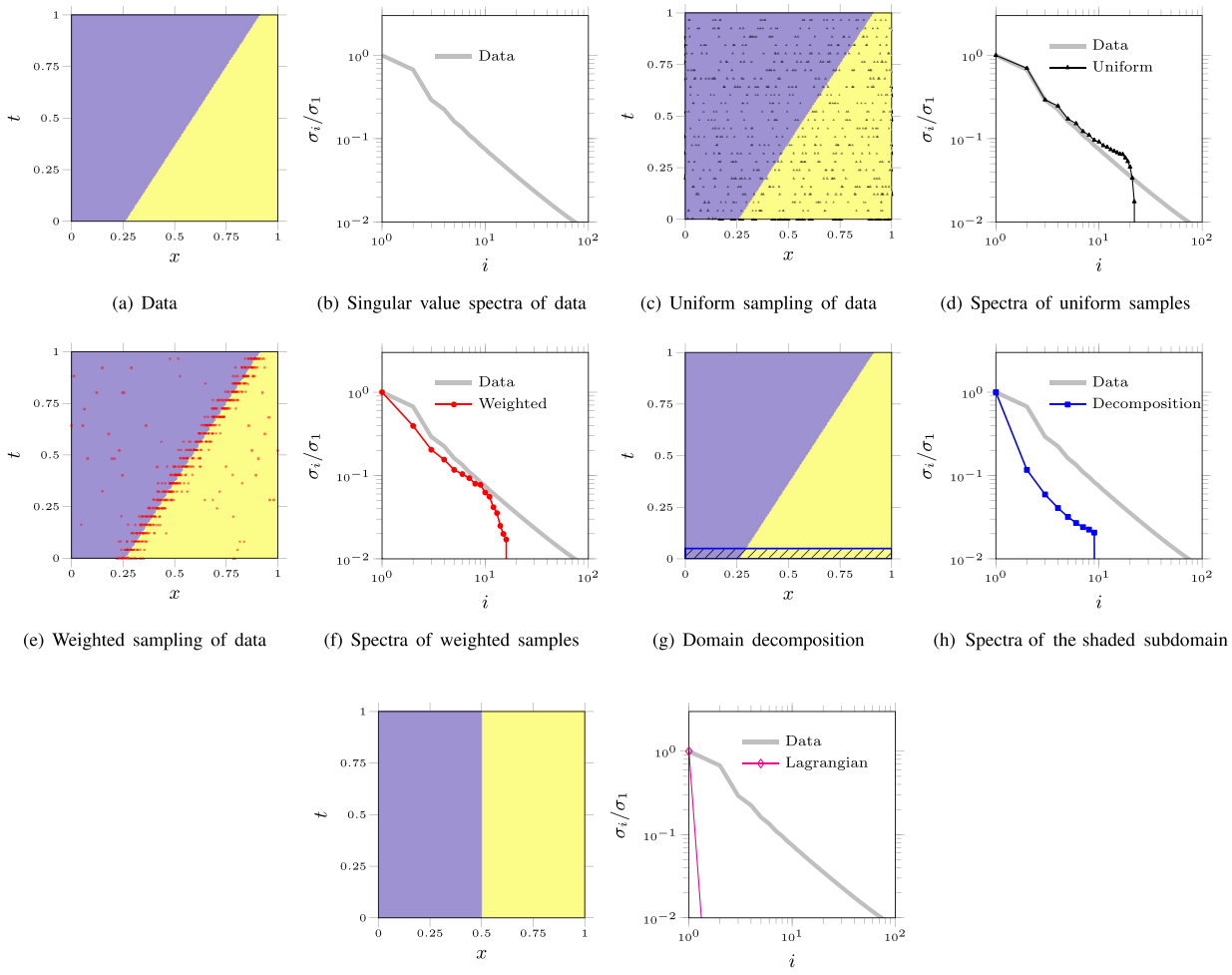
of the domain similar to [9], and (iii) decomposition of the temporal domain similar to [12,13]. Some of the aforementioned methods in training of PINNs for convection equation are compared with our proposed LPINNs in Appendix B. In this section, we focus on a representative problem to evaluate such remedies through the lens of Kolmogorov n–width. Consider a synthetic traveling shock/interface (Fig. 6(a)). The snapshot is constructed on $N_x = 256$ spatial grid points and $N_t = 500$ time steps. The spectra of singular values of the full data, in Fig. 6(b), shows that $\mathcal{O}\left(10^2\right)$ bases are required to reconstruct the data to the error of $\mathcal{O}\left(10^{-2}\right)$ on the subspace spanned by singular vectors.

Firstly, at each time level 25 data points are sampled randomly and uniformly. As in Fig. 6(b), the sampled data has a lower rank compared to the full data, demonstrating the distance in accuracy of approximation of the full data. However, the leading singular values remains unchanged (a property used in randomized singular value decomposition).

Secondly, the data points are selected randomly but weighted such that the probability of sampling is proportional to the absolute value of the gradient of the data. Specifically, `randsample` function in MATLAB [55] is used, where the weight vector is the absolute value of the difference between two consecutive data points. In this case, the rate of decay of singular values is increased, reducing the approximation error given the same number of singular vectors.

Thirdly, the temporal domain is decomposed by selecting 25 consecutive time steps (i.e., the domain is divided into 20 consecutive subdomains, Fig. 6(g)), significantly increasing the rate of decay of singular values as in Fig. 6(h). This strategy is effectively equivalent to parallel-in-time decomposition [13] and sequence-to-sequence learning [12], where the temporal domain is decomposed into short time intervals. For an effective reduction in the Kolmogorov n–width, the subdomains' time horizon are very limited, leading to an inefficient and repeated training in PINNs. For example in the case of our synthetic data, 20 different networks/training passes are required to cover the full temporal domain. This strategy is similar to principal interval decomposition (PID) (in linear subspace), applied to ROMs [47] and Long short-term memory (LSTM) networks [40], and reduces the Kolmogorov n–width of the data at the cost of localizing of the solution. Subsequently, increasing the temporal domain, decreases the rate of decay. Decomposition of the computational domain, in both space and time, is possible using eXtended PINNs (XPINNs), originally developed to tackle the scalability of PINNs [4].

Given the numerical evidence in this section, one paradigm to tame the training of PINNs is to reformulate the problem on a manifold such that Kolmogorov n–width is decreased, or equivalently, the rate of decay of the singular values is increased. For non–linear convection–diffusion flows, where convection dominates diffusion, such goal is achievable by reformulating the governing equations on the characteristics curves [56]. Accordingly, we expect a similar strategy, when applicable, can reduce the challenges in training of PINNs. We reconsider the synthetic data in Fig. 6. When the characteristics are followed or the observer is re-framed on the traveling shock Fig. 6(a) is transformed into a stationary shock of Fig. 6(i). Consequently, the rank of the transformed snapshots is exactly equal to 1, Fig. 6(j). In the cases where the transformed state parameter does not remain constant, e.g., in the presence



(a) Data (b) Singular value spectra of data (c) Uniform sampling of data (d) Spectra of uniform samples

(e) Weighted sampling of data (f) Spectra of weighted samples (g) Domain decomposition (h) Spectra of the shaded subdomain

(i) Data on Lagrangian frame (j) Spectra of data on Lagrangian frame

Fig. 6. A comparison of different methods reducing the Kolmogorov n–width. The synthetic data depicts a traveling shock, Fig. 6(a), and Fig. 6(b) is the corresponding singular value spectra. Figures 6(c) to 6(d) represent uniform sampling (black triangles), and Figs. 6(e) to 6(f) represent weighted sampling (red circles), Figs. 6(g) to 6(h) represent the effect of domain decomposition sampling (red circles), b. The corresponding singular value spectra of the uniformly sampled data (black triangles), the weighted sampled data (blue squares and shaded subdomain). Finally, Fig. 6(g) depicts the data represented on a frame following the traveling shock (i.e., Lagrangian frame), and Fig. 6(h) shows the optimal reduction of rank to 1. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of non–linear convection, or diffusion, the rank of the transformed snapshots is greater than 1, nevertheless, it is significantly smaller compared to the snapshots in the original, Eulerian frame of reference. See [56] for a more detailed discussion.

4. Proposed Lagrangian PINNs for non-linear convection–diffusion

Consider the following scalar, one-dimensional convection–diffusion equation

$$R := \frac{\partial w(x,t)}{\partial t} + f_1(x,t,w)\frac{\partial w(x,t)}{\partial x} - f_2(x,t,w)\frac{\partial^2 w(x,t)}{\partial x^2} = 0, \tag{11}$$

in the domain $(x,t) \in [x_a, x_b] \times [0, T]$, with initial conditions $w(x,0) = w_0(x)$, and appropriate boundary conditions at x_a and x_b.

As motivated in Section 3, to address the complexity of the training, the governing equation Eq. (11) is reformulated in the Lagrangian frame of reference, i.e.,

$$R_\chi := \frac{d\chi(t)}{dt} - f_1(w(\chi(t), t)) = 0, \tag{12a}$$

$$R_w := \frac{\partial w(x,t)}{\partial t}|_{x=\chi(t)} - f_2(x,t,w)\frac{\partial^2 w(x,t)}{\partial x^2}|_{x=\chi(t)} = 0. \tag{12b}$$

where χ denotes the characteristics, and w is the state variable on the characteristics.

Here, we describe the additional changes to the architecture of the traditional PINNs to conform with the Lagrangian formulation to satisfy Eq. (12). We propose a parallel architecture comprised of two branches, corresponding to the trajectory of the characteristics and the state parameter on the characteristics, respectively denoted by $\mathcal{N}_\chi(\boldsymbol{u})$ and $\mathcal{N}_w(\boldsymbol{u})$. The output of the first branch is the trajectory of the characteristic curves, i.e.,

$$\boldsymbol{\chi}(\boldsymbol{x}_i, t_i) = \mathcal{N}_\chi(\boldsymbol{u}) = \phi_m(\boldsymbol{W}_m\phi_{m-1}(\boldsymbol{W}_{m-1}\cdots\phi_1(\boldsymbol{W}_1\boldsymbol{u}+\boldsymbol{b}_1)\cdots+\boldsymbol{b}_{m-1})+\boldsymbol{b}_m), \tag{13}$$

and the second branch outputs the state parameter on the characteristics curves, i.e.,

$$\boldsymbol{w}(\boldsymbol{x}_i, t_i) = \mathcal{N}_w(\boldsymbol{u}) = \phi_m(\boldsymbol{W}_m\phi_{m-1}(\boldsymbol{W}_{m-1}\cdots\phi_1(\boldsymbol{W}_1\boldsymbol{u}+\boldsymbol{b}_1)\cdots+\boldsymbol{b}_{m-1})+\boldsymbol{b}_m), \tag{14}$$

where all the parameters are defined similar to the network in Section 2. The two branches can be of different width, and depth. The output of the network is the state parameter on the characteristics curves minimizing the loss, i.e.,

$$\mathcal{L} = \mathcal{L}_{r_\chi} + \mathcal{L}_{r_w} + \mathcal{L}_{ic}, \tag{15}$$

where $\mathcal{L}_{r_\chi}$, and $\mathcal{L}_{r_\chi}$ are the loss associated with the residuals in Eq. (12), i.e.,

$$\mathcal{L}_{r_\chi} = \lambda_{r_\chi}\frac{1}{N_r}\sum_{i=1}^{N_r}\left|R_\chi(\boldsymbol{x}_i, t_i)\right|_2, \tag{16a}$$

$$\mathcal{L}_{r_w} = \lambda_{r_w}\frac{1}{N_r}\sum_{i=1}^{N_r}|R_w(\boldsymbol{x}_i, t_i)|_2, \tag{16b}$$

and $\mathcal{L}_{ic}$ is the loss associated with initial condition of both the state and grid, i.e.,

$$\mathcal{L}_{ic} = \lambda_{ic}\frac{1}{N_{ic}}\sum_{i=1}^{N_{ic}}\left(\left|\boldsymbol{w}(\boldsymbol{x},0) - g_w(\boldsymbol{x}_{ic}^i)\right|_2 + \left|\chi(\boldsymbol{x},0) - g_\chi(\boldsymbol{x}_{ic}^i)\right|_2\right), \tag{17}$$

where $g_w(\boldsymbol{x}_{ic}^i)$ and $g_\chi(\boldsymbol{x}_{ic}^i)$ are the known initial conditions for the state and grid, respectively. The proposed architecture is depicted in Fig. 7.

The output of the proposed network is the state variable on the Lagrangian frame, i.e., $\boldsymbol{w}(\chi(t), t)$. The solution on the Lagrangian frame of reference is consistent with the solution on the Eulerian frame. Finally, one can interpolate the states from the Lagrangian to the Eulerian frame of reference.

We recognize the residual equations in Eq. (12) can also be solved in an architecture similar to that of the traditional PINNs. However, the proposed two-branch architecture leverages the inherent low-dimensionality of the characteristics [56], and employ a shallow and compact network to solve Eq. (12a).

In the cases where the characteristics are independent of the state variable, e.g., linear convection where $d\chi(t)/dt = c$ in Eq. (12a), the equations for state and characteristics are decoupled. Subsequently, the two branches of LPINNs can be trained separately. In this paper, we use the same coupled architecture of Fig. 7 for all discussed problems.

The proposed LPINNs reformulation and architecture is causality conforming. Characteristics are curves on which the state variables are governed by ordinary differential equations. Therefore, the evolution of the state variable is only "influenced" or "caused" by the state on the corresponding characteristics curve. In other words, the evolution of the state variables in time is only influenced/caused by their history and on the corresponding characteristics, i.e., the domain of dependence. Similarly, the current state can only influence/cause the future state on the corresponding characteristic curve, i.e., the region of influence. Therefore, by redefining the state variable on the curves, the direction of travel of information is by construction enforced.

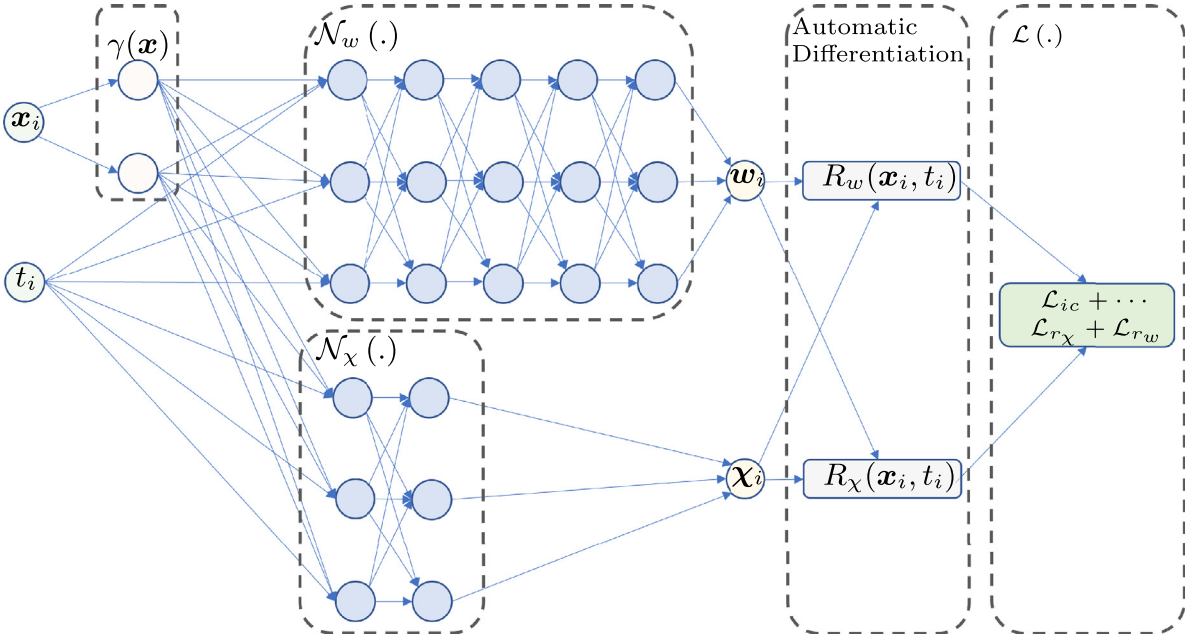


Fig. 7. The proposed LPINNs architecture with periodic boundary condition.

5. Numerical Experiments

In this section, the traditional PINN and the proposed LPINN are compared. The equidistant collocation points in the spatio-temporal are of size $(N_x, N_t) = (256, 100)$ for $(x, t) \in [0, 2\pi] \times [0, 1]$. In the convection and convection–diffusion cases, the initial condition is $w_0 = \sin(x)$, and the solutions in Table 2 are considered as the truth. In the Burgers' case, the initial condition is $w_0 = \sin(x) + c$.

In all cases, the PINNs have 4 hidden-layers, and the LPINNs have an additional shallow branch with 2 hidden-layers. All the hidden layers have 50 neurons. All activation functions are $\phi(.) = \tanh(.)$, except those of the last layer, where activation is linear (identity). The hyper-parameters in Eq. (4) of PINNs are $[\lambda_r, \lambda_{ic}, \lambda_{bc}] = [1, 15, 0]$ (hard constraint for the periodic boundary), and are chosen via the explorations described in Appendix A. Similarly, the hyper-parameters in Eq. (15) of LPINNs are $\left[\lambda_{r_w}, \lambda_{r_\chi}, \lambda_{ic}, \lambda_{bc}\right] = [1, 10, 1000, 0]$ (hard constraint for the periodic boundary), unless otherwise stated. Adam optimizer [57] with 2×10^4 iterations and learning rate of 0.01 are used for all the cases, as we observe more consistent converged solutions of PINNs using Adam (see Appendix A). As described in Appendix A, an ensemble of 10 random seeds are trained for both PINNs and LPINNs, and the probability distribution and lowest values of the error are compared.

The relative error is defined as,

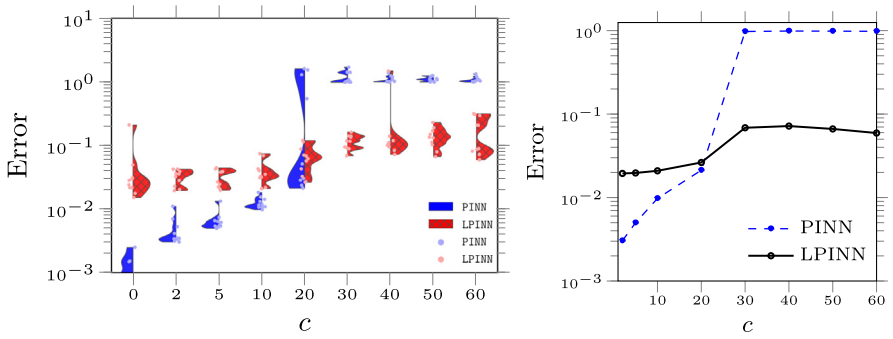
$$\text{Error} = \frac{\|\boldsymbol{w} - \boldsymbol{w}^*\|_2}{\|\boldsymbol{w}\|_2}, \tag{18}$$

where $\boldsymbol{w}$ is the truth, and $\boldsymbol{w}^*$ is the output of the network (interpolated) on the Eulerian grid, both after removing the boundary condition, to remove synthetic deflation of the reported error due to c in the cases of Burgers' equation. A quadratic scheme is used to interpolate the output of LPINN to the Eulerian grid.

5.1. Convection

Consider the inviscid convection equation,

$$\frac{\partial w(x,t)}{\partial t} + c\frac{\partial w(x,t)}{\partial x} = 0, \tag{19}$$



(a) Probability distribution of error in PINN (blue, left) vs. LPINNs (right, red)

(b) Minimum error

Fig. 8. Comparison of the error in PINN (with hard constraint) vs. the proposed LPINNs (with hard constraint) for the convection equation, Eq. (19).

and its reformulation in the Lagrangian frame of reference,

$$\frac{d\chi}{dt} = c, \tag{20a}$$

$$\frac{\partial w}{\partial t} = 0. \tag{20b}$$

The solution to Eq. (20) is straightforward. Eq. (20b) dictates the grid points to move with the constant convection velocity, c, while the state variable remains constant along the moving points, Eq. (20b). The accuracy of PINN and LPINN are compared for different convection velocity in Fig. 8. Similar to [12], the error of PINN increases for larger values of c, such that for $c = 20$ many of the random initialization seeds do not converge to an accurate solution. For $c \geq 30$, none of the ensemble members are converged to a small error. In case of the proposed LPINN, where the problem is simply reformulated on the Lagrangian frame of reference, the minimum error for all cases remains below 5%. Note that the reported error is also comprised of the error originating from interpolating the predicted state from the moving grid of the Lagrangian frame to the stationary grid of the Eulerian frame, and in its current implementation has lead to a limit in the accuracy of the proposed LPINNs.

To evaluate the optimality of the trained network, the loss landscape of the network at the end of the training phase is often used as a qualitative measure [8,12,15,58]. To compute the loss landscape, the two dominant eigenvectors of the Hessian of the loss with respect to the trainable parameters of the networks, $\boldsymbol{\delta}$ and $\boldsymbol{\eta}$, are computed using an efficient approximation [59]. Subsequently, the network is perturbed along the eigenvectors and its loss, $\mathcal{L}'$, is evaluated, i.e.,

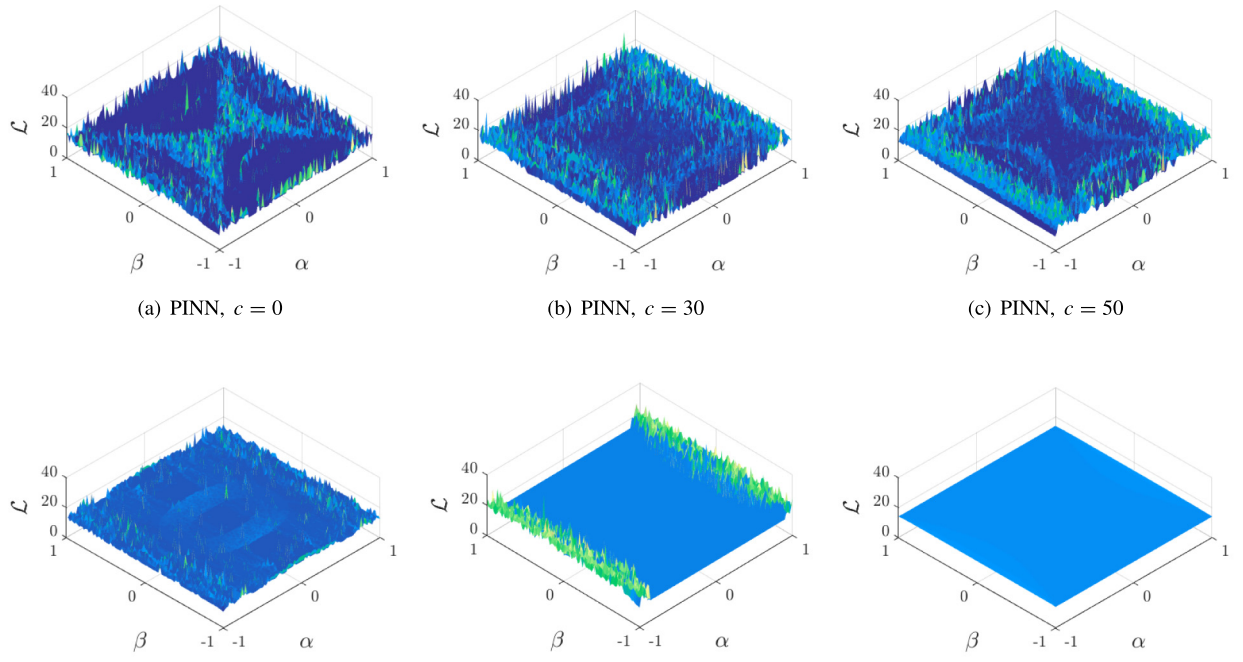
$$\mathcal{L}'(\alpha, \beta) = \mathcal{L}(\boldsymbol{\theta} + \alpha\boldsymbol{\delta} + \beta\boldsymbol{\eta}), \tag{21}$$

where $(\alpha, \beta) \in [-\alpha_0, \alpha_0] \times [-\beta_0, \beta_0]$. Finally, $\log\left(\mathcal{L}'(\alpha, \beta)\right)$ is visualized in Fig. 9, for $c = \{0, 30, 50\}$ and in both PINN and LPINN architectures. In Fig. 9(a), we recover the saddle shape of the loss landscape for small convection speed as reported for PINN in [12]. Similarly, by increasing c, the landscape becomes less smooth (sharper, or more rugged), implying the trained network is not at a minimizer (Figs. 9(b) to 9(c)). In the case of LPINN (Figs. 9(d) to 9(f)), the loss landscapes are significantly smoother compared to their PINN counterparts (Figs. 9(a) to 9(c)). Moreover, the landscape is smooth (flat), even at high c, increasing the confidence that the obtained minimizer is a global one. In LPINNs, explaining why the landscape for $c = 50$ (Fig. 9(f)) is smoother compared to $c = 30$ (Fig. 9(e)) or $c = 0$ (Fig. 9(d)) requires further investigations.

5.2. Convection–diffusion

Consider the viscous convection–diffusion equation,

$$\frac{\partial w(x,t)}{\partial t} + c\frac{\partial w(x,t)}{\partial x} = \nu\frac{\partial^2 w(x,t)}{\partial x^2}, \tag{22}$$



(a) PINN, $c = 0$ (b) PINN, $c = 30$ (c) PINN, $c = 50$

(d) LPINN, $c = 0$ (e) LPINN, $c = 30$ (f) LPINN, $c = 50$

Fig. 9. The (log of) loss landscape of convection equation given different convection speeds, $c \in \{0, 30, 50\}$. a-c. PINN,d-e. LPINN.

and its reformulation in the Lagrangian frame of reference,

$$\frac{d\chi}{dt} = c, \tag{23a}$$

$$\frac{\partial w}{\partial t} = \nu \frac{\partial^2 w}{\partial x^2}. \tag{23b}$$

Fig. 10 compares the accuracy of PINN and the proposed LPINNs. Similar to the inviscid case discussed in Section 5.1, the error in PINNs increases by increasing c. However, the minimum error in LPINNs is relatively insensitive with respect to c, and the error remains below 10% in all of the cases.

5.3. Burgers' equation

Consider the viscous Burgers' equation,

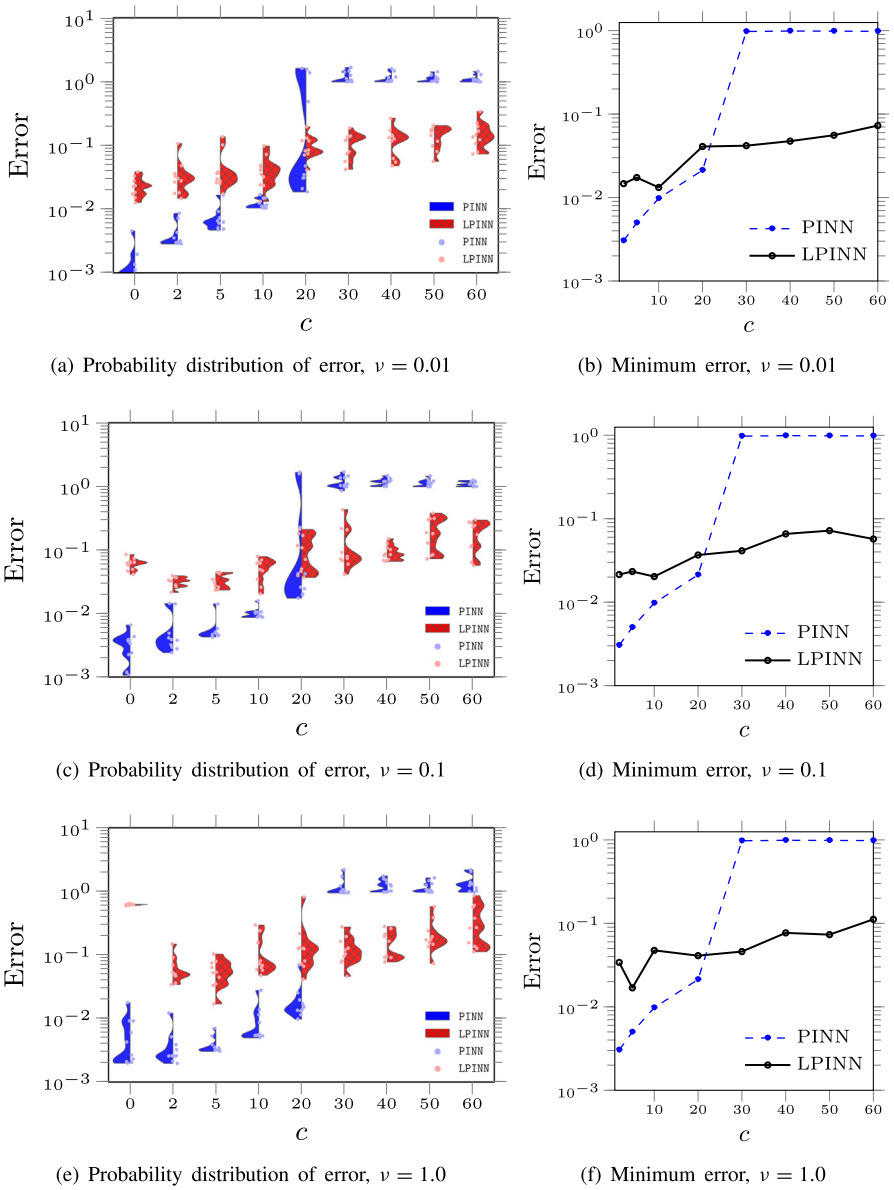
$$\frac{\partial w(x,t)}{\partial t} + w(x,t)\frac{\partial w(x,t)}{\partial x} = \nu \frac{\partial^2 w(x,t)}{\partial x^2}, \tag{24}$$

and its representation on the Lagrangian frame,

$$\frac{d\chi}{dt} = w(x,t), \tag{25a}$$

$$\frac{\partial w}{\partial t} = \nu \frac{\partial^2 w}{\partial x^2}. \tag{25b}$$

While the traditional formulation of PINN is successfully demonstrated for Burgers' equation [2], the examined problem lacks the main property of challenging cases for training, i.e., the large Kolmogorov n–width associated with the travel of the shock as in Fig. 5. In Burgers' equation, similar to convection–diffusion equation, the PINN is slower to train for larger c, while the LPINN is trained for all cases (Fig. 11). The higher error in this case compared to the convection–diffusion equation is due to the higher interpolation error close to the shock. Note that



(a) Probability distribution of error, $\nu = 0.01$

(b) Minimum error, $\nu = 0.01$

(c) Probability distribution of error, $\nu = 0.1$

(d) Minimum error, $\nu = 0.1$

(e) Probability distribution of error, $\nu = 1.0$

(f) Minimum error, $\nu = 1.0$

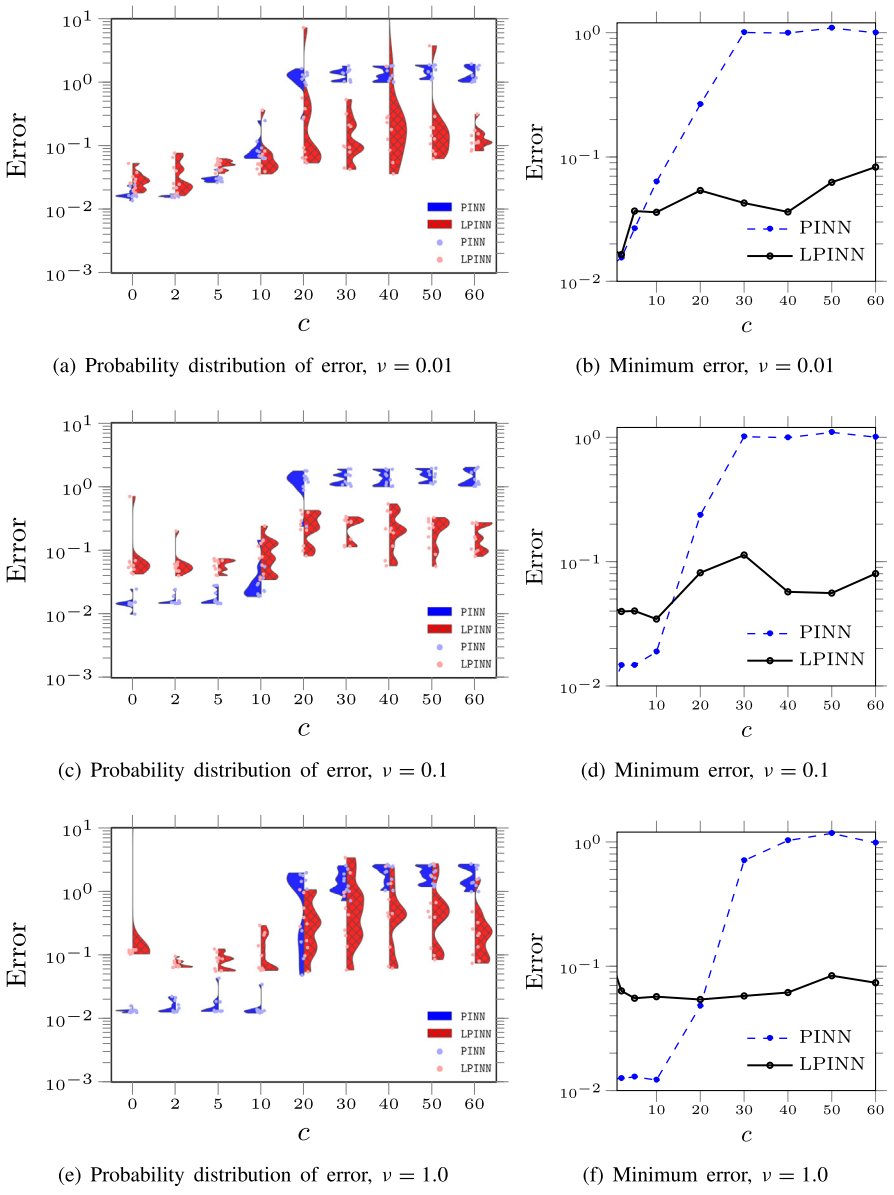
Fig. 10. Comparison of the (probability distribution and minimum) error in PINN vs. the proposed LPINNs for the convection–diffusion equation Eq. (22) for $\nu \in \{0.01, 0.1, 1.0\}$. For the probabilistic plots, PINNs are denoted by blue (left) and LPINNs are denoted by red (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in these cases the viscosity, ν, is small enough to form the high gradient shock and is large enough to avoid the intersecting characteristics. Fig. 12 shows the accuracy of the proposed LPINN compared to the numerical solver at different simulation times.

5.4. 2D convection–diffusion

Consider the 2D viscous convection–diffusion equation,

$$\frac{\partial w(x_1, x_2, t)}{\partial t} + c_1 \frac{\partial w(x_1, x_2, t)}{\partial x_1} + c_2 \frac{\partial w(x_1, x_2, t)}{\partial x_2} = \nu \left(\frac{\partial^2 w(x_1, x_2, t)}{\partial x_1^2} + \frac{\partial^2 w(x_1, x_2, t)}{\partial x_2^2} \right), \tag{26}$$



(a) Probability distribution of error, $\nu = 0.01$

(b) Minimum error, $\nu = 0.01$

(c) Probability distribution of error, $\nu = 0.1$

(d) Minimum error, $\nu = 0.1$

(e) Probability distribution of error, $\nu = 1.0$

(f) Minimum error, $\nu = 1.0$

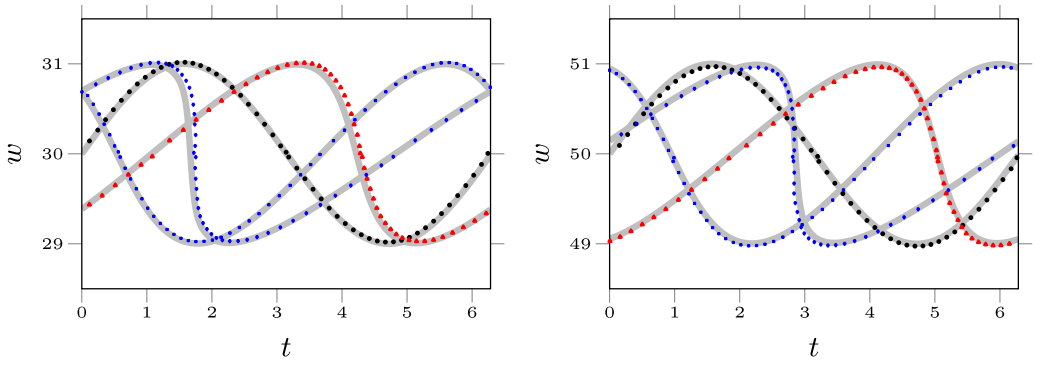
Fig. 11. Comparison of the (probability distribution and minimum) error in PINN vs. the proposed LPINNs for the Burgers' equation Eq. (22) for $\nu \in \{0.01, 0.1, 1.0\}$. For the probabilistic plots, PINNs are denoted by blue (left) and LPINNs are denoted by red (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and its representation on the Lagrangian frame

$$\frac{d\chi_1}{dt} = c_1, \tag{27a}$$

$$\frac{d\chi_2}{dt} = c_2, \tag{27b}$$

$$\frac{\partial w}{\partial t} = \nu \left(\frac{\partial^2 w(x_1, x_2, t)}{\partial x_1^2} + \frac{\partial^2 w(x_1, x_2, t)}{\partial x_2^2} \right), \tag{27c}$$



(a) $c = 30$ (b) $c = 50$

Fig. 12. Comparison of the proposed LPINN with pseudo-spectral solver for the viscous Burgers' equation of Eq. (24) ($\nu = 0.01$) at $t \in \{0, 1/3, 2/3, 1\}$ (black circle, blue square, red triangle, blue diamond) for $c = \{30, 50\}$. PINN are not converged to a low error in both regimes.

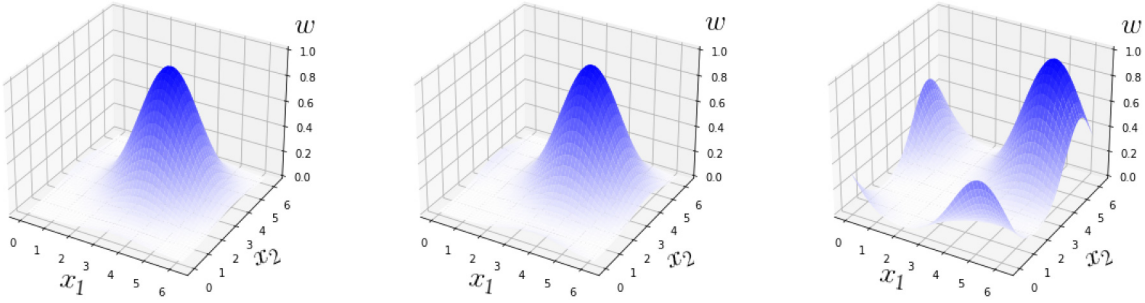
where χ_1 and χ_2 are the characteristics path on horizontal and vertical directions. The difficulty in training of PINNs for this case is reported in [60].

In the Eulerian frame, the domain is $(x_1, x_2, t) \in [0, 2\pi] \times [0, 2\pi] \times [0, 1]$ and doubly periodic boundary conditions are imposed. The initial condition is set to $w(x_1, x_2, 0) = \exp\left(-\frac{(x_1-\pi)^2+(x_2-\pi)^2}{1.5^2}\right)$ and $\nu = 0.01$. The network is similar to that of the 1D case, and the hyper-parameters are set to $\lambda_{bc} = 0$, $\lambda_{ic} = 1$, and $\lambda_r = 1$. The equidistant collocation points in the spatio-temporal are of size $(N_1, N_2, N_t) = (128, 128, 100)$. The Adam optimizer [57] with 10^4 iterations and learning rate of 0.01 are used for all the cases. The exact solution, i.e., $\mathcal{F}^{-1}\left(\mathcal{F}(w(x_1, x_2, 0))\, e^{-\nu\left(\kappa_1^2+\kappa_2^2\right)t} e^{-ic_1\kappa_1 t} e^{-ic_2\kappa_2 t}\right)$, the solution of PINN, and LPINN at $t = 1$ are compared for different cases of $c_1 = c_2 \in \{0.5, 0.75, 2.0\}$ in Fig. 13. While the PINNs is hard to train for convection speed higher than ≈ 0.5, our proposed LPINN is easily trained independent of the convection speed. Note that the solution of LPINNs is on a translating grid, χ_1 and χ_2. The periodic boundary condition, by definition, is easily imposed by removing the whole period of the domain.

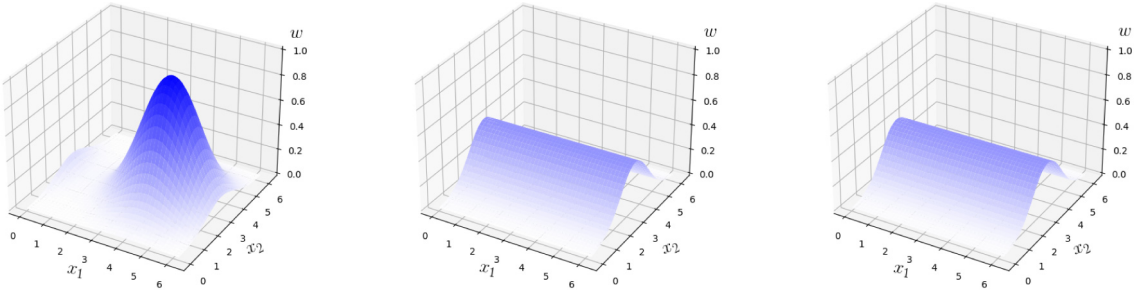
6. Potentials, challenges, and limitations

In the previous sections, we motivated reformulating convection-dominated problems on a Lagrangian frame of reference. While sequence-to-sequence learning [12] and in parallel in time decomposition [13] require decomposition of time in training and training/inference regimes, LPINNs provides the solution independent of the temporal domain length and in a global time-continuous domain. Moreover, LPINNs do not require solution of intermediate problems (in contrast to [12]). LPINN is also independent of data sampling and therefore can be applied in a no-data regime, where the network is used as a forward solver. This independence is particularly important since one does not necessarily have access to location of the provided data and refinement/adaptation the location of data/collocation points requires some *a priori* knowledge of the solution or repeating the experiment, where sampling methods such as those in [9,22] might not be practically efficient. The change of frame of reference is mathematically consistent with the initial problem and does not require ad-hoc engineering of the problem [21] or tuning of the system parameters [22].

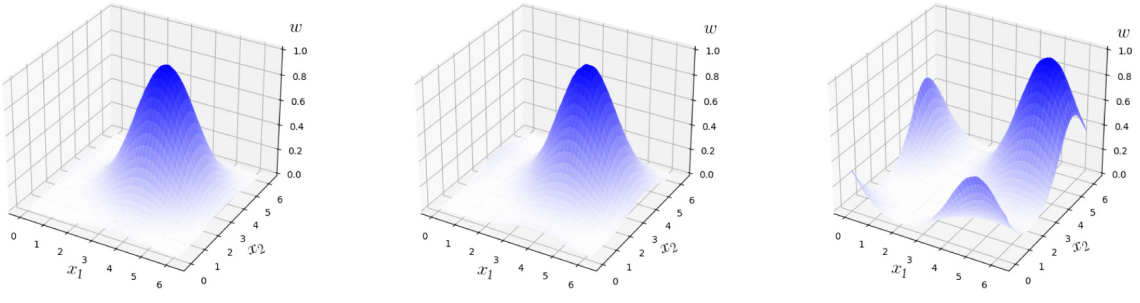
In summary, the inherent reduced dimension of the solution on a frame of reference following the traveling features is consistent with the original problem in the Eulerian frame and enables training of a single network (one set of weights and biases), and in a global time continuous domain without requiring solutions intermediate problems. While the discussed advantages make LPINNs a promising method for highly convection-dominated problems and, in principle, can be extended to high dimensional multi-scale problems, there are additional practical challenges that have to be considered. In the following subsections, the challenge of grid management and interpolation is discussed.



(a) Exact, $c_1 = c_2 = 0.50$ (b) Exact, $c_1 = c_2 = 0.75$ (c) Exact, $c_1 = c_2 = 2.00$



(d) PINN, $c_1 = c_2 = 0.50$ (e) PINN, $c_1 = c_2 = 0.75$ (f) PINN, $c_1 = c_2 = 2.00$



(g) LPINN, $c_1 = c_2 = 0.50$ (h) LPINN, $c_1 = c_2 = 0.75$ (i) LPINN, $c_1 = c_2 = 2.00$

Fig. 13. Comparison of the exact solution of 2D convection–diffusion equation with convection speed of $c_1 = c_2 \in \{0.5, 0.75, 2.0\}$ and $\nu = 0.01$ at $T = 1$ (Figs. 13(a) to 13(c)) with output of PINN (Figs. 13(d) to 13(f)), and our proposed LPINNs (Figs. 13(g) to 13(i)).

6.1. Grid management

Grid management is one of the well-known challenges of solvers in traditional numerical Lagrangian solvers, where the grid points, by definition, convect with the local flow velocity. Depending on the local velocity the adjacent grid points can depart and depreciate the accuracy of local gradients or they can move towards each other and eventually cross such that the grid volume becomes negative, i.e., grid tanglement (mesh imprinting),

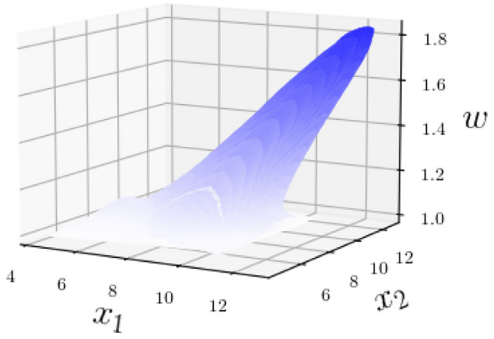


Fig. 14. Solution of LPINN for inviscid Burgers' equation on a 2D infinite domain, showing formation of a strong solution and grid tanglement.

or small enough to lead to unacceptably small time-steps (due to the Courant–Friedrichs–Lewy (CFL) limit) [61]. The challenge of accurate and strictly positive grid volume has been recognized since the early days of Lagrangian solvers [62] and has been continually an active topic of research [63–65]. Despite these challenges, the Lagrangian solvers are extended to three-dimensional (3D) flows with highly complicated shocks [66].

To examine the capabilities of LPINNs for a case where the characteristics intersect, i.e., grid tanglement, we consider Lagrangian representation of inviscid Burgers' equation in an unbounded 2D domain,

$$\frac{d\chi_1}{dt} = w(x_1, x_2, t), \tag{28a}$$

$$\frac{d\chi_2}{dt} = w(x_1, x_2, t), \tag{28b}$$

$$\frac{\partial w}{\partial t} = 0, \tag{28c}$$

with the initial condition of $w\,(x_1, x_2, 0) = c + \exp\left(-\frac{(x_1-\pi)^2+(x_2-\pi)^2}{1.5^2}\right)$, where c is set to 1.0 to induce a traveling shock. The LPINN architecture is similar to that of 2D convection–diffusion problem presented in Section 5.4. The LPINN is trained for a long time horizon of $T = 5$, and the solution in the last time step is shown in Fig. 14. The grids in this case simply intersect and pass, but such grid tanglement does not lead to any instabilities or challenge in the training phase. However, the output is the strong solution of inviscid Burgers' equation, and to obtain physical weak solution of the Burgers' the so called entropy condition must be introduced.

Another classical approach to avoid the challenges associated with grid tanglement is to relax the assumption of the Lagrangian frame. In the arbitrary Lagrangian–Eulerian (ALE) frame, the grid can move in an arbitrary and prescribed manner [67]. Therefore, the grid velocity can be set to be independent of flow velocity. The governing equations have to be adjusted to account for convective velocity, i.e., the difference between the grid velocity and the local velocity, e.g, [68]. Such a generalization can alleviate the challenge of grid tanglement. The optimality and effectiveness of an arbitrary grid for reducing the Kolmogorov n–width of convection-dominated problems are shown in [39], where an optimization problem is solved to discover a grid trajectory such that the rank of the mapped solution is small. The application of a similar strategy in the context of PINNs remains a topic for future work.

Moreover, the PINN formulation provides unique opportunities to address the aforementioned challenges. Firstly, the accuracy of derivatives calculated via AD, in contrast to traditional methods, does not degrade with what traditionally is defined as the grid quality measures, e.g. skewness, and aspect ratio. Accordingly, the inviscid Burgers' problem discussed above and the developments of Neural Particle Methods (NPMs) [69,70] show that a PINN-based implementations remain stable even when the collaboration points become highly irregular. Secondly, the stability in PINNs seems not to be restricted by the CFL condition, therefore small grid volumes do not restrict the choice of refinement of collocation points in time. Finally, an arbitrary grid trajectory can directly be learned by simple addition of constraints to the NN.

6.2. Interpolation error

The immediate consequence of reformulating the problem on the Lagrangian frame is detachment of the solution from the (stationary) Eulerian grid/collocation points. While the choice of a frame of reference is arbitrary, there are cases where the solution is sought on a stationary frame, e.g., coupling with other Eulerian solvers, or a quantity of interest (QoI) is defined in the Eulerian frame. In such cases, an additional post-processing step is necessary to map the solution from the Lagrangian to the Eulerian grid. Although the interpolation step can introduce error, its convergence and error bounds are well studied, and typically negligible. There is no one correct choice of an interpolation scheme, an appropriate scheme is a delicate compromise between tolerable error, cost of search for the neighboring grid points, dimension, the scale of the problem, and the assigned computational budget. This can be more pronounced in presence of high gradient features in the solution, e.g., shocks.

7. Conclusions

In this paper, we explored the slow convergence of PINN for convection-dominated problems. It is demonstrated that regardless of the choice of commonly used optimization schemes (Adam or L-BFGS) or distribution of the collocation points (equidistant or uniform), convergence of PINN is significantly more challenging as convection dominates. Our contributions are threefold.

1. We described the challenges of training traditional PINNs through the lens of approximation theory using Kolmogorov n–width and the rate of decay of singular values of the solution. Accordingly, we explained many of the successful remedies in training of PINNs. Moreover, this lens can lead to identifying new challenging problems and novel methods to address them.
2. We identified Burgers' equation in the presence of *traveling shocks* as another challenging case. The complexity of training is explained based on our discussion of the irreducibly on a linear space, i.e., large Kolmogorov n–width.
3. Finally, we demonstrated our proposed architecture, i.e., LPINNs for linear and non-linear convection–diffusion equations. The reformulation of the equations on the characteristics automatically conforms to the direction of travel of information in the domain, and satisfies the expected causality. More importantly, the solution on the manifold is of lower dimension, i.e., low Kolmogorov n–width, and is less sensitive to the system parameters. Using the inherent low-dimensionality of the characteristics [56], only a shallow branch is added to the traditional PINN to minimize the composite loss comprised of residual equations of characteristics, and the state variable on the characteristics. While it is suggested that the condition number of the loss is a probable source of complexity in training of PINNs [12], the proposed LPINN architecture is robust with respect to the condition number and good convergence is demonstrated regardless.

The advantage of LPINN is most highly evident in very strongly convection-dominated problems, where traveling features dominates the solution. Such problems often surface additional challenges that should be taken into account when developing the proposed approach. For example, in the nonlinear equations governing compressible fluid flows, multiple features may from a single point, e.g., Sod's shock tube problem [71], where the shock and expansion waves are initiated instantly, from a single point and are emanated in different directions.

So far we have only discussed time dependent problems where the solution depicts convection with respect to time. Another category of problems with large Kolmogorov n–width occurs in steady state parametric systems, where different system parameters lead to snapshots depicting similar features at different spatial locations [72]. In practice, the number of such parameters is often fewer than the number of time steps in unsteady problems, however, we predict a similar challenge in training of PINNs. The proposed Lagrangian approach is not applicable to such cases, however, a similar idea can be beneficial, i.e., to learn a mapping that reduces the Kolmogorov n–width of the problem by removing traveling features between the cases, e.g., [39,72].

We have demonstrated viability of LPINN for inviscid Burgers' equation where the characteristics intersect. The strong solution is found without any challenge in training or additional considerations of grid tanglement. In such cases, a vanishing viscosity approach or removing the triple point value using Rankine–Hugoniot condition (entropy condition) seem to be straightforward to solve for the weak solutions [73]. Similar strategies are applied to PINNs in Eulerian frame and in the presence of shocks in [21,22]. An extension of the proposed LPINN to higher spatial

dimensions is possible using the Radon transform [74]. In cases where characteristics curves are not real, e.g., wave equation, a manifold can be identified by registration-based or feature tracking approaches, e.g., [37,39,46,75,76], and in particle method formulation of PINNs [69]. Specifically, an optimal and low-rank manifold can be constructed offline by identifying an optimally morphing grid [39]. Subsequently, PINNs' architecture provides an opportunity for one-shot discovery of an optimal manifold defined on an ALE grid.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Shared on https://github.com/rmojgani/LPINNs/.

Acknowledgments

We also would like to thank the anonymous reviewers for their constructive comments. Especially we thank one of the reviewers for sharing their implementation of traditional PINNs with traditional optimization strategies, which we based our further implementations required for the comparison studies. This work was supported by an award from the ONR Young Investigator Program (N00014-20-1-2722) and a grant from the NSF CSSI program (OAC-2005123). Computational resources were provided by NSF XSEDE (allocation ATM170020) and NCAR's CISL (allocation URIC0004). Our codes and data are available at https://github.com/rmojgani/LPINNs.

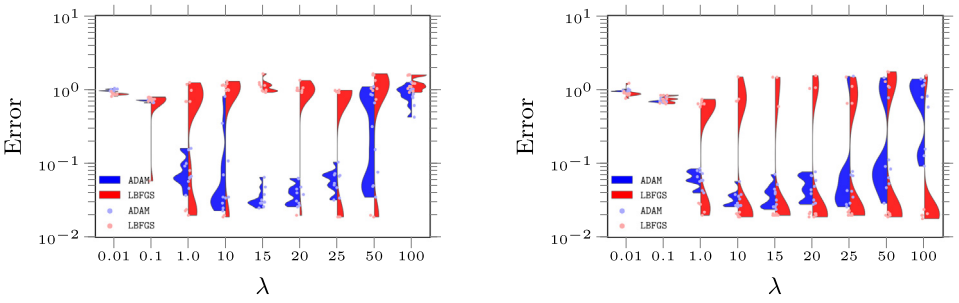
Appendix A. Convergence of PINNs for convection equation

In this section, convergence behavior of traditional PINNs with respect to choice of hyper-parameters, optimization schemes, distribution of the collocation points, and types of enforcing of periodic boundary conditions (soft versus hard enforcement) are investigated. Specifically, we compare training dynamics of PINNs using two commonly used optimization schemes [2], i.e., Adam [57] and L-BFGS, [77], sweeping a range of hyper-parameters of the composite loss, i.e., $\lambda := \lambda_{bc} = \lambda_{ic} \in \{0.01, 0.1, 1.0, 10, 15, 20, 25, 50, 100\}$, and $\lambda_r = 1$. The search for an optimal hyper-parameter is carried out with the soft enforcement of the boundary conditions, as the most common architecture of PINNs in Appendix A.1 and Appendix A.2. A comparison of soft versus hard imposing of the boundary conditions is made in Appendix A.3. Each instance is repeated for 10 different random seeds to study the probabilistic behavior in training with respect to random initialization of the networks, i.e., a multi-start global search commonly used in non-convex optimization problems, e.g., for control [78]. As the metric of comparison, relative error (ε) of the solution of the trained PINNs is compared with the exact solution (see Table 2). The maximum number of iterations is 2×10^4, unless otherwise stated. The probability distribution of the error over the random seeds are approximated with kernel density estimations (KDEs), and are plotted between the minima and maxima of the occurrences of the error. The effect of two different distribution of collocation points are also compared. The equidistant collocation points are of $(N_x, N_t) = (45, 45)$ (total of 2025 points). The random collocation points are sampled from a Latin hypercube distribution [79] (total of 2000 points). The domain is $(x, t) \in [0, 2\pi] \times [0, 1]$.

A.1. Choice of hyper-parameters

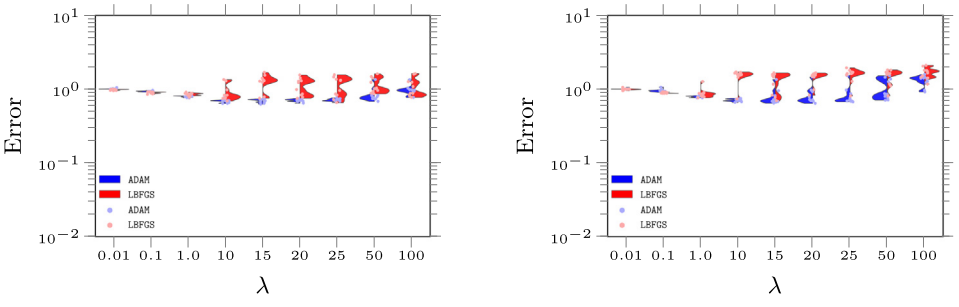
In Fig. A.1 the error of the trained PINNs using the different optimization schemes and grids are compared for $c = 30$. In all cases, it is observed that the accuracy of the converged solution is sensitive to the choice of the hyper-parameters used to scale the loss components, see Eq. (5) in Section 2.

A bifurcation in the accuracy is apparent in PINNs trained using L-BFGS regardless of the type of the grid, i.e., one branch of the trained networks is stuck in local minima with high error (around 100%), while the other branch is converged to error of approximately 3%. The type of the grid affects the probability of occurrence of each these branches. Random collocation points are more likely to lead to inaccurate solutions, meaning that, for an accurate solution of a PINN on random collocation points, a large ensemble of networks must be trained.



(a) Random grid, comparison of optimizers (b) Uniform grid, comparison of optimizers

Fig. A.1. The stability map of convergence of training of PINNs to solve convection equation with $c = 30$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Random grid, comparison of optimizers (b) Uniform grid, comparison of optimizers

Fig. A.2. The stability map of convergence of training of PINNs to solve convection equation with $c = 50$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

However, on the uniform collocation points, the chances of accurate solution and high error are almost equal (given an acceptable λ).

This bifurcated behavior is less pronounced using the Adam optimizer, however, it similarly suffers from large variance in accuracy and sensitivity to the hyper-parameter (e.g., while $\varepsilon \in [3\%, 7\%]$ with $\lambda = 15$, with $\lambda = 10$, $\varepsilon \in [3\%, 25\%]$).

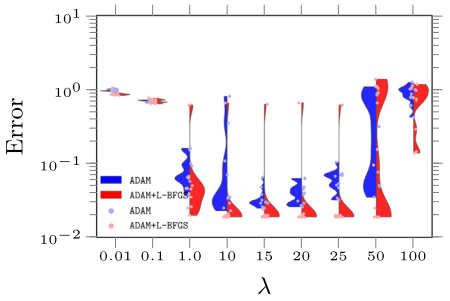
Similar experiments for $c = 50$ are shown in Fig. A.2. In this case, regardless of the grid type or the optimization scheme, the error remains consistently high over the range of hyper-parameters.

A commonly used strategy in training of PINNs, starting an optimization using Adam followed by L-BFGS is shown in Fig. A.3 and Fig. A.4. In the case of $c = 30$, the addition of L-BFGS reduces the error in most cases (in the range of acceptable hyper-parameters), however, there are still rare incidences of poor convergence. Moreover, for $c = 50$, such a strategy does not effectively reduce the error.

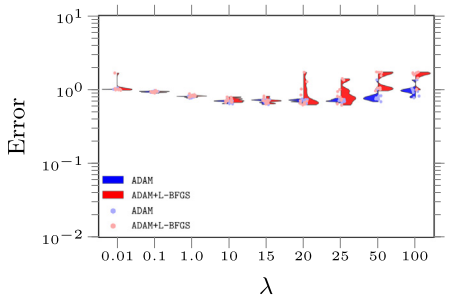
Note that given these experiments, one cannot claim that accurate PINNs are unattainable in such regimes. The expressivity of the network is shown by training the same architecture with different optimization and scheduling (see Appendix B). However, it is clear that convergence of PINNs becomes slower as c increases. Clearly, this complexity is *not* due to higher order derivatives or chaos in the solution, as they are simply absent from the showcased convection equation. This realization motivates our proposed architecture, i.e., LPINNs, where the complexity due to the convection is reduced by transformation of convection-dominated problems to diffusion-dominated ones. The details of our proposed approach is presented in Section 4.

A.2. Choice of collocation distribution

In this section, the effect of collocation point strategies are compared for convection problems of $c = 30$ and $c = 50$ in Fig. A.5 and Fig. A.5, respectively. For the case of $c = 30$, the L-BFGS scheme is more sensitive to the distribution of collocation points, where higher accuracy is more probable on a uniform grid. However, the Adam



(a) Random grid, comparison of optimizers



(b) Uniform grid, comparison of optimizers

Fig. A.3. The stability map of convergence of training of PINNs to solve convection equation. The results are shown for Adam versus Adam (blue, left) followed by L-BFGS, i.e., Adam+L-BFGS (red, right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

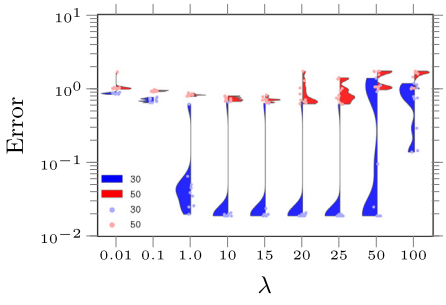
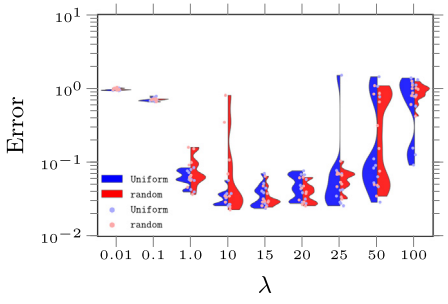
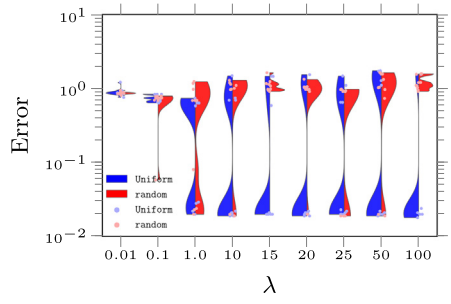


Fig. A.4. The stability map of convergence of training of PINNs using Adam followed by L-BFGS to solve convection equation with $c = 30$ (blue, left), and $c = 50$ (red, right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) Adam optimizer, comparison of collocation points



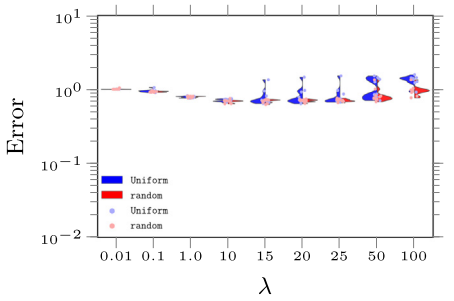
(b) L-BFGS optimizer, comparison of collocation points

Fig. A.5. The effect of collocation points and optimization scheme. Uniform and random distributions are designated with blue (left) and red (right), respectively. The stability map of convergence of training of PINNs to solve convection equation with $c = 30$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

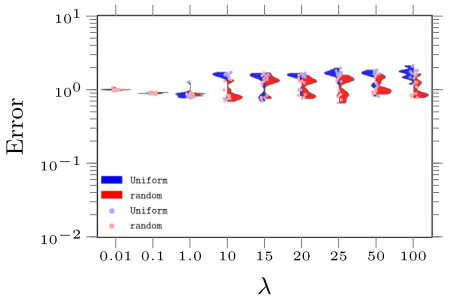
optimizer reaches acceptable error range more frequently. For the case of $c = 50$, all approaches fail to converge (see Fig. A.6).

A.3. Choice of constraints

In Section 2 two approaches of enforcing periodic boundary conditions of PINNs are discussed, (i) adding periodicity of the state at the boundaries in the loss function (soft constraint and $\lambda_{bc} \neq 0$), and (ii) addition of



(a) Adam optimizer, comparison of collocation points



(b) L-BFGS optimizer, comparison of collocation points

Fig. A.6. The effect of collocation points and optimization scheme. Uniform and random distributions are designated with blue (left) and red (right), respectively. The stability map of convergence of training of PINNs to solve convection equation with $c = 50$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

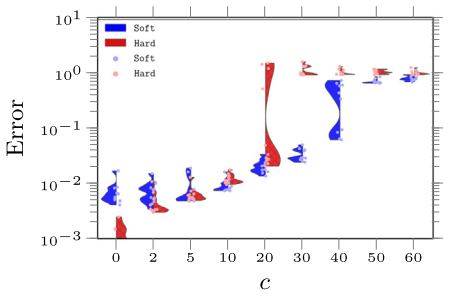


Fig. A.7. The effect of soft (left, blue) and hard (red, right) enforcing of periodic boundary conditions on accuracy of PINNs of convection equation for a range of c. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a non-trainable periodic layer to the architecture of PINNs (hard constraint and $\lambda_{bc} = 0$). In Fig. A.7, the effect of this choice on accuracy of PINNs is compared for a range of c of the convection equation. The hyper-parameters are $\lambda_{ic} = 15$ (both cases) and $\lambda_{bc} = 15$ (soft constraint) and the results are shown for Adam optimizer with uniform collocation points. Hard constraints increase solution accuracy for small c, however, as c is increased both methods go through a transition phase, where accurate and inaccurate solutions are similarly probable. This transition occurs in $c \in [10, 20]$ and $c \in [30, 40]$ for hard and soft constraints, respectively. For c larger than of the transition value, the PINNs show high error, regardless of the type of enforcing of the periodic boundary conditions.

Appendix B. Remedies to train PINNs for convection

In this section, several approaches for improving training performance are compared. Specifically, we focus on methods that (i) are demonstrated or claimed to be applicable to convection-dominated problems, (ii) are independent of sampling strategies, and (iii) can be implemented using commonly used and off-the-shelf numerical libraries. Finally, the existing methods are compared with our proposed LPINNs.

B.1. Sequence-to-sequence learning [12]

In sequence-to-sequence [12] (or sometimes referred to parallel in time decomposition [13], the temporal domain is decomposed into shorter non-overlapping subdomains, i.e., each subdomain is $\mathcal{T}_i = [(i-1)\,\Delta t, i\,\Delta t]$ such that $\cup_{i=1}^{\tau} \mathcal{T}_i = [0, T]$ with no overlapping subdomains, $\mathcal{T}_i \cap \mathcal{T}_j = \varnothing, \forall i \neq j$, where τ is the number of subdomains, Δt is the length of each subdomain, and $T = \tau \Delta t$. Subsequently, a PINN is trained on each subdomain such that it satisfies the solution of the PINN in the previous subdomain as the initial condition. This procedure is repeated so

as to cover the full temporal domain. Generally, this method does *not* guarantee smoothness of the solution and its derivatives across the subdomains. For this purpose, XPINNs can be employed to impose smoothness across the subdomains [4].

B.1.1. Sequence-to-sequence learning with warm initialization

In the previously discussed seq-to-seq learning, the network is initialized with random weights in the beginning of training for each of the subdomains (cold initialization). However, a warm initialization strategy [80] can be used, i.e., the weights in the subsequent networks can be initialized from the previous subdomain. Note that warm initialization is not equivalent to transfer learning, where often the deepest layers are retrained. However, recent studies claim such a choice might not be optimal for physical systems [81]. In our comparison, we simply retrain the same network with newly introduced initial conditions, i.e., warm initialization.

B.1.2. Curriculum learning [12]

In curriculum learning or curriculum regularization [12], a network is trained to solve the PDE at various different levels of modeling complexity, i.e., the trained weights and biases are transferred (used in warm initialization, to be precise) from simpler regimes to progressively harder regimes [12]. In the case of convection equation, the network is first trained to model small convection speed and then, training is progressed towards the target problem with high convection speed.

B.1.3. Extended sequence-to-sequence learning

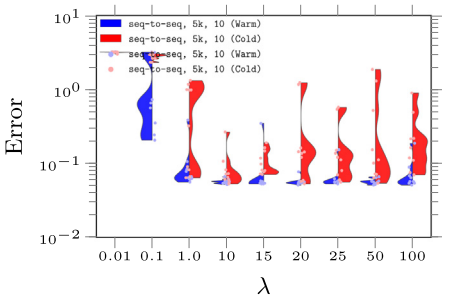
We introduce a natural extension to the seq-to-seq method discussed above. In this method, the temporal domain is decomposed to subdomains, however, each subdomain is an extension of all the previous subdomains, i.e., $\mathcal{T}_i = [0, i\,\Delta t]$ such that $\cup_{i=1}^{\tau} \mathcal{T}_j = [0, T]$ with overlapping subdomains of $\mathcal{T}_i \cap \mathcal{T}_{i+1} = [0, i\,\Delta t]\,, \forall i \in \{1, \tau - 1\}$. In this method, by progressively extending the length of the initial subdomain (instead of progressing to the next subdomain), the boundary of subdomains become interior points in the subsequent training of the subdomains and the smoothness considerations become irrelevant.

Remark. The extended sequence-to-sequence learning approach becomes progressively more expensive to train as extending the temporal domain also increases the number of collocation points. This backward-compatible PINN (bc–PINN) method [82] is expected to have a more favorable scalability by removing some portion of collocation points from previously learned subdomains. More specifically, at each subsequent subdomain, bc–PINN minimizes the difference between the prediction of the previously trained PINN with the current network at fraction of the initial collocation points. Although this strategy reduces the number of function evaluations, we note that the extended sequence-to-sequence learning proposed above is at most as large as training a network in the full spatiotemporal domain and is still is affordable in our experiments. We expect bc–PINN to be beneficial for larger problems with hardware limitations.

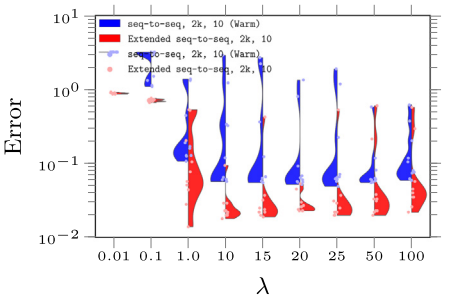
B.2. Comparison

In this section, the aforementioned approaches are compared. The boundary conditions are imposed through the composite loss (soft constraint) for sequence-to-sequence, extended sequence-to-sequence, and curriculum learning. For the traditional PINNs on Eulerian frame, both soft and hard constraints (using the custom layer described in Section 2) enforcing of the periodic boundary condition are compared. Hard constraint periodic boundary condition is always imposed on LPINNs.

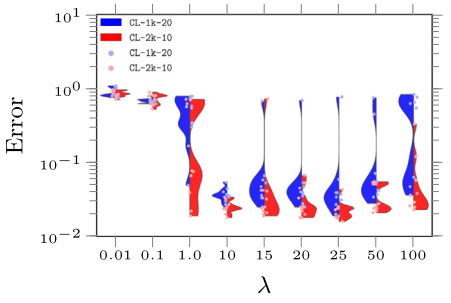
In Fig. B.8 the results for $c = 30$ are summarized. The collocation points are random and the optimizer is L-BFGS. In Fig. B.8(a), the effect of warm initialization on the seq-to-seq approach is compared. Both approaches reduces the error compared to direct optimization on the full temporal domain (compare to Fig. A.1(a)). Moreover, the warm initialization leads to more consistent convergence, while the variance of the error without the warm initialization strategy remains high. In Fig. B.8(b) the seq-to-seq approach (with warm initialization) is compared to our proposed extended seq-to-seq approach. Both the overall error and sensitivity to the hyper-parameters are decreased in our proposed approach. This can be explained by the smoothness on the boundary of domains and avoiding error accumulations over the subdomains. In Fig. B.8(c) the curriculum learning approach is studied, where



(a) Seq-to-seq with warm (blue, left) and cold (red, right) initialization with 5×10^3 iterations per subdomain, and $\tau = 10$ subdomains)



(b) Seq-to-seq with warm initialization (blue, left) compared to extended seq-to-seq (red, right) with 2×10^3 iterations on $\tau = 10$ subdomains



(c) Curriculum learning, $\tau = 10$ (blue, left) and $\tau = 20$ (red, right) subdomains with total budget of 20×10^3 iterations

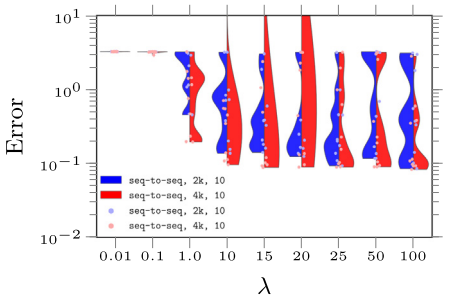
Fig. B.8. Comparison of different strategies of training of PINNs for convection equation ($c = 30$). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

a total of 20×10^3 iterations are budgeted in $\tau = 10$ and $\tau = 20$. In this case, the $\tau = 10$ shows lower error compared to $\tau = 10$ and $\tau = 1$ (Fig. A.1(a)), suggesting balancing the number of subdomains and iterations per subdomain is a necessary task.

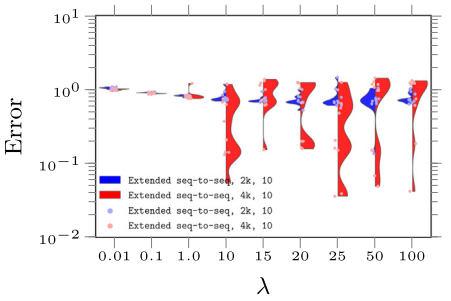
Similar experiments are carried out for $c = 50$ in Fig. B.9. The seq-to-seq with warm initialization may reach 10% error in some instances, but with high uncertainty and less probability (Fig. B.9(a)). The extended seq-to-seq approach also suffers from slower convergence and requires more iterations Fig. B.9(b), compared to Fig. B.8(b). The curriculum learning approach reaches more accurate solutions and is less sensitive to the hyper-parameters Fig. B.9(c), compared to extended seq-to-seq (Fig. B.9(b)).

Finally, in Fig. B.10 the methods discussed in Appendix B are compared with our proposed LPINNs. The budget for each of the methods is set to 2×10^4. In the case of $c = 30$, the traditional formulation of PINNs with soft constraint has the lowest error compared to all other formulations. However, in the case of enforcing the boundary condition with hard constraint, LPINN outperforms PINN. Moreover, the accuracy of the best solution of LPINNs is close to that of seq-to-seq learning, while extended and seq-to-seq and curriculum reaches more accurate solutions. However, given the high variance in the accuracy, it is also probable for those methods to show poor convergence within the assigned budget. For the higher convection speed of $c = 50$, LPINNs is shown to be more accurate than all other methods and achieves this accuracy for the majority of random seeds.

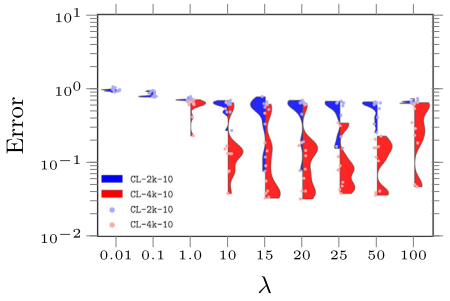
To summarize, although many of the remedies proposed in the literature provide some improvements in training convergence, these methods remain highly sensitive to additional hyper-parameters (e.g., number of sub-problems such as subdomains in seq-to-seq training or intermediate problems in curriculum learning). More importantly, in the aforementioned methods, the probability of achieving an accurate solution is reduced at higher convection speed, such that large ensemble of training over randomized seeds has to be generated to achieve an accurate enough solution. This is in contrast to our proposed method, LPINNs, where the accuracy shows small variance across a broad range of convection speeds.



(a) Seq-to-seq with warm initialization with 2×10^3 (left, blue) and 4×10^3 (right, red) iterations on $\tau = 10$ subdomains)

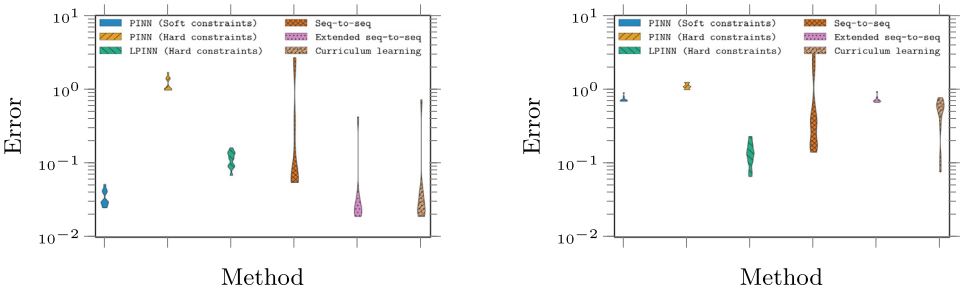


(b) Extended seq-to-seq with 2×10^3 (left, blue) and 4×10^3 (right, red) iterations on $\tau = 10$ subdomains)



(c) Curriculum learning (CL), $\tau = 10$ subdomains with 2×10^3 (left, blue) and 4×10^3 (right, red) iterations per subdomain

Fig. B.9. Comparison of different strategies of training of PINNs for convection equation ($c = 50$). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $c = 30$ (b) $c = 50$

Fig. B.10. Comparison of the proposed LPINNs with traditional PINNs, and different strategies of training of convection equation ($c = \{30, 50\}$) with the budget of 2×10^4 iterations per random seed. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

References

[1] J. Han, A. Jentzen, W. E., Solving high-dimensional partial differential equations using deep learning, Proc. Natl. Acad. Sci. 115 (34) (2018) 8505–8510.

[2] M. Raissi, P. Perdikaris, G. Karniadakis, Physics–informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.

[3] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics–informed machine learning, Nat. Rev. Phys. 3 (6) (2021) 422–440.

[4] A.D. Jagtap, G.E. Karniadakis, Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations, Commun. Comput. Phys. 28 (5) (2020) 2002–2041.
[5] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo, S.G. Johnson, Physics–Informed neural networks with hard constraints for inverse design, SIAM J. Sci. Comput. 43 (6) (2021) B1105–B1132.
[6] A. Arzani, J.-X. Wang, R.M. D'Souza, Uncovering near-wall blood flow from sparse data with physics-informed neural networks, Phys. Fluids 33 (7) (2021) 071905.
[7] S. Mowlavi, S. Nabi, Optimal control of PDEs using physics-informed neural networks, J. Comput. Phys. 473 (2023) 111731.
[8] O. Fuks, H.A. Tchelepi, Limitations of physics informed machine learning for nonlinear two-phase transport in porous media, J. Mach. Learn. Model. Comput. 1 (1) (2020) 19–37.
[9] Z. Mao, A.D. Jagtap, G.E. Karniadakis, Physics–informed neural networks for high-speed flows, Comput. Methods Appl. Mech. Engrg. 360 (2020) 112789.
[10] S. Wang, X. Yu, P. Perdikaris, When and why PINNs fail to train: A neural tangent kernel perspective, J. Comput. Phys. 449 (2022) 110768.
[11] S. Wang, Y. Teng, P. Perdikaris, Understanding and mitigating gradient flow pathologies in physics-informed neural networks, SIAM J. Sci. Comput. 43 (5) (2021) A3055–A3081.
[12] A.S. Krishnapriyan, A. Gholami, S. Zhe, R.M. Kirby, M.W. Mahoney, Characterizing possible failure modes in physics-informed neural networks, Adv. Neural Inf. Process. Syst. (Ml) (2021) 1–22.
[13] A. Bihlo, R.O. Popovych, Physics–informed neural networks for the shallow–water equations on the sphere, J. Comput. Phys. 456 (2021) 111024.
[14] S. Wang, S. Sankaran, P. Perdikaris, Respecting causality is all you need for training physics–informed neural networks, 2022, preprint arXiv:2203.07404.
[15] S. Basir, I. Senocak, Critical investigation of failure modes in physics-informed neural networks, in: AIAA SCITECH 2022 Forum, American Institute of Aeronautics and Astronautics, Reston, Virginia, 2022, pp. 1–12.
[16] Colby, L. Wight, J. Zhao, Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics informed neural networks, Commun. Comput. Phys. 29 (3) (2021) 930–954.
[17] M.R. Hestenes, Multiplier and gradient methods, J. Optim. Theory Appl. 4 (5) (1969) 303–320.
[18] R.G. Patel, I. Manickam, N.A. Trask, M.A. Wood, M. Lee, I. Tomas, E.C. Cyr, Thermodynamically consistent physics-informed neural networks for hyperbolic systems, J. Comput. Phys. 449 (2022) 110754.
[19] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, K. Saenko, Sequence to sequence - video to text, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4534–4542.
[20] X. Meng, Z. Li, D. Zhang, G.E. Karniadakis, PPINN: Parareal physics–informed neural network for time–dependent PDEs, Comput. Methods Appl. Mech. Engrg. 370 (2020) 113250.
[21] C.G. Fraces, H. Tchelepi, Physics informed deep learning for flow and transport in porous media, 2021, preprint arXiv:2104.02629.
[22] E. Abreu, J.B. Florindo, A study on a feedforward neural network to solve partial differential equations in hyperbolic-transport problems, in: M. Paszynski, D. Kranzlmüller, V.V. Krzhizhanovskaya, J.J. Dongarra, P.M.A. Sloot (Eds.), Computational Science – ICCS 2021, Springer International Publishing, Cham, 2021, pp. 398–411.
[23] K. Haitsiukevich, A. Ilin, Improved training of physics-informed neural networks with model ensembles, 2022, preprint arXiv:2204.05108.
[24] A. Daw, J. Bu, S. Wang, P. Perdikaris, A. Karpatne, Rethinking the importance of sampling in physics-informed neural networks, 2022, preprint arXiv:2207.02338.
[25] Y. Shin, J. Darbon, G.E. Karniadakis, On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs, Commun. Comput. Phys. 28 (5) (2020) 2042–2074.
[26] B. Hillebrecht, B. Unger, Certified machine learning: A posteriori error estimation for physics–informed neural networks, 2022, pp. 1–8, preprint arXiv:2203.17055.
[27] S. Dong, N. Ni, A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks, J. Comput. Phys. 435 (2021) 110242.
[28] W. E, S. Wojtowytsch, Kolmogorov width decay and poor approximators in machine learning: shallow neural networks, random feature models and neural tangent kernels, Res. Math. Sci. 8 (1) (2021) 5.
[29] S. Wojtowytsch, W. E, Can shallow neural networks beat the curse of dimensionality? A mean field training perspective, IEEE Trans. Artif. Intell. 1 (2) (2021) 121–129.
[30] A. Quarteroni, A. Manzoni, F. Negri, Reduced Basis Methods for Partial Differential Equations, in: UNITEXT, Vol. 92, Springer International Publishing, 2016.
[31] A. Pinkus, N-WIdths in Approximation Theory, Springer, Berlin, Heidelberg, 1985.
[32] J.M. Melenk, On n-widths for elliptic problems, J. Math. Anal. Appl. 247 (1) (2000) 272–289.
[33] S.M. Djouadi, On the optimality of the proper orthogonal decomposition and balanced truncation, in: Proceedings of the IEEE Conference on Decision and Control, (3) IEEE, 2008, pp. 4221–4226.
[34] S.M. Djouadi, On the connection between balanced proper orthogonal decomposition, balanced truncation, and metric complexity theory for infinite dimensional systems, in: Proceedings of the 2010 American Control Conference, ACC 2010, IEEE, 2010, pp. 4911–4916.
[35] B. Unger, S. Gugercin, Kolmogorov n-widths for linear dynamical systems, Adv. Comput. Math. (2019).
[36] J.A. Evans, Y. Bazilevs, I. Babuška, T.J. Hughes, N-widths, sup–infs, and optimality ratios for the k-version of the isogeometric finite element method, Comput. Methods Appl. Mech. Engrg. 198 (21) (2009) 1726–1741.
[37] M.A. Mirhoseini, M.J. Zahr, Model reduction of convection-dominated partial differential equations via optimization-based implicit feature tracking, 2021, pp. 1–38, preprint arXiv:2109.14694.

[38] T. Taddei, A.T. Patera, A localization strategy for data assimilation; application to state estimation and parameter estimation, SIAM J. Sci. Comput. 40 (2) (2018) B611–B636.
[39] R. Mojgani, M. Balajewicz, Low-rank registration based manifolds for convection-dominated PDEs, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 399–407.
[40] S.E. Ahmed, S.M. Rahman, O. San, A. Rasheed, I.M. Navon, Memory embedded non-intrusive reduced order modeling of non-ergodic flows, Phys. Fluids 31 (12) (2019) 126602.
[41] S. Dutta, P. Rivera-Casillas, B. Styles, M.W. Farthing, Reduced order modeling using advection-aware autoencoders, Math. Comput. Appl. 27 (3) (2022).
[42] R. Mojgani, Reduced order modeling of convection-dominated flows, dimensionality reduction and stabilization (Ph.D. thesis), University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2020, p. 140.
[43] M. Nonino, F. Ballarin, G. Rozza, Y. Maday, Overcoming slowly decaying Kolmogorov n-width by transport maps: application to model order reduction of fluid dynamics and fluid–structure interaction problems, 2019, preprint arXiv:1911.06598.
[44] B. Peherstorfer, Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling, SIAM J. Sci. Comput. 42 (5) (2020) A2803–A2836.
[45] D. Rim, B. Peherstorfer, K.T. Mandli, Manifold approximations via transported subspaces: Model reduction for transport-dominated problems, 2020, pp. 1–30, preprint arXiv:1912.13024.
[46] T. Taddei, A registration method for model order reduction: Data compression and geometry reduction, SIAM J. Sci. Comput. 42 (2) (2020) A997–A1027.
[47] S.E. Ahmed, O. San, Breaking the Kolmogorov barrier in model reduction of fluid flows, Fluids 5 (1) (2020).
[48] J. Barnett, C. Farhat, Quadratic approximation manifold for mitigating the Kolmogorov barrier in nonlinear projection-based model order reduction, 2022, pp. 1–38, arXiv preprint arXiv:2204.02462.
[49] P. Krah, S. Büchholz, M. Häringer, J. Reiss, Front transport reduction for complex moving fronts, 2022, pp. 1–26, preprint arXiv:2202.08208.
[50] J. Ren, W.R. Wolf, X. Mao, Model reduction of traveling-wave problems via radon cumulative distribution transform, Phys. Rev. Fluids 6 (2021) L082501.
[51] S. Venturi, T. Casey, SVD perspectives for augmenting DeepONet flexibility and interpretability, Comput. Methods Appl. Mech. Engrg. 403 (2023) 115718.
[52] H. Owhadi, Gaussian process hydrodynamics, 2022, preprint arXiv:2209.10707.
[53] K. Lee, K.T. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comput. Phys. 404 (2020) 108973.
[54] Y. Kim, Y. Choi, D. Widemann, T. Zohdi, A fast and accurate physics–informed neural network reduced order model with shallow masked autoencoder, J. Comput. Phys. 451 (2022) 110841.
[55] MATLAB, Version 9.0.0 (R2016a), The MathWorks Inc., Natick, Massachusetts, 2016.
[56] R. Mojgani, M. Balajewicz, Lagrangian basis method for dimensionality reduction of convection dominated nonlinear flows, 2017, preprint arXiv:1701.04343.
[57] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, preprint arXiv:1412.6980.
[58] F.M. Rohrhofer, S. Posch, C. Gößnitzer, B.C. Geiger, Understanding the difficulty of training physics-informed neural networks on dynamical systems, 2022, pp. 1–12, preprint arXiv:2203.13648.
[59] Z. Yao, A. Gholami, K. Keutzer, M.W. Mahoney, PyHessian: Neural networks through the lens of the hessian, in: 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 581–590.
[60] V. Dwivedi, B. Srinivasan, Physics informed extreme learning machine (PIELM)–a rapid method for the numerical solution of partial differential equations, Neurocomputing 391 (2020) 96–118.
[61] V.A. Dobrev, T.V. Kolev, R.N. Rieben, High-order curvilinear finite element methods for Lagrangian hydrodynamics, SIAM J. Sci. Comput. 34 (5) (2012).
[62] E.S. Oran, J.P. Boris, Numerical Simulation of Reactive Flow, 1987.
[63] Y. Bazilevs, M. Hsu, J. Kiendl, R. Wüchner, K. Bletzinger, 3D simulation of wind turbine rotors at full scale. Part II: Fluid – structure interaction modeling with composite blades, Internat. J. Numer. Methods Fluids 65 (October 2010) (2011) 236–253.
[64] R. Ramani, S. Shkoller, A fast dynamic smooth adaptive meshing scheme with applications to compressible flow, 2022, pp. 1–58, preprint arXiv:10.48550/arxiv.2205.09463.
[65] X. Liu, N.R. Morgan, E.J. Lieberman, D.E. Burton, A fourth-order Lagrangian discontinuous Galerkin method using a hierarchical orthogonal basis on curvilinear grids, J. Comput. Appl. Math. 404 (2022) 113890.
[66] C.Y. Loh, M.S. Liou, Three-dimensional steady supersonic duct flow using Lagrangian formulation, Shock Waves 3 (3) (1994) 239–248.
[67] C. Hirt, A. Amsden, J. Cook, An arbitrary Lagrangian–Eulerian computing method for all flow speeds, J. Comput. Phys. 135 (2) (1997) 203–216.
[68] H. Braess, P. Wriggers, Arbitrary Lagrangian Eulerian finite element analysis of free surface flow, Comput. Methods Appl. Mech. Engrg. 190 (1) (2000) 95–109.
[69] H. Wessels, C. Weißenfels, P. Wriggers, The neural particle method – an updated Lagrangian physics informed neural network for computational fluid dynamics, Comput. Methods Appl. Mech. Engrg. 368 (2020) 113127.
[70] J. Bai, Y. Zhou, Y. Ma, H. Jeong, H. Zhan, C. Rathnayaka, E. Sauret, Y. Gu, A general neural particle method for hydrodynamics modeling, Comput. Methods Appl. Mech. Engrg. 393 (2022) 114740.
[71] G.A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, J. Comput. Phys. 27 (1) (1978) 1–31.

[72] N.J. Nair, M. Balajewicz, Transported snapshot model order reduction approach for parametric, steady-state fluid flows containing parameter-dependent shocks, Int. J. Numer. Methods Eng. 117 (12) (2019) 1234–1262.
[73] R.J. LeVeque, Numerical Methods for Conservation Laws, Vol. 214, Springer, 1992.
[74] D. Rim, Dimensional splitting of hyperbolic partial differential equations using the radon transform, SIAM J. Sci. Comput. 40 (6) (2018) A4184–A4207.
[75] T. Taddei, L. Zhang, Space-time registration-based model reduction of parameterized one-dimensional hyperbolic PDEs, ESAIM Math. Model. Numer. Anal. 55 (1) (2021) 99–130.
[76] N. Sarna, P. Benner, Data-driven model order reduction for problems with parameter-dependent jump-discontinuities, Comput. Methods Appl. Mech. Engrg. 387 (2021) 114168.
[77] D.C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. 45 (1–3) (1989) 503–528.
[78] R. Mojgani, M. Balajewicz, Stabilization of linear time-varying reduced order models, a feedback controller approach, Internat. J. Numer. Methods Engrg. (2020).
[79] M. Stein, Large sample properties of simulations using latin hypercube sampling, Technometrics 29 (2) (1987) 143–151.
[80] J. Ash, R.P. Adams, On warm-starting neural network training, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 3884–3894.
[81] A. Subel, Y. Guan, A. Chattopadhyay, P. Hassanzadeh, Explaining the physics of transfer learning a data-driven subgrid-scale closure to a different turbulent flow, 2022, arXiv preprint arXiv:2206.03198.
[82] R. Mattey, S. Ghosh, A novel sequential method to train physics informed neural networks for Allen Cahn and Cahn Hilliard equations, Comput. Methods Appl. Mech. Engrg. 390 (2022) 114474.