INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 21, Number 5, Pages 609–628

doi: 10.4208/ijnam2024-1024

LEAST-SQUARES NEURAL NETWORK (LSNN) METHOD FOR LINEAR ADVECTION-REACTION EQUATION: NON-CONSTANT JUMPS

ZHIQIANG CAI, JUNPYO CHOI*, AND MIN LIU

Abstract. The least-squares ReLU neural network (LSNN) method was introduced and studied for solving linear advection-reaction equation with discontinuous solution in [4, 5]. The method is based on an equivalent least-squares formulation and [5] employs ReLU neural network (NN) functions with $\lceil \log_2(d+1) \rceil + 1$-layer representations for approximating solutions. In this paper, we show theoretically that the method is also capable of accurately approximating non-constant jumps along discontinuous interfaces that are not necessarily straight lines. Theoretical results are confirmed through multiple numerical examples with $d = 2, 3$ and various non-constant jumps and interface shapes, showing that the LSNN method with $\lceil \log_2(d+1) \rceil + 1$ layers approximates solutions accurately with degrees of freedom less than that of mesh-based methods and without the common Gibbs phenomena along discontinuous interfaces having non-constant jumps.

Key words. Least-squares method, ReLU neural network, linear advection-reaction equation, discontinuous solution.

1. Introduction

For decades, extensive research has been conducted on numerical methods for linear advection-reaction equations to develop precise and efficient numerical schemes. A major challenge in numerical simulation is that the solution of the equation is discontinuous along an interface because of a discontinuous inflow boundary condition, where the discontinuous interface can be the streamline from the inflow boundary. Traditional mesh-based numerical methods often exhibit oscillations near the discontinuity (called the Gibbs phenomena), which are not suitable for many applications and may not be extended to nonlinear hyperbolic conservation laws.

Recently, the application of neural networks (NNs) for solving partial differential equations have achieved significant accomplishments. For the linear advection-reaction problem, the least-squares ReLU neural network (LSNN) method was introduced and studied in [4, 5]. The method is based on an equivalent least-squares formulation studied in [2, 6] and [5] employs ReLU neural network (NN) functions with $\lceil \log_2(d+1) \rceil + 1$-layer representations for approximating the solution. The LSNN method is capable of automatically approximating the discontinuous solution accurately since the free hyperplanes of ReLU NN functions adapt to the solution (see [3, 4, 5]). Moreover, for problems with unknown locations of discontinuity interfaces, it is quite easy to see that the LSNN method uses much fewer number of degrees of freedom than the mesh-based methods (see, e.g., [3, 4]).

Approximation properties of ReLU NN functions to step functions were examined and employed in [4, 5]. In particular, it was shown theoretically that two- or $\lceil \log_2(d+1) \rceil + 1$-layer ReLU NN functions are necessary and sufficient to approximate a step function with any given accuracy $\varepsilon > 0$ when the discontinuous interface is a hyperplane or general hyper-surface, respectively. This approximation

Received by the editors on May 30, 2023 and, accepted on December 15, 2023.
2000 *Mathematics Subject Classification.* 65N15, 65N99.
*Corresponding author.

property was used to establish *a priori* error estimates of the LSNN method for the linear advection-reaction problem.

The jump of the discontinuous solution of the problem, however, is generally non-constant when the reaction coefficient is non-zero. The main purpose of this paper is to establish *a priori* error estimates (see Theorem 3.2) for the LSNN method without making the assumption that the jump is constant as in [5]. To this end, we decompose the solution as the sum of discontinuous and continuous parts (see (10)), so that the discontinuous part of the solution can be described as a cylindrical surface on one subdomain and zero otherwise. Then we construct a continuous piecewise linear (CPWL) function with a sharp transition layer along the discontinuous interface to approximate the discontinuous piecewise cylindrical surface accurately. From [10, 11, 1, 5], we know that the CPWL function is a ReLU NN function $\mathbb{R}^d \to \mathbb{R}$ with a $\lceil \log_2(d+1) \rceil + 1$-layer representation, from which it follows that the discontinuous part of the solution can be approximated by this class of functions for any prescribed accuracy. Then Theorem 3.2 follows.

The rest of the paper is organized as follows. In Section 2, we introduce the linear advection-reaction problem, and briefly review and discuss properties of ReLU NN functions and the LSNN method in [5]. Then theoretical convergence analysis is conducted in Section 3, showing that discretization error of the method for the problem mainly depends on the continuous part of the solution. Finally, to demonstrate the effectiveness of the method, we provide numerical results for test problems with various non-constant jumps in Section 4. Section 5 summarizes the work.

2. Problem formulation and the LSNN method

Let Ω be a bounded domain in $\mathbb{R}^d$ ($d \geq 2$) with Lipschitz boundary $\partial\Omega$, and denote the advective velocity field by $\boldsymbol{\beta}(\mathbf{x}) = (\beta_1, \cdots, \beta_d)^T \in C^0(\bar{\Omega})^d$. Define the inflow part of the boundary $\Gamma = \partial\Omega$ by

$$\Gamma_- = \{\mathbf{x} \in \Gamma : \boldsymbol{\beta}(\mathbf{x}) \cdot \boldsymbol{n}(\mathbf{x}) < 0\} \tag{1}$$

where $\boldsymbol{n}(\mathbf{x})$ is the unit outward normal vector to Γ at $\mathbf{x} \in \Gamma$. Consider the linear advection-reaction equation

$$\begin{cases} u_{\boldsymbol{\beta}} + \gamma\, u &= f, \quad \text{in } \Omega, \\ u &= g, \quad \text{on } \Gamma_-, \end{cases} \tag{2}$$

where $u_{\boldsymbol{\beta}}$ denotes the directional derivative of u along $\boldsymbol{\beta}$. Assume that $\gamma \in C^0(\bar{\Omega})$, $f \in L^2(\Omega)$, and $g \in L^2(\Gamma_-)$ are given scalar-valued functions.

For the convenience of the reader, this section briefly reviews properties of ReLU neural network (NN) functions and the least-squares ReLU neural network (LSNN) method in [5]. A function $\mathcal{N} : \mathbb{R}^d \to \mathbb{R}^c$ is called a ReLU neural network (NN) function if it can be expressed as a composition of functions

$$N^{(L)} \circ \cdots \circ N^{(2)} \circ N^{(1)} \text{ with } L > 1, \tag{3}$$

where $N^{(l)} : \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$ ($n_0 = d$, $n_L = c$) is affine linear when $l = L$, and affine linear with the rectified linear unit (ReLU) activation function σ applied to each component when $1 \leq l \leq L-1$. Each affine linear function takes the form $\boldsymbol{\omega}^{(l)}\mathbf{x} - \mathbf{b}^{(l)}$ for $\mathbf{x} \in \mathbb{R}^{n_{l-1}}$ where $\boldsymbol{\omega}^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$, $\mathbf{b}^{(l)} \in \mathbb{R}^{n_l}$ are weight and bias matrices, respectively. For $n \in \mathbb{N}$, denote the collection of all ReLU NN functions from $\mathbb{R}^d$ to $\mathbb{R}$ with depth L and the number of hidden neurons $n(= n_1 + \cdots + n_{L-1})$

by $\mathcal{M}(d,1,L,n)$ (1 being the output dimension), and the collection of all ReLU NN functions from $\mathbb{R}^d$ to $\mathbb{R}$ with depth L by $\mathcal{M}(d,1,L)$. Then we have

$$\mathcal{M}(d,1,L) = \bigcup_{n\in\mathbb{N}} \mathcal{M}(d,1,L,n). \tag{4}$$

The following proposition justifies our use of $\lceil \log_2(d+1)\rceil + 1$-layer ReLU NN functions.

Proposition 2.1 (see [1, 5])**.** *The collection of all continuous piecewise linear* (CPWL) *functions on* $\mathbb{R}^d$ *is equal to* $\mathcal{M}(d,1,\lceil \log_2(d+1)\rceil+1)$*, i.e., the collection of all* ReLU NN *functions from* $\mathbb{R}^d$ *to* $\mathbb{R}$ *that have representations with depth* $\lceil \log_2(d+1)\rceil + 1$.

Proposition 2.2. $\mathcal{M}(d,1,L,n) \subseteq \mathcal{M}(d,1,L,n+1)$.

Proposition 2.2 implies that as we increase n, $\mathcal{M}(d,1,L,n)$ approaches $\mathcal{M}(d,1,L)$ and the approximation class gets larger.

As in [5], breaking hyperplanes are depicted in Figures 2 to 7 to better understand the graphs of ReLU NN function approximations using domain partitions (on each element in a given partition, the ReLU NN function is affine linear; see [5]). More specifically, the l^{th}- (hidden) layer breaking hyperplanes of a given ReLU NN function (with output dimension 1) representation as in (3) are defined as the zero sets of the layer without the activation function: when $l=1$, $\mathbf{w}_i^{(1)}\mathbf{x} - b_i^{(1)} = 0$, and when $2 \le l < L$,

$$\mathbf{w}_i^{(l)}\left(N^{(l-1)} \circ \cdots \circ N^{(2)} \circ N^{(1)}(\mathbf{x})\right) - b_i^{(l)} = 0,$$

where

$$\boldsymbol{\omega}^{(l)} = (\mathbf{w}_1^{(l)}, \ldots, \mathbf{w}_{n_l}^{(l)})^T \in \mathbb{R}^{n_l \times n_{l-1}}, \quad \text{and} \quad \mathbf{b}^{(l)} = (b_1^{(l)}, \ldots, b_{n_l}^{(l)})^T.$$

Define the least-squares (LS) functional

$$\mathcal{L}(v;\mathbf{f}) = \|v_{\boldsymbol{\beta}} + \gamma\, v - f\|_{0,\Omega}^2 + \|v-g\|_{-\boldsymbol{\beta}}^2, \tag{5}$$

where $\mathbf{f} = (f,g)$ and $\|\cdot\|_{-\boldsymbol{\beta}}$ is given by

$$\|v\|_{-\boldsymbol{\beta}} = \langle v, v\rangle_{-\boldsymbol{\beta}}^{1/2} = \left(\int_{\Gamma_-} |\boldsymbol{\beta}\cdot\boldsymbol{n}|\, v^2\, ds\right)^{1/2}.$$

The LS formulation of problem (2) is to seek $u \in V_{\boldsymbol{\beta}}$ such that

$$\mathcal{L}(u;\mathbf{f}) = \min_{v\in V_{\boldsymbol{\beta}}} \mathcal{L}(v;\mathbf{f}), \tag{6}$$

where $V_{\boldsymbol{\beta}} = \{v \in L^2(\Omega) : v_{\boldsymbol{\beta}} \in L^2(\Omega)\}$ is a Hilbert space that is equipped with the norm

$$|||v|||_{\boldsymbol{\beta}} = \left(\|v\|_{0,\Omega}^2 + \|v_{\boldsymbol{\beta}}\|_{0,\Omega}^2\right)^{1/2}.$$

Then the corresponding LS and discrete LS approximations are, respectively, to find $u_{_N} \in \mathcal{M}(d,1,L,n)$ such that

$$\mathcal{L}\big(u_{_N};\mathbf{f}\big) = \min_{v\in\mathcal{M}(d,1,L,n)} \mathcal{L}\big(v;\mathbf{f}\big), \tag{7}$$

and to find $u_{\mathcal{T}}^{N} \in \mathcal{M}(d,1,L,n)$ such that

$$\mathcal{L}_{\mathcal{T}}\big(u_{\mathcal{T}}^{N};\mathbf{f}\big) = \min_{v\in\mathcal{M}(d,1,L,n)} \mathcal{L}_{\mathcal{T}}\big(v;\mathbf{f}\big), \tag{8}$$

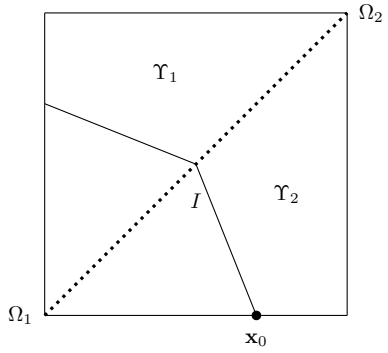
where $\mathcal{L}_{\mathcal{T}}\big(v;\mathbf{f}\big)$ is the discrete LS functional (see [4, 5]).

3. Error estimates

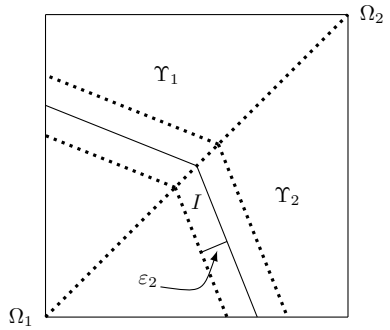
In this section, we establish error estimates of the LSNN method for the linear advection-reaction equation with a non-constant jump along a discontinuous interface. For simplicity, we will restrict our attention to two dimensions.

To this end, assume the advection velocity field $\boldsymbol{\beta}$ is piecewise constant. That is, there exists a partition of the domain Ω such that $\boldsymbol{\beta}$ has the same direction but possibly a different magnitude at each interior point of each subdomain. Without loss of generality, assume that there are only two sub-domains: $\Omega = \Upsilon_1 \cup \Upsilon_2$ and that the inflow boundary data $g(\mathbf{x})$ is discontinuous at only one point $\mathbf{x}_0 \in \Gamma_-$ with $g(\mathbf{x}_0^-) = \alpha_1$ and $g(\mathbf{x}_0^+) = \alpha_2$ from different sides. (Figure 1(a) depicts Υ_1 and Υ_2 as the left-upper and the right-lower triangles, respectively.)

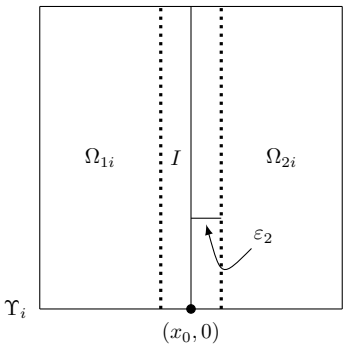
Let I be the streamline emanating from $\mathbf{x}_0$; then the discontinuous interface I divides the domain Ω into two sub-domains: $\Omega = \Omega_1 \cup \Omega_2$, where Ω_1 and Ω_2 are the left-lower and the right-upper subdomains separated by the discontinuous interface I, respectively (see Figure 1(a)). The corresponding solution u of (2) is discontinuous across the interface I and is piecewise smooth with respect to the partition $\{\Omega_1, \Omega_2\}$. For a given $\varepsilon_2 > 0$, take an ε_2 neighborhood around the interface I in the direction of $\boldsymbol{\beta}$ as in Figure 1(b).



(a) Subdomains Υ_1, Υ_2 with respect to $\boldsymbol{\beta}$



(b) An ε_2 neighborhood around the interface I in the direction of $\boldsymbol{\beta}$



(c) A subdomain Υ_i

FIGURE 1. A domain decomposition for the case that $\boldsymbol{\beta}$ is piecewise constant.

Next, we estimate the error in the sub-domain Υ_i, say, Υ_2. To further simplify the error estimate, we assume that

$$\Upsilon_i = (0,1)\times(0,1), \quad \boldsymbol{\beta}(\mathbf{x}) = (0, v_2(\mathbf{x}))^T, \quad \text{and} \quad \mathbf{x}_0 = (x_0, 0) \text{ for } x_0 \in (0,1).$$

These assumptions imply that the restriction of the interface I to Υ_i is a vertical line segment

$$I = \{(x_0, y) \in \Upsilon_i : y \in (0,1)\},$$

and that Υ_i is partitioned into two sub-domains

$$\Omega_{1i} = \{\mathbf{x} = (x,y) \in \Upsilon_i : x < x_0\} \quad \text{and} \quad \Omega_{2i} = \{\mathbf{x} = (x,y) \in \Upsilon_i : x > x_0\}$$

(see Figure 1(c)).

In $\Upsilon_i = \Omega_{1i} \cup \Omega_{2i} \cup I$, let u_1 and u_2 be the solutions of (2) defined only on Ω_{1i} with the constant inflow boundary conditions $g = \alpha_1$ and α_2 on $\{(x,0) : x \in [0, x_0]\}$, respectively. (When Υ_i is different from Υ_2, the discontinuous point is not $\mathbf{x}_0$ but an interior point of the domain Ω, and the values of the solution u at that discontinuous point from different sides are taken as the constant inflow boundary conditions.) We set $a(\mathbf{x}) = u_1(\mathbf{x}) - u_2(\mathbf{x})$ and let $\chi(\mathbf{x})$ be the piecewise discontinuous function defined by

$$\chi(\mathbf{x}) = \begin{cases} a(\mathbf{x}), & \mathbf{x} \in \Omega_{1i}, \\ 0, & \mathbf{x} \in \Omega_{2i}; \end{cases} \tag{9}$$

then the solution u of (2) has the following decomposition (see [5])

$$u(\mathbf{x}) = \hat{u}(\mathbf{x}) + \chi(\mathbf{x}) \text{ in } \Upsilon_i. \tag{10}$$

Here, $\hat{u}(\mathbf{x}) = u(\mathbf{x}) - \chi(\mathbf{x})$ is clearly piecewise smooth; moreover, it is also continuous in Υ_i since $\hat{u}\big|_I = u_2\big|_I$ from both sides. Then we have the following error estimate and postpone its proof to Appendix.

Theorem 3.1. *For any $\varepsilon_2 > 0$ and $\varepsilon_3 > 0$, on Υ_i, there exists a CPWL function $p_i(\mathbf{x})$ such that*

$$|||\chi - p_i|||_{\boldsymbol{\beta}} \le D_1\sqrt{\varepsilon_2} + D_2\sqrt{\varepsilon_3}. \tag{11}$$

Remark 3.1. *We now construct the CPWL function $p(\mathbf{x})$ on Ω defined by*

$$p(\mathbf{x}) = p_i(\mathbf{x}), \ \mathbf{x} \in \Upsilon_i,$$

such that $p_i(\mathbf{x}) = p_{i+1}(\mathbf{x})$ on the intersection of Υ_i and Υ_{i+1}. Using the triangle inequality, Theorem 3.1 can be extended to the case that $\boldsymbol{\beta}$ is piecewise constant to establish the error estimate on the whole domain Ω.

Theorem 3.2. *Let u and u_N be the solutions of problems (6) and (7), respectively. If the depth of ReLU NN functions in (7) is at least $\lceil \log_2(d+1) \rceil + 1$, then for a sufficiently large integer n, there exists an integer $\hat{n} \le n$ such that*

$$|||u - u_N|||_{\boldsymbol{\beta}} \le C\left(\sqrt{\varepsilon_2} + \sqrt{\varepsilon_3} + \inf_{v \in \mathcal{M}(d, n-\hat{n})} |||\hat{u} - v|||_{\boldsymbol{\beta}}\right), \tag{12}$$

where $\mathcal{M}(d, n-\hat{n}) = \mathcal{M}(d, 1, \lceil \log_2(d+1) \rceil + 1, n - \hat{n})$.

Proof. The proof is similar to that of Theorem 4.5 in [5]. □

Lemma 3.1. *Let u, $u_{\mathcal{N}}$, and $u_{\mathcal{T}}^{N}$ be the solutions of problems (6), (7), and (8), respectively. Then there exist positive constants C_1 and C_2 such that*

$$\begin{aligned}\left|\!\left|\!\left| u-u_{\mathcal{T}}^{N}\right|\!\right|\!\right|_{\boldsymbol{\beta}} \le C_1 \left(\left|(\mathcal{L}-\mathcal{L}_{\mathcal{T}})(u_{\mathcal{N}}-u_{\mathcal{T}}^{N},\mathbf{0})\right| + \left|(\mathcal{L}-\mathcal{L}_{\mathcal{T}})(u-u_{\mathcal{N}},\mathbf{0})\right|\right)^{1/2} \\ + C_2\left(\sqrt{\varepsilon_2}+\sqrt{\varepsilon_3}+\inf_{v\in\mathcal{M}(d,n-\hat{n})}\left|\!\left|\!\left|\hat{u}-v\right|\!\right|\!\right|_{\boldsymbol{\beta}}\right).\end{aligned} \tag{13}$$

Proof. The proof is similar to that of Lemma 4.9 in [5]. □

4. Numerical experiments

In this section, we demonstrate the performance of the LSNN method across different settings, incorporating various non-constant jumps. The discrete LS functional was minimized by the Adam optimization algorithm [7] on a uniform mesh with mesh size $h = 10^{-2}$. As in [5], the directional derivative $v_{\boldsymbol{\beta}}$ was approximated by the backward finite difference quotient multiplied by $|\boldsymbol{\beta}|$

$$v_{\boldsymbol{\beta}}(\mathbf{x}_{K}) \approx |\boldsymbol{\beta}|\frac{v(\mathbf{x}_{K})-v\big(\mathbf{x}_{K}-\rho\bar{\boldsymbol{\beta}}(\mathbf{x}_{K})\big)}{\rho}, \tag{14}$$

where $\bar{\boldsymbol{\beta}}=\frac{\boldsymbol{\beta}}{|\boldsymbol{\beta}|}$ and $\rho = h/4$ (except for the fifth test problem, which used $\rho = h/15$). The LSNN method was implemented with an adaptive learning rate that started with 0.004 and was reduced by half for every 50000 iterations (except for the fourth test problem, which reduced for every 100000 iterations). For each experiment, to avoid local minima, 10 ReLU NN functions were trained for 5000 iterations each, and then the experiment began with one of the pretrained network functions that gave the minimum loss.

Tables 1 and 6 report numerical errors in the relative L^2, $V_{\boldsymbol{\beta}}$, and the LS functional with parameters being the total number of weights and biases. Since the input dimensions $d = 2, 3$ and the depth $\lceil \log_2(d+1)\rceil + 1 = 3$ for $d = 2, 3$, we employed ReLU NN functions with 2–n_1–n_2–1 or 3–n_1–n_2–1 representations or structures, which means that the representations have two-hidden-layers with n_1, n_2 neurons, respectively (here 2,3 mean the input dimensions and 1 is the output dimension.) In the fourth test problem for which the discontinuous interface is not a straight line, we also have the approximation of the 2-layer NN known as a universal approximator (see, e.g., [8, 9]) to show how the depth of a neural network impacts the approximation (see [5] for more examples).

All the test problems are defined on the domain $\Omega = (0,1)^2$ or $(0,1)^3$ with $\gamma = 1$ ($f = 1$ for the first three and the last test problems, and $f = 0$ for the remaining test problems).

4.1. A problem with a constant advection velocity field. The first test problem has the constant advective velocity field $\boldsymbol{\beta}(x,y) = (0,1)$, $(x,y) \in \Omega$, and the inflow boundary of the problem is $\Gamma_- = \{(x,0) : x \in (0,1)\}$. The inflow boundary condition is given by

$$g(x,y) = \begin{cases} 1, & (x,y) \in \Gamma_-^1 \equiv \{(x,0) : x \in (0,1/2)\}, \\ 2, & (x,y) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1. \end{cases}$$

The exact solution of this test problem is

$$u(x,y) = \begin{cases} 1, & (x,y) \in \Omega_1 = \{(x,y) \in \Omega : x < 1/2\}, \\ 1+e^{-y}, & (x,y) \in \Omega \setminus \Omega_1. \end{cases}$$

The LSNN method was implemented with 50000 iterations for 2–20–20–1 ReLU NN functions. We report the numerical results in Figure 2 and Table 1. The traces (Figure 2(b)) of the exact and numerical solutions on the plane $y = 0.5$ show no difference or oscillation. The exact solution (Figure 2(c)), which has a non-constant jump along the vertical interface (Figure 2(a)) is accurately approximated by a 3-layer ReLU NN function (Figure 2(d) and Table 1). We note that the solution of this test problem takes the same form as $\chi(\mathbf{x})$ in (9), which was approximated by a CPWL function constructed by partitioning the domain into rectangles stacking on top of each other. It appears from Figure 2(e) that the 3-layer ReLU NN function approximation has a similar partition, and the second-layer breaking hyperplanes were generated for approximating the jump along the discontinuous interface and the non-constant part of the solution, which is consistent with our theoretical analysis on the convergence of the method.

TABLE 1. Relative errors of the problem in Subsection 4.1.

Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
2–20–20–1	0.037881	0.007044	0.005391	501

4.2. A problem with a piecewise smooth solution. This example is a modification of Subsection 4.1 by changing the inflow boundary condition to

$$g(x,y) = \begin{cases} 0, & (x,y) \in \Gamma_-^1 \equiv \{(x,0) : x \in (0,1/2)\}, \\ 2, & (x,y) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1. \end{cases}$$

The exact solution of this test problem is

$$u(x,y) = \begin{cases} 1 - e^{-y}, & (x,y) \in \Omega_1, \\ 1 + e^{-y}, & (x,y) \in \Omega_2. \end{cases} \tag{15}$$

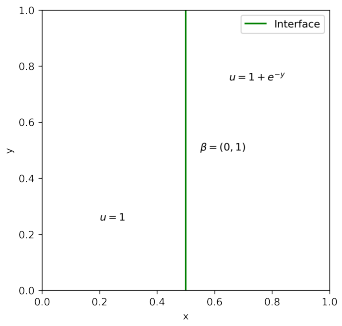
The LSNN method was implemented with 50000 iterations for 2–20–20–1 ReLU NN functions. We report the numerical results in Figure 3 and Table 2. Unlike the previous test problem, the exact solution (Figure 3(c)) consists of two non-constant smooth parts. The LSNN method is capable of approximating the solution accurately without oscillation (Figures 3(b) to 3(d) and Table 2). The 3-layer ReLU NN function approximation has a partition (Figure 3(e)) similar to that in Subsection 4.1 with the second-layer breaking hyperplanes on both sides for approximating the two non-constant smooth parts of the solution.

TABLE 2. Relative errors of the problem in Subsection 4.2.

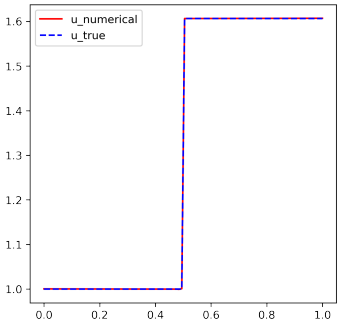
Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
2–20–20–1	0.078036	0.013157	0.010386	501

4.3. A problem with a piecewise smooth inflow boundary. This example is again a modification of Subsection 4.1 by changing the inflow boundary condition to

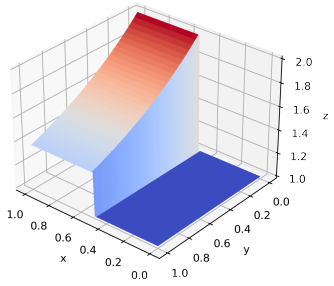
$$g(x,y) = \begin{cases} 1 - \sin(2\pi x), & (x,y) \in \Gamma_-^1 \equiv \{(x,0) : x \in (0,1/2)\}, \\ 5/2 - x, & (x,y) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1. \end{cases}$$



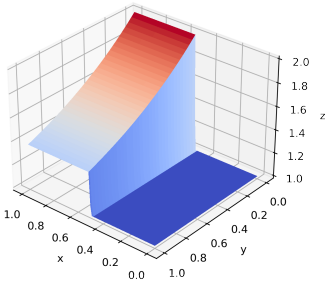
(a) The interface



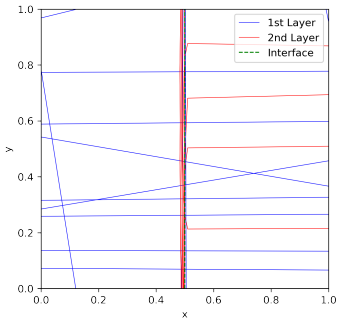
(b) The trace of Figure 2(d) on $y = 0.5$



(c) The exact solution



(d) A 2–20–20–1 ReLU NN function approximation



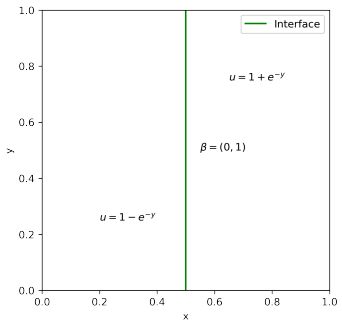
(e) The breaking hyperplanes of the approximation in Figure 2(d)

FIGURE 2. Approximation results of the problem in Subsection 4.1.

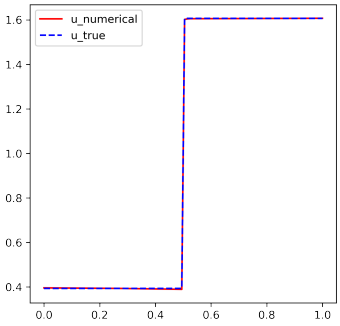
The exact solution of this test problem is

$$u(x,y) = \begin{cases} 1 - \sin(2\pi x)e^{-y}, & (x,y) \in \Omega_1, \\ 1 + (3/2 - x)e^{-y}, & (x,y) \in \Omega_2. \end{cases} \tag{16}$$

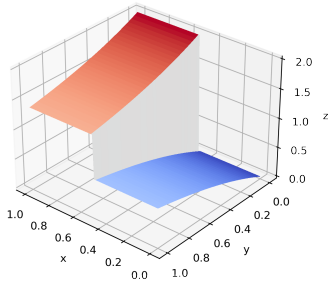
The LSNN method was implemented with 100000 iterations for 2–40–40–1 ReLU NN functions. We report the numerical results in Figure 4 and Table 3. Since the solution on the inflow boundary consists of two non-constant smooth curves, we increased the size of the neural network to obtain a more accurate solution. Figures 4(c) and 4(d) and Table 3 show that the approximation is accurate pointwisely and



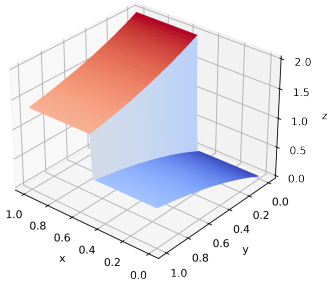
(a) The interface



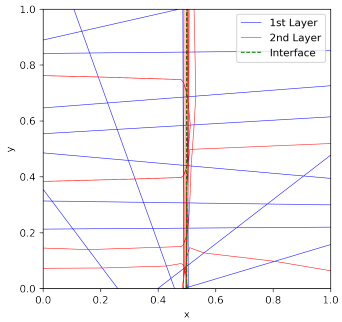
(b) The trace of Figure 3(d) on $y = 0.5$



(c) The exact solution



(d) A 2–20–20–1 ReLU NN function approximation



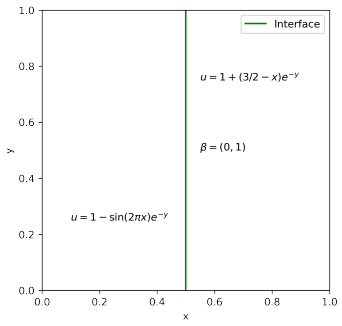
(e) The breaking hyperplanes of the approximation in Figure 3(d)

FIGURE 3. Approximation results of the problem in Subsection 4.2.

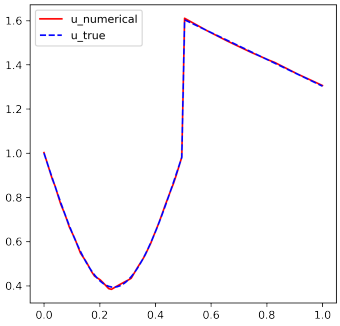
in average. The traces (Figure 4(b)) on $y = 0.5$ exhibit no oscillation and we note a few corners on the curve, verifying that the ReLU NN function approximation is a CPWL function. The partition generated by the breaking hyperplanes (Figure 4(e)) of the approximation shows how the exact solution was approximated.

4.4. A problem with a piecewise constant advection velocity field. Let $\bar{\Omega} = \bar{\Upsilon}_1 \cup \bar{\Upsilon}_2$ and

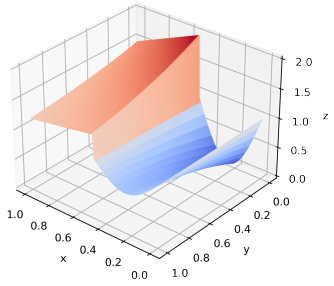
$$\Upsilon_1 = \{(x, y) \in \Omega : y \geq x\} \text{ and } \Upsilon_2 = \Omega \setminus \Upsilon_1.$$



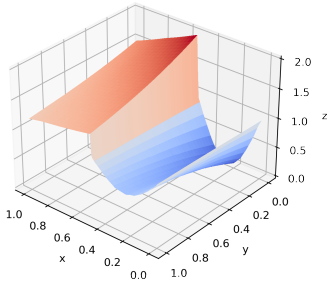
(a) The interface



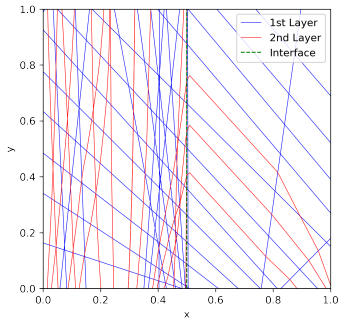
(b) The trace of Figure 4(d) on $y = 0.5$



(c) The exact solution



(d) A 2–40–40–1 ReLU NN function approximation



(e) The breaking hyperplanes of the approximation in Figure 4(d)

FIGURE 4. Approximation results of the problem in Subsection 4.3.

TABLE 3. Relative errors of the problem in Subsection 4.3.

Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
2–40–40–1	0.041491	0.016480	0.012733	1801

This test problem has a piecewise constant advective velocity field given by

$$\boldsymbol{\beta}(x,y) = \begin{cases} (-1, \sqrt{2}-1)^T, & (x,y) \in \Upsilon_1, \\ (1-\sqrt{2}, 1)^T, & (x,y) \in \Upsilon_2, \end{cases} \tag{17}$$

and the inflow boundary $\Gamma_- = \{(1,y) : y \in (0,1)\} \cup \{(x,0) : x \in (0,1)\}$. For the inflow boundary condition,

$$g(x,y) = \begin{cases} x\exp(x/(\sqrt{2}-1)), & (x,y) \in \Gamma_-^1 \equiv \{(x,0) : x \in (0,1)\}, \\ (11+(\sqrt{2}-1)y)\exp(1/(\sqrt{2}-1)), & (x,y) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1, \end{cases}$$

the exact solution of this test problem is

$$u(x,y) = \begin{cases} (y+(\sqrt{2}-1)x)\exp(\sqrt{2}x+y), & (x,y) \in \widehat{\Upsilon}_{11}, \\ (y+(\sqrt{2}-1)x+10)\exp(\sqrt{2}x+y), & (x,y) \in \widehat{\Upsilon}_{12}, \\ (x+(\sqrt{2}-1)y)\exp(x/(\sqrt{2}-1)), & (x,y) \in \widehat{\Upsilon}_{21}, \\ (x+(\sqrt{2}-1)y+10)\exp(x/(\sqrt{2}-1)), & (x,y) \in \widehat{\Upsilon}_{22}, \end{cases} \tag{18}$$

where

$$\widehat{\Upsilon}_{11} = \{(x,y) \in \Upsilon_1 : y < (1-\sqrt{2})x+1\},\ \widehat{\Upsilon}_{12} = \Upsilon_1 \setminus \widehat{\Upsilon}_{11},$$

$$\widehat{\Upsilon}_{21} = \{(x,y) \in \Upsilon_2 : y < \tfrac{1}{1-\sqrt{2}}(x-1)\},\ \text{and}\ \widehat{\Upsilon}_{22} = \Upsilon_2 \setminus \widehat{\Upsilon}_{21}.$$

The LSNN method was implemented with 300000 iterations for 2–450–1 and 2–40–40–1 ReLU NN functions. We report the numerical results in Figure 5 and Table 4. This example compares the approximation differences with the one-hidden-layer NN (a universal approximator) with the same number of degrees of freedom. The exact solution (Figure 5(b)) consists of four non-constant smooth parts and has a non-constant jump along two connected line segments (Figure 5(a)). The traces (Figure 5(f)) of the exact and numerical solutions, the approximation (Figure 5(d)), and Table 4 indicate that the 3-layer ReLU NN function approximation is accurate pointwisely and in average. Most of the second-layer breaking hyperplanes (Figure 5(h)) are along the discontinuous interface, which correspond to the sharp transition layer of the approximation for the jump. On the other hand, Figures 5(c), 5(e), and 5(g) and Table 4 show that the one-hidden-layer NN failed to approximate the solution, especially around the interface.

TABLE 4. Relative errors of the problem in Subsection 4.4.

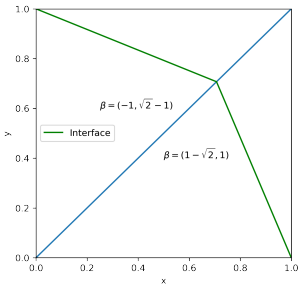
Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
2–40–40–1	0.058884	0.073169	0.038245	1801
2–450–1	0.243956	0.258038	0.225756	1801

4.5. A problem with a variable advection velocity field. This test problem has the variable advective velocity field $\boldsymbol{\beta}(x,y) = (1,2x),\ (x,y) \in \Omega$, and the inflow boundary of the problem is $\Gamma_- = \{(0,y) : y \in (0,1)\} \cup \{(x,0) : x \in (0,1)\}$. The inflow boundary condition is given by

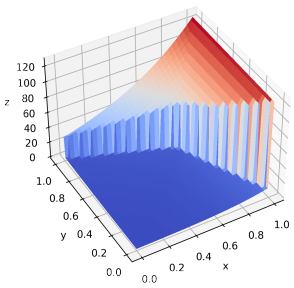
$$g(x,y) = \begin{cases} y+2, & (x,y) \in \Gamma_-^1 \equiv \{(0,y) : y \in [\frac{1}{5},1)\}, \\ (y-x^2)e^{-x}, & (x,y) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1. \end{cases}$$

The exact solution of this test problem is

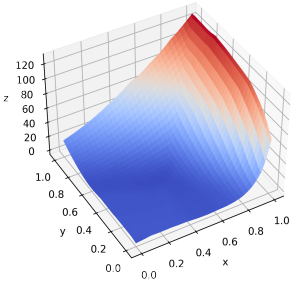
$$u(x,y) = \begin{cases} (y-x^2)e^{-x}, & (x,y) \in \Omega_1 \equiv \{(x,y) \in \Omega : y < x^2 + \frac{1}{5}\}, \\ (y-x^2+2)e^{-x}, & (x,y) \in \Omega_2 = \Omega \setminus \Omega_1. \end{cases} \tag{19}$$



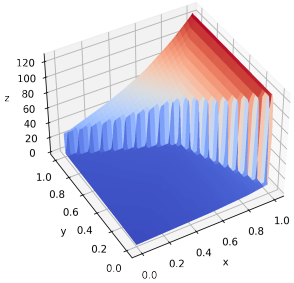
(a) The interface



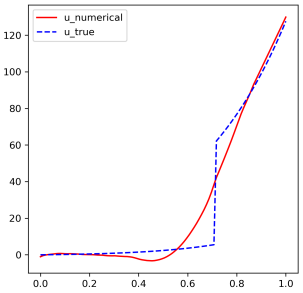
(b) The exact solution



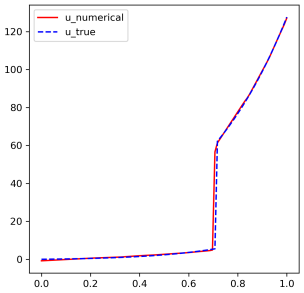
(c) A 2–450–1 ReLU NN function approximation



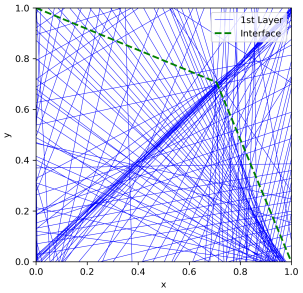
(d) A 2–40–40–1 ReLU NN function approximation



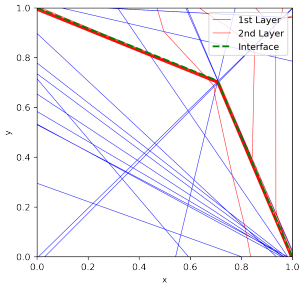
(e) The trace of Figure 5(c) on $y = x$



(f) The trace of Figure 5(d) on $y = x$



(g) The breaking hyperplanes of the approximation in Figure 5(c)



(h) The breaking hyperplanes of the approximation in Figure 5(d)

FIGURE 5. Approximation results of the problem in Subsection 4.4.

The LSNN method was implemented with 300000 iterations for 2–60–60–1 ReLU NN functions. We report the numerical results in Figure 6 and Table 5. Since the advective velocity field is a variable field, we increased the size of the neural network and $\rho = h/15$ was used to compute the finite difference quotient in (14) to take values in one subdomain of the partition. Although theoretical analysis on the convergence of the method in the case of a smooth interface was not conducted, Figures 6(b) to 6(d) and Table 5 show that the LSNN method is still capable of approximating the discontinuous solution with the curved interface accurately without oscillation. Finally, again, most of the second-layer breaking hyperplanes are along the interface (Figure 6(e)) to approximate the discontinuous jump.

TABLE 5. Relative errors of the problem in Subsection 4.5.

Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
2–60–60–1	0.046528	0.049423	0.019995	3901

4.6. A problem with a constant advection velocity field ($d = 3$). The last test problem is a three-dimensional problem defined on the domain $\Omega = (0,1)^3$, and approximation results are depicted on $z = 0.505$. The advective velocity field is the constant field $\boldsymbol{\beta}(x,y,z) = (1,0,0)$, $(x,y,z) \in \Omega$, and the inflow boundary of the problem is $\Gamma_- = \{(0,y,z) : y,z \in (0,1)\}$. The inflow boundary condition is given by

$$g(x,y,z) = \begin{cases} 1-4y, & (x,y,z) \in \Gamma_-^1 \equiv \{(0,y,z) \in \Gamma_- : y < \frac{1}{2}\}, \\ 5-4y, & (x,y,z) \in \Gamma_-^2 = \Gamma_- \setminus \Gamma_-^1. \end{cases}$$

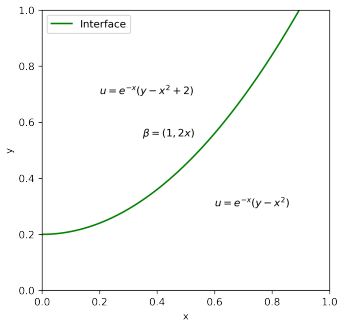
The exact solution of this test problem is

$$u(x,y,z) = \begin{cases} 1-4ye^{-x}, & (x,y,z) \in \Omega_1 \equiv \{(x,y,z) \in \Omega : y < \frac{1}{2}\}, \\ 1+(4-4y)e^{-x}, & (x,y,z) \in \Omega_2 = \Omega \setminus \Omega_1. \end{cases} \tag{20}$$

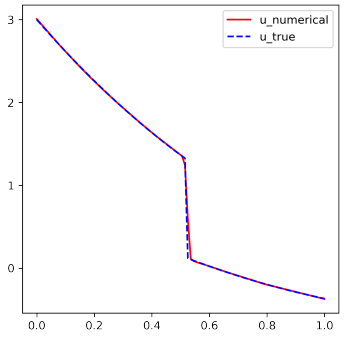
The LSNN method was implemented with 100000 iterations for 3–30–30–1 ReLU NN functions. Note that we still employ the two-hidden-layer NN since $\lceil \log_2(d+1) \rceil + 1 = 3$ for $d = 3$. We report the numerical results in Figure 7 and Table 6. The exact solution (Figure 7(c)) is piecewise smooth along the plane segment $y = 0.5$ (Figure 7(a)) with a non-constant jump, and was approximated accurately by the 3-layer NN (Figures 7(b) and 7(d) and Table 6). The behavior of the breaking hyperplanes (Figure 7(e)) is similar to those of the previous examples around the discontinuous interface and on the subdomains.

TABLE 6. Relative errors of the problem in Subsection 4.6.

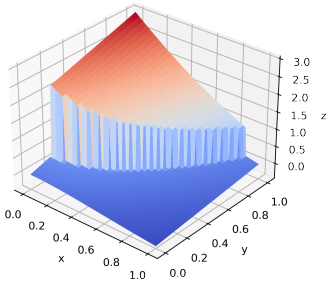
Network structure	$\frac{\|u-u_{\mathcal{T}}^N\|_0}{\|u\|_0}$	$\frac{\|\|\|u-u_{\mathcal{T}}^N\|\|\|_{\boldsymbol{\beta}}}{\|\|\|u\|\|\|_{\boldsymbol{\beta}}}$	$\frac{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{f})}{\mathcal{L}^{1/2}(u_{\mathcal{T}}^N,\mathbf{0})}$	Parameters
3–30–30–1	0.005082	0.027765	0.022519	1081



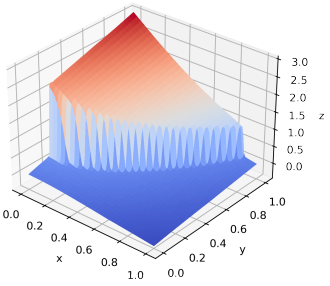
(a) The interface



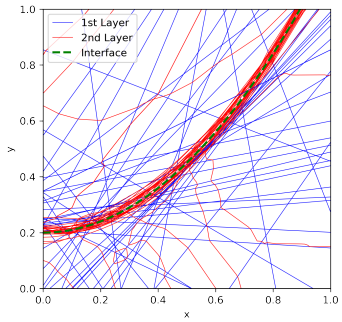
(b) The trace of Figure 6(d) on $y = 1-x$



(c) The exact solution



(d) A 2–60–60–1 ReLU NN function approximation

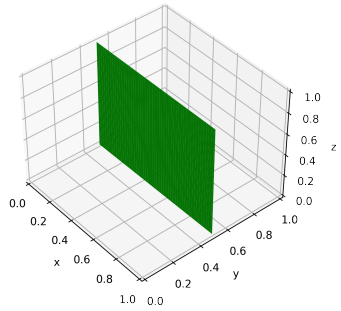


(e) The breaking hyperplanes of the approximation in Figure 6(d)

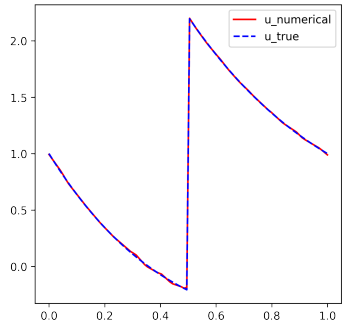
FIGURE 6. Approximation results of the problem in Subsection 4.5.

5. Conclusion

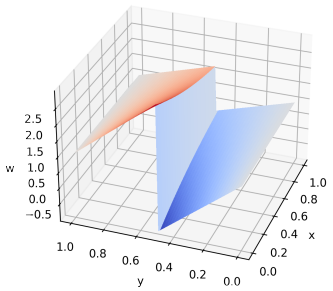
In this paper, we used the least-squares ReLU neural network (LSNN) method for solving linear advection-reaction equation with discontinuous solution having non-constant jumps. The method, being mesh-free, requires no mesh to resolve the interfacial discontinuity. We proved theoretically that ReLU neural network (NN) functions with $\lceil \log_2(d+1) \rceil + 1$-layer representations are capable of accurately approximating solutions with non-constant jumps along discontinuous interfaces that are not necessarily straight lines. Our theoretical findings were validated by



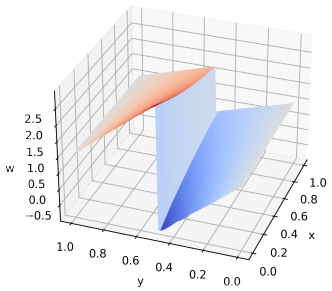
(a) The interface



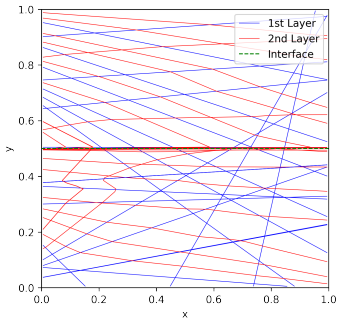
(b) The trace of Figure 7(d) on $y = x$



(c) The exact solution



(d) A 3–30–30–1 ReLU NN function approximation



(e) The breaking hyperplanes of the approximation in Figure 7(d)

FIGURE 7. Approximation results of the problem in Subsection 4.6.

multiple numerical examples with $d = 2, 3$ and various non-constant jumps and interface shapes, demonstrating accurate performance of the method.

The approximation of discontinuous functions by NNs is also encountered in classification tasks, and our results suggest that we can achieve accurate predictions with properly designed neural network architectures. However, as in [5], in this paper, we mainly focused on the depth of NNs. The approximation of discontinuous classification functions by NNs with fixed depth and width will be addressed in a forthcoming paper.

Acknowledgments

The authors thank the anonymous reviewers for multiple valuable comments and suggestions that improved the paper. This work was supported in part by the National Science Foundation under grant DMS-2110571.

References

[1] R. Arora, A. Basu, P. Mianjy, and A. Mukherjee. Understanding deep neural networks with rectified linear units. International Conference on Learning Representations, 2018.
[2] P. Bochev and J. Choi. Improved least-squares error estimates for scalar hyperbolic problems. Comput. Methods Appl. Math., 1:115–124, 2001.
[3] Z. Cai, J. Chen, and M. Liu. Least-squares neural network (LSNN) method for scalar nonlinear hyperbolic conservation laws: discrete divergence operator. J. Comput. Appl. Math., 433:115298, 2023.
[4] Z. Cai, J. Chen, and M. Liu. Least-squares ReLU neural network (LSNN) method for linear advection-reaction equation. J. Comput. Phys., 443:110514, 2021.
[5] Z. Cai, J. Choi, and M. Liu. Least-squares neural network (LSNN) method for linear advection-reaction equation: discontinuity interface. SIAM J. Sci. Comput., 46:C448-C478, 2024.
[6] H. De Sterck, T. A. Manteuffel, S. F. McCormick, and L. Olson. Least-squares finite element methods and algebraic multigrid solvers for linear hyperbolic PDEs. SIAM J. Sci. Comput., 26:31–54, 2004.
[7] D. P. Kingma and J. Ba. ADAM: A method for stochastic optimization. International Conference on Learning Representations, 2015.
[8] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks, 6:861–867, 1993.
[9] A. Pinkus. Approximation theory of the MLP model in neural networks. Acta Numer., 8:143–195, 1999.
[10] J. Tarela and M. Martinez. Region configurations for realizability of lattice piecewise-linear models. Math. Comput. Modelling., 30:17–27, 1999.
[11] S. Wang and X. Sun. Generalization of hinging hyperplanes. IEEE Trans. Inform. Theory., 51:4425–4431, 2005.

Appendix A. The proof of Theorem 3.1

In this section, we provide a proof of Theorem 3.1 by constructing a CPWL function to approximate $\chi(\mathbf{x})$ in (9). We note that $a(\mathbf{x})$ is generally a cylindrical surface and that the jump of $\chi(\mathbf{x})$ is non-constant with

$$\chi(x,0) = a(x,0) = \alpha_1 - \alpha_2 \text{ for } x \in [0, x_0]$$

(see Figure 8(a)).

For a given $\varepsilon_1 > 0$, we take $\widehat{\Upsilon} = (0,1) \times (y_0, y_0 + \varepsilon_1)$ (see Figure 8(b)). Without loss of generality, we let $\alpha_1 = 1$, $\alpha_2 = 0$, and $\widehat{\Upsilon} = (0,1) \times (0, \varepsilon_1)$.

Hence,

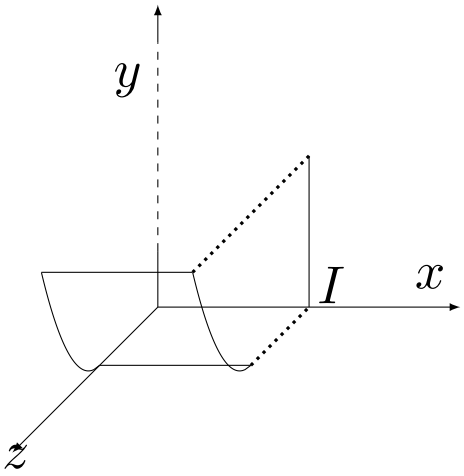
$$\chi(\mathbf{x}) = \chi_0(\mathbf{x}) + \chi_1(\mathbf{x}) \text{ on } \widehat{\Upsilon}, \tag{A.1}$$

where $\chi_0(\mathbf{x})$ is a step function and $\chi_1(\mathbf{x})$ vanishes on the inflow boundary given by

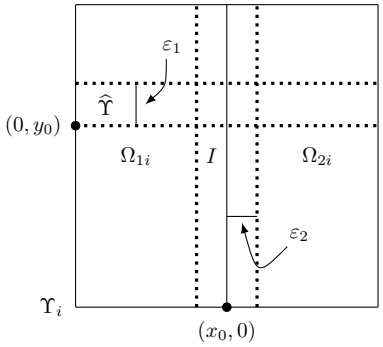
$$\chi_0(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}, \\ 0, & \mathbf{x} \in \Omega_{2i} \cap \widehat{\Upsilon}, \end{cases} \quad \text{and} \quad \chi_1(\mathbf{x}) = \begin{cases} a(\mathbf{x}) - 1, & \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}, \\ 0, & \mathbf{x} \in \Omega_{2i} \cap \widehat{\Upsilon}. \end{cases}$$

Lemma A.1. *Let*

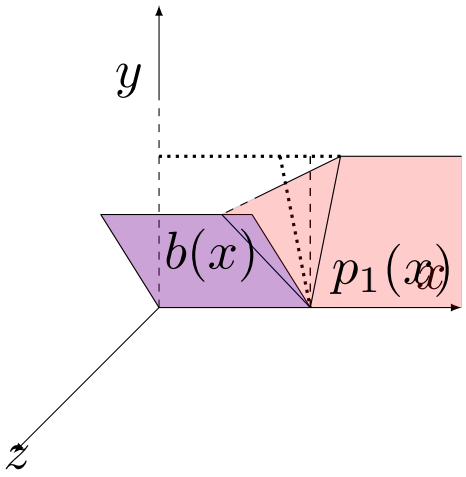
$$b(\mathbf{x}) = \begin{cases} \boldsymbol{b} \cdot (\mathbf{x} - (x_0, 0)), & \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}, \\ 0, & \mathbf{x} \in \Omega_{2i} \cap \widehat{\Upsilon}, \end{cases}$$



(a) A cylindrical surface $a(\mathbf{x})$



(b) $(0, 1) \times (y_0, y_0 + \varepsilon_1)$



(c) $b(x)$ in purple and $p_1(x)$ in pink

FIGURE 8. An illustration of the convergence analysis on one subdomain Υ_i.

where $\boldsymbol{b} = (0, d)^T$ *is a constant vector, and let* $p_1(\mathbf{x})$ *be the two-layer neural network function on* $\widehat{\Upsilon}$ *defined by*

$$p_1(\mathbf{x}) = -c\,\sigma(\mathbf{w}_1 \cdot \mathbf{x} + x_0) + c\,\sigma(\mathbf{w}_2 \cdot \mathbf{x} + x_0)$$

with the weights and coefficient

$$\mathbf{w}_1 = \begin{pmatrix} -1 \\ -\varepsilon_2 \end{pmatrix}, \quad \mathbf{w}_2 = \begin{pmatrix} -1 \\ \varepsilon_2 \end{pmatrix}, \quad c = \frac{d}{2\varepsilon_2}$$

(see Figure 8(c)). Then we have on $\widehat{\Upsilon}$*,*

$$\text{(A.2)} \qquad |||b - p_1|||_{\boldsymbol{\beta}} = \left(\|b - p_1\|^2_{0,\widehat{\Upsilon}} + \|b_{\boldsymbol{\beta}} - p_{1\boldsymbol{\beta}}\|^2_{0,\widehat{\Upsilon}} \right)^{1/2} \leq \sqrt{\frac{\varepsilon_1^3}{24} + \frac{B^2}{4}}\,|d|\,\sqrt{\varepsilon_1\varepsilon_2},$$

where we assume $|v_2(\mathbf{x})| \leq B$ *(*$\boldsymbol{\beta}(\mathbf{x}) = (0, v_2(\mathbf{x}))$*).*

Proof. Let us denote

$$\widehat{\Upsilon}_{\varepsilon_2} \equiv \widehat{\Upsilon}_{1,\varepsilon_2} \cup \widehat{\Upsilon}_{2,\varepsilon_2} \equiv \{\mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon} : \mathbf{w}_1 \cdot \mathbf{x} + x_0 < 0\} \cup \{\mathbf{x} \in \Omega_{2i} \cap \widehat{\Upsilon} : \mathbf{w}_2 \cdot \mathbf{x} + x_0 > 0\}.$$

Then we have

$$b(\mathbf{x}) - p_1(\mathbf{x}) = \begin{cases} -c(\mathbf{w}_1 \cdot \mathbf{x} + x_0) & \mathbf{x} \in \widehat{\Upsilon}_{1,\varepsilon_2}, \\ -c(\mathbf{w}_2 \cdot \mathbf{x} + x_0) & \mathbf{x} \in \widehat{\Upsilon}_{2,\varepsilon_2}, \\ 0, & \mathbf{x} \in \widehat{\Upsilon} \setminus \widehat{\Upsilon}_{\varepsilon_2}. \end{cases}$$

Calculating the double integral,

$$\text{(A.3)} \qquad \|b - p_1\|^2_{0,\widehat{\Upsilon}} = \|b - p_1\|^2_{0,\widehat{\Upsilon}_{1,\varepsilon_2}} + \|b - p_1\|^2_{0,\widehat{\Upsilon}_{2,\varepsilon_2}} = \frac{d^2}{24}\varepsilon_1^4\varepsilon_2,$$

and using the directional derivative,

$$\text{(A.4)} \qquad \|b_{\boldsymbol{\beta}} - p_{1\boldsymbol{\beta}}\|^2_{0,\widehat{\Upsilon}} = \int_{\widehat{\Upsilon}_{1,\varepsilon_2}} (c\mathbf{w}_1 \cdot \boldsymbol{\beta})^2\,d\mathbf{x} + \int_{\widehat{\Upsilon}_{2,\varepsilon_2}} (c\mathbf{w}_2 \cdot \boldsymbol{\beta})^2\,d\mathbf{x} \leq \frac{(dB)^2}{4}\varepsilon_1\varepsilon_2.$$

Now (A.2) follows from (A.3) and (A.4). □

Lemma A.2. *Let* $\widehat{\Upsilon}$*,* I*,* $b(\mathbf{x})$*,* $p_1(\mathbf{x})$*, and* $\boldsymbol{\beta}(\mathbf{x})$ *be as in* Lemma A.1 *with* $d = \chi_1(0, \varepsilon_1)/\varepsilon_1$*, and let* $p_0(\mathbf{x})$ *be the two-layer neural network function on* $\widehat{\Upsilon}$ *defined by*

$$\text{(A.5)} \qquad p_0(\mathbf{x}) = \frac{1}{2\varepsilon_2}\left(\sigma(x - x_0 + \varepsilon_2) - \sigma(x - x_0 - \varepsilon_2)\right).$$

Then we have on $\widehat{\Upsilon}$*,*

$$\text{(A.6)} \qquad |||\chi - (p_0 + p_1)|||_{\boldsymbol{\beta}} \leq C_1\sqrt{\varepsilon_1\varepsilon_2} + C_2\sqrt{\varepsilon_1},$$

where C_2 *is given by the square root of*

$$\text{(A.7)} \quad \sup\{(u_1(\mathbf{x}) - u_2(\mathbf{x}) - 1 - b(\mathbf{x}))^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}\}x_0 + \sup\{2\gamma(\mathbf{x})^2(u_2(\mathbf{x}) - u_1(\mathbf{x}))^2 + 2(dB)^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}\}x_0.$$

Proof. From the triangle inequality,

$$\text{(A.8)} \quad |||\chi - (p_0 + p_1)|||_{\boldsymbol{\beta}} = |||\chi_0 + \chi_1 - (p_0 + p_1)|||_{\boldsymbol{\beta}} \leq |||\chi_0 - p_0|||_{\boldsymbol{\beta}} + |||\chi_1 - p_1|||_{\boldsymbol{\beta}}.$$

Since $\|\chi_{0\boldsymbol{\beta}} - p_{0\boldsymbol{\beta}}\| = 0$, calculating the double integral,

$$\begin{aligned} |||\chi_0 - p_0|||_{\boldsymbol{\beta}} &= \left(\|\chi_0 - p_0\|^2_{0,\widehat{\Upsilon}} + \|\chi_{0\boldsymbol{\beta}} - p_{0\boldsymbol{\beta}}\|^2_{0,\widehat{\Upsilon}}\right)^{1/2} \\ &= \|\chi_0 - p_0\|_{0,\widehat{\Upsilon}} \\ &= \frac{1}{\sqrt{6}}\sqrt{\varepsilon_1\varepsilon_2}. \end{aligned} \tag{A.9}$$

Next, again, by the triangle inequality,

$$|||\chi_1 - p_1|||_{\boldsymbol{\beta}} \le |||\chi_1 - b|||_{\boldsymbol{\beta}} + |||b - p_1|||_{\boldsymbol{\beta}}.$$

By Lemma A.1

$$|||b - p_1|||_{\boldsymbol{\beta}} \le \sqrt{\frac{\varepsilon_1^3}{24} + \frac{B^2}{4}}\,|d|\,\sqrt{\varepsilon_1\varepsilon_2}.$$

To bound $|||\chi_1 - b|||_{\boldsymbol{\beta}}$, we recall the definition of the graph norm,

$$|||\chi_1 - b|||_{\boldsymbol{\beta}} = \left(\|\chi_1 - b\|^2_{0,\widehat{\Upsilon}} + \|\chi_{1\boldsymbol{\beta}} - b_{\boldsymbol{\beta}}\|^2_{0,\widehat{\Upsilon}}\right)^{1/2}.$$

First we have

$$\begin{aligned} \|\chi_1 - b\|^2_{0,\widehat{\Upsilon}} &= \|\chi_1 - b\|^2_{0,\Omega_{1i}\cap\widehat{\Upsilon}} \\ &\le \sup\{(\chi_1(\mathbf{x}) - b(\mathbf{x}))^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}\}\varepsilon_1 x_0 \\ &= \sup\{(u_1(\mathbf{x}) - u_2(\mathbf{x}) - 1 - b(\mathbf{x}))^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}\}\varepsilon_1 x_0. \end{aligned}$$

Next, observing $b_{\boldsymbol{\beta}} = dv_2$ and from (2),

$$\chi_{1\boldsymbol{\beta}} = (u_1 - u_2 - 1)_{\boldsymbol{\beta}} = (u_1 - u_2)_{\boldsymbol{\beta}} = \gamma(u_2 - u_1),$$

we have

$$\begin{aligned} \|\chi_{1\boldsymbol{\beta}} - b_{\boldsymbol{\beta}}\|^2_{0,\widehat{\Upsilon}} &= \|\chi_{1\boldsymbol{\beta}} - b_{\boldsymbol{\beta}}\|^2_{0,\Omega_{1i}\cap\widehat{\Upsilon}} \\ &= \|\gamma(u_2 - u_1) - dv_2\|^2_{0,\Omega_{1i}\cap\widehat{\Upsilon}} \\ &\le \left\|\sqrt{2\gamma^2(u_2 - u_1)^2 + 2(dv_2)^2}\right\|^2_{0,\Omega_{1i}\cap\widehat{\Upsilon}} \\ &\le \sup\{2\gamma(\mathbf{x})^2(u_2(\mathbf{x}) - u_1(\mathbf{x}))^2 + 2(dB)^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}\}\varepsilon_1 x_0. \end{aligned}$$

Now (A.6) follows from combining the above inequalities. □

Given $\varepsilon_3 > 0$, let us choose $\varepsilon_1 = 1/m$ such that

$$\begin{aligned} &\sup\{(u_1(\mathbf{x}) - u_2(\mathbf{x}) - \chi(0, j/m) - b_j(\mathbf{x}))^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}_j\}, \\ &\qquad \sup\{2\gamma(\mathbf{x})^2(u_2(\mathbf{x}) - u_1(\mathbf{x}))^2 + 2(d_j B)^2 : \mathbf{x} \in \Omega_{1i} \cap \widehat{\Upsilon}_j\} < \varepsilon_3, \end{aligned} \tag{A.10}$$

where $\widehat{\Upsilon}_j = (0,1) \times (j/m, (j+1)/m)$ for $j = 0, \ldots, m-1$, and $b_j(\mathbf{x})$ and d_j are as in Lemma A.2. Then we define on each $\widehat{\Upsilon}_j$, $p_{0j}(\mathbf{x}), p_{1j}(\mathbf{x})$ as in Lemma A.2, and construct the CPWL function $p_i(\mathbf{x})$ on Υ_i defined by

$$p_i(\mathbf{x}) = p_{0j}(\mathbf{x}) + p_{1j}(\mathbf{x}),\ \mathbf{x} \in \widehat{\Upsilon}_j.$$

Proof of Theorem 3.1. By Lemma A.2 and the given condition,

$$
\begin{aligned}
|||\chi - p_i|||_{\boldsymbol{\beta}} &= \left(\sum_{j=0}^{m-1} |||\chi - (p_{0j} + p_{1j})|||_{\boldsymbol{\beta}}^2 \right)^{1/2} \\
&\leq \left(\sum_{j=0}^{m-1} \left(D_{1j}\sqrt{\varepsilon_1\varepsilon_2} + D_{2j}\sqrt{\varepsilon_1\varepsilon_3} \right)^2 \right)^{1/2} \\
&\leq \left(m \max_{0\leq j\leq m-1} \left(D_{1j}\sqrt{\varepsilon_1\varepsilon_2} + D_{2j}\sqrt{\varepsilon_1\varepsilon_3} \right)^2 \right)^{1/2} \\
&= \sqrt{m} \max_{0\leq j\leq m-1} \left(D_{1j}\sqrt{\varepsilon_1\varepsilon_2} + D_{2j}\sqrt{\varepsilon_1\varepsilon_3} \right) \\
&= \max_{0\leq j\leq m-1} \left(D_{1j}\sqrt{\varepsilon_2} + D_{2j}\sqrt{\varepsilon_3} \right),
\end{aligned}
\tag{A.11}
$$

where for the first identity, each norm on the right-hand side is taken over $\widehat{\Upsilon}_j$. Now $D_1 = D_{1k}$ and $D_2 = D_{2k}$ for some $0 \leq k \leq m-1$. □

Department of Mathematics, Purdue University, 150 N. University Street, West Lafayette, IN 47907-2067

E-mail: caiz@purdue.edu and choi508@purdue.edu

School of Mechanical Engineering, Purdue University, 585 Purdue Mall, West Lafayette, IN 47907-2088

E-mail: liu66@purdue.edu