ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Seminar for Applied Mathematics SAM

Weak physics informed neural networks for approximating entropy solutions of hyperbolic conservation laws

T. De Ryck and S. Mishra and R. Molinaro

Research Report No. 2022-35
July 2022

Seminar für Angewandte Mathematik
Eidgenössische Technische Hochschule
CH-8092 Zürich
Switzerland

Funding ERC: 770880

WPINNS: *WEAK* PHYSICS INFORMED NEURAL NETWORKS FOR APPROXIMATING ENTROPY SOLUTIONS OF HYPERBOLIC CONSERVATION LAWS

T. DE RYCK, S. MISHRA, AND R. MOLINARO

Abstract. Physics informed neural networks (PINNs) require regularity of solutions of the underlying PDE to guarantee accurate approximation. Consequently, they may fail at approximating discontinuous solutions of PDEs such as nonlinear hyperbolic equations. To ameliorate this, we propose a novel variant of PINNs, termed as weak PINNs (*wPINNs*) for accurate approximation of entropy solutions of scalar conservation laws. *wPINNs* are based on approximating the solution of a min-max optimization problem for a residual, defined in terms of Kruzkhov entropies, to determine parameters for the neural networks approximating the entropy solution as well as test functions. We prove rigorous bounds on the error incurred by *wPINNs* and illustrate their performance through numerical experiments to demonstrate that *wPINNs* can approximate entropy solutions accurately.

1. Introduction

Driven by their well-documented successes in fields ranging from computer vision and natural language understanding to robotics and autonomous vehicles, machine learning techniques, particularly deep learning [23], are being increasingly used in scientific computing. Given the fact that *deep neural networks* (DNNs) are universal function approximators, they have proved to be a popular choice for ansatz spaces for the construction of fast surrogates for approximating a variety of partial differential equations (PDEs) including elliptic [38, 20], parabolic [10] and hyperbolic PDEs [26, 27]. Even, the underlying solution operators can be *learned* using DNN based operator learning frameworks such as DeepONets [25] or Fourier Neural Operators [24].

All the afore-mentioned approaches fall under the rubric of *supervised learning* and require significant amounts of data for training the underlying DNNs. However, this training data is generated from either observations (experiments) or numerical simulations and can be very expensive to access and store. Hence, in many contexts, one needs a learning framework that can approximate the underlying PDE solutions with very little or possibly *no data*. Physics informed neural networks (PINNs) provide a very popular example of such an *unsupervised learning* framework. In contrast to supervised learning approaches where the mismatch between the ground truth and neural network predictions is minimized during training, training of PINNs relies on the minimization of an underlying (pointwise) residual associated with the PDE in suitable hypotheses spaces of neural networks.

PINNs were first proposed, albeit in slightly different form, in [9, 22, 21] but they were resurrected and popularized more recently in [35, 36] as an efficient alternative to traditional numerical methods, for solving both forward as well as inverse problems for PDEs. Since their reintroduction, the use of PINNs has grown exponentially in different areas of scientific computing and a very selected list of references include [37, 28, 33, 44, 14, 15, 16, 18, 31, 29, 30, 2, 41, 17, 13, 42, 31, 29, 30, 5, 6] and references therein, see also [4] for an extensive recent review of PINNs.

Although less advanced than the applications of PINNs in various domains, there has been significant development of the theory for PINNs recently, particularly in the form of rigorous bounds on various sources of the underlying error. These include [39] where the authors show consistency of PINNs with the underlying linear elliptic and parabolic PDE under stringent assumptions and in [40] where similar estimates are derived for linear advection equations. In [31, 29], the authors provided a roadmap for deriving error estimates for PINNs and applied it for both the forward and inverse problems in a variety of elliptic, parabolic and convection-dominated PDEs. This strategy was further refined and adapted to very-high dimensional Kolmogorov PDEs in [7] as well to the Navier-Stokes equations in [5], among others. The key elements of this strategy, as also identified in [8], are as follows,

i.) *Regularity* of the solutions of the underlying PDEs, which enables one to apply estimates on the error (in high-enough Sobolev norms), incurred by neural networks in approximating smooth

(T. De Ryck, S. Mishra and R. Molinaro) Seminar for Applied Mathematics, ETH Zürich, Rämistrasse 101, 8092 Zürich, Switzerland

functions. This regularity is leveraged into proving that the PDE residuals, which are to be minimized during the training process, can be made arbitrarily small.

ii.) *Coercivity* (or stability) of the underlying PDEs which ensure that the total error can be estimated in terms of the residuals. For nonlinear PDEs, the constants in these coercivity estimates often depend on the regularity of the underlying solutuions.

iii.) *Quadrature error* bounds for estimating the so-called generalization gap between the continuous and discrete versions of the PDE residual [6].

These elements also bring forth the *limitations of PINNs* by highlighting PDEs where PINNs might not provide an accurate approximation. In particular, there are a large number of contexts in which solutions of PDEs might not be smooth, even for smooth inputs. The analysis of [31, 5, 8] suggests that conventional forms of PINNs will fail to accurately approximate solutions of these PDEs.

A prototypical example of a class of PDEs for which the underlying solutions are not smooth, is provided by nonlinear hyperbolic systems of conservation laws such as the Euler, shallow-water and ideal Magneto-Hydrodynamics (MHD) equations [12]. Even in the simplest case of a *scalar conservation law*, e.g. inviscid Burgers' equation, it is well-known that, even if the initial datum is smooth, discontinuities in the form of *shock waves* form within a finite time. Hence, the underlying equation can no longer be interpreted in a (pointwise) *strong* sense, rather *weak solutions* need to be considered [12]. Moreover, these weak solutions are no longer unique. Additional admissibility criteria or *entropy conditions* have to be imposed in order to restore uniqueness [12].

As the solutions of hyperbolic conservation laws can be discontinuous, the analysis of [31, 8] and references therein suggests that PINNs cannot accurately approximating the underlying weak solutions of these PDEs. This fact is also empirically verified in [31] (Figure 6 (d)) where unacceptably large errors were observed for PINNs approximating scalar conservation laws. This can be explained by the fact that the pointwise residual blows up for smooth approximations of the underlying exact solution and minimizing it in the class of neural networks is futile.

Given this context, we ask if one can modify PINNs to design an *unsupervised learning framework* for approximating (entropy) weak solutions of hyperbolic conservation laws and related equations, such that the resulting error can be rigorously proved to be arbitrarily small. Answering this question affirmatively constitutes the central rationale of the current paper.

To this end, we will focus on the simple yet prototypical case of scalar conservation laws here. As mentioned earlier, the pointwise residuals associated with (approximations of) weak solutions can blow up. Hence, we need to replace these pointwise PDE residuals with suitable *weak* versions. This can be achieved by mimicking the weak formulation of the underlying conservation laws [12] and integrating by parts with respect to smooth *test functions* to define a weak form of the PDE residual. Such weak versions of PINNs have already been considered in the context of so-called *variational PINNs* [19], where the test functions are selected as suitable basis functions, such as orthogonal polynomials. Such an approach can indeed be considered in our context. However, we refrain from doing so here as a key advantage for PINNs is that it is a *meshless* approach and does not require any underlying grid. However, variational PINNs are often defined in terms of locally supported functions on a mesh. Instead, we will leverage the universal approximation properties of neural networks and choose parametrized neural networks as our test functions. Moreover, neural networks are also used as approximations to the underlying solution i.e., as trial functions. Hence, our weak formulation leads to a *min-max* optimization problem where the neural network parameters (weights and biases) are maximized with respect to test functions and minimized with respect to trial functions. Such min-max problems arise in machine learning when training generative adversarial networks or GANs, [11, 1] and references therein.

However, working with the weak formulation alone does not suffice to build an accurate approximation strategy for scalar conservation laws as one also needs to incorporate entropy conditions. To this end, we will define a novel *entropy residual*, based on the well-known family of *Kruzkhov entropies* [12] and solve the corresponding min-max optimization problem for training the neural networks that will approximate the entropy solution accurately. We term the resulting construction as *wPINNs* or weak PINNs and prove rigorous error bounds on them. In particular, we will prove that the entropy residuals, associated with *wPINNs* can be made arbitrarily small. We rely on Kruzkhov's method of *doubling of variables* to prove that the error of the *wPINN* with respect to the entropy solution can be bounded in terms of the (continuous version) of the entropy residual. Finally, statistical learning theory arguments, already considered in the context of PINNs in [6], are adapted to yield a bound on the generalization gap. Thus, taken together, these bounds provide rigorous estimates on the error for *wPINNs* and prove that *wPINNs* can approximate entropy solutions of scalar conservation laws accurately. We also provide numerical experiments to illustrate the efficient approximation of entropy solutions by *wPINNs*. Hence,

we design a novel variant of PINNs to approximate discontinuous entropy solutions of conservation laws and prove that the approximation is accurate.

2. *wPINNs* for Scalar conservation laws

In this section, we will describe the construction of *wPINNs* for approximating the entropy solutions of scalar conservation laws. We start by recapitulating some basic concepts about conservation laws.

2.1. **Scalar Conservation Laws.** For simplicity of exposition, we will focus on the case of one spatial dimension while mentioning that extending the results to several space dimensions is straightforward. Without loss of generality, we fix $D = [0,1] \subset \mathbb{R}$ as the spatial domain and consider the scalar conservation law,

$$\begin{aligned} u_t + f(u)_x &= 0 \quad \text{in } D \times [0,T] \\ u &= u_0 \quad \text{on } D \times \{0\}. \end{aligned} \tag{2.1}$$

Here, $u \in L^1(D \times (0,T))$ is the conserved quantity and f is the so-called flux function with u_0 being the initial data. Moreover, the PDE (2.1) needs to be supplemented with suitable boundary conditions. We mostly consider periodic boundary conditions in this paper.

Following [12], one defines *weak solutions* of (2.1) as follows,

Definition 2.1. *A function $u \in L^\infty(\mathbb{R} \times \mathbb{R}_+)$ is a weak solution of (2.1) with initial data $u_0 \in L^\infty(\mathbb{R})$ if*

$$\int_{\mathbb{R}_+} \int_{\mathbb{R}} \left(u\varphi_t + f(u)\varphi_x \right) dxdt + \int_{\mathbb{R}} u_0(x)\varphi(x,0)dx = 0, \tag{2.2}$$

holds for all test functions $\varphi \in C_c^1(\mathbb{R} \times \mathbb{R}_+)$.

However, weak solutions are not unique [12]. To recover uniqueness, one needs to impose additional admissibility criteria or *entropy conditions*. To this end, we consider the so-called *Kruzkhov entropy functions*, given by $|u-c|$, for any $c \in \mathbb{R}$ and the resulting entropy flux functions,

$$\partial_t |u-c| + \partial_x Q[u;c] \leq 0 \qquad \text{where} \quad Q : \mathbb{R}^2 \to \mathbb{R} : (u,c) \mapsto \operatorname{sgn}(u-c)\left(f(u) - f(c)\right). \tag{2.3}$$

With this notation, we have the following definition of *entropy solutions*,

Definition 2.2. *We say that a function $u \in L^\infty(\mathbb{R} \times \mathbb{R}_+)$ is an entropy solution of (2.1) with initial data $u_0 \in L^\infty(\mathbb{R})$ if u is a weak solution of (2.1) and if u satisfies that*

$$\int_0^T \int_{\mathbb{R}} \left(|u-c|\varphi_t + Q[u;c]\varphi_x \right) dxdt - \int_{\mathbb{R}} \Big(\big|u(x,T) - c\big|\varphi(x,T) - \big|u_0(x) - c\big|\varphi(x,0) \Big) dx \geq 0 \tag{2.4}$$

for all $\varphi \in C_c^1(\mathbb{R} \times \mathbb{R}_+)$, $c \in \mathbb{R}$ and $T > 0$.

It holds that these entropy solutions are unique and continuous in time, as formulated below [12], where $\|\cdot\|_{TV}$ denotes the total variation seminorm.

Theorem 2.3. *Assume that $f \in C^1$ and $u_0 \in L^\infty \cap L^1$. Then there exists a unique entropy solution u of (2.1) and if $\|u_0\|_{TV} < \infty$ then u satisfies the following,*

$$\big\|u(t) - u(s)\big\|_{L^1} \leq |t-s| M \|u_0\|_{TV} \qquad \text{and} \qquad \big\|u(t)\big\|_{L^\infty} \leq \|u_0\|_{L^\infty}, \ \big\|u(t)\big\|_{BV} \leq \|u_0\|_{BV}, \tag{2.5}$$

where $M = M(u_0) = \max_{\operatorname{ess\,inf}_x u_0(x) \leq u \leq \operatorname{ess\,sup}_x u_0(x)} \big|f'(u)\big|$.

2.2. **Neural networks.** Our aim in this paper is to approximate entropy solutions of (2.1) with neural networks, which we formally define next.

Definition 2.4. *Let $R \in (0,\infty]$, $L, W \in \mathbb{N}$ and $l_0, \dots, l_L \in \mathbb{N}$. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a measurable function, the so-called* activation function*, and define*

$$\Theta = \Theta_{L,W,R} := \bigcup_{L' \in \mathbb{N}, L' \leq L} \ \bigcup_{l_0,\dots,l_L \in \{1,\dots,W\}} \ \bigtimes_{k=1}^{L'} \left([-R,R]^{l_k \times l_{k-1}} \times [-R,R]^{l_k} \right). \tag{2.6}$$

For $\theta \in \Theta_{L,W,R}$, we define $(W_k, b_k) := \theta_k$ and $\mathcal{A}_k^\theta : \mathbb{R}^{l_{k-1}} \to \mathbb{R}^{l_k} : z \mapsto W_k z + b_k$ for $1 \leq k \leq L$ and we denote by $u_\theta : \mathbb{R}^{l_0} \to \mathbb{R}^{l_L}$ the function that satisfies for all $z \in \mathbb{R}^{l_0}$ that

$$u_\theta(z) = \left(\mathcal{A}_L^\theta \circ \sigma \circ \mathcal{A}_{L-1}^\theta \circ \cdots \circ \sigma \circ \mathcal{A}_1^\theta \right)(z), \tag{2.7}$$

where in the setting of approximating solutions to PDEs we set $l_0 = d+1$ and $z = (x,t)$.

We refer to u_θ as the realization of the neural network associated to the parameter θ with L layers with widths $(l_0, l_1, \dots, l_L)$, of which the middle $L-1$ layers are called hidden layers. For $1 \leq k \leq L$, we say that layer k has width l_k i.e., we say that it consists of l_k neurons, and we refer to W_k and b_k as the weights and biases corresponding to layer k. If $L \geq 3$, we say that u_θ is a deep neural network (DNN). The total number of neurons in the network is given by the sum of the layer widths, $\sum_{k=0}^{L} l_k$. Note that the weights and biases of neural network u_θ with $\theta \in \Theta_{L,W,R}$ are bounded by R.

2.3. **Physics informed neural networks (PINNs).** We briefly introduce PINNs and how these neural networks could be used to approximate the solutions of the scalar conservation law (2.1). Given that we are on a finite domain, we (formally) write down the boundary conditions supplementing (2.1) as, $u = g$ on $\partial D \times [0,T]$. Then, one can consider the following (pointwise) residuals associated with the PDE (2.1) and initial (and boundary) conditions,

$$r_{int}[v](x,t) = v_t + f(v)_x, \qquad r_{sb}[v](y,t) = v(y,t) - g(y,t), \qquad r_{tb}[v](x) = v(x,0) - u_0(x), \tag{2.8}$$

where $v \in C^1(D \times [0,T])$ and where r_{int} is the PDE residual, r_{sb} is the (spatial) boundary residual and r_{tb} is the temporal boundary residual stemming from the initial condition. Ideally, one would then solve the following minimization problem,

$$\widehat{\theta} := \underset{\theta \in \Theta}{\operatorname{argmin}} \left(\big\| r_{int}[u_\theta] \big\|^2_{L^2(D\times[0,T])} + \lambda_{sb} \big\| r_{sb}[u_\theta] \big\|^2_{L^2(\partial D \times [0,T])} + \lambda_{tb} \big\| r_{tb}[u_\theta] \big\|^2_{L^2(D)} \right), \tag{2.9}$$

and define the PINN as $u_{\widehat{\theta}}$, with θ in (2.9) denoting tunable parameters (weights and biases) of the underlying neural networks (2.7). In practice, one cannot exactly solve the above minimization problem. Instead, one approximates the integrals in (2.9) using a numerical quadrature and one tries to find an approximate minimizer using a (stochastic) gradient descent algorithm.

One can already observe from the form of the PDE residual in (2.8) that it is imposed pointwise. Although well-defined as long as the activation function σ in (2.7) is smooth, one expects this residual to blow up as the residual corresponding to smooth approximations of entropy solutions of conservation laws, such as viscous profiles, blows up as the profile width vanishes [12, 31]. Hence, we cannot expect that PINNs will approximate weak solutions of (2.1) accurately, particularly near shocks. This is indeed observed empirically in [31] (Figure 6 (d)).

2.4. **Weak PINNs (*wPINNs*).** As mentioned in the introduction, we will circumvent the failure of conventional PINNs that minimized the pointwise PDE residual (2.9) by searching for neural networks that minimize a residual, related to the Kruzkhov entropy condition instead. To this end, we define for $v \in (L^\infty \cap L^1)(D \times [0,T])$, $\varphi \in W^{1,\infty}_0(D \times [0,T])$ and $c \in \mathbb{R}$ the following *Kruzhov entropy residual*,

$$\mathcal{R}(v, \varphi, c) := -\int_D \int_{[0,T]} \Big(\big| v(x,t) - c \big| \partial_t \varphi(x,t) + Q[v(x,t); c] \partial_x \varphi(x,t) \Big) \, dx dt. \tag{2.10}$$

Note that if u is an entropy solution of (2.1), then it holds that $\mathcal{R}(u, \varphi, c) \leq 0$. This suggests solving the following min-max counterpart to the PINN minimization problem (2.9),

$$\theta^* := \underset{\theta \in \Theta}{\operatorname{argmin}} \max_{\varphi \in W^{1,\infty}_0(D\times[0,T]), c \in \mathbb{R}} \left(\mathcal{R}(u_\theta, \varphi, c) + \lambda_{sb} \big\| r_{sb}[u_\theta] \big\|^2_{L^2(\partial D \times [0,T])} + \lambda_{tb} \big\| r_{tb}[u_\theta] \big\|^2_{L^2(D)} \right), \tag{2.11}$$

The DNN associated with the u_{θ^*} is defined as the weak PINN, *wPINN* for short, and we will show that it provides an accurate approximation of the entropy solution u of (2.1).

We observe that in practice, the test function space $W^{1,\infty}_0$ needs to be replaced by a finite-dimensional approximation. One possibility is to use locally supported (piecewise) polynomials or orthogonal polynomials such as Legendre polynomials. Such choices lead to what is often termed as variational PINNs. However, we will depart from this choice and leverage the universal approximation property of neural networks to approximate $W^{1,\infty}_0$ by parametrized neural networks of the form (2.7) with a C^1-activation function. This leads to a min-max optimization problem where maximum as well as the minimum are taken with respect to neural network training parameters. We refer to Section 4 for details on how the min-max problem (2.11) can be solved in practice.

3. Error Analysis for *wPINNs*

In this section, we will provide rigorous bounds on different components of the error incurred by *wPINNs* in approximating the entropy solution of the conservation law (2.1).

3.1. **Bounds on the Entropy Residual.** We start the rigorous analysis of *wPINNs* by following the layout of [6] and asking if the entropy residual (2.10) can be made arbitrarily small, within the class of neural networks. An affirmative answer to this question will confirm that the choice of the loss function for *wPINN* is suitable. To investigate this question, we start with the following observation,

Lemma 3.1. *Let $f : \mathbb{R} \to \mathbb{R}$ be Lipschitz with constant L_f, let Q be as in (2.3) and let $c, u, v \in \mathbb{R}$. It holds that $\big|Q[u;c] - Q[v;c]\big| \leq 3L_f|u-v|$.*

Proof. We calculate,

$$\begin{aligned} Q[u;c] - Q[v;c] &= \operatorname{sgn}(u-c)\,(f(u)-f(c)) - \operatorname{sgn}(v-c)\,(f(v)-f(c)) \\ &= (\operatorname{sgn}(u-c) - \operatorname{sgn}(v-c))(f(u)-f(c)) + \operatorname{sgn}(v-c)\,(f(u)-f(v)) \\ &=: T_1 + T_2. \end{aligned} \tag{3.1}$$

Note that if $\operatorname{sgn}(u-c) \neq \operatorname{sgn}(v-c)$ then either $u < c < v$ or $v < c < u$ pointwise almost everywhere. Consequently, if $\operatorname{sgn}(u-c) \neq \operatorname{sgn}(v-c)$ then $|u-c| \leq |u-v|$. We therefore find that

$$|T_1| \leq L_f\big|\operatorname{sgn}(u-c) - \operatorname{sgn}(v-c)\big||u-c| \leq 2L_f|u-v|. \tag{3.2}$$

Moreover, it holds that $|T_2| \leq L_f|u-v|$. In total we therefore have that $\big|Q[u;c] - Q[v;c]\big| \leq 3L_f|u-v|$. □

Next, we prove that for any test function $\varphi \in W_0^{1,\infty}$, any $q \geq 1$ and any $c \in \mathbb{R}$ the quantity $\mathcal{R}(v,\varphi,c)$ can be bounded in terms of the L^q-norm of $v-u$.

Theorem 3.2. *Let $p, q > 1$ be such that $\frac{1}{p} + \frac{1}{q} = 1$ or let $p = \infty$ and $q = 1$. Let u be the entropy solution of (2.1) and let $v \in L^q(D \times [0,T])$. Assume that f has Lipschitz constant L_f. Then it holds for any $\varphi \in W_0^{1,\infty}(D \times [0,T])$ that*

$$\mathcal{R}(v,\varphi,c) \leq (1+3L_f)|\varphi|_{W^{1,p}(D\times[0,T])}\|u-v\|_{L^q(D\times[0,T])}. \tag{3.3}$$

Proof. Let c and φ be arbitrary. In the following, we will write L^p for $L^p(D \times [0,T])$.

$$\begin{aligned} \mathcal{R}(v,\varphi,c) &= \mathcal{R}(v,\varphi,c) - \mathcal{R}(u,\varphi,c) + \mathcal{R}(u,\varphi,c) \\ &\leq \mathcal{R}(v,\varphi,c) - \mathcal{R}(u,\varphi,c) \\ &= \int_D \int_{[0,T]} \Big(\big|u(x,t)-c\big| - \big|v(x,t)-c\big|\Big)\,\partial_t\varphi(x,t) + \big(Q[u(x,t);c] - Q[v(x,t);c]\big)\,\partial_x\varphi(x,t)dxdt \\ &\leq \|\partial_t\varphi\|_{L^p}\|u-v\|_{L^q} + \|\partial_x\varphi\|_{L^p}\big\|Q[u;c] - Q[v;c]\big\|_{L^q} \end{aligned} \tag{3.4}$$

Using Lemma 3.1 we find that $\big\|Q[u;c] - Q[v;c]\big\|_{L^q} \leq 3L_f\|u-v\|_{L^q}$ and therefore,

$$\mathcal{R}(v,\varphi,c) \leq (1+3L_f)|\varphi|_{W^{1,p}(D\times[0,T])}\|u-v\|_{L^q(D\times[0,T])}. \tag{3.5}$$

□

The above result implies that we have to show that the entropy solution of (2.1) can be approximated by neural networks (2.7) for the entropy residual (2.10) to be sufficiently small, within the class of neural networks. In the following, we focus on the hyperbolic tangent (tanh) activation function,

$$\sigma(x) := \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}, \tag{3.6}$$

for the sake of definiteness, while observing that extensions to other smooth activation functions can be similarly considered. We have the following result on the approximation of BV-functions by tanh neural networks,

Lemma 3.3. *For every $u \in BV([0,1] \times [0,T])$ and $\varepsilon > 0$ there is a tanh neural network $\widehat{u}$ with two hidden layers and at most $O(\varepsilon^{-2})$ neurons such that*

$$\|u - \widehat{u}\|_{L^1([0,1]\times[0,T])} \leq \varepsilon. \tag{3.7}$$

Proof. Write $\Omega = [0,1] \times [0,T]$. For every $u \in BV(\Omega)$ and $\varepsilon > 0$ there exists a function $\widetilde{u} \in C^\infty(\Omega) \cap BV(\Omega)$ such that $\|u - \widetilde{u}\|_{L^1(\Omega)} \lesssim \varepsilon$ and $\|\nabla\widetilde{u}\|_{L^1(\Omega)} \lesssim \|u\|_{BV(\Omega)} + \varepsilon$ [3]. Then use the approximation techniques of [6, 5] and the fact that $\|\widetilde{u}\|_{W^{1,1}(\Omega)}$ can be uniformly bounded in ε to find the existence of a tanh neural network $\widehat{u}$ with two hidden layers and at most $O(\varepsilon^{-2})$ neurons that satisfies the mentioned error estimate. □

If one additionally knows that u is piecewise smooth, for instance as in the solutions of convex scalar conservation laws [12], then one can use the results of [34] to obtain the following result.

Lemma 3.4. *Let $m, n \in \mathbb{N}$, $1 \leq q < \infty$ and let $u : [0,1] \times [0,T] \to \mathbb{R}$ be a function that is piecewise C^m smooth and with a C^n smooth discontinuity surface. Then there is a tanh neural network $\widehat{u}$ with at most three hidden layers and $\mathcal{O}(\varepsilon^{-2/m} + \varepsilon^{-q/n})$ neurons such that*

$$\|u - \widehat{u}\|_{L^q([0,1]\times[0,T])} \leq \varepsilon. \tag{3.8}$$

Proof. The proof follows the lines of [34, Appendix A] with a few adaptations. We approximate the Heaviside function by the function $x \mapsto \frac{1}{2}(\sigma(\beta x) + 1)$, where $\beta = \frac{1}{2\varepsilon}\ln(1/\varepsilon)$ for some $\varepsilon > 0$. This replaces [34, Lemma A.2]. Moreover, we replace [34, Lemma A.3], which discusses approximation rates for the multiplication operator, by [6, Corollary 3.7] and we replace [34, Lemma A.9], which discusses approximation rates for smooth functions, by [6, Theorem 5.1]. □

Finally, from Lemma A.1, stated and proved in the Appendix, we find that one has to consider test functions φ that might grow as $|\varphi|_{W^{1,p}} \sim \varepsilon^{-3(1+2(p-1)/p)}$. Consequently, we will need to use Lemma 3.3 with $\varepsilon \leftarrow \varepsilon^{-4+6(p-1)/p}$, leading to the following corollary.

Corollary 3.5. *Assume the setting of Lemma 3.4, assume that $mq > 2n$ and let $p \in [1, \infty]$ be such that $\frac{1}{p} + \frac{1}{q} = 1$. There is a fixed depth tanh neural network $\widehat{u}$ with size $\mathcal{O}(\varepsilon^{-(4q+6)n})$ such that*

$$\max_{c\in\mathcal{C}} \sup_{\varphi\in\overline{\Phi}_\varepsilon} \mathcal{R}(\widehat{u}, \varphi, c) + \lambda_{sb}\big\|r_{sb}[\widehat{u}]\big\|^2_{L^2(\partial D\times[0,T])} + \lambda_{tb}\big\|r_{tb}[\widehat{u}]\big\|^2_{L^2(D)} \leq \varepsilon, \tag{3.9}$$

where $\overline{\Phi}_\varepsilon = \{\varphi \,:\, |\varphi|_{W^{1,p}} = \mathcal{O}(\varepsilon^{-3(1+2(p-1)/p))}\}$.

Proof. We find that $\widehat{u}$ will need to have a size of $\mathcal{O}(\varepsilon^{-(4q+6)/\beta})$ such that $\max_{c\in\mathcal{C}} \sup_{\varphi\in\Phi_\varepsilon} \mathcal{R}(\widehat{u}, \varphi, c) \leq \varepsilon/3$, where we used that $q = p/(p-1)$. Since in the proof of Lemma 3.4 the network $\widehat{u}$ is constructed as an approximation of piecewise Taylor polynomials, the spatial and temporal boundary residuals (r_{sb} and r_{tb}) are automatically minimized as well, given that Taylor polynomials provide approximations in C^0-norm. □

3.2. **Bounds on Error in terms of Residuals.** In the previous subsection (Corollary 3.5), we showed that the residuals appearing in the loss function (2.11) can be made arbitrarily small. Hence, this loss function constitutes a suitable target for the training process. However, as argued in [6], the smallness of residual, does not necessarily imply that the total error, with respect to PINNs, can be made arbitrarily small as the optimization process might converge to local saddle-points of (2.11). Hence, we need to bound the error in terms of the corresponding residuals. We obtain such bounds in this section.

To start with, we define the following set of test functions,

Definition 3.6. *Let for any $(y, s) \in [0,1] \times [0,T]$ and $\varepsilon > 0$ the function $\overline{\varphi}^{y,s}_\varepsilon : [0,1] \times [0,T] \to [0,\infty)$ be given by,*

$$\begin{aligned}
\overline{\varphi}^{y,s}_\varepsilon(x,t) &= \chi_\varepsilon\left(\frac{t+s}{2}\right)\rho_\varepsilon(x-y)\rho_\varepsilon(t-s),\\
\chi_\varepsilon(t) &= \frac{1}{2\sigma(\alpha\varepsilon)}(\sigma(\alpha(t-2\varepsilon)) - \sigma(\alpha(t-T+2\varepsilon))), \qquad \alpha = 3\ln(1/\varepsilon)/\varepsilon,\\
\rho_\varepsilon(x) &= \frac{\sigma(\beta(x+\varepsilon^6)) - \sigma(\beta(x-\varepsilon^6))}{2\varepsilon^6}, \qquad \beta = 9\ln(1/\varepsilon)/\varepsilon^3,
\end{aligned} \tag{3.10}$$

for $(x,t) \in [0,1] \times [0,T]$. Furthermore we define the set Φ_ε by,

$$\Phi_\varepsilon = \left\{\overline{\varphi}^{y,s}_\varepsilon \,:\, (y,s) \in [0,1] \times [0,T]\right\}. \tag{3.11}$$

Now, we will modify the famous doubling of variables argument of Kruzkhov to obtain the following bound on the L^1-error with *wPINNs*,

Theorem 3.7. *Assume that u is the piecewise smooth entropy solution of (2.1) with essential range $\mathcal{C}$ and that $u(0,t) = u(1,t)$ for all $t \in [0,T]$. There is a constant $C > 0$ such that for every $\varepsilon > 0$ and $v \in C^1(D \times [0,T])$, it holds that*

$$\begin{aligned}
\int_0^1 \big|v(x,T) - u(x,T)\big|dx \leq C\Bigg(&\int_0^1 \big|v(x,0) - u(x,0)\big|dx + \max_{c\in\mathcal{C},\varphi\in\Phi_\varepsilon} \mathcal{R}(v,\varphi,c)\\
&+ (1 + \|v\|_{C^1})\ln(1/\varepsilon)^3\varepsilon + \int_0^T \big|v(1,t) - v(0,t)\big|dt\Bigg).
\end{aligned} \tag{3.12}$$

Proof. Since u is an entropy solution, inequality (2.4) in Definition 2.2 is satisfied in particular for $c \leftarrow v(y,s)$ for any $(y,s) \in D \times [0,T]$. We now apply Lemma A.6 with $z \leftarrow \big|u(x,t) - v(y,s)\big|$ to obtain that,

$$\begin{aligned} &-\int_0^1\int_0^T\int_0^1\int_0^T \big|u(x,t)-v(y,s)\big|\partial_t \overline{\varphi}_\varepsilon^{y,s}(x,t)dtdxdsdy \\ &\leq \int_0^1\int_0^T\int_0^1\int_0^T \partial_t\big|u(x,t)-v(y,s)\big|\overline{\varphi}_\varepsilon^{y,s}(x,t)dtdxdsdy + CB\varepsilon, \end{aligned} \tag{3.13}$$

and Lemma A.7 to obtain that,

$$\begin{aligned} &-\int_0^1\int_0^T\int_0^1\int_0^T Q[u(x,t);v(y,s)]\partial_x \overline{\varphi}_\varepsilon^{y,s}(x,t)dtdxdsdy \\ &\leq \int_0^1\int_0^T\int_0^1\int_0^T \partial_x Q[u(x,t);v(y,s)]\overline{\varphi}_\varepsilon^{y,s}(x,t)dtdxdsdy \\ &\quad + 12L_f\int_0^T \big|v(1,t)-v(0,t)\big|dt + CL_f\|v_x\|_\infty(1-\ln(\varepsilon))\varepsilon, \end{aligned} \tag{3.14}$$

Next, we observe that for $v \in C^1(D\times[0,T])$ and $(y,s),(x,t) \in D\times[0,T]$ it holds that,

$$\mathcal{R}(v,\overline{\varphi}_\varepsilon^{x,t},u(x,t)) \leq \max_{c,\varphi}\mathcal{R}(v,\varphi,c), \tag{3.15}$$

where $\mathcal{C}$ is the essential range of u. As a result, we find using the symmetry of Q that,

$$\begin{aligned} &-\int_0^1\int_0^T\int_0^1\int_0^T \Big(\big|u(x,t)-v(y,s)\big|\partial_s\overline{\varphi}_\varepsilon^{x,t}(y,s) + Q[u(x,t);v(y,s)]\partial_y\overline{\varphi}_\varepsilon^{x,t}(y,s)\Big)\,dsdydtdx \\ &\leq T\max_{c,\varphi}\mathcal{R}(v,\varphi,c). \end{aligned} \tag{3.16}$$

Summing (3.13), (3.14) and (3.16) and using Fubini's theorem and the fact that $\overline{\varphi}_\varepsilon^{x,t}(y,s) = \overline{\varphi}_\varepsilon^{y,s}(x,t)$ leads us to,

$$\begin{aligned} &-\int_D\int_{[0,T]}\int_D\int_{[0,T]} \big|v(x,t)-u(y,s)\big|(\partial_t\overline{\varphi}_\varepsilon^{y,s}(x,t)+\partial_s\overline{\varphi}_\varepsilon^{y,s}(x,t))dxdtdyds \quad =:T_1 \\ &-\int_D\int_{[0,T]}\int_D\int_{[0,T]} Q[v(x,t);u(y,s)](\partial_x\overline{\varphi}_\varepsilon^{y,s}(x,t)+\partial_y\overline{\varphi}_\varepsilon^{y,s}(x,t))dxdtdyds \quad =:T_2 \\ &\leq T\max_{c\in\mathcal{C},\varphi\in\Phi}\mathcal{R}(v,\varphi,c) + C(1+\|v_x\|_\infty)(1-\ln(\varepsilon))\varepsilon + C\int_0^T\big|v(1,t)-v(0,t)\big|dt. \end{aligned} \tag{3.17}$$

One can observe that $T_2 = 0$ since $\partial_x\overline{\varphi}_\varepsilon^{y,s} = -\partial_y\overline{\varphi}_\varepsilon^{y,s}$. In what follows, we will prove that T_1 is a good approximation of $\big\|v(T)-u(T)\big\|_{L^1} - \big\|v(0)-u(0)\big\|_{L^1}$. By replacing the absolute value $|\cdot|$ in T_1 by its smoothened counterpart $|\cdot|_\eta$, $\eta \geq 0$, (as defined in Lemma A.5) and by using the definition of $\overline{\varphi}_\varepsilon^{y,s}(x,t)$ we find that $T_1 = T_1^0$ where,

$$T_1^\eta := -\int_D\int_{[0,T]}\int_D\int_{[0,T]} \big|v(y,s)-u(x,t)\big|_\eta\chi_\varepsilon'\left(\frac{t+s}{2}\right)\rho_\varepsilon(x-y)\rho_\varepsilon(t-s)dxdtdyds. \tag{3.18}$$

Using Lemma A.5, we find for $\eta > 0$ that $\big|T_1 - T_1^\eta\big| \leq C\eta$ for some absolute constant $C > 0$. For every x,y,t, we now want to apply Lemma A.4 with $f \leftarrow (s \mapsto \big|v(y,s)-u(x,t)\big|_\eta\chi_\varepsilon'\left(\frac{t+s}{2}\right))$. Define $\omega(t) = \sigma(\beta(t-\max\{0,t-\varepsilon^3\})) - \sigma(\beta(t-\min\{b,t+\varepsilon^3\}))$. We find using Lemma A.5 that

$$\begin{aligned} \Big|T_1^\eta - \widetilde{T}_1^\eta\Big| &:= \left|T_1^\eta + \int_D\int_{[0,T]}\int_D \big|v(y,t)-u(x,t)\big|_\eta\chi_\varepsilon'(t)\,\rho_\varepsilon(x-y)\omega(t)dxdydt\right| \\ &\leq 10(4B\alpha^2+4\|v_t\|_\infty)(T-3\ln(\varepsilon))\varepsilon^3. \end{aligned} \tag{3.19}$$

Next, for every x,t, we now want to apply Lemma A.4 with $f \leftarrow (y \mapsto \big|v(y,t)-u(x,t)\big|_\eta)$. We find using Lemma A.5 that,

$$\begin{aligned} \Big|\widetilde{T}_1^\eta - \widehat{T}_1^\eta(D)\Big| &:= \left|\widetilde{T}_1^\eta + \int_D\int_{[0,T]} \big|v(x,t)-u(x,t)\big|_\eta\omega(x)\omega(t)\chi_\varepsilon'(t)\,dxdt\right| \\ &\leq 10\cdot 4\|v_t\|_\infty(1-3\ln(\varepsilon))\varepsilon^3. \end{aligned} \tag{3.20}$$

Let $A = \{x \in D|$ the function $[0, 3\varepsilon] \cup [T - 3\varepsilon, T] \to \mathbb{R} : t \mapsto u(x,t)$ is cont. diff.$\}$. As u is piecewise smooth, we find for some constant $C > 0$ that $\left|\widehat{T}_1^\eta(D) - \widehat{T}_1^\eta(A)\right| \leq C\varepsilon$. We can now use Lemma A.3 for every $x \in A$ with $f \leftarrow (t \mapsto \left|v(x,t) - u(x,t)\right|_\eta \omega(t))$. Let $B = [\varepsilon, 3\varepsilon] \cup [T - 3\varepsilon, T - \varepsilon]$. Note that $\omega(t)$ is constant on B under the assumption that $\varepsilon < 1$ and therefore $\varepsilon^3 < \varepsilon$. This gives us, writing $T_\varepsilon := T - 2\varepsilon$,

$$\begin{aligned}
&\left|\widehat{T}_1^\eta(A) - \mathcal{T}_1^\eta(A)\right| := \\
&= \left|\widehat{T}_1^\eta(A) - \int_A \left|v(x,T_\varepsilon) - u(y,T_\varepsilon)\right|_\eta \omega(x)\omega(T_\varepsilon)dx + \int_A \left|v(x,2\varepsilon) - u(y,2\varepsilon)\right|_\eta \omega(x)\omega(2\varepsilon)dx\right| \\
&\leq C(T|\mathcal{C}| + (\|v_t\|_\infty + |u|_{W^{1,\infty}(B)})\ln\big(1/\varepsilon\big))\frac{\varepsilon}{1-\varepsilon}.
\end{aligned} \tag{3.21}$$

We find also that $\left|\mathcal{T}_1^\eta(A) - \mathcal{T}_1^\eta(D)\right| \leq C\varepsilon$. Next, using the time continuity of entropy solutions (Lemma 2.3),

$$\begin{aligned}
\left\|v(T) - u(T)\right\|_{L^1} &\leq \left\|v(T) - v(T_\varepsilon)\right\|_{L^1} + \left\|v(T_\varepsilon) - u(T_\varepsilon)\right\|_{L^1} + \left\|u(T_\varepsilon) - u(T)\right\|_{L^1} \\
&\leq 2|D|\|v_t\|_\infty \varepsilon + \left\|v(T_\varepsilon) - u(T_\varepsilon)\right\|_{L^1} + 2\varepsilon M\|u_0\|_{TV},
\end{aligned} \tag{3.22}$$

where $M = \max_{c\in\mathcal{C}} \left|f'(c)\right|$. Similarly,

$$\begin{aligned}
\left\|v(2\varepsilon) - u(2\varepsilon)\right\|_{L^1} &\leq \left\|v(2\varepsilon) - v(0)\right\|_{L^1} + \left\|v(0) - u(0)\right\|_{L^1} + \left\|u(0) - u(2\varepsilon)\right\|_{L^1} \\
&\leq 2|D|\|v_t\|_\infty \varepsilon + \left\|v(0) - u(0)\right\|_{L^1} + 2\varepsilon M\|u_0\|_{TV}.
\end{aligned} \tag{3.23}$$

We can now bring everything together. We will only quantify the dependence on v and aggregate all other constants in the constant $C > 0$, which we update progressively. If we set $\eta \leftarrow \varepsilon$ we find that,

$$\begin{aligned}
\left\|v(T) - u(T)\right\|_{L^1} &\leq C\left(\left\|v(0) - u(0)\right\|_{L^1} + T_1 + \eta + (\alpha^2 + \|v_t\|_\infty)\ln\Big(1/\varepsilon^3\Big)\frac{\varepsilon^3}{1-\varepsilon}\right) \\
&\leq C\left(\left\|v(0) - u(0)\right\|_{L^1} + T_1 + (1 + \|v_t\|_\infty)\ln\big(1/\varepsilon\big)^3\varepsilon\right).
\end{aligned} \tag{3.24}$$

Combining this with (3.16) and the fact that $T_2 = 0$ brings us,

$$\begin{aligned}
\int_0^1 \left|v(x,T) - u(x,T)\right|dx \leq C\Bigg(&\int_0^1 \left|v(x,0) - u(x,0)\right|dx + \max_{c\in\mathcal{C},\varphi\in\Phi_\varepsilon} \mathcal{R}(v,\varphi,c) \\
&+ (1 + \|v\|_{C^1})\ln\big(1/\varepsilon\big)^3\varepsilon + \int_0^T \left|v(1,t) - v(0,t)\right|dt\Bigg).
\end{aligned} \tag{3.25}$$

□

Remark 3.8. *In experiments (cf. Section 4), one can replace $\mathcal{R}$ with the following alternative,*

$$\widetilde{\mathcal{R}}(v,\varphi,c) := \int_D \int_{[0,T]} \Big(\varphi(x,t)\partial_t\left|v(x,t) - c\right| - Q[v(x,t);c]\partial_x\varphi(x,t)\Big)\,dxdt. \tag{3.26}$$

The only difference with $\mathcal{R}$ is that in $\widetilde{\mathcal{R}}$ the time derivative is with $\left|v(x,t) - c\right|$ and not with the test function. Because of Lemma A.6 it holds that $\left|\widetilde{\mathcal{R}}(v,\varphi,c) - \mathcal{R}(v,\varphi,c)\right| = \mathcal{O}(\varepsilon)$ if $\varphi \in \Phi_\varepsilon$.

3.3. **Bounds on the Generalization Gap.** The main point of Theorem 3.7 was to provide an upper bound on the L^1-error in terms of the residuals. However, in practice, one cannot evaluate the integrals in the residuals, for instance in (2.10), exactly and has to resort to quadrature. To this end, we consider the simplest case of random (Monte Carlo) quadrature and generate a set of collocation points, $\mathcal{S} = \{(x_i,t_i)\}_{i=1}^M \subset D \times [0,T]$, where all (x_i,t_i) are iid drawn from the uniform distribution on $D \times [0,T]$. For a fixed $\theta \in \Theta$, $\varphi \in \Phi_\varepsilon$, $c \in \mathcal{C}$ and for this data set $\mathcal{S}$ we can then define the *training error*,

$$\begin{aligned}
\mathcal{E}_T(\theta,\mathcal{S},\varphi,c) = &-\frac{T}{M}\sum_{i=1}^M \Big(\left|u_\theta(x_i,t_i) - c\right|\partial_t\varphi(x_i,t_i) + Q[u_\theta(x_i,t_i);c]\partial_x\varphi(x_i,t_i)\Big) \\
&+ \frac{T}{M}\sum_{i=1}^M \left|u_\theta(x_i,0) - u(x_i,0)\right| + \frac{T}{M}\sum_{i=1}^M \left|u_\theta(0,t_i) - u_\theta(1,t_i)\right|.
\end{aligned} \tag{3.27}$$

During training, one then aims to obtain neural network parameters $\theta_\mathcal{S}^*$, a test function $\varphi_\mathcal{S}^*$ and a scalar $c_\mathcal{S}^*$ such that

$$\mathcal{E}_T(\theta_\mathcal{S}^*,\mathcal{S},\varphi_\mathcal{S}^*,c_\mathcal{S}^*) \approx \min_{\theta\in\Theta}\max_{\varphi\in\Phi_\varepsilon}\max_{c\in\mathcal{C}} \mathcal{E}_T(\theta,\mathcal{S},\varphi,c), \tag{3.28}$$

for some $\varepsilon > 0$. We call the resulting neural network $u^* := u_{\theta^*_{\mathcal{S}}}$ a weak PINN (*wPINN*).

Note that the training error (3.27) is a *quadrature approximation* of the corresponding (total) residual,

$$\begin{aligned}\mathcal{E}_G(\theta,\varphi,c) = -\int_0^1\int_0^T \left(\Big|u_{\theta^*(\mathcal{S})}(x,t) - c\Big|\partial_t\varphi(x,t) + Q[u_{\theta^*(\mathcal{S})}(x,t);c]\partial_x\varphi(x,t)\right)dxdt \\ + \int_0^1 \big|u_\theta(x,0) - u(x,0)\big|dx + \int_0^T \big|u_\theta(0,t) - u_\theta(1,t)\big|dt\end{aligned} \tag{3.29}$$

The next step in the analysis of *wPINNs* is to provide a bound on the so-called *generalization gap* i.e., the difference between $\mathcal{E}_G$ and $\mathcal{E}_T$. To this end, we have the following theorem,

Theorem 3.9. *Let $M, L, W \in \mathbb{N}$, $R \geq \max\{1, T, |\mathcal{C}|\}$ with $L \geq 2$ and $M \geq 3$. Moreover let $u_\theta : D \times [0,T] \to \mathbb{R}$, $\theta \in \Theta$, be tanh neural networks with at most $L-1$ hidden layers, width at most W and weights and biases bounded by R. Assume that $\mathcal{E}_G$ and $\mathcal{E}_T$ are bounded by $B \geq 1$. It holds with a probability of at least $1-\delta$ that,*

$$\mathcal{E}_G(\theta^*_{\mathcal{S}},\varphi^*_{\mathcal{S}},c^*_{\mathcal{S}}) \leq \mathcal{E}_T(\theta^*_{\mathcal{S}},\mathcal{S},\varphi^*_{\mathcal{S}},c^*_{\mathcal{S}}) + \frac{3BdLW}{\sqrt{M}}\sqrt{\ln\left(\frac{C\ln(1/\varepsilon)WRM}{\varepsilon^3\delta B}\right)} \tag{3.30}$$

where $C > 0$ is a constant that only depends on u and f.

Proof. Note that $\mathcal{E}_G$ (3.29) is defined as the sum of the error related to the entropy residual $\mathcal{R}$, the initial condition and the boundary condition. We will bound (with a high probability) each of these terms separately in terms of the corresponding term in the definition of the training error (3.27) using Lemma A.8.

We start by calculating the Lipschitz constant of $\mathcal{R}$. From the proof of Theorem 3.2 and from [7, Lemma 11] we find that,

$$\begin{aligned}\big|\mathcal{R}(u_\theta,\varphi,c) - \mathcal{R}(u_\vartheta,\varphi,c)\big| &\leq T|\varphi|_{W^{1,\infty}}(1+3L_f)\max_{x,t}\big|(u_\theta - u_\vartheta)(x,t)\big| \\ &\leq T|\varphi|_{W^{1,\infty}}(1+3L_f)(d+5)(WR)^{L-1}\|\theta-\vartheta\|_\infty.\end{aligned} \tag{3.31}$$

Following the steps of the proof of Lemma 3.1 we also find that,

$$\big|\mathcal{R}(v,\varphi,c) - \mathcal{R}(v,\varphi,b)\big| \leq (1+3L_f)|\varphi|_{W^{1,1}(D\times[0,T])}|b-c|. \tag{3.32}$$

Using that $\partial_t\overline{\varphi}_\varepsilon = \alpha\overline{\varphi}_\varepsilon + \beta\overline{\varphi}_\varepsilon$ and $\partial_t\overline{\varphi}_\varepsilon = \beta\overline{\varphi}_\varepsilon$ we find,

$$\begin{aligned}&\big|\mathcal{R}(v,\overline{\varphi}^{y,s}_\varepsilon,c) - \mathcal{R}(v,\overline{\varphi}^{z,\tau}_\varepsilon,c)\big| \\ &\leq \int_0^1\int_0^T \Big|\big|v(x,t) - c\big|\partial_t(\overline{\varphi}^{y,s}_\varepsilon - \overline{\varphi}^{z,\tau}_\varepsilon)(x,t) + Q[v(x,t);c]\partial_x(\overline{\varphi}^{y,s}_\varepsilon - \overline{\varphi}^{z,\tau}_\varepsilon)(x,t)\Big|dtdx \\ &\leq C(\alpha+\beta)\int_0^1\int_0^T \big|(\overline{\varphi}^{y,s}_\varepsilon - \overline{\varphi}^{z,\tau}_\varepsilon)(x,t)\big|dtdx,\end{aligned} \tag{3.33}$$

where $C > 0$ is a constant depending on u and f. Combining the previous calculations with $\alpha < \beta$ and $|\varphi|_{W^{1,\infty}} = \mathcal{O}(\beta^3)$ (Lemma A.1) we find that,

$$\big|\mathcal{R}(v,\overline{\varphi}^{y,s}_\varepsilon,c) - \mathcal{R}(v,\overline{\varphi}^{z,\tau}_\varepsilon,c)\big| \leq C\beta^4(|y-z| + |s-\tau|). \tag{3.34}$$

Putting everything together, we find that,

$$\big|\mathcal{R}(u_\theta,\overline{\varphi}^{y,s}_\varepsilon,c) - \mathcal{R}(u_\vartheta,\overline{\varphi}^{z,\tau}_\varepsilon,b)\big| \leq C\beta^4(d+5)(WR)^{L-1}\big\|(\theta,y,s,c) - (\vartheta,z,\tau,b)\big\|_\infty. \tag{3.35}$$

Similarly, we also find that

$$\left|\int_0^1 \big|u_\theta(x,0) - u(x,0)\big|dx - \int_0^1 \big|u_\vartheta(x,0) - u(x,0)\big|dx\right| \leq \max_x\big|(u_\theta - u_\vartheta)(x,0)\big| \leq (d+5)(WR)^{L-1}\|\theta-\vartheta\|_\infty \tag{3.36}$$

and also,

$$\left|\int_0^T \big|u_\theta(0,t) - u_\theta(1,t)\big|dt - \int_0^T \big|u_\vartheta(0,t) - u_\vartheta(1,t)\big|dt\right| \leq T(d+5)(WR)^{L-1}\|\theta-\vartheta\|_\infty. \tag{3.37}$$

In total, we conclude that

$$\begin{aligned}\big|\mathcal{E}_G(\theta,y,s,c) - \mathcal{E}_G(\vartheta,z,\tau,b)\big| &\leq C\beta^4(d+5)(WR)^{L-1}\big\|(\theta,y,s,c) - (\vartheta,z,\tau,b)\big\|_\infty \\ &=: \mathfrak{L}\big\|(\theta,y,s,c) - (\vartheta,z,\tau,b)\big\|_\infty.\end{aligned} \tag{3.38}$$

The same inequality holds for the training error (3.27).

Next, one can calculate that every u_θ has at most $(d+1+(L-2)W+1)W$ weights and $(L-1)W+1$ biases. Together with y, s, b we find that there are at most $k \leftarrow 4(d+1)LW^2 \leq 9dLW^2$ parameters. We can now obtain the inequality from the statement by using Lemma A.8 with $\Theta \leftarrow \Theta \times [0,1] \times [0,T] \times \mathcal{C}$, the calculated values of $\mathfrak{L}, k$ and $a \leftarrow R$ and the estimate,

$$\sqrt{\frac{B^2 k}{M} \ln\left(\frac{a\mathfrak{L}\sqrt{M}}{\sqrt[k]{\delta} B}\right)} \leq \frac{3BdLW}{\sqrt{M}} \sqrt{\ln\left(\frac{C\ln\big(1/\varepsilon\big)WRM}{\varepsilon^3 \delta B}\right)}, \tag{3.39}$$

where we used that $\beta = \mathcal{O}(\ln\big(1/\varepsilon\big)\varepsilon^{-3})$. Finally, our chosen value of k leads to the implication that $M \geq 3 \implies M \geq e^{16/k}$, as is required by Lemma A.8. □

Using Theorem 3.7 and the above bound on the generalization gap, one can prove the following rigorous upper bound on the total L^1-error of the weak PINN, which we denote as,

$$\mathcal{E}^T(\theta) = \int_D \big|u_\theta(x,T) - u(x,T)\big| dx. \tag{3.40}$$

Corollary 3.10. *Assume the setting of Theorem 3.9. It holds with a probability of at least $1-\delta$ that*

(3.41)

$$\mathcal{E}^T(\theta^*_{\mathcal{S}}) \leq C\Bigg[\underbrace{\mathcal{E}_T(\theta^*_{\mathcal{S}}, \mathcal{S}, \varphi^*_{\mathcal{S}}, c^*_{\mathcal{S}}), c^*_{\mathcal{S}})}_{\text{training error}} + \underbrace{\frac{3BdLW}{\sqrt{M}} \sqrt{\ln\left(\frac{C\ln\big(1/\varepsilon\big)WRM}{\varepsilon^3\delta B}\right)}}_{\text{generalization gap}} + \underbrace{\big(1+\|u^*\|_{C^1}\big)\ln\big(1/\varepsilon\big)^3 \varepsilon}_{\text{limited test space } \Phi_\varepsilon \subsetneq H^1}\Bigg].$$

Proof. Using Theorem 3.7 we find that,

$$\begin{aligned}
\mathcal{E}^T(\theta^*_{\mathcal{S}}) &\leq C\Big[\big\|u^*(\cdot,0) - u(\cdot,0)\big\|_{L^1} + \max_{c\in\mathcal{C}, \varphi\in\Phi_\varepsilon} \mathcal{R}(u^*, \varphi, c) \\
&\qquad + \big(1+\|u^*\|_{C^1}\big)\ln\big(1/\varepsilon\big)^3\varepsilon + \big\|u^*(1,\cdot) - u(0,\cdot)\big\|_{L^1}\Big] \\
&\leq C\Big[\mathcal{R}(u^*, \varphi^*_{\mathcal{S}}, c^*_{\mathcal{S}}) + \big\|u^*(\cdot,0) - u(\cdot,0)\big\|_{L^1} + \big(1+\|u^*\|_{C^1}\big)\ln\big(1/\varepsilon\big)^3\varepsilon \\
&\qquad + \Big(\max_{c\in\mathcal{C}, \varphi\in\Phi_\varepsilon} \mathcal{R}(u^*, \varphi, c) - \mathcal{R}(u^*, \varphi^*_{\mathcal{S}}, c^*_{\mathcal{S}})\Big) + \big\|u^*(1,\cdot) - u(0,\cdot)\big\|_{L^1}\Big] \\
&= C\Big[\mathcal{E}_G(\theta^*_{\mathcal{S}}, \varphi^*_{\mathcal{S}}, c^*_{\mathcal{S}}) + \big(1+\|u^*\|_{C^1}\big)\ln\big(1/\varepsilon\big)^3\varepsilon + \underbrace{\max_{c\in\mathcal{C}, \varphi\in\Phi_\varepsilon} \mathcal{R}(u^*, \varphi, c) - \mathcal{R}(u^*, \varphi^*_{\mathcal{S}}, c^*_{\mathcal{S}})}_{\leq 0}\Big]
\end{aligned} \tag{3.42}$$

Combining the previous inequality with Theorem 3.9 we then find the statement of the corollary. □

Thus, the estimate (3.41) provides a rigorous bound on the total error of a *wPINN* in approximating the entropy solution of the scalar conservation law (2.1), in terms of the training error, size of the underlying neural networks and the number of collocation points. In particular, this error can be made as small as desired by choosing sufficient number of collocation points and networks of suitable size.

Remark 3.11. *Instead of using random training points, one can choose a training set based on low-discrepancy sequences. This will improve the convergence rate in terms of M to $\mathcal{O}(V_{HK}\ln(M)^d M^{-1})$ by using arguments from* [32]*, where V_{HK} is the Hardy-Krause variation of the integrand in the definition of $\mathcal{E}_G(\theta, c)$. However, if one also considers the ε-dependence of V_{HK} through $\overline{\varphi}_\varepsilon$ then it is better to use random points as the bound in Corollary 3.10 only depends sublogarithmically on ε^{-1} whereas upper bounds for V_{HK} depend polynomially on ε^{-1}.*

Remark 3.12. *Throughout the entire section, we have focused on calculating the L^1-error of the weak PINN, as the L^1-norm is a natural choice for scalar conservation laws. In practice, however, we will see that it is easier to train the weak PINN using an L^2-based loss function. The developed theory can be easily extended to L^p-based loss functions for $p > 1$. First, one can use Hölder's inequality to prove an L^2-based upper bound for the RHS of the stability result* (3.12) *of Theorem 3.7. Second, one needs to make an adaptation to Theorem 3.9. This will lead to a theoretical deterioration of the generalization gap from $\mathcal{O}\left(M^{-1/2}\sqrt{\ln(M)}\right)$ to $\mathcal{O}\left(M^{-1/4}\sqrt{\ln(M)}\right)$, as is common for these type of estimates.*

4. Training and Implementation of *wPINNs*

In this section, we describe how *wPINNs*, designed in Section 2.4, are implemented and trained in practice. To this end, we need the following elements,

4.1. **Set of Collocation Points.** The set of collocation points $\mathcal{S}$, also called as *training set* in the literature on PINNs, was already introduced in Section 3.3. We will divide $\mathcal{S} \subset D \times [0,T]$, into the following three parts,

- Interior collocation points $\mathcal{S}_{int} = \{y_m\}$ for $1 \leq m \leq M_{int}$, with each $y_m = (x,t)_m \in D \times [0,T]$.
- Spatial boundary collocation points $\mathcal{S}_{sb} = \{z_m\}$ for $1 \leq m \leq M_{sb}$ with each $z_m = (x,t)_m$ and $z_m \in \partial D \times [0,T]$.
- Temporal boundary collocation points $\mathcal{S}_{tb} = \{x_m\}$, with $1 \leq m \leq M_{tb}$ and $x_m \in D$.

The full set of collocation points is $\mathcal{S} = \mathcal{S}_{int} \cup \mathcal{S}_{sb} \cup \mathcal{S}_{tb}$ and $M = M_{int} + M_{sb} + M_{tb}$.

4.2. **Residuals.** We recapitulate the definitions of different residuals, already introduced in (2.11) and further refined in Theorem 3.7 (see also Remark 3.8) below,

- *Interior residual* given by,

$$r_{int}[u_\theta, \varphi](x,t,c) := \varphi(x,t)\partial_t \big|u_\theta(x,t) - c\big| - Q[u_\theta(x,t); c]\partial_x \varphi(x,t). \tag{4.1}$$

  Observe that the residual depends on the test function $\varphi(x,t)$. We restrict the choice of the test function to the parametrized family of functions $\varphi_\eta(x,t)$ defined as $\varphi_\eta(x,t) = \omega(x,t)\xi_\eta(x,t)$. Here, $\omega : D \mapsto \mathbb{R}$ is a *cutoff* function satisfying the following properties:
  (1) $\omega(x) = 1, \quad x \in D_\varepsilon$,
  (2) $\omega(x) = 0, \quad x \in \partial D$,

$$D_\varepsilon = \{x \in D : dist(x, \partial D) < \varepsilon\}, \tag{4.2}$$

  and $\xi_\eta(x,t)$ a neural network with trainable parameters η. The choice of the cutoff function guarantees that the test function has compact support. Then, the interior residual becomes,

$$r_{int}[u_\theta, \varphi_\eta] := \varphi_\eta(x,t)\partial_t \big|u_\theta(x,t) - c\big| - Q[u_\theta(x,t); c]\partial_x \varphi_\eta(x,t). \tag{4.3}$$

- *Spatial boundary residual* given by,

$$r_{sb}[u_\theta](x,t) := u_\theta(x,t) - g(x,t). \quad \forall x \in \partial D,\ t \in (0,T], \tag{4.4}$$

  Although the estimates above were derived assuming periodic boundary conditions, the numerical experiments are carried out with Dirichlet boundary conditions corresponding to the boundary trace of exact solutions i.e., with $g(x,t) = u|_{\partial D \times (0,T)}$.
- *Temporal boundary residual* given by,

$$r_{tb}[u_\theta](x) := u_\theta(x,0) - u(x,0), \quad \forall x \in D. \tag{4.5}$$

4.3. **Loss function.** Given the definitions above, we consider the following loss function,

$$J(\theta, \eta) = J_{pde}(\theta, \eta, c) + \lambda J_u(\theta) \tag{4.6}$$

with

$$J_{pde}(\theta,\eta,c) = \frac{\Big(ReLU\Big(\sum_{m=1}^{M_{int}} r_{int}[u_\theta, \varphi_\eta](y_m, c)\Big)\Big)^2}{\sum_{m=1}^{M_{int}} \partial_x \varphi_\eta(y_m)^2}, \quad J_u(\theta) = \sum_{m=1}^{M_{tb}} r_{tb}[u_\theta](x_m)^2 + \sum_{m=1}^{M_{sb}} r_{sb}[u_\theta](z_m)^2. \tag{4.7}$$

Here, λ is a hyperparameter for balancing the role of PDE and data residuals, and the denominator of J_{pde} a Monte Carlo approximation of the H^1-seminorm of $\varphi_\eta(x,t)$. The following remarks about the specific form of the loss function (4.6) are in order,

Remark 4.1. *The terms appearing in the loss function terms are Monte Carlo approximations of the integrals in the error estimate, cf. Theorem 3.7, where the L^1 norm has been replaced by the L^2 (Remark 3.12). This was done to facilitate optimization of the resulting min-max problem.*

Remark 4.2. *We observe that additional terms, i.e., use of the ReLU function and test function H^1-seminorm, have been introduced in the definition of the loss function* (4.6)*, when compared to the terms in the error estimate in Theorem 3.7. These are introduced to facilitate training and the estimates can be readily extended to incorporate the contributions of these terms.*

Remark 4.3. *In practice, the maximization problem with respect to the scalar c is solved by computing* $J_{max,C}(\theta,\eta) = \max_{c_i \in C} J_{max}(\theta,\eta,c_i)$*, for a discrete set of values* $C = \{c_i\}_{i=1}^{M}, c_i \in [c_{min}, c_{max}]$*, whereas the optimization problems with respect to the neural network parameters* θ *and* η *is approximated with gradient descent and ascent, respectively.*

4.4. **Weak PINNs Algorithm.** Given the above description of its constituents, we summarize the *wPINNs* algorithm in Algorithm 1.

Algorithm 1: Training of *wPINNs*

Result: $\theta^*_{\mathcal{S}}, \eta^*_{\mathcal{S}}, c^*_{\mathcal{S}}$
Initialize the networks $u_\theta, \varphi_\eta : D \times [0,T] \mapsto \mathbb{R}$ and C;
for $e = 1, ..., N$ **do**
  for $k = 1, ..., N_{max}$ **do**
    Compute $J_{max,C}(\theta,\eta) = \max_{c_i \in C} J_{max}(\theta,\eta,c_i)$;
    Update $\eta \leftarrow \eta + \tau_\eta \nabla J_{max,C}(\theta,\eta)$;
  end
  for $k = 1, ..., N_{min}$ **do**
    Compute $J_{max,C}(\theta,\eta) = \max_{c_i \in C} J_{max}(\theta,\eta,c_i)$;
    Update $\theta \leftarrow \theta - \tau_\theta \nabla(\lambda J_{max,C} + J_u)(\theta,\eta)$;
  end
end

4.5. **Implementation of *wPINNs*.** The implementation of the *wPINNs* algorithm is carried out with a collection of Python scripts, realized with the support of PyTorch `https://pytorch.org/`. The scripts can be downloaded from `https://github.com/mroberto166/wpinns`. Below, we describe key implementation details.

4.5.1. *Ensemble Training. wPINNs* include several hyperparameters, including number of hidden layers and neurons of the networks, L_θ, L_η, l_θ, l_η, the number of iterations K_{max}, K_{min}, number of epochs e, residual parameter λ, etc. A user is always confronted with the question of which parameter to choose. It is standard practice in machine learning to perform a systematic hyperparameter search. To this end, we follow the *ensemble training* procedure of [26]: for each configuration of the model hyperparameters we *retrain* the *wPINN* n_θ times, each with different initialisation of the networks hyperparameters, and select the hyperparameter configuration that minimises the average value over the retrainings of the training error 3.27, computed in the L^2 norm:

$$\begin{aligned}\mathcal{E}_T(\theta^*_{\mathcal{S}}, \eta^*_{\mathcal{S}}, c^*_{\mathcal{S}}) = &\sum_{m=1}^{M_{int}} \Big(\varphi^*(y_m)\partial_t\big|u^*(y_m) - c^*_{\mathcal{S}}\big| - Q[u_\theta(y_m); c^*_{\mathcal{S}}]\partial_x\varphi^*(y_m)\Big)^2 \\ &+ \sum_{m=1}^{M_{sb}} \big|u_\theta(x_m,0) - u(x_m,0)\big|^2 + \sum_{m=1}^{M_{tb}} \big|u^*(z_m) - u(z_m)\big|^2.\end{aligned} \tag{4.8}$$

4.5.2. *Random Reinitialization of the Test function Parameters.* (Approximate) solutions of min-max problems are significantly harder to reach, when compared to standard minimization (or maximization) problems, as they correspond to saddle points of the underlying loss function. One essential ingredient for improving the numerical stability of the algorithm is the random reinitialization of the trainable parameters η, corresponding to the test function neural network in (4.6). This can be performed with frequency r_f. This *reset frequency* can be suitably chosen as any other model hyperparameters through ensemble training. On account of this random reinitialization of the test function parameters η, the algorithm 1 can be readily modified to yield algorithm 2, that is used in practice.

4.5.3. *Averages of retrainings.* The final *wPINN* approximation to the solution of the scalar conservation law (2.1) at any given input (x,t), denoted as $u_{av}(x,t)$, is defined as the average over retrainings,

$$u_{av}(x,t) = \frac{1}{n_\theta}\sum_{i}^{n_\theta} u_i^*(x,t), \tag{4.9}$$

where $u_i^*(x,t)$ denotes the predictions of the underlying *wPINN* at (x,t), trained via algorithm 2, with initial parameters θ_i, η_i. This averaging is performed to yield more robust predictions as well as to

provide an estimate of the underlying uncertainty in predictions, due to the random initializations of the neural network parameters during training.

Algorithm 2: *Weak PINN* training with random reset of the test function parameters

Result: $\theta^*_{\mathcal{S}}, \eta^*_{\mathcal{S}}, c^*_{\mathcal{S}}$
Initialize the networks $u_\theta, \varphi_\eta : D \times [0,T] \mapsto \mathbb{R}$ and C;
for $e = 1, ..., N$ **do**
  if $e \ \% \ (r_f N) = 0$ **then**
    Randomly initialize η;
  end
  for $k = 1, ..., N_{max}$ **do**
    Compute $J_{max,C}(\theta, \eta) = \max_{c_i \in C} J_{max}(\theta, \eta, c_i)$;
    Update $\eta \leftarrow \eta + \tau_\eta \nabla J_{max,C}(\theta, \eta)$;
  end
  for $k = 1, ..., N_{min}$ **do**
    Compute $J_{max,C}(\theta, \eta) = \max_{c_i \in C} J_{max}(\theta, \eta, c_i)$;
    Update $\theta \leftarrow \theta - \tau_\theta \nabla (\lambda J_{max,C} + J_u)(\theta, \eta)$;
  end
end

5. Numerical Results

In this section, we present numerical experiments to illustrate the performance of *wPINNs*. To this end, we consider the scalar conservation law (2.1) in the domain $D = [-1, 1]$, with the flux function $f(u) = \frac{1}{2}u^2$. Note that this amounts to considering the well-known inviscid Burgers' equation. We evaluate the performance of *wPINNs*, implemented through algorithm 2, by computing the (relative) total error at a final time T,

$$\mathcal{E}_r^T(\theta^*_{\mathcal{S}}) = \frac{\int_D \left|u^*(x,T) - u(x,T)\right| dx}{\int_D \left|u(x,T)\right| dx}, \tag{5.1}$$

where u^* is the prediction of the *wPINN* algorithm 2. We also compute the space-time relative error,

$$\mathcal{E}_r(\theta^*_{\mathcal{S}}) = \frac{\int_{D\times[0,T]} \left|u^*(x,t) - u(x,t)\right| dxdt}{\int_{D\times[0,T]} \left|u(x,t)\right| dxdt}, \tag{5.2}$$

to assess the performance of *wPINNs* over the entire evolution of the entropy solution. We remark that the integrals in the above error expressions can be readily approximated with Monte Carlo quadratures. We consider the following numerical experiments,

5.1. Standing and Moving Shock.

As a first numerical example, we consider the Burgers' equation in $[-1, 1] \times [0, 0.5]$ with initial conditions:

$$u_0(x) = \begin{cases} 1 & x \leq 0 \\ -1 & x > 0 \end{cases}, \quad u_0(x) = \begin{cases} 1 & x \leq 0 \\ 0 & x > 0 \end{cases} \tag{5.3}$$

which result into a standing shock located at $x = 0$ and a shock moving with speed 0.5, respectively:

$$u(x,t) = \begin{cases} 1 & x \leq 0 \\ -1 & x > 0 \end{cases}, \quad u(x,t) = \begin{cases} 1 & x \leq \frac{t}{2} \\ 0 & x > \frac{t}{2} \end{cases} \tag{5.4}$$

We perform an ensemble training, as outlined in the previous section, to find the best set of hyperparameters among those mentioned in Tables 1 and 2. On the other hand, we fix $l_\theta = 20$, $l_\eta = 10$, $N_{min} = 1$, $\lambda = 10$ $e = 5000$, $\tau_\theta = 0.01$ and $\tau_\eta = 0.015$, $n_\theta = 10$ and the sin activation function as the activation function σ_θ for the neural network approximating the solution of (2.1).

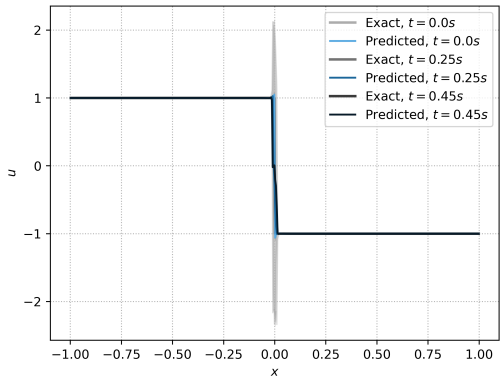
With this setting, the *wPINNs* algorithm 2 is run and its average prediction (as described in Section 4.5.3) is plotted in Figures 1a and 1b, respectively, where we also compare the predictions with the exact solutions (5.4) at different times. From these figures, we observe that the *wPINNs* average prediction accurately approximates both the standing as well as the moving shock. There is some variance in the predictions of multiple retrainings. This is completely expected as a highly non-convex min-max

L_θ	L_η	σ_η	N_{max}	r_f
4,6	2,4	*sin*,*tanh*	6, 8	0.001, 0.005, 0.025, 0.05

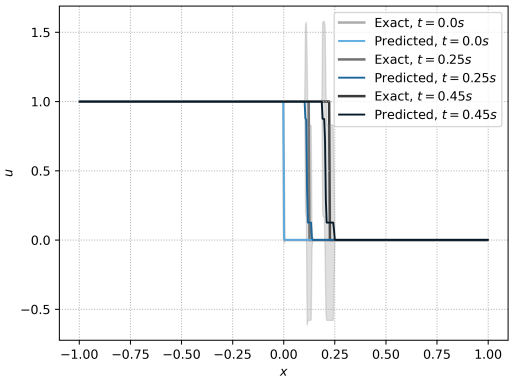
TABLE 1. Hyperparameter configurations and number of retrainings employed in the ensemble training of *wPINN* for moving shock.

L_θ	L_η	σ_η	N_{max}	r_f
4,6	2,4	*sin*,*tanh*	6, 8	0.025, 0.05, 0.25

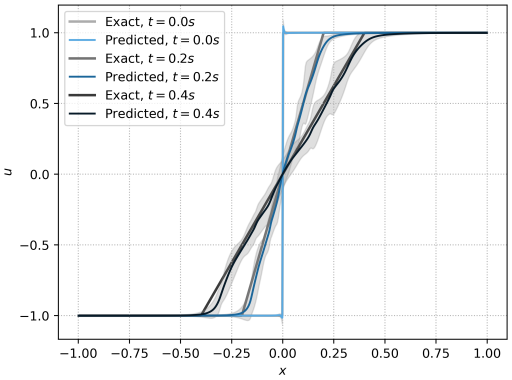
TABLE 2. Hyperparameter configurations and number of retrainings employed in the ensemble training of *wPINN* for standing shock, rarefaction wave and initial sine wave.



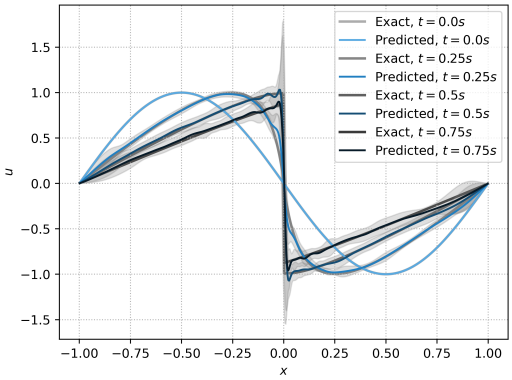
(A) Standing Shock Solution , $\mathcal{E}_r^T(\theta_S^*) = 0.01$



(B) Moving Shock Solution, $\mathcal{E}_r^T(\theta_S^*) = 0.019$



(C) Rarefaction Wave, $\mathcal{E}_r^T(\theta_S^*) = 0.022$



(D) Initial Sine Wave, $\mathcal{E}_r^T(\theta_S^*) = 0.057$

FIGURE 1. Exact solutions and average predictions obtained with *wPINN* for the Burgers' equation. Retraining average and standard deviation are plotted.

optimization problem is being approximated and it is possible to be trapped at local saddle points. Nevertheless, the quantitative predictions of the relative total errors $\mathcal{E}_r(\theta_S^*)$ and $\mathcal{E}_r^T(\theta_S^*)$, presented in Table 3, are very accurate, with even errors for the whole time-history of evolution, being below 2%. Finally, in Figure 2, we also plot the *wPINN* prediction that corresponds to the hyperparameter configuration that leads to smallest overall error among all tested hyperparameter configurations. We term it as the *best hyperparameter configuration*. This *best* prediction is extremely accurate, with the largest error below 0.2%. However, in practice, one does not have access to exact (or reference) solutions and needs to choose hyperparameter configurations that correspond to the smallest values of the loss function.

5.2. **Rarefaction Wave.** We further test the performance of *wPINNs* by considering the Burgers' equation with initial data,

$$u_0(x) = \begin{cases} -1 & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{5.5}$$

The exact solution, given by,

$$u(x,t) = \begin{cases} -1 & x \leq -t \\ \frac{x}{t} & -t < x \leq t \\ 1 & x > t \end{cases} \tag{5.6}$$

corresponds to a *rarefaction wave*. We observe that this initial datum is often used to illustrate the

	M_{int}	M_{tb}	M_{sb}	$\mathcal{E}_r$	$\mathcal{E}_r^T$
Standing Shock	16384	4096	4096	0.005	0.01
Moving Shock	16384	4096	4096	0.011	0.019
Rarefaction Wave	16384	4096	4096	0.013	0.022
Initial Sine Wave	16384	4096	4096	0.03	0.057

TABLE 3. Number of training samples, total error (at final time) and total error over time, obtained with *wPINNs* (average) predictions in the numerical experiments for the Burgers' equation

multiplicity of weak solutions of hyperbolic conservation laws as the *standing shock*, corresponding to the initial datum is clearly a weak solution but does not satisfy the entropy conditions. To illustrate the rationale behind considering *entropy residuals* (2.10) in our definitions of the loss function (4.6) in the *wPINNs* algorithm 2, we first run the same algorithm but replace the entropy residual (2.10) in Algorithm 2 with the following residuals,

$$r_{int,\theta,\eta} = \int_D \int_{[0,T]} \Big(u_{\theta,t}(x,t)\varphi_\eta(x,t) - f(u_\theta(x,t))\varphi_{\eta,x}(x,t) \Big) dtdx \tag{5.7}$$

and

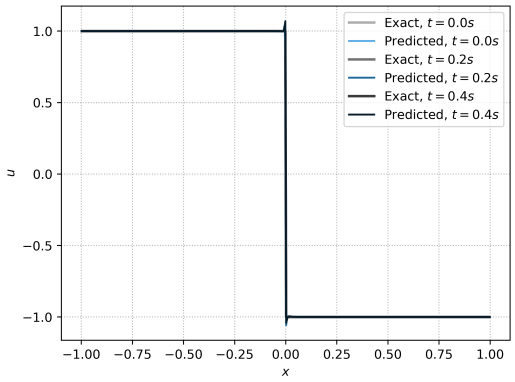
$$J_{pde}(\theta,\eta) = \frac{\Big(\sum_{m=1}^{M_{int}} r_{int,\theta,\eta}(y_m,c) \Big)^2}{\sum_{m=1}^{M_{int}} \partial_x \varphi_\eta(y_m)^2}, \tag{5.8}$$

that correspond to the standard weak formulation (see definition 2.1) of the scalar conservation law. The resulting predictions are plotted in figure 3 and show that the resulting *wPINN* only approximated the *non-entropic* standing shock solution corresponding to the initial datum. Thus, a naive weak formulation of PINNs does not suffice in accurate approximations of scalar conservation laws. On the other hand, the *wPINN* average predictions of algorithm 2, with the entropy residual (2.10), provide accurate approximation of the rarefaction wave entropy solution, as shown in Figure 1c and Table 3. In particular, the error at the final time T is approximately 2% on average, whereas the error corresponding to the best hyperparameter configuration (see Figure 2) is only slightly lower (1.9%).

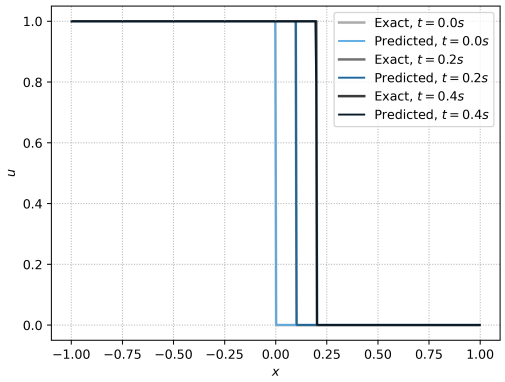
5.3. **Sine Wave Initial Datum.** As a final numerical experiment, we consider the Burgers' equation with the initial data,

$$u_0(x) = -\sin(\pi x)$$

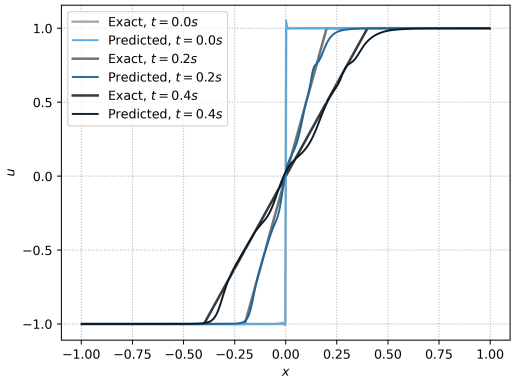
and zero Dirichlet boundary conditions in the spatio-temporal domain $[-1,1] \times [0,1]$. The exact solution (approximated in Figure 1d with a high-resolution finite volume scheme), shows a complex evolution with both steepening as well as expansions of the sine wave that eventually form into a shock wave that separates two rarefactions. We run the *wPINNs* algorithm 2, with low-discrepancy Sobol points [32], instead of random collocation points. An ensemble training procedure, based on the hyperparameters presented in Table 2. The remaining parameters are set as follows: $L_\theta = 4$, $L_\eta = 2$, $l_\theta = 20$, $l_\eta = 10$, $\tau_\eta = 0.015$, $\tau_\theta = 0.01$ and sin activation function for both the networks. The networks are trained for $e = 75000$ epochs and parameters reinitialized $n_\theta = 15$ times, on account of the more complex underlying solution.



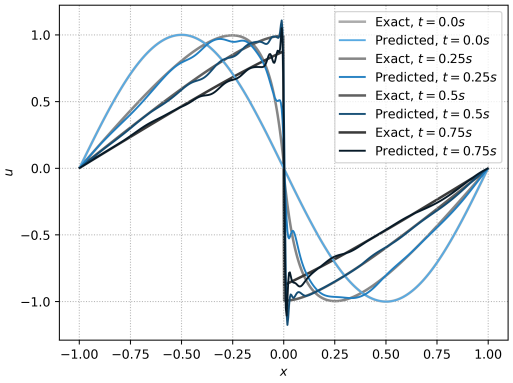
(A) Standing Shock Solution , $\mathcal{E}_r^T(\theta_{\mathcal{S}}^*) = 0.0004$



(B) Moving Shock Solution, $\mathcal{E}_r^T(\theta_{\mathcal{S}}^*) = 0.002$



(C) Rarefaction Wave, $\mathcal{E}_r^T(\theta_{\mathcal{S}}^*) = 0.019$



(D) Initial Sine Wave, $\mathcal{E}_r^T(\theta_{\mathcal{S}}^*) = 0.047$

FIGURE 2. Exact solutions and best predictions obtained with *wPINN* for the Burgers' equation.

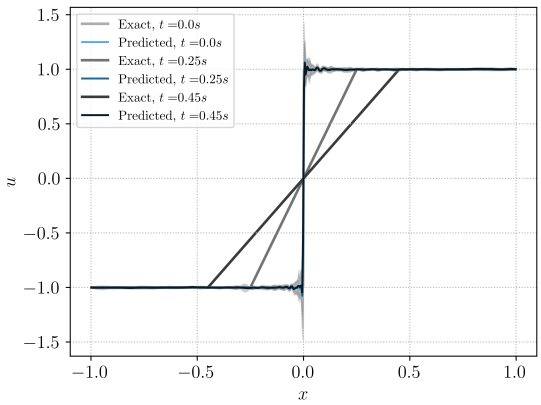


FIGURE 3. Exact rarefaction wave solution and prediction obtained with weak PINNs, without entropy conditions incorporated into the weak residual.

The (average) predictions with the *wPINNs* algorithm 2 are depicted in Figure 1d and show that this complicated underlying solution is approximated accurately with *wPINNs*, although there are very small spurious oscillations in the approximation. These might be further eliminated by adding additional regularization terms, such as on the BV-norm into the loss function (4.6). Nevertheless, as shown in Table 3, the error over the entire time period is approximately 3%, whereas the error at final time is understandably higher. This should also be contrasted with the relative error of approximately 24%, obtained

for this particular test case with conventional PINNs, as observed in [31] (Figure 6 (d)). Moreover, the error $\mathcal{E}_r^T$ is even smaller and the approximation significantly more accurate, with the best performing hyperparameter configuration shown in Figure 2.

6. Discussion

Physics informed neural networks (PINNs) have been extremely successful in approximating both forward and inverse problems, in an unsupervised manner, for a very diverse set of PDEs. Recent theoretical work on PINNs, for instance in [31, 6], suggest that *regularity* of solutions of the underlying PDE is a key requirement in the derivation of estimates on PINN error. In particular, this analysis suggests that the conventional form of PINNs can fail at accurately approximating nonlinear PDEs such as hyperbolic conservation laws, whose solutions are not sufficiently regular. This was further verified in numerical experiments, such as the ones presented in [31].

Our main aim in this paper was to design a novel variant of PINNs for the accurate approximation of entropy solutions of scalar conservation laws. To this end,

- We base the PDE residual on the weak form of the Kruzkhov entropy conditions (2.10) and solve the resulting min-max optimization problem for determining parameters (weights and biases), for the neural networks approximating the entropy solution as well as test functions. The resulting PINN is termed as a weak PINN (*wPINN*).
- We prove rigorous bounds on different sources of error associated with *wPINNs* in Section 3 to show that the resulting errors can be made arbitrarily small by ensuring a small enough training error, choosing neural networks of suitable size and enoough (random) collocation points.
- We present numerical experiments with the Burgers' equation to illustrate that *wPINNs*, with suitable choices of loss functions and training protocol (see section 4), can approximate the entropy solutions of scalar conservation laws accurately.

Thus, we propose a novel unsupervised learning algorithm for approximating scalar conservation laws and show, both theoretically as well as empirically, that it provides an accurate approximation to entropy solutions. Given that, the algorithm was fully unsupervised i.e., no labelled data (solution values in the interior of the space-time domain) are used, *wPINNs* can serve as an alternative to existing high-resolution finite volume, finite difference and discontinuous Galerkin finite element methods for approximating conservation laws.

Below, we discuss possible shortcomings and extensions of *wPINNs*,

- In comparison to conventional PINNs, *wPINNs* entail the (approximate) solution of a min-max optimization problem. Thus, training *wPINNs* is clearly more computationally expensive than training PINNs as only a minimization problem is solved in the latter. However, given that conventional PINNs can fail to accurately approximate discontinuous solutions of conservation laws, it is imperative to use *wPINNs* in this context. We aim to explore faster training algorithms for solving the min-max optimization problem and plan to take inspiration from the extensive literature on training GANs in this regard.
- A key advantage of machine learning approaches such as conventional PINNs and *wPINNs* is their ability to serve as a fast surrogate, particularly for parametric PDEs (see [2] for an example of PINNs solving a parametrized KdV equation). Once the *wPINN* has been trained for a (random) sampling of the parameter space, it can infer solutions with respect to other parameters at practically zero computational cost. Moreover, one can also use *wPINNs* within a physics informed operator learning framework [43, 8] to approximate the semi-group of the underlying solution operator. The use of *wPINNs* in the context of parametric PDEs and operator learning will be considered in future work.
- Lastly, we presented the algorithm and provided error estimates for scalar conservation laws in one space dimension. The extension of the algorithm and estimates to multi-dimensional scalar conservation laws is straightforward as Kruzkhov entropies are well-defined in this case. The extension of *wPINNs* to systems of conservation laws is non-trivial. In particular, there is no analogue of (infinitely many) Kruzkhov entropies for systems. Rather, one has to work with, usually, a single entropy family (for instance the thermodynamic entropy for the compressible Euler equations or total energy for shallow-water equations). Adding such entropies to the residual is straightforward. However, it is unclear if a single entropy will drive the algorithm towards a physically relevant solution. This extension will also be considered in the future.

Acknowledgments

The research of RM and SM was partly performed under a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 770880). The authors thank Prof. Ulrik S. Fjordholm (University of Oslo, Norway) and Prof. Ujjwal Koley (TIFR-CAM, Bangalore, India) for insightful discussions.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875v3*, 2017.
[2] G. Bai, U. Koley, S. Mishra, and R. Molinaro. Physics informed neural networks (PINNs) for approximating nonlinear dispersive PDEs. *arXiv preprint arXiv:2104.05584*, 2021.
[3] S. Bartels. Total variation minimization with finite elements: convergence and iterative solution. *SIAM Journal on Numerical Analysis*, 50(3):1162–1180, 2012.
[4] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. Scientific machine learning through physics-informed neural networks: Where we are and what's next. *arXiv preprint arXiv:2201.05624*, 2022.
[5] T. De Ryck, A. D. Jagtap, and S. Mishra. Error analysis for PINNs approximating the Navier-Stokes equations. *arXiv preprint arXiv:2203.09346*, 2022.
[6] T. De Ryck, S. Lanthaler, and S. Mishra. On the approximation of functions by tanh neural networks. *Neural Networks*, 2021.
[7] T. De Ryck and S. Mishra. Error analysis for physics informed neural networks (PINNs) approximating Kolmogorov PDEs. *arXiv preprint arXiv:2106.14473*, 2021.
[8] T. De Ryck and S. Mishra. Generic bounds on the approximation error for physics-informed (and) operator learning. *arXiv preprint arXiv:2205.11393*, 2022.
[9] M. Dissanayake and N. Phan-Thien. Neural-network-based approximations for solving partial differential equations. *Communications in Numerical Methods in Engineering*, 1994.
[10] W. E, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, 2017.
[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:*, 2014.
[12] H. Holden and N. H. Risebro. *Front tracking for hyperbolic conservation laws*, volume 152. Springer, 2015.
[13] Z. Hu, A. D. Jagtap, G. E. Karniadakis, and K. Kawaguchi. When do extended physics-informed neural networks (XPINNs) improve generalization? *arXiv preprint arXiv:2109.09444*, 2021.
[14] A. D. Jagtap and G. E. Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.
[15] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.
[16] A. D. Jagtap, Z. Mao, N. Adams, and G. E. Karniadakis. Physics-informed neural networks for inverse problems in supersonic flows. *arXiv preprint arXiv:2202.11821*, 2022.
[17] A. D. Jagtap, D. Mitsotakis, and G. E. Karniadakis. Deep learning of inverse water waves problems using multi-fidelity data: Application to Serre–Green–Naghdi equations. *Ocean Engineering*, 248:110775, 2022.
[18] X. Jin, S. Cai, H. Li, and G. E. Karniadakis. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 426:109951, 2021.
[19] E. Kharazmi, Z. Zhang, and G. E. Karniadakis. Variational physics informed neural networks for solving partial differential equations. *arXiv preprint arXiv:1912.00873*, 2019.
[20] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider. A theoretical analysis of deep neural networks and parametric pdes. *Constructive Approximation*, pages 1–53, 2021.
[21] I. E. Lagaris, A. Likas, and P. G. D. Neural-network methods for boundary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11:1041–1049, 2000.
[22] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 2000.
[23] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
[24] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations, 2020.
[25] L. Lu, P. Jin, and G. E. Karniadakis. DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.
[26] K. O. Lye, S. Mishra, and D. Ray. Deep learning observables in computational fluid dynamics. *Journal of Computational Physics*, page 109339, 2020.
[27] K. O. Lye, S. Mishra, D. Ray, and P. Chandrashekar. Iterative surrogate model optimization (ISMO): An active learning algorithm for pde constrained optimization with deep neural networks. *Computer Methods in Applied Mechanics and Engineering*, 374:113575, 2021.
[28] Z. Mao, A. D. Jagtap, and G. E. Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.
[29] S. Mishra and R. Molinaro. Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for pdes. *IMA Journal of Numerical Analysis*, 2021.
[30] S. Mishra and R. Molinaro. Physics informed neural networks for simulating radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 270:107705, 2021.

[31] S. Mishra and R. Molinaro. Estimates on the generalization error of physics-informed neural networks for approximating PDEs. *IMA Journal of Numerical Analysis*, 01 2022. drab093.
[32] S. Mishra and T. K. Rusch. Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences. *SIAM Journal on Numerical Analysis*, 59(3):1811–1834, 2021.
[33] G. Pang, L. Lu, and G. E. Karniadakis. fPINNs: Fractional physics-informed neural networks. *SIAM journal of Scientific computing*, 41:A2603–A2626, 2019.
[34] P. Petersen and F. Voigtlaender. Optimal approximation of piecewise smooth functions using deep relu neural networks. *Neural Networks*, 108:296–330, 2018.
[35] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.
[36] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
[37] M. Raissi, A. Yazdani, and G. E. Karniadakis. Hidden fluid mechanics: A Navier-Stokes informed deep learning framework for assimilating flow visualization data. *arXiv preprint arXiv:1808.04327*, 2018.
[38] C. Schwab and J. Zech. Deep learning in high dimension: Neural network expression rates for generalized polynomial chaos expansions in uq. *Analysis and Applications*, 17(01):19–55, 2019.
[39] Y. Shin, J. Darbon, and G. E. Karniadakis. On the convergence and generalization of physics informed neural networks. *arXiv preprint arXiv:2004.01806*, 2020.
[40] Y. Shin, Z. Zhang, and G. E. Karniadakis. Error estimates of residual minimization using neural networks for linear equations. *arXiv preprint arXiv:2010.08019*, 2020.
[41] K. Shukla, A. D. Jagtap, J. L. Blackshire, D. Sparkman, and G. E. Karniadakis. A physics-informed neural network for quantifying the microstructural properties of polycrystalline nickel using ultrasound data: A promising approach for solving inverse problems. *IEEE Signal Processing Magazine*, 39(1):68–77, 2021.
[42] K. Shukla, A. D. Jagtap, and G. E. Karniadakis. Parallel physics-informed neural networks via domain decomposition. *Journal of Computational Physics*, 447:110683, 2021.
[43] S. Wang and P. Perdikaris. Long-time integration of parametric evolution equations with physics-informed DeepONets. *arXiv preprint arXiv:2106.05384*, 2021.
[44] L. Yang, X. Meng, and G. E. Karniadakis. B-PINNs: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.

Appendix A. Auxiliary results

A.1. **Auxiliary results for Section 3.2.** We will use the partition of unity construction using tanh neural networks from [6].

Lemma A.1. *Let $\varepsilon > 0$ and $1 \leq p < \infty$. For any $\varphi \in \Phi_\varepsilon$ it holds that $|\varphi|_{W^{1,p}} = \mathcal{O}(\beta^{1+2(p-1)/p})$, $p \in \mathbb{N}$, and $|\varphi|_{W^{1,\infty}} = \mathcal{O}(\beta^3)$.*

Proof. One can easily calculate that $\partial_t \overline{\varphi}_\varepsilon = \alpha\overline{\varphi}_\varepsilon + \beta\overline{\varphi}_\varepsilon$ and $\partial_x \overline{\varphi}_\varepsilon = \beta\overline{\varphi}_\varepsilon$. Moreover, one can easily find that $\|\overline{\varphi}_\varepsilon\|_{L^\infty} = \mathcal{O}(\beta^2)$ and $\|\overline{\varphi}_\varepsilon\|_{L^1} = \mathcal{O}(1)$. As a result, one can find that for $p \in \mathbb{N}$ it holds that $|\overline{\varphi}_\varepsilon|_{W^{1,p}} = \mathcal{O}(\beta^{2(p-1)/p})$. The bound from the statement follows immediately. □

Lemma A.2. *Let $N \in \mathbb{N}$ and $0 < \varepsilon < 1$. If we set $\alpha = N \ln\big(N^2/\varepsilon\big)$ then it holds that,*

$$\alpha/N \geq 1, \quad 1 - \sigma(\alpha/N) \leq \varepsilon, \quad \alpha^m \left|\sigma^{(m)}(\alpha/N)\right| \leq \varepsilon \text{ for } m = 1, 2. \tag{A.1}$$

Proof. The statement follows directly from [5, Lemma B.4] by using that $4/e^2 \leq 1$. □

The following auxiliary results are needed in the proof of Theorem 3.7.

Lemma A.3. *Let $\delta, T > 0$, $0 < \delta < T/2$, $3\delta \leq z \leq T - 3\delta$ and let $f \in C^1([0,T] \setminus \{z\})$. Define $\chi_\delta(t) = \gamma(\sigma(\alpha(t-2\delta)) - \sigma(\alpha(t-T+2\delta)))$ where $\alpha = \ln\big(1/\delta^3\big)/\delta$ and $\gamma^{-1} = \sigma(\alpha\delta)$. Then it holds that*

$$\left|\int_0^T f(t)\chi_\delta'(t)dt - f(2\delta) + f(T-2\delta)\right| \leq \Big(T\|f\|_{L^\infty([0,T])} + 3\ln\big(1/\delta\big)\big\|f'\big\|_{L^\infty(B)}\Big)\frac{4\delta}{1-\delta}, \tag{A.2}$$

where $B = [\delta, 3\delta] \cup [T-3\delta, T-\delta]$.

Proof. Define $\alpha = \ln\big(1/\delta^3\big)/\delta$ and $\gamma^{-1} = \alpha\int_{-\delta}^{\delta}\sigma'(\alpha s)ds = 2\sigma(\alpha\delta) \geq 2(1-\delta)$, where the last inequality follows from Lemma A.2 with $N = \varepsilon \leftarrow \delta$. Assume that $z \geq 3\delta$. Taylor's theorem then guarantees the existence of $\zeta_t \in [\delta, 3\delta]$ such that,

$$\int_\delta^{3\delta} f(t)\gamma\alpha\sigma'(\alpha(t-2\delta))dt = f(2\delta)\gamma\alpha\int_\delta^{3\delta}\sigma'(\alpha(t-2\delta))dt + \gamma\alpha\int_\delta^{3\delta} f'(\zeta_t)(t-2\delta)\sigma'(\alpha(t-2\delta))dt. \tag{A.3}$$

From this, it follows using the definition of γ that,

$$\left|\int_\delta^{3\delta} f(t)\gamma\alpha\sigma'(\alpha(t-2\delta))dt - f(2\delta)\right| \leq \gamma\alpha\delta^2\big\|f'\big\|_{L^\infty([\delta,3\delta])}. \tag{A.4}$$

Moreover, it follows from Lemma A.2 that,

$$\left|\int_{[0,\delta]\cup[3\delta,T]} f(t)\gamma\alpha\sigma'(\alpha(t-2\delta))dt\right| \leq T\gamma\|f\|_\infty\delta. \tag{A.5}$$

Similarly we can find for $A=[T-3\delta,T-\delta]$ that,

$$\left|\int_0^T f(t)\gamma\alpha\sigma'(\alpha(t-T+2\delta))dt - f(T-2\delta)\right| \leq \gamma\alpha\delta^2\left\|f'\right\|_{L^\infty(A)} + T\gamma\|f\|_\infty\delta. \tag{A.6}$$

Conclude by combining all the previous inequalities. □

Lemma A.4. *Let $0<\varepsilon<\min\{1,b-a\}$ and $f\in C^1([a,b])$. Define*

$$\rho_\varepsilon(x) = \frac{\sigma(\beta(x+\varepsilon^2)) - \sigma(\beta(x-\varepsilon^2))}{2\varepsilon^2}, \tag{A.7}$$

where $\beta=-3\ln(\varepsilon)/\varepsilon$ and $\omega(t)=\sigma(\beta(t-\max\{a,t-\varepsilon\})) - \sigma(\beta(t-\min\{b,t+\varepsilon\}))$, for which it holds that $1-\varepsilon\leq\omega(t)\leq 2$. Then it holds that,

$$\left|\int_a^b f(s)\rho_\varepsilon(t-s)ds - \omega(t)f(t)\right| \leq 20\|f\|_{C^1([a,b])}(b-a-\ln(\varepsilon))\varepsilon. \tag{A.8}$$

Proof. First we observe that for any $\varepsilon>0$ it holds that,

$$\begin{aligned}\left|\int_a^b f(s)\beta\sigma'(\beta(t-s))ds - \int_a^b f(s)\frac{\sigma(\beta(t-s+\varepsilon^2)) - \sigma(\beta(t-s-\varepsilon^2))}{2\varepsilon^2}ds\right| \\ \leq 2(\varepsilon^2\beta)^2\left\|\sigma'''\right\|_\infty\|f\|_{L^1} \leq 20(b-a)\|f\|_\infty\varepsilon,\end{aligned} \tag{A.9}$$

where for the last inequality we used that $\left\|\sigma'''\right\|_\infty\leq 2$ and $\varepsilon\ln\left(1/\varepsilon^3\right)^2\leq 5$ for $0<\varepsilon<1$.

Next, we let $A=[a,b]\cap[t-\varepsilon,t+\varepsilon]$ and $B=[a,b]\setminus[t-\varepsilon,t+\varepsilon]$. Taylor's theorem then guarantees the existence of $\zeta_t\in A$ such that,

$$\int_A f(s)\beta\sigma'(\beta(t-s))ds = f(t)\int_A \beta\sigma'(\beta(t-s))ds + \beta\int_A f'(\zeta_t)(s-t)\sigma'(\beta(t-s))ds \tag{A.10}$$

Using that if $\varepsilon\leq b-a$ then,

$$2\geq\omega(t)=\beta\int_A\sigma'(\beta(t-s))ds \geq \beta\int_0^\varepsilon\sigma'(\beta s)ds = \sigma(\beta\varepsilon)\geq 1-\varepsilon, \tag{A.11}$$

together with Lemma A.2 gives us,

$$\begin{aligned}&\left|\int_a^b f(s)\beta\sigma'(\beta(t-s))ds - \omega(t)f(t)\right| \\ &\quad\leq \beta\int_A\left|f'(\zeta_t)(s-t)\sigma'(\beta(t-s))\right|ds + \beta\int_B\left|f(s)\sigma'(\beta(t-s))\right|ds \\ &\quad\leq \frac{\beta\left\|f'\right\|_{L^\infty(A)}}{N^2} + (b-a)\|f\|_\infty\varepsilon \\ &\quad\leq \|f\|_{C^1([a,b])}(b-a+\ln\left(1/\varepsilon^3\right))\varepsilon.\end{aligned} \tag{A.12}$$

□

Lemma A.5. *Let $\eta>0$. Define the function $|\cdot|_\eta:\mathbb{R}\to[0,\infty):x\mapsto|x|_\eta=\sqrt{x^2+\eta^2}$. It holds that*

$$0\leq|x|_\eta-|x|\leq\eta, \qquad \left|\frac{d}{dx}|x|_\eta\right|\leq 1. \tag{A.13}$$

Proof. The first set of inequalities follows from

$$0\leq(|x|_\eta-|x|)(|x|_\eta+|x|) = |x|_\eta^2-|x|^2=\eta^2 \qquad\text{and}\qquad |x|_\eta+|x|\geq\eta, \tag{A.14}$$

and the other inequality follows from $\left|\frac{d}{dx}|x|_\eta\right| = \frac{|x|}{\sqrt{x^2+\eta^2}}\leq 1$. □

Lemma A.6. *Let $B > 0$, $0 < \varepsilon < 1$, $T > 4\varepsilon$, $z \in W^{1,\infty}(([0,1] \times [0,T])^2)$ with $\left|z(x,t,y,s)\right| \leq B$ for all $(x,t),(y,s) \in [0,1] \times [0,T]$ and $\overline{\varphi}_\varepsilon^{y,s}$, $(y,s) \in [0,1] \times [0,T]$, as defined in* (3.10)*. There exists an constant $C > 0$ (independent of z and ε) such that,*

$$\left| \int_0^1 \int_0^T \int_0^1 \int_0^T \left(\overline{\varphi}_\varepsilon^{y,s}(x,t) \partial_t z(x,t,y,s) + z(x,t,y,s) \partial_t \overline{\varphi}_\varepsilon^{y,s}(x,t) \right) dt dx ds dy \right| \leq CB\varepsilon. \tag{A.15}$$

Proof. Using integration by parts and Fubini's theorem we find that,

$$\begin{aligned} &\int_0^1 \int_0^T \int_0^1 \int_0^T \left(\overline{\varphi}_\varepsilon^{y,s}(x,t) \partial_t z(x,t) + z(x,t) \partial_t \overline{\varphi}_\varepsilon^{y,s}(x,t) \right) dt dx ds dy \\ &\quad = \int_0^1 \int_0^1 \int_0^T \left(z(x,T) \overline{\varphi}_\varepsilon^{y,s}(x,T) - z(x,0) \overline{\varphi}_\varepsilon^{y,s}(x,0) \right) ds dx dy. \end{aligned} \tag{A.16}$$

We will now bound the absolute value of both terms on the RHS of the above equation. Both terms can be bounded in a similar way, we only show the proof for the upper bound of the second term. Observe that χ_ε is increasing on $[0,\varepsilon]$ and ρ_ε is decreasing on $[0,T]$. Moreover, it holds that $\overline{\varphi}_\varepsilon^{y,s} \geq 0$. Using this information we find for any $x, y \in [0,1]$ that,

$$\begin{aligned} \left| \int_0^T z(x,0) \overline{\varphi}_\varepsilon^{y,s}(x,0) ds \right| &\leq B \rho_\varepsilon(x-y) \int_0^T \chi_\varepsilon \left(\frac{s}{2} \right) \rho_\varepsilon(s) ds \\ &\leq B \rho_\varepsilon(x-y) \left[\chi_\varepsilon(\varepsilon) \int_0^\varepsilon \rho_\varepsilon(s) ds + \rho_\varepsilon(\varepsilon) \int_\varepsilon^T \chi_\varepsilon \left(\frac{s}{2} \right) ds \right]. \end{aligned} \tag{A.17}$$

From Lemma A.3 and Lemma A.4 it follows that the two integrals above are at most 2. Furthermore if $T > 4\varepsilon$ it holds that,

$$\chi_\varepsilon(\varepsilon) = \frac{1}{2\sigma(\alpha\varepsilon)} (\sigma(-\alpha\varepsilon) - \sigma(\alpha(3\varepsilon - T))) \leq (1 - (1 - \varepsilon)) = \varepsilon. \tag{A.18}$$

By using a Taylor approximation (cf. (A.9) in the proof of Lemma A.4) we find that,

$$\rho_\varepsilon(\varepsilon) \leq \rho_\varepsilon(\varepsilon^3) \leq \beta \sigma'(\beta \varepsilon^3) + 20\varepsilon^3 \leq 21\varepsilon^3, \tag{A.19}$$

where the second inequality follows from the definition of β and Lemma A.2. As a result we find from (A.17) that there exists an absolute constant $C > 0$ such that,

$$\left| \int_0^T z(x,0) \overline{\varphi}_\varepsilon^{y,s}(x,0) ds \right| \leq CB \rho_\varepsilon(x-y) \varepsilon \tag{A.20}$$

Combining the above with Lemma A.4 gives,

$$\left| \int_0^1 \int_0^1 \int_0^T z(x,0) \overline{\varphi}_\varepsilon^{y,s}(x,0) ds dx dy \right| \leq CB\varepsilon \int_0^1 \int_0^1 \rho_\varepsilon(x-y) dx dy \leq 2CB\varepsilon. \tag{A.21}$$

Similarly it holds that,

$$\left| \int_0^1 \int_0^1 \int_0^T z(x,T) \overline{\varphi}_\varepsilon^{y,s}(x,T) ds dx dy \right| \leq 2CB\varepsilon. \tag{A.22}$$

Combining the two above equation with (A.16) and redefining C then concludes the proof of the lemma. □

Lemma A.7. *Let $T, \varepsilon > 0$, $u \in W^{1,\infty}([0,1] \times [0,T])$ with $u(0,t) = u(1,t)$ for all $t \in [0,T]$ and $\overline{\varphi}_\varepsilon^{y,s}$, $(y,s) \in [0,1] \times [0,T]$, as defined in* (3.10)*. There exists an constant $C > 0$ (independent of u, v and ε) such that,*

$$\begin{aligned} &\int_0^1 \int_0^T \int_0^1 \int_0^T \left(\overline{\varphi}_\varepsilon^{y,s}(x,t) \partial_x Q[u(x,t); v(y,s)] + Q[u(x,t); v(y,s)] \partial_x \overline{\varphi}_\varepsilon^{y,s}(x,t) \right) dt dx ds dy \\ &\quad \leq 12 L_f \int_0^T \left| v(1,t) - v(0,t) \right| dt + C L_f \|v_x\|_\infty (1 - \ln(\varepsilon)) \varepsilon \end{aligned} \tag{A.23}$$

Proof. Using integration by parts and Fubini's theorem we find that,

$$\begin{aligned}
&\int_0^1\int_0^1 \left(\overline{\varphi}_\varepsilon^{y,s}(x,t)\partial_x Q[u(x,t);v(y,s)] + Q[u(x,t);v(y,s)]\partial_x\overline{\varphi}_\varepsilon^{y,s}(x,t)\right)dxdy\\
&\quad= \int_0^1 \left(Q[u(1,t);v(y,s)]\overline{\varphi}_\varepsilon^{y,s}(1,t) - Q[u(0,t);v(y,s)]\overline{\varphi}_\varepsilon^{y,s}(0,t)\right)dy\\
&\quad= \int_0^1 \left(Q[u(1,t);v(1,s)]\overline{\varphi}_\varepsilon^{y,s}(1,t) - Q[u(0,t);v(0,s)]\overline{\varphi}_\varepsilon^{y,s}(0,t)\right)dy \quad =:(A)\\
&\qquad+ \int_0^1 \left(Q[u(1,t);v(y,s)] - Q[u(1,t);v(1,s)]\right)\overline{\varphi}_\varepsilon^{y,s}(1,t)dy \quad =:(B)\\
&\qquad+ \int_0^1 \left(Q[u(0,t);v(0,s)] - Q[u(0,t);v(y,s)]\right)\overline{\varphi}_\varepsilon^{y,s}(0,t)dy \quad =:(C).
\end{aligned} \tag{A.24}$$

We first bound (A). Recall that $\overline{\varphi}_\varepsilon^{y,s}(x,t) = \chi_\varepsilon\left(\frac{t+s}{2}\right)\rho_\varepsilon(t-s)\rho_\varepsilon(x-y) =: z(t,s)\rho_\varepsilon(x-y)$ and that $u(0,t) = u(1,t)$. We calculate,

$$\begin{aligned}
(A) = z(t,s)Q[u(1,t);v(1,s)]\int_0^1 \left(\rho_\varepsilon(1-y) - \rho_\varepsilon(y)\right)dy\\
+ z(t,s)\left(Q[u(1,t);v(1,s)] - Q[u(0,t);v(0,s)]\right)\int_0^1 \rho_\varepsilon(y)dy.
\end{aligned} \tag{A.25}$$

Using Lemma 3.1, Lemma A.4 and the fact that $u(0,t) = u(1,t)$ we find that,

$$\left|(A)\right| \leq z(t,s)\cdot 3L_f\left|v(1,s) - v(0,s)\right|\cdot 2 \tag{A.26}$$

and as a result,

$$\left|\int_0^T\int_0^T (A)\,dtds\right| \leq 12L_f\int_0^T \left|v(1,s) - v(0,s)\right|ds, \tag{A.27}$$

where we used that $\int_0^T \rho_\varepsilon(t-s)dt \leq 2$ (Lemma A.4).

The terms (B) and (C) are similar to each other and can be bounded in the same way. We will prove an upper bound on (C). Using Lemma 3.1 and the non-negativity of ρ_ε we find that,

$$\left|(C)\right| \leq 3L_f z(t,s)\int_0^1 \left|v(y,s) - v(0,s)\right|\rho_\varepsilon(y)dy. \tag{A.28}$$

In addition, using Lemma A.5 and Lemma A.4 (for any $\eta > 0$),

$$\begin{aligned}
\int_0^1 \left|v(y,s) - v(0,s)\right|\rho_\varepsilon(y)dy &\leq \int_0^1 \left|v(y,s) - v(0,s)\right|_\eta \rho_\varepsilon(y)dy + C\eta\\
&\leq 20\|v_x\|_\infty(1 - \ln(\varepsilon))\varepsilon + C\eta.
\end{aligned} \tag{A.29}$$

Finally we can easily see that $\int_0^T\int_0^T z(t,s)dsdt \leq 4$. Combining everything then proves the lemma. □

A.2. **Auxiliary results for Section 3.3.**

Lemma A.8. *Let $a, B, \mathfrak{L} \geq 1$, $k, d, M \in \mathbb{N}$, D a set, $(\Omega, \mathcal{A}, \mathbb{P})$ a probability space, $\Theta = [-a,a]^k$ and let $f : D \to \mathbb{R}$ and $f_\theta : D \to \mathbb{R}$ be functions for all $\theta \in \Theta$. Let $X_i : \Omega \to D$, $1 \leq i \leq M$ be iid random variables, $\mathcal{S} = \{X_1, \dots X_M\}$, let $\mathcal{E}_T : \Theta \times D^M \to [0,B]$ and $\mathcal{E}_G : \Theta \to [0,B]$ be given by*

$$\mathcal{E}_T(\theta,\mathcal{S}) = \frac{1}{M}\sum_{i=1}^M \left|f_\theta(z_i) - f(z_i)\right|, \quad \mathcal{E}_G(\theta)^2 = \int_D \left|f_\theta(z) - f(z)\right|d\mu(z), \tag{A.30}$$

and let $\theta^* : D^M \to \Theta$ be a function. Let $\theta \mapsto \mathcal{E}_G(\theta,\mathcal{S})$ and $\theta \mapsto \mathcal{E}_T(\theta)$ be Lipschitz continuous with Lipschitz constant $\mathfrak{L}$. If $M \geq e^{16/k}$ then it holds that

$$\mathbb{P}\left(\mathcal{E}_G(\theta^*(\mathcal{S})) > \mathcal{E}_T(\theta^*(\mathcal{S}),\mathcal{S}) + \sqrt{\frac{B^2k}{2M}\ln\left(\frac{a\mathfrak{L}\sqrt{M}}{\sqrt[k]{\delta}B}\right)}\right) \leq \delta. \tag{A.31}$$

Proof. Let $\varepsilon > 0$ be arbitrary, let $\{\theta_i\}_{i=1}^N$ be a δ-covering of Θ with respect to the supremum norm and define the random variable $Y = \mathcal{E}_G(\theta^*(\mathcal{S})) - \mathcal{E}_T(\theta^*(\mathcal{S}), \mathcal{S})$. Then if follows from equation (4.8) in the proof of [7, Theorem 5] that

$$\mathbb{P}(Y > \varepsilon) \leq \left(\frac{2a\mathfrak{L}}{\varepsilon}\right)^k \exp\left(\frac{-2\varepsilon^2 M}{B^2}\right), \tag{A.32}$$

since $\mathbb{P}(Y > \varepsilon) = 1 - \mathbb{P}(\mathcal{A})$, where $\mathcal{A}$ is as defined in the proof of [7, Theorem 5]. Setting $\delta = \mathbb{P}(Y > \varepsilon)$ leads to

$$\varepsilon = \sqrt{\frac{B^2}{2M} \ln\left(\frac{1}{\delta}\left(\frac{2a\mathfrak{L}}{\varepsilon}\right)^k\right)}. \tag{A.33}$$

If $\delta\varepsilon^k \leq Be^{-8}(2a\mathfrak{L})^k$ then it holds that

$$\left[\ln\left(\frac{1}{\delta}\left(\frac{2a\mathfrak{L}}{\varepsilon}\right)^k\right)\right]^{-1/2} \leq \frac{1}{2\sqrt{2}}. \tag{A.34}$$

Using (A.33) and (A.34) gives us,

$$\begin{aligned} \varepsilon &\leq \sqrt{\frac{B^2}{2M} \ln\left(\frac{(2a\mathfrak{L})^k}{\delta}\left(\frac{\sqrt{2M}}{B}\left[\ln\left(\frac{1}{\delta}\left(\frac{2a\mathfrak{L}}{\varepsilon}\right)^k\right)\right]^{-1/2}\right)^k\right)} \\ &\leq \sqrt{\frac{B^2}{2M} \ln\left(\frac{(a\mathfrak{L}\sqrt{M})^k}{\delta B^k}\right)} = \sqrt{\frac{B^2 k}{2M} \ln\left(\frac{a\mathfrak{L}\sqrt{M}}{\sqrt[k]{\delta} B}\right)}. \end{aligned} \tag{A.35}$$

Finally, we can use (A.33) to observe that the condition $\delta\varepsilon^k \leq Be^{-8}(2a\mathfrak{L})^k$ is met if $\delta < 1 \leq B$ and $M \geq e^{16/k}$ because

$$\left(\frac{e^{8/k}}{2a\mathcal{L}}\right)^2 \frac{2}{B^2 \ln\big((2a\mathcal{L}/\varepsilon)^k/\delta\big)} \leq e^{16/k}. \tag{A.36}$$

□